PIC18F97J94 FAMILY

8-Bit LCD Flash Microcontroller with USB and XLP Technology

eXtreme Low-Power Features

- Multiple Power Management Options for Extreme Power Reduction:
  - VBAT allows for lowest power consumption on back-up battery (with or without RTCC)
  - Deep Sleep allows near total power-down with the ability to wake-up on external triggers
  - Sleep and Idle modes selectively shut down peripherals and/or core for substantial power reduction and fast wake-up
- Alternate Clock modes Allow On-the-Fly Switching to a Lower Clock Speed for Selective Power Reduction
- Extreme Low-Power Current Consumption for Deep Sleep:
  - WDT: 650 nA @ 2V typical
  - RTCC: 650 nA @ 32 kHz, 2V typical
  - Deep Sleep current, 80 nA typical

Universal Serial Bus Features

- USB V2.0 Compliant
- Low Speed (1.5 Mb/s) and Full Speed (12 Mb/s)
- Supports Control, Interrupt, Isochronous and Bulk Transfers
- Supports up to 32 Endpoints (16 bidirectional)
- USB module can use Any RAM Location on the Device as USB Endpoint Buffers
- On-Chip USB Transceiver

Peripheral Features

- LCD Display Controller:
  - Up to 60 segments by 8 commons
  - Internal charge pump and low-power, internal resistor biasing
  - Operation in Sleep mode
- Up to Four External Interrupt Sources
- Peripheral Pin Select Lite (PPS-Lite):
  - Allows independent I/O mapping of many peripherals
- Four 16-Bit and Four 8-Bit Timers/Counters with Prescaler
- Seven Capture/Compare/PWM (CCP) modules
- Three Enhanced Capture/Compare/PWM (ECCP) modules:
  - One, two or four PWM outputs
  - Selectable polarity
  - Programmable dead time
  - Auto-shutdown and auto-restart
  - Pulse steering control
- Hardware Real-Time Clock/Calendar (RTCC):
  - Runs in Deep Sleep and VBAT modes
- Two Master Synchronous Serial Ports (MSSP) modules Featuring:
  - 3-Wire/4-Wire SPI (all 4 modes)
  - SPI Direct Memory Access (DMA) channel w/1024 byte count
  - Two I^2C modules Support Multi-Master/Slave mode and 7-Bit/10-Bit Addressing
- Four Enhanced Addressable USART modules:
  - Support RS-485, RS-232 and LIN/J2602
  - On-chip hardware encoder/decoder for IrDA®
  - Auto-wake-up on Auto-Baud Detect
- Digital Signal Modulator Provides On-Chip OOK, FSK and PSK Modulation for a Digital Signal Stream
- High-Current Sink/Source 18 mA/18 mA on all Digital I/O
- Configurable Open-Drain Outputs on ECCP/CCP/USART/MSSP
- Extended Microcontroller mode Using 12, 16 or 20-Bit Addressing mode

Analog Features

- 10/12-Bit, 24-Channel Analog-to-Digital (A/D) Converter:
  - Conversion rate of 500 ksps (10-bit), 200 kbps (12-bit)
  - Conversion available during Sleep and Idle
- Three Rail-to-Rail Enhanced Analog Comparators with Programmable Input/Output Configuration
- On-Chip Programmable Voltage Reference
- Charge Time Measurement Unit (CTMU):
  - Used for capacitive touch sensing, up to 24 channels
  - Time measurement down to 1 ns resolution
  - CTMU temperature sensing

High-Performance CPU

- High-Precision PLL for USB
- Two External Clock modes, Up to 64 MHz (16 MIPS®)
- Internal 31 kHz Oscillator
- High-Precision Internal Oscillator with Clock Recovery from SOSC to Achieve 0.15% Precision, 31 kHz to 8 MHz or 64 MHz w/PLL, ±0.15% Typical, ±1.5% Max.
- Secondary Oscillator using Timer1 @ 32 kHz
- C Compiler Optimized Instruction Set Architecture
- Two Address Generation Units for Separate Read and Write Addressing of Data Memory

Special Microcontroller Features

- Operating Voltage Range of 2.0V to 3.6V
- Two On-Chip Voltage Regulators (1.8V and 1.2V) for Regular and Extreme Low-Power Operation
- 20,000 Erase/Write Cycle Endurance Flash Program Memory, Typical
- Flash Data Retention: 10 Years Minimum
- Self-Programmable under Software Control
- Two Configurable Reference Clock Outputs (REFO1 and REFO2)
- In-Circuit Serial Programming™ (ICSP™)
- Fail-Safe Clock Monitor Operation:
  - Detects clock failure and switches to on-chip, low-power RC oscillator
- Power-on Reset (POR), Power-up Timer (PWRT) and Oscillator Start-up Timer (OST)
- Brown-out Reset (BOR) with Operation Below V_{BOR}, with Regulator Enabled
- High/Low-Voltage Detect (HLVD)
- Flexible Watchdog Timer (WDT) with its Own RC Oscillator for Reliable Operation
- Standard and Ultra Low-Power Watchdog Timers (WDT) for Reliable Operation in Standard and Deep Sleep modes

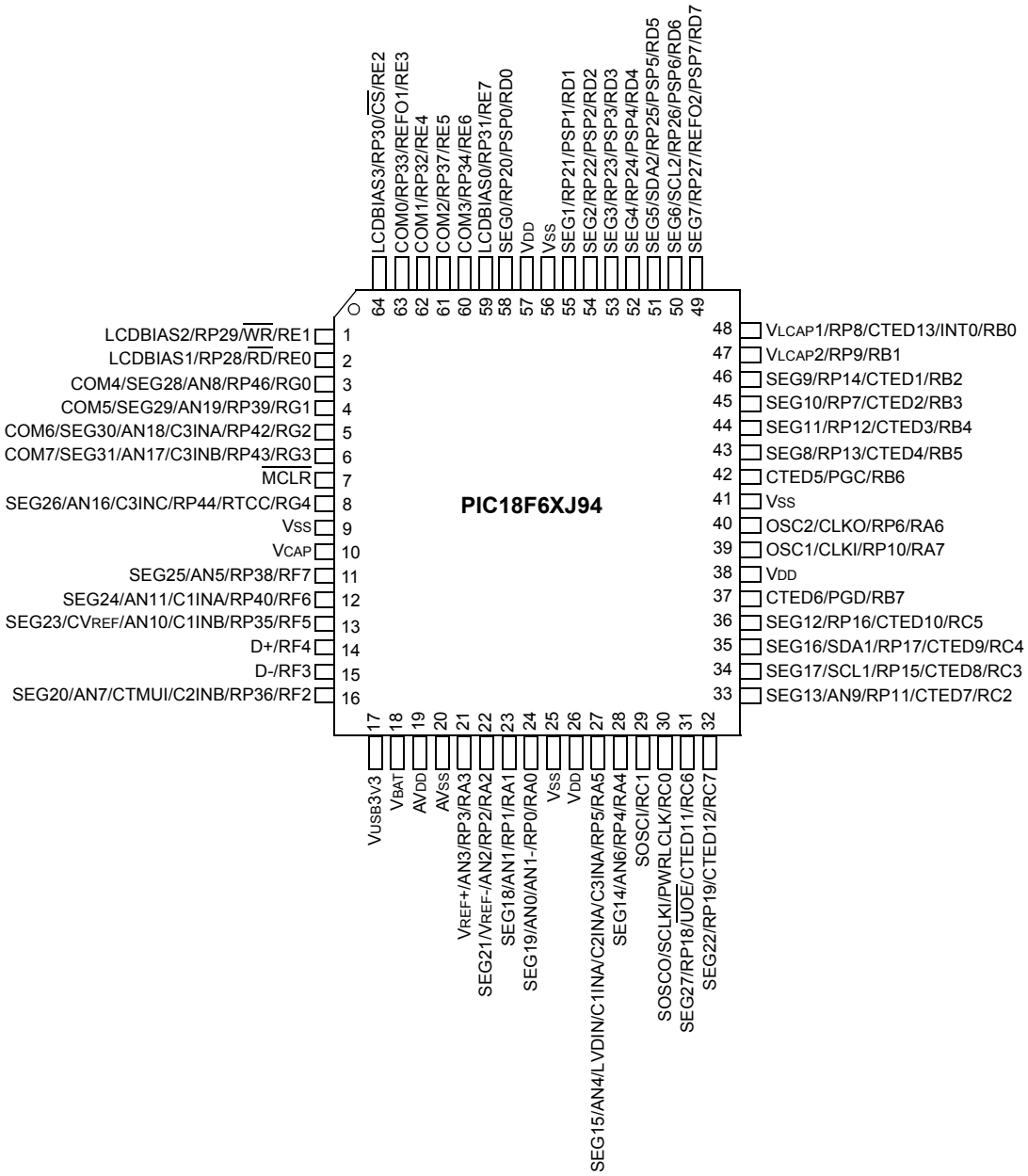
TABLE 1: PIC18F97J94 FAMILY TYPES

Device	Pins	Memory		Remappable Peripherals					I^2C	10/12-Bit A/D (ch)	CTMU	LCD (pixels)	USB	Deep Sleep w/V_{BAT}	PPS (Lite)
		Flash Program (bytes)	Data SRAM (bytes)	Timers 8-Bit/16-Bit	USART w/IrDA®	SPI w/ DMA	Comparators	CCP/ECCP							
PIC18F97J94	100	128K	4K	4	4	2	3	Y	2	24	Y	480	Y	Y	Lite
PIC18F87J94	80	128K	4K	4	4	2	3	Y	2	24	Y	352	Y	Y	Lite
PIC18F67J94	64	128K	4K	4	4	2	3	Y	2	16	Y	224	Y	Y	Lite
PIC18F96J94	100	64K	4K	4	4	2	3	Y	2	24	Y	480	Y	Y	Lite
PIC18F86J94	80	64K	4K	4	4	2	3	Y	2	24	Y	352	Y	Y	Lite
PIC18F66J94	64	64K	4K	4	4	2	3	Y	2	16	Y	224	Y	Y	Lite
PIC18F95J94	100	32K	4K	4	4	2	3	Y	2	24	Y	480	Y	Y	Lite
PIC18F85J94	80	32K	4K	4	4	2	3	Y	2	24	Y	352	Y	Y	Lite
PIC18F65J94	64	32K	4K	4	4	2	3	Y	2	16	Y	224	Y	Y	Lite

For other small form-factor package availability and marking information, visit http://www.microchip.com/packaging or contact your local sales office.

PIN DIAGRAMS

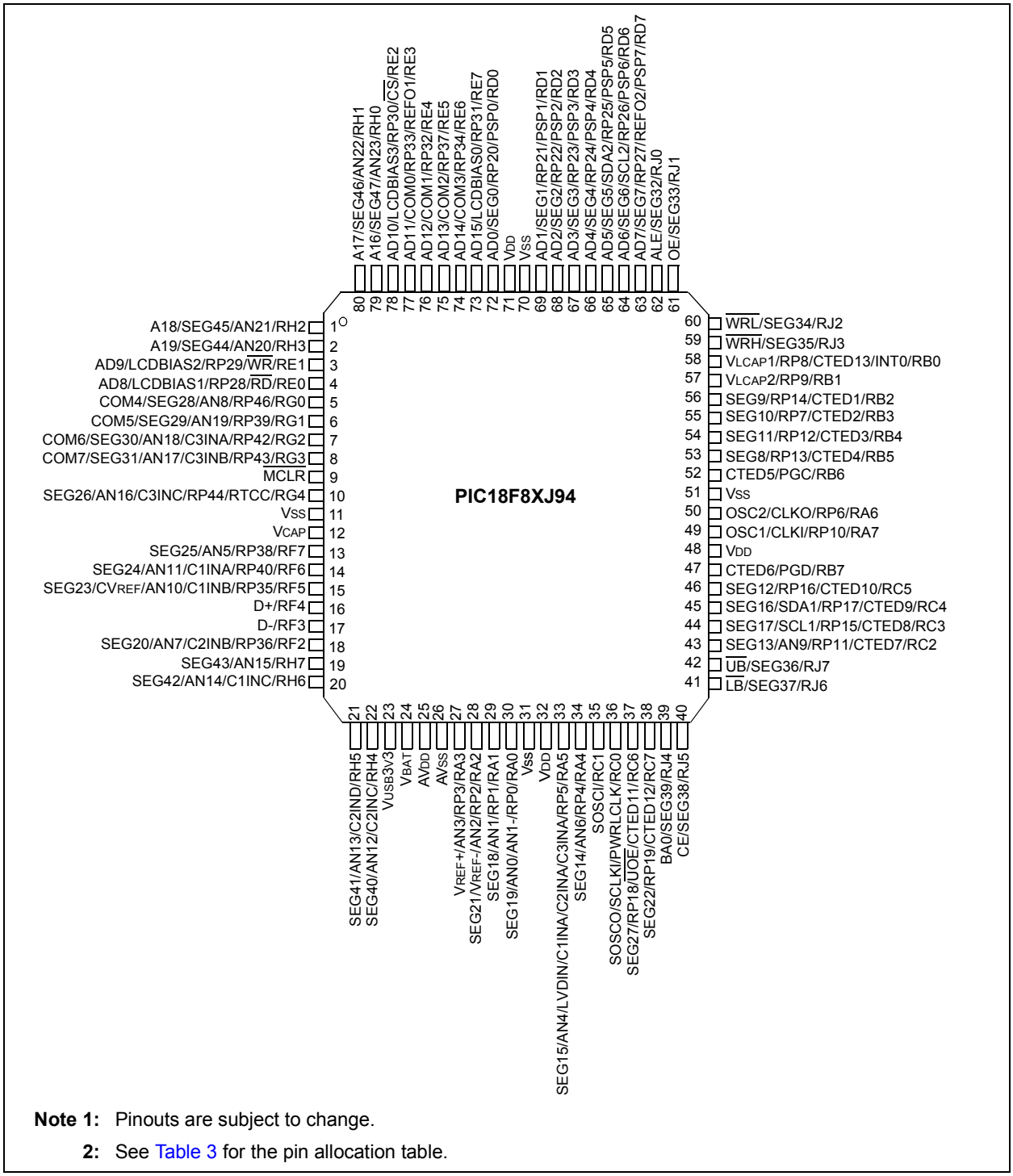
FIGURE 1: 64-PIN TQFP, QFN DIAGRAM FOR PIC18F6XJ94



Note 1: Pinouts are subject to change.

2: See Table 2 for the pin allocation table.

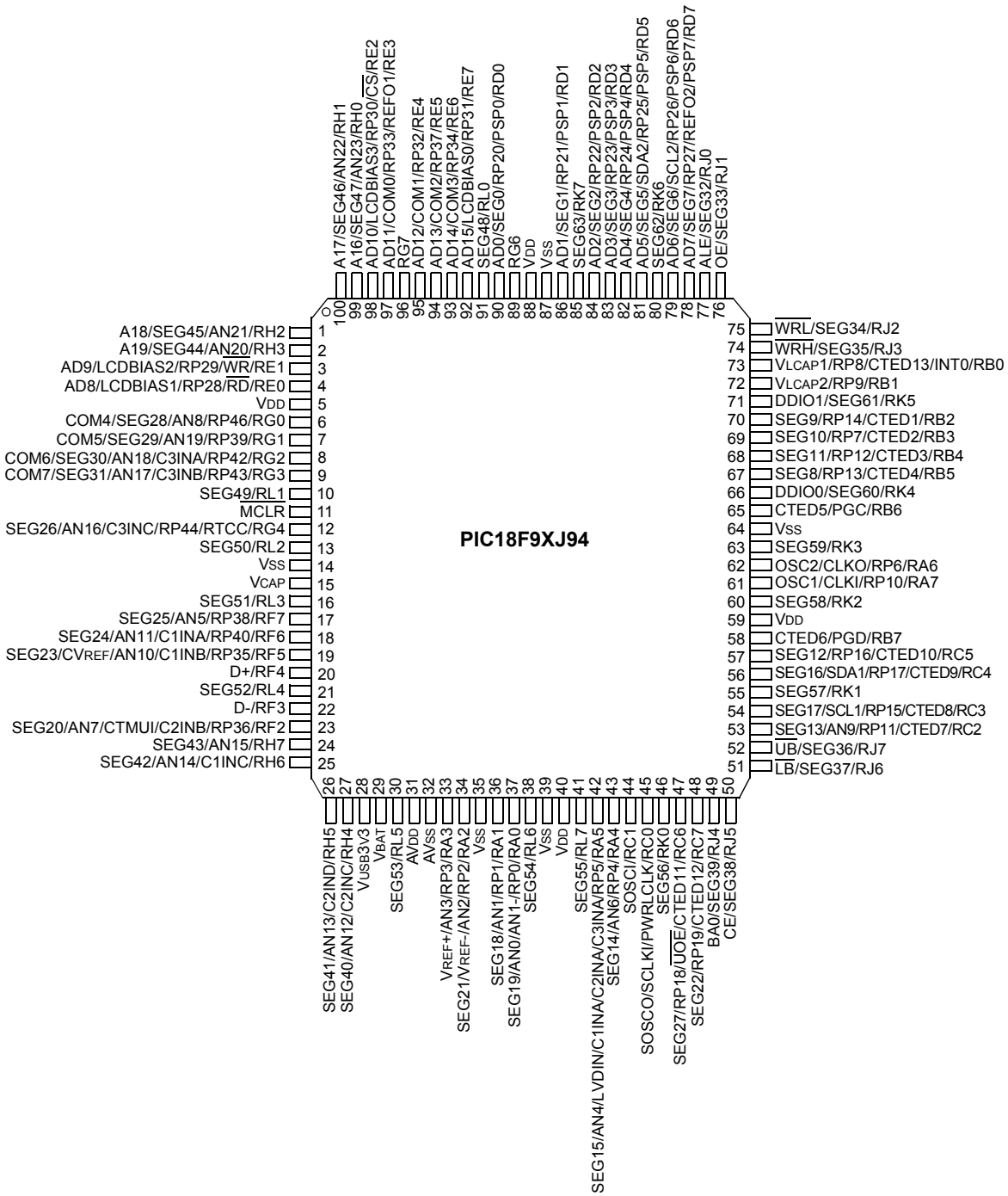
FIGURE 2: 80-PIN TQFP DIAGRAM FOR PIC18F8XJ94



Note 1: Pinouts are subject to change.

2: See Table 3 for the pin allocation table.

FIGURE 3: 100-PIN TQFP DIAGRAM FOR PIC18F9XJ94



Note 1: Pinouts are subject to change.

2: See Table 4 for the pin allocation table.

PIN ALLOCATION TABLES

TABLE 2: 64-PIN ALLOCATION TABLE (PIC18F6XJ94)

I/O	64-Pin TQFP/QFN	ADC	Comparator	HLVD	CTMU	USB	LCD	MSSP	PSP	Interrupt	REFO	PPS-Lite(1)	Pull-up	Basic
RA0	24	AN0/ AN1-	—	—	—	—	SEG19	—	—	—	—	RP0	—	—
RA1	23	AN1	—	—	—	—	SEG18	—	—	—	—	RP1	—	—
RA2	22	AN2/ VREF-	—	—	—	—	SEG21	—	—	—	—	RP2	—	—
RA3	21	AN3/ VREF+	—	—	—	—	—	—	—	—	—	RP3	—	—
RA4	28	AN6	—	—	—	—	SEG14	—	—	—	—	RP4	—	—
RA5	27	AN4	C1INA/ C2INA/ C3INA	LVDIN	—	—	SEG15	—	—	—	—	RP5	—	—
RA6	40	—	—	—	—	—	—	—	—	—	—	RP6	—	OSC2/ CLKO
RA7	39	—	—	—	—	—	—	—	—	—	—	RP10	—	OSC1/ CLKI
RB0	48	—	—	—	CTED13	—	VLCAP1	—	—	INT0	—	RP8	—	—
RB1	47	—	—	—	—	—	VLCAP2	—	—	—	—	RP9	—	—
RB2	46	—	—	—	CTED1	—	SEG9	—	—	—	—	RP14	—	—
RB3	45	—	—	—	CTED2	—	SEG10	—	—	—	—	RP7	—	—
RB4	44	—	—	—	CTED3	—	SEG11	—	—	—	—	RP12	—	—
RB5	43	—	—	—	CTED4	—	SEG8	—	—	—	—	RP13	—	—
RB6	42	—	—	—	CTED5	—	—	—	—	—	—	—	—	PGC
RB7	37	—	—	—	CTED6	—	—	—	—	—	—	—	—	PGD
RC0	30	—	—	—	—	—	—	—	—	—	—	—	—	SOSCO/ SCKI/ PWRCLK
RC1	29	—	—	—	—	—	—	—	—	—	—	—	—	SOSCI
RC2	33	AN9	—	—	CTED7	—	SEG13	—	—	—	—	RP11	—	—
RC3	34	—	—	—	CTED8	—	SEG17	SCL1	—	—	—	RP15	—	—
RC4	35	—	—	—	CTED9	—	SEG16	SDA1	—	—	—	RP17	—	—
RC5	36	—	—	—	CTED10	—	SEG12	—	—	—	—	RP16	—	—
RC6	31	—	—	—	CTED11	$\overline{UOE}$	SEG27	—	—	—	—	RP18	—	—
RC7	32	—	—	—	CTED12	—	SEG22	—	—	—	—	RP19	—	—
RD0	58	—	—	—	—	—	SEG0	—	PSP0	—	—	RP20	Y	—
RD1	55	—	—	—	—	—	SEG1	—	PSP1	—	—	RP21	Y	—
RD2	54	—	—	—	—	—	SEG2	—	PSP2	—	—	RP22	Y	—
RD3	53	—	—	—	—	—	SEG3	—	PSP3	—	—	RP23	Y	—
RD4	52	—	—	—	—	—	SEG4	—	PSP4	—	—	RP24	Y	—
RD5	51	—	—	—	—	—	SEG5	SDA2	PSP5	—	—	RP25	Y	—
RD6	50	—	—	—	—	—	SEG6	SCL2	PSP6	—	—	RP26	Y	—
RD7	49	—	—	—	—	—	SEG7	—	PSP7	—	REFO2	RP27	Y	—
RE0	2	—	—	—	—	—	LCDBIAS1	—	$\overline{RD}$	—	—	RP28	Y	—
RE1	1	—	—	—	—	—	LCDBIAS2	—	$\overline{WR}$	—	—	RP29	Y	—
RE2	64	—	—	—	—	—	LCDBIAS3	—	$\overline{CS}$	—	—	RP30	Y	—
RE3	63	—	—	—	—	—	COM0	—	—	—	REFO1	RP33	Y	—
RE4	62	—	—	—	—	—	COM1	—	—	—	—	RP32	Y	—
RE5	61	—	—	—	—	—	COM2	—	—	—	—	RP37	Y	—

TABLE 2: 64-PIN ALLOCATION TABLE (PIC18F6XJ94) (CONTINUED)

I/O	64-Pin TQFP/QFN	ADC	Comparator	HLVD	CTMU	USB	LCD	MSSP	PSP	Interrupt	REFO	PPS-Lite(1)	Pull-up	Basic
RE6	60	—	—	—	—	—	COM3	—	—	—	—	RP34	Y	—
RE7	59	—	—	—	—	—	LCDBIAS0	—	—	—	—	RP31	Y	—
RF2	16	AN7	C2INB	—	CTMUI	—	SEG20	—	—	—	—	RP36	Y	—
RF3	15	—	—	—	—	D-	—	—	—	—	—	—	Y	—
RF4	14	—	—	—	—	D+	—	—	—	—	—	—	Y	—
RF5	13	AN10	C1INB/ CVREF	—	—	—	SEG23	—	—	—	—	RP35	Y	—
RF6	12	AN11	C1INA	—	—	—	SEG24	—	—	—	—	RP40	Y	—
RF7	11	AN5	—	—	—	—	SEG25	—	—	—	—	RP38	Y	—
RG0	3	AN8	—	—	—	—	COM4/ SEG28	—	—	—	—	RP46	Y	—
RG1	4	AN19	—	—	—	—	COM5/ SEG29	—	—	—	—	RP39	Y	—
RG2	5	AN18	C3INA	—	—	—	COM6/ SEG30	—	—	—	—	RP42	Y	—
RG3	6	AN17	C3INB	—	—	—	COM7/ SEG31	—	—	—	—	RP43	Y	—
RG4	8	AN16	C3INC	—	—	—	SEG26	—	—	—	—	RP44	Y	—
RG5/ MCLR	7	—	—	—	—	—	—	—	—	—	—	—	Y	$\overline{\text{MCLR}}$
AVDD	19	AVDD	—	—	—	—	—	—	—	—	—	—	—	—
AVSS	20	AVSS	—	—	—	—	—	—	—	—	—	—	—	—
VBAT	18	—	—	—	—	—	—	—	—	—	—	—	—	VBAT
VCAP/ VDDCORE	10	—	—	—	—	—	—	—	—	—	—	—	—	VCAP/ VDDCORE
VDD	26, 38, 57	—	—	—	—	—	—	—	—	—	—	—	—	VDD
VSS	9, 25, 41, 56	—	—	—	—	—	—	—	—	—	—	—	—	VSS
VUSB3V3	17	—	—	—	—	—	—	—	—	—	—	—	—	VUSB3V3

Note 1: The peripheral inputs and outputs that support PPS have no default pins.

TABLE 3: 80-PIN ALLOCATION TABLE (PIC18F8XJ94)

I/O	80-Pin TQFP	ADC	Comparator	HLVD	CTMU	USB	LCD	MSSP	PSP	Interrupt	REFO	EMB	PPS-Lite(1)	Pull-up	Basic
RA0	30	AN0/ AN1-	—	—	—	—	SEG19	—	—	—	—	—	RP0	—	—
RA1	29	AN1	—	—	—	—	SEG18	—	—	—	—	—	RP1	—	—
RA2	28	AN2/ VREF-	—	—	—	—	SEG21	—	—	—	—	—	RP2	—	—
RA3	27	AN3/ VREF+	—	—	—	—	—	—	—	—	—	—	RP3	—	—
RA4	34	AN6	—	—	—	—	SEG14	—	—	—	—	—	RP4	—	—
RA5	33	AN4	C1INA/ C2INA/ C3INA	LVDIN	—	—	SEG15	—	—	—	—	—	RP5	—	—
RA6	50	—	—	—	—	—	—	—	—	—	—	—	RP6	—	OSC2/ CLKO
RA7	49	—	—	—	—	—	—	—	—	—	—	—	RP10	—	OSC1/ CLKI
RB0	58	—	—	—	CTED13	—	VLCAP1	—	—	INT0	—	—	RP8	—	—
RB1	57	—	—	—	—	—	VLCAP2	—	—	—	—	—	RP9	—	—
RB2	56	—	—	—	CTED1	—	SEG9	—	—	—	—	—	RP14	—	—
RB3	55	—	—	—	CTED2	—	SEG10	—	—	—	—	—	RP7	—	—
RB4	54	—	—	—	CTED3	—	SEG11	—	—	—	—	—	RP12	—	—
RB5	53	—	—	—	CTED4	—	SEG8	—	—	—	—	—	RP13	—	—
RB6	52	—	—	—	CTED5	—	—	—	—	—	—	—	—	—	PGC
RB7	47	—	—	—	CTED6	—	—	—	—	—	—	—	—	—	PGD
RC0	36	—	—	—	—	—	—	—	—	—	—	—	—	—	SOSCO/ SCKI/ PWRCLK
RC1	35	—	—	—	—	—	—	—	—	—	—	—	—	—	SOSCI
RC2	43	AN9	—	—	CTED7	—	SEG13	—	—	—	—	—	RP11	—	—
RC3	44	—	—	—	CTED8	—	SEG17	SCL1	—	—	—	—	RP15	—	—
RC4	45	—	—	—	CTED9	—	SEG16	SDA1	—	—	—	—	RP17	—	—
RC5	46	—	—	—	CTED10	—	SEG12	—	—	—	—	—	RP16		
RC6	37	—	—	—	CTED11	$\overline{UOE}$	SEG27	—	—	—	—	—	RP18	—	—
RC7	38	—	—	—	CTED12	—	SEG22	—	—	—	—	—	RP19	—	—
RD0	72	—	—	—	—	—	SEG0	—	PSP0	—	—	AD0	RP20	Y	—
RD1	69	—	—	—	—	—	SEG1	—	PSP1	—	—	AD1	RP21	Y	—
RD2	68	—	—	—	—	—	SEG2	—	PSP2	—	—	AD2	RP22	Y	—
RD3	67						SEG3		PSP3			AD3	RP23	Y	—
RD4	66	—	—	—	—	—	SEG4	—	PSP4	—	—	AD4	RP24	Y	—
RD5	65	—	—	—	—	—	SEG5	SDA2	PSP5	—	—	AD5	RP25	Y	—
RD6	64	—	—	—	—	—	SEG6	SCL2	PSP6	—	—	AD6	RP26	Y	—
RD7	63	—	—	—	—	—	SEG7	—	PSP7	—	REFO2	AD7	RP27	Y	—
RE0	4	—	—	—	—	—	LCDBIAS1	—	$\overline{RD}$	—	—	AD8	RP28	Y	—
RE1	3	—	—	—	—	—	LCDBIAS2	—	$\overline{WR}$	—	—	AD9	RP29	Y	—
RE2	78	—	—	—	—	—	LCDBIAS3	—	$\overline{CS}$	—	—	AD10	RP30	Y	—
RE3	77	—	—	—	—	—	COM0	—	—	—	REFO1	AD11	RP33	Y	—
RE4	76	—	—	—	—	—	COM1	—	—	—	—	AD12	RP32	Y	—
RE5	75	—	—	—	—	—	COM2	—	—	—	—	AD13	RP37	Y	—
RE6	74	—	—	—	—	—	COM3	—	—	—	—	AD14	RP34	Y	—
RE7	73	—	—	—	—	—	LCDBIAS0	—	—	—	—	AD15	RP31	Y	—
RF2	18	AN7	C2INB		CTMUI		SEG20	—	—	—	—	—	RP36	Y	—

TABLE 3: 80-PIN ALLOCATION TABLE (PIC18F8XJ94) (CONTINUED)

I/O	80-Pin TQFP	ADC	Comparator	HLVD	CTMU	USB	LCD	MSSP	PSP	Interrupt	REFO	EMB	PPS-Lite(1)	Pull-up	Basic
RF3	17	—	—	—	—	D-	—	—	—	—	—	—	—	Y	—
RF4	16	—	—	—	—	D+	—	—	—	—	—	—	—	Y	—
RF5	15	AN10	C1INB/ CVREF	—	—	—	SEG23	—	—	—	—	—	RP35	Y	—
RF6	14	AN11	C1INA	—	—	—	SEG24	—	—	—	—	—	RP40	Y	—
RF7	13	AN5	—	—	—	—	SEG25	—	—	—	—	—	RP38	Y	—
RG0	5	AN8	—	—	—	—	COM4/ SEG28	—	—	—	—	—	RP46	Y	—
RG1	6	AN19	—	—	—	—	COM5/ SEG29	—	—	—	—	—	RP39	Y	—
RG2	7	AN18	C3INA	—	—	—	COM6/ SEG30	—	—	—	—	—	RP42	Y	—
RG3	8	AN17	C3INB	—	—	—	COM7/ SEG31	—	—	—	—	—	RP43	Y	—
RG4	10	AN16	C3INC	—	—	—	SEG26	—	—	—	—	—	RP44	Y	—
RG5/ MCLR	9	—	—	—	—	—	—	—	—	—	—	—	—	Y	$\overline{MCLR}$
RH0	79	AN23	—	—	—	—	SEG47	—	—	—	—	A16	—	Y	—
RH1	80	AN22	—	—	—	—	SEG46	—	—	—	—	A17	—	Y	—
RH2	1	AN21	—	—	—	—	SEG45	—	—	—	—	A18	—	Y	—
RH3	2	AN20	—	—	—	—	SEG44	—	—	—	—	A19	—	Y	—
RH4	22	AN12	C2INC	—	—	—	SEG40	—	—	—	—	—	—	Y	—
RH5	21	AN13	C2IND	—	—	—	SEG41	—	—	—	—	—	—	Y	—
RH6	20	AN14	C1INC	—	—	—	SEG42	—	—	—	—	—	—	Y	—
RH7	19	AN15	—	—	—	—	SEG43	—	—	—	—	—	—	Y	—
RJ0	62	—	—	—	—	—	SEG32	—	—	—	—	ALE	—	Y	—
RJ1	61	—	—	—	—	—	SEG33	—	—	—	—	$\overline{OE}$	—	Y	—
RJ2	60	—	—	—	—	—	SEG34	—	—	—	—	$\overline{WRL}$	—	Y	—
RJ3	59	—	—	—	—	—	SEG35	—	—	—	—	$\overline{WRH}$	—	Y	—
RJ4	39	—	—	—	—	—	SEG39	—	—	—	—	BA0	—	Y	—
RJ5	40	—	—	—	—	—	SEG38	—	—	—	—	$\overline{CE}$	—	Y	—
RJ6	41	—	—	—	—	—	SEG37	—	—	—	—	$\overline{LB}$	—	Y	—
RJ7	42	—	—	—	—	—	SEG36	—	—	—	—	$\overline{UB}$	—	Y	—
AVDD	25	AVDD	—	—	—	—	—	—	—	—	—	—	—	—	—
AVSS	26	AVSS	—	—	—	—	—	—	—	—	—	—	—	—	—
VBAT	24	—	—	—	—	—	—	—	—	—	—	—	—	—	VBAT
VCAP/ VDDCORE	12	—	—	—	—	—	—	—	—	—	—	—	—	—	VCAP/ VDDCORE
VDD	32, 48, 71	—	—	—	—	—	—	—	—	—	—	—	—	—	VDD
VSS	11, 31, 51, 70	—	—	—	—	—	—	—	—	—	—	—	—	—	VSS
VUSB3V3	23	—	—	—	—	—	—	—	—	—	—	—	—	—	VUSB3V3

Note 1: The peripheral inputs and outputs that support PPS have no default pins.

TABLE 4: 100-PIN ALLOCATION TABLE (PIC18F9XJ94)

I/O	100-Pin TQFP	ADC	Comparator	HLVD	CTMU	USB	LCD	MSSP	PSP	Interrupt	REFO	EMB	PPS-Lite(1)	Pull-up	Basic
RA0	37	AN0/ AN1-	—	—	—	—	SEG19	—	—	—	—	—	RP0	—	—
RA1	36	AN1	—	—	—	—	SEG18	—	—	—	—	—	RP1	—	—
RA2	34	AN2/ VREF-	—	—	—	—	SEG21	—	—	—	—	—	RP2	—	—
RA3	33	AN3/ VREF+	—	—	—	—	—	—	—	—	—	—	RP3	—	—
RA4	43	AN6	—	—	—	—	SEG14	—	—	—	—	—	RP4	—	—
RA5	42	AN4	C1INA/ C2INA/ C3INA	LVDIN	—	—	SEG15	—	—	—	—	—	RP5	—	—
RA6	62	—	—	—	—	—	—	—	—	—	—	—	RP6	—	OSC2/ CLKO
RA7	61	—	—	—	—	—	—	—	—	—	—	—	RP10	—	OSC1/ CLKI
RB0	73	—	—	—	CTED13	—	VLCAP1	—	—	INT0	—	—	RP8	—	—
RB1	72	—	—	—	—	—	VLCAP2	—	—	—	—	—	RP9	—	—
RB2	70	—	—	—	CTED1	—	SEG9	—	—	—	—	—	RP14	—	—
RB3	69	—	—	—	CTED2	—	SEG10	—	—	—	—	—	RP7	—	—
RB4	68	—	—	—	CTED3	—	SEG11	—	—	—	—	—	RP12	—	—
RB5	67	—	—	—	CTED4	—	SEG8	—	—	—	—	—	RP13	—	—
RB6	65	—	—	—	CTED5	—	—	—	—	—	—	—	—	—	PGC
RB7	58	—	—	—	CTED6	—	—	—	—	—	—	—	—	—	PGD
RC0	45	—	—	—	—	—	—	—	—	—	—	—	—	—	SOSCO/ SCKI/ PWRCLK
RC1	44	—	—	—		—	—	—	—	—	—	—	—	—	SOSCI
RC2	53	AN9	—	—	CTED7	—	SEG13	—	—	—	—	—	RP11	—	—
RC3	54	—	—	—	CTED8	—	SEG17	SCL1	—	—	—	—	RP15	—	—
RC4	56	—	—	—	CTED9	—	SEG16	SDA1	—	—	—	—	RP17	—	—
RC5	57	—	—	—	CTED10	—	SEG12	—	—	—	—	—	RP16	—	—
RC6	47	—	—	—	CTED11	$\overline{\text{UOE}}$	SEG27	—	—	—	—	—	RP18	—	—
RC7	48	—	—	—	CTED12	—	SEG22	—	—	—	—	—	RP19	—	—
RD0	90	—	—	—	—	—	SEG0	—	PSP0	—	—	AD0	RP20	Y	—
RD1	86	—	—	—	—	—	SEG1	—	PSP1	—	—	AD1	RP21	Y	—
RD2	84	—	—	—	—	—	SEG2	—	PSP2	—	—	AD2	RP22	Y	—
RD3	83	—	—	—	—	—	SEG3	—	PSP3	—	—	AD3	RP23	Y	—
RD4	82	—	—	—	—	—	SEG4	—	PSP4	—	—	AD4	RP24	Y	—
RD5	81	—	—	—	—	—	SEG5	SDA2	PSP5	—	—	AD5	RP25	Y	—
RD6	79	—	—	—	—	—	SEG6	SCL2	PSP6	—	—	AD6	RP26	Y	—
RD7	78	—	—	—	—	—	SEG7	—	PSP7	—	REFO2	AD7	RP27	Y	—
RE0	4	—	—	—	—	—	LCDBIAS1	—	RD-bar	—	—	AD8	RP28	Y	—
RE1	3	—	—	—	—	—	LCDBIAS2	—	WR-bar	—	—	AD9	RP29	Y	—
RE2	98	—	—	—	—	—	LCDBIAS3	—	CS-bar	—	—	AD10	RP30	Y	—
RE3	97	—	—	—	—	—	COM0	—	—	—	REFO1	AD11	RP33	Y	—
RE4	95	—	—	—	—	—	COM1	—	—	—	—	AD12	RP32	Y	—
RE5	94	—	—	—	—	—	COM2	—	—	—	—	AD13	RP37	Y	—
RE6	93	—	—	—	—	—	COM3	—	—	—	—	AD14	RP34	Y	—
RE7	92	—	—	—	—	—	LCDBIAS0	—	—	—	—	AD15	RP31	Y	—

TABLE 4: 100-PIN ALLOCATION TABLE (PIC18F9XJ94) (CONTINUED)

I/O	100-Pin TQFP	ADC	Comparator	HLVD	CTMU	USB	LCD	MSSP	PSP	Interrupt	REFO	EMB	PPS-Lite(1)	Pull-up	Basic
RF2	23	AN7	C2INB	—	CTMUI	—	SEG20	—	—	—	—	—	RP36	Y	—
RF3	22	—	—	—	—	D-	—	—	—	—	—	—	—	Y	—
RF4	20	—	—	—	—	D+	—	—	—	—	—	—	—	Y	—
RF5	19	AN10	C1INB/ CVREF	—	—	—	SEG23	—	—	—	—	—	RP35	Y	—
RF6	18	AN11	C1INA	—	—	—	SEG24	—	—	—	—	—	RP40	Y	—
RF7	17	AN5	—	—	—	—	SEG25	—	—	—	—	—	RP38	Y	—
RG0	6	AN8	—	—	—	—	COM4/ SEG28	—	—	—	—	—	RP46	Y	—
RG1	7	AN19	—	—	—	—	COM5/ SEG29	—	—	—	—	—	RP39	Y	—
RG2	8	AN18	C3INA	—	—	—	COM6/ SEG30	—	—	—	—	—	RP42	Y	—
RG3	9	AN17	C3INB	—	—	—	COM7/ SEG31	—	—	—	—	—	RP43	Y	—
RG4	12	AN16	C3INC	—	—	—	SEG26	—	—	—	—	—	RP44	Y	—
RG5/ MCLR	11	—	—	—	—	—	—	—	—	—	—	—	—	Y	$\overline{\text{MCLR}}$
RG6	89	—	—	—	—	—	—	—	—	—	—	—	—	Y	—
RG7	96	—	—	—	—	—	—	—	—	—	—	—	—	Y	—
RH0	99	AN23	—	—	—	—	SEG47	—	—	—	—	A16	—	Y	—
RH1	100	AN22	—	—	—	—	SEG46	—	—	—	—	A17	—	Y	—
RH2	1	AN21	—	—	—	—	SEG45	—	—	—	—	A18	—	Y	—
RH3	2	AN20	—	—	—	—	SEG44	—	—	—	—	A19	—	Y	—
RH4	27	AN12	C2INC	—	—	—	SEG40	—	—	—	—	—	—	Y	—
RH5	26	AN13	C2IND	—	—	—	SEG41	—	—	—	—	—	—	Y	—
RH6	25	AN14	C1INC	—	—	—	SEG42	—	—	—	—	—	—	Y	—
RH7	24	AN15	—	—	—	—	SEG43	—	—	—	—	—	—	Y	—
RJ0	77	—	—	—	—	—	SEG32	—	—	—	—	ALE	—	Y	—
RJ1	76	—	—	—	—	—	SEG33	—	—	—	—	$\overline{\text{OE}}$	—	Y	—
RJ2	75	—	—	—	—	—	SEG34	—	—	—	—	$\overline{\text{WRL}}$	—	Y	—
RJ3	74	—	—	—	—	—	SEG35	—	—	—	—	$\overline{\text{WRH}}$	—	Y	—
RJ4	49	—	—	—	—	—	SEG39	—	—	—	—	BA0	—	Y	—
RJ5	50	—	—	—	—	—	SEG38	—	—	—	—	$\overline{\text{CE}}$	—	Y	—
RJ6	51	—	—	—	—	—	SEG37	—	—	—	—	$\overline{\text{LB}}$	—	Y	—
RJ7	52	—	—	—	—	—	SEG36	—	—	—	—	$\overline{\text{UB}}$	—	Y	—
RK0	46	—	—	—	—	—	SEG56	—	—	—	—	—	—	Y	—
RK1	55	—	—	—	—	—	SEG57	—	—	—	—	—	—	Y	—
RK2	60	—	—	—	—	—	SEG58	—	—	—	—	—	—	Y	—
RK3	63	—	—	—	—	—	SEG59	—	—	—	—	—	—	Y	—
RK4	66	—	—	—	—	—	SEG60	—	—	—	—	—	—	Y	—
RK5	71	—	—	—	—	—	SEG61	—	—	—	—	—	—	Y	—
RK6	80	—	—	—	—	—	SEG62	—	—	—	—	—	—	Y	—
RK7	85	—	—	—	—	—	SEG63	—	—	—	—	—	—	Y	—
RL0	91	—	—	—	—	—	SEG48	—	—	—	—	—	—	Y	—
RL1	10	—	—	—	—	—	SEG49	—	—	—	—	—	—	Y	—
RL2	13	—	—	—	—	—	SEG50	—	—	—	—	—	—	Y	—
RL3	16	—	—	—	—	—	SEG51	—	—	—	—	—	—	Y	—
RL4	21	—	—	—	—	—	SEG52	—	—	—	—	—	—	Y	—
RL5	30	—	—	—	—	—	SEG53	—	—	—	—	—	—	Y	—

TABLE 4: 100-PIN ALLOCATION TABLE (PIC18F9XJ94) (CONTINUED)

I/O	100-Pin TQFP	ADC	Comparator	HLVD	CTMU	USB	LCD	MSSP	PSP	Interrupt	REFO	EMB	PPS-Lite(1)	Pull-up	Basic
RL6	38	—	—	—	—	—	SEG54	—	—	—	—	—	—	Y	—
RL7	41	—	—	—	—	—	SEG55	—	—	—	—	—	—	Y	—
AVDD	31	AVDD	—	—	—	—	—	—	—	—	—	—	—	—	—
AVSS	32	AVSS	—	—	—	—	—	—	—	—	—	—	—	—	—
VBAT	29	—	—	—	—	—	—	—	—	—	—	—	—	—	VBAT
VCAP/ VDDCORE	15	—	—	—	—	—	—	—	—	—	—	—	—	—	VCAP/ VDDCORE
VDD	5, 40, 59, 88	—	—	—	—	—	—	—	—	—	—	—	—	—	VDD
VSS	14, 35, 39, 64, 87	—	—	—	—	—	—	—	—	—	—	—	—	—	VSS
VUSB3V3	28	—	—	—	—	—	—	—	—	—	—	—	—	—	VUSB3V3

Note 1: The peripheral inputs and outputs that support PPS have no default pins.

Table of Contents

TO OUR VALUED CUSTOMERS

It is our intention to provide our valued customers with the best documentation possible to ensure successful use of your Microchip products. To this end, we will continue to improve our publications to better suit your needs. Our publications will be refined and enhanced as new volumes and updates are introduced.

If you have any questions or comments regarding this publication, please contact the Marketing Communications Department via E-mail at **docerrors@microchip.com** or fax the **Reader Response Form** in the back of this data sheet to (480) 792-4150. We welcome your feedback.

Most Current Data Sheet

To obtain the most up-to-date version of this data sheet, please register at our Worldwide Website at:

http://www.microchip.com

You can determine the version of a data sheet by examining its literature number found on the bottom outside corner of any page. The last character of the literature number is the version number, (e.g., DS30000A is version A of document DS30000).

Errata

An errata sheet, describing minor operational differences from the data sheet and recommended workarounds, may exist for current devices. As device/documentation issues become known to us, we will publish an errata sheet. The errata will specify the revision of silicon and revision of document to which it applies.

To determine if an errata sheet exists for a particular device, please check with one of the following:

- Microchip's Worldwide Website; **http://www.microchip.com**
- Your local Microchip sales office (see last page)

When contacting a sales office, please specify which device, revision of silicon and data sheet (include literature number) you are using.

Customer Notification System

Register on our website at **www.microchip.com** to receive the most current information on all of our products.

1.0 DEVICE OVERVIEW

This document contains device-specific information for the following devices:

- PIC18F97J94
- PIC18F87J94
- PIC18F67J94
- PIC18F96J94
- PIC18F86J94
- PIC18F66J94
- PIC18F95J94
- PIC18F85J94
- PIC18F65J94

This family introduces a new line of low-voltage LCD microcontrollers with Universal Serial Bus (USB). It combines all the main traditional advantage of all PIC18 microcontrollers, namely, high computational performance and a rich feature set at an extremely competitive price point. These features make the PIC18F9XJ94 family a logical choice for many high-performance applications, where cost is a primary consideration.

1.1 Core Features

1.1.1 TECHNOLOGY

All of the devices in the PIC18F9XJ94 family incorporate a range of features that can significantly reduce power consumption during operation. Key items include:

- **Alternate Run Modes:** By clocking the controller from the Timer1 source or the Internal RC oscillator, power consumption during code execution can be reduced.
- **Multiple Idle Modes:** The controller can also run with its CPU core disabled but the peripherals still active. In these states, power consumption can be reduced even further.
- **On-the-Fly Mode Switching:** The power-managed modes are invoked by user code during operation, allowing the user to incorporate power-saving ideas into their application's software design.
- **XLP:** An extra low-power Sleep, BOR, RTCC and Watchdog Timer.

1.1.2 OSCILLATOR OPTIONS AND FEATURES

All of the devices in the PIC18F9XJ94 family offer different oscillator options, allowing users a range of choices in developing application hardware. These include:

- Two Crystal modes (HS, MS)
- One External Clock mode (EC)
- A Phase Lock Loop (PLL) frequency multiplier, which allows clock speeds of up to 64 MHz.
- A fast Internal Oscillator (FRC) block that provides an 8 MHz clock (±0.15% accuracy) with Active Clock Tuning (ACT) from USB or SOSC source.
  - Offers multiple divider options from 8 MHz to 500 kHz
  - Frees the two oscillator pins for use as additional general purpose I/O
- A separate Low-Power Internal RC Oscillator (LPRC) (31 kHz nominal) for low-power, timing-insensitive applications.

The internal oscillator block provides a stable reference source that gives the family additional features for robust operation:

- **Fail-Safe Clock Monitor (FSCM):** This option constantly monitors the main clock source against a reference signal provided by the internal oscillator. If a clock failure occurs, the controller is switched to the internal oscillator, allowing for continued low-speed operation or a safe application shutdown.
- **Two-Speed Start-up (IESO):** This option allows the internal oscillator to serve as the clock source from Power-on Reset, or wake-up from Sleep mode, until the primary clock source is available.

1.1.3 MEMORY OPTIONS

The PIC18F9XJ94 family provides ample room for application code, from 32 Kbytes to 128 Kbytes of code space. The Flash cells for program memory are rated to last up to 20,000 erase/write cycles. Data retention without refresh is conservatively estimated to be greater than 10 years.

The Flash program memory is readable and writable. During normal operation, the PIC18F9XJ94 family also provides plenty of room for dynamic application data with up to 3,578 bytes of data RAM.

1.1.4 UNIVERSAL SERIAL BUS (USB)

Devices in the PIC18F9XJ94 family incorporate a fully-featured USB communications module with a built-in transceiver that is compliant with the USB Specification Revision 2.0. The module supports both low-speed and full-speed communication for all supported data transfer types.

1.1.5 EXTERNAL MEMORY BUS

Should 128 Kbytes of memory be inadequate for an application, the 80-pin and 100-pin members of the PIC18F9XJ94 family have an External Memory Bus (EMB), enabling the controller's internal Program Counter to address a memory space of up to 2 Mbytes. This is a level of data access that few 8-bit devices can claim and enables:

- Using combinations of on-chip and external memory of up to 2 Mbytes
- Using external Flash memory for reprogrammable application code or large data tables
- Using external RAM devices for storing large amounts of variable data

1.1.6 EXTENDED INSTRUCTION SET

The PIC18F9XJ94 family implements the optional extension to the PIC18 instruction set, adding eight new instructions and an Indexed Addressing mode. Enabled as a device configuration option, the extension has been specifically designed to optimize re-entrant application code originally developed in high-level languages, such as 'C'.

1.1.7 EASY MIGRATION

All devices share the same rich set of peripherals. This provides a smooth migration path within the device family as applications evolve and grow.

The consistent pinout scheme, used throughout the entire family, also aids in migrating to the next larger device. This is true when moving between the 64-pin members, between the 80-pin members, between the 100-pin members or even jumping from 64-pin to 80-pin to 100-pin devices.

The PIC18F9XJ94 family is also largely pin compatible with other PIC18 families, such as the PIC18F87J90, PIC18F87J11 and the PIC18F87J50. This allows a new dimension to the evolution of applications, allowing developers to select different price points within Microchip's PIC18 portfolio, while maintaining a similar feature set.

1.2 LCD Controller

The on-chip LCD driver includes many features that make the integration of displays in low-power applications easier. These include an integrated voltage regulator with charge pump and an integrated internal resistor ladder that allows contrast control in software and display operation above device V_{DD}.

1.3 Other Special Features

- **Communications:** The PIC18F9XJ94 family incorporates a range of serial communication peripherals, including USB, four Enhanced Addressable USARTs with IrDA, and two Master Synchronous Serial Port MSSP modules capable of both SPI and I^2C (Master and Slave) modes of operation.
- **CCP Modules:** PIC18F9XJ94 family devices incorporate up to seven Capture/Compare/PWM (CCP) modules. Up to six different time bases can be used to perform several different operations at once.
- **ECCP Modules:** The PIC18F9XJ94 family has three Enhanced CCP (ECCP) modules to maximize flexibility in control applications:
  - Up to eight different time bases for performing several different operations at once
  - Up to four PWM outputs for each module – for a total of 12 PWMs
  - Other beneficial features, such as polarity selection, programmable dead time, auto-shutdown and restart, and Half-Bridge and Full-Bridge Output modes
- **12-Bit A/D Converter:** The PIC18F9XJ94 family has a software selectable, 10/12-bit Analog-to-Digital (A/D) Converter. It incorporates programmable acquisition time, allowing for a channel to be selected and a conversion to be initiated without waiting for a sampling period, and thus, reducing code overhead.
- **Charge Time Measurement Unit (CTMU):** The CTMU is a flexible analog module that provides accurate differential time measurement between pulse sources, as well as asynchronous pulse generation.
- Together with other on-chip analog modules, the CTMU can precisely measure time, measure capacitance or relative changes in capacitance, or generate output pulses that are independent of the system clock.
- **LP Watchdog Timer (WDT):** This enhanced version incorporates a 22-bit prescaler, allowing an extended time-out range that is stable across operating voltage and temperature. See **Section 30.0 "Electrical Specifications"** for time-out periods.
- **Real-Time Clock and Calendar Module (RTCC):** The RTCC module is intended for applications requiring that accurate time be maintained for extended periods of time, with minimum to no intervention from the CPU.
- The module is a 100-year clock and calendar with automatic leap year detection. The range of the clock is from 00:00:00 (midnight) on January 1, 2000 to 23:59:59 on December 31, 2099.

1.4 Details on Individual Family Members

Devices in the PIC18F9XJ94 family are available in 64-pin, 80-pin and 100-pin packages. Block diagrams for the two groups are shown in Figure 1-1, Figure 1-2 and Figure 1-3.

The devices are differentiated from each other in these ways:

- Flash Program Memory:
  - PIC18FX5J94 – 32 Kbytes
  - PIC18FX6J94 – 64 Kbytes
  - PIC18FX7J94 – 128 Kbytes
- Data RAM:
  - All devices – 4 Kbytes
- I/O Ports:
  - PIC18F6XJ9X (64-pin devices) – seven bidirectional ports
  - PIC18F8XJ9X (80-pin devices) – nine bidirectional ports
  - PIC18F9XJ9X (100-pin devices) – eleven bidirectional ports
- A/D Channels:
  - PIC18F6XJXX (64-pin devices) – 16 channels
  - PIC18F8XJXX (80-pin devices) – 24 channels
  - PIC18F9XJXX (100-pin devices) – 24 channels

All other features for devices in this family are identical. These are summarized in Table 1-1, Table 1-2 and Table 1-3.

The pinouts for all devices are listed in Table 1-4.

TABLE 1-1: DEVICE FEATURES FOR THE 64-PIN DEVICES

Features	PIC18F65J94	PIC18F66J94	PIC18F67J94
Operating Frequency	DC – 64 MHz		
Program Memory (Bytes)	32K	64K	128K
Program Memory (Instructions)	16,384	32,768	65,536
Data Memory (Bytes)	4K	4K	4K
Interrupt Sources	42	48	
I/O Ports	Ports A, B, C, D, E, F, G		
Parallel Communications	Parallel Slave Port (PSP)		
Timers	8		
Comparators	3		
LCD	224 pixels		
CTMU	Yes		
RTCC	Yes		
Enhanced Capture/Compare/PWM Modules	3 ECCPs and 7 CCPs		
Serial Communications	Two MSSPs, Four Enhanced USARTs (EUSART) and USB		
10/12-Bit Analog-to-Digital Module	16 Input Channels		
Resets (and Delays)	POR, BOR, CM RESET Instruction, Stack Full, Stack Underflow, $\overline{\text{MCLR}}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	64-Pin QFN, 64-Pin TQFP		

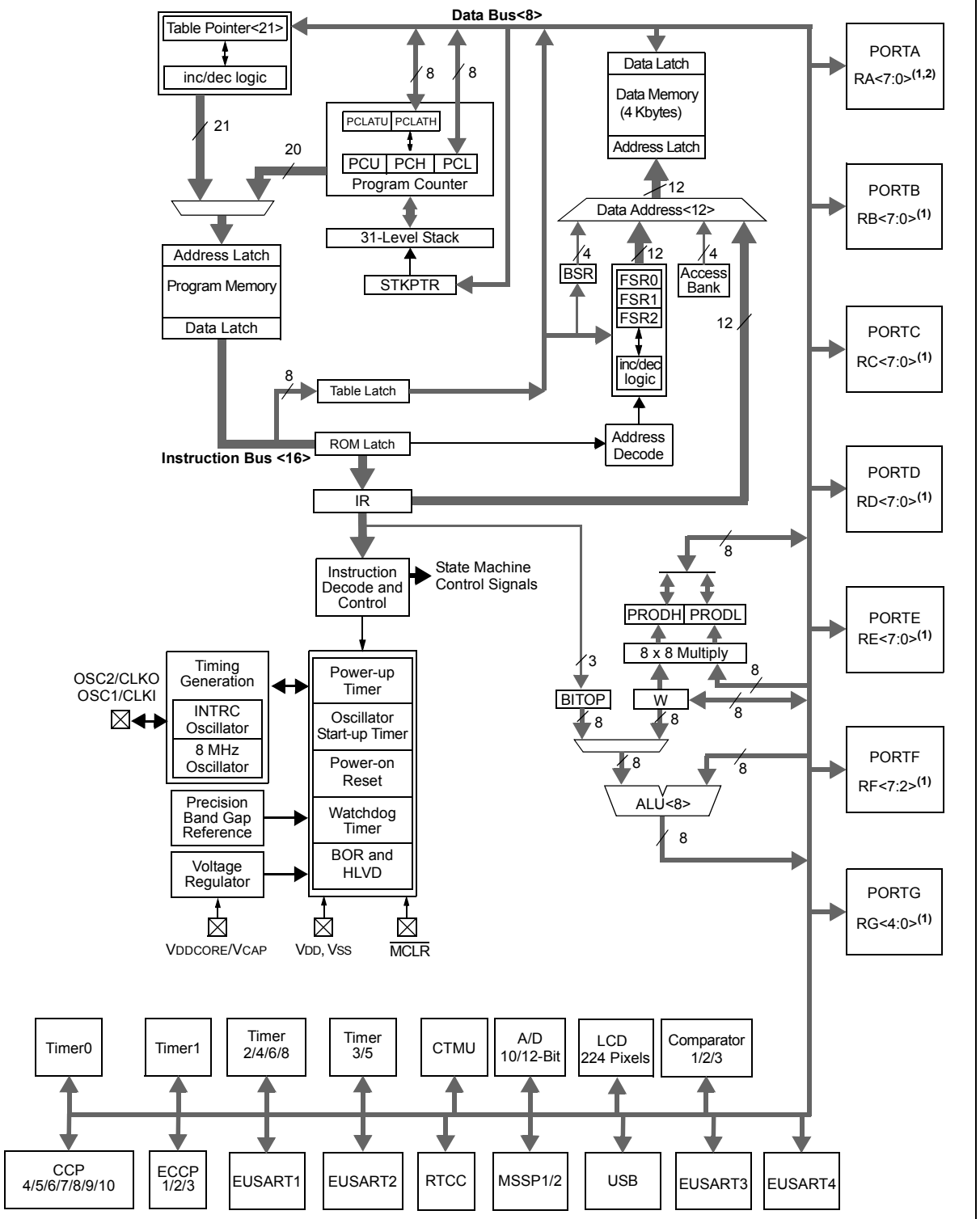
TABLE 1-2: DEVICE FEATURES FOR THE 80-PIN DEVICES

Features	PIC18F85J94	PIC18F86J94	PIC18F87J94
Operating Frequency	DC – 64 MHz		
Program Memory (Bytes)	32 K	64K	128K
	(Up to 2 Mbytes with Extended Memory)		
Program Memory (Instructions)	16,384	32,768	65,536
Data Memory (Bytes)	4K	4K	4K
Interrupt Sources	42	48	
I/O Ports	Ports A, B, C, D, E, F, G, H, J		
Parallel Communications	Parallel Slave Port (PSP)		
Timers	8		
Comparators	3		
LCD	352 pixels		
CTMU	Yes		
RTCC	Yes		
Enhanced Capture/Compare/PWM Modules	3 ECCPs and 7 CCPs		
Serial Communications	Two MSSPs, Four Enhanced USARTs (EUSART) and USB		
12-Bit Analog-to-Digital Module	24 Input Channels		
Resets (and Delays)	POR, BOR, CM RESET Instruction, Stack Full, Stack Underflow, $\overline{\text{MCLR}}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	80-Pin TQFP		

TABLE 1-3: DEVICE FEATURES FOR THE 100-PIN DEVICES

Features	PIC18F95J94	PIC18F96J94	PIC18F97J94
Operating Frequency	DC – 64 MHz		
Program Memory (Bytes)	32 K	64K	128K
	(Up to 2 Mbytes with Extended Memory)		
Program Memory (Instructions)	16,384	32,768	65,536
Data Memory (Bytes)	4K	4K	4K
Interrupt Sources	42	48	
I/O Ports	Ports A, B, C, D, E, F, G, H, J, K, L		
Parallel Communications	Parallel Slave Port (PSP)		
Timers	8		
Comparators	3		
LCD	480 pixels		
CTMU	Yes		
RTCC	Yes		
Enhanced Capture/Compare/PWM Modules	3 ECCPs and 7 CCPs		
Serial Communications	Two MSSPs, Four Enhanced USARTs (EUSART) and USB		
12-Bit Analog-to-Digital Module	24 Input Channels		
Resets (and Delays)	POR, BOR, CM RESET Instruction, Stack Full, Stack Underflow, $\overline{\text{MCLR}}$, WDT (PWRT, OST)		
Instruction Set	75 Instructions, 83 with Extended Instruction Set Enabled		
Packages	100-Pin TQFP		

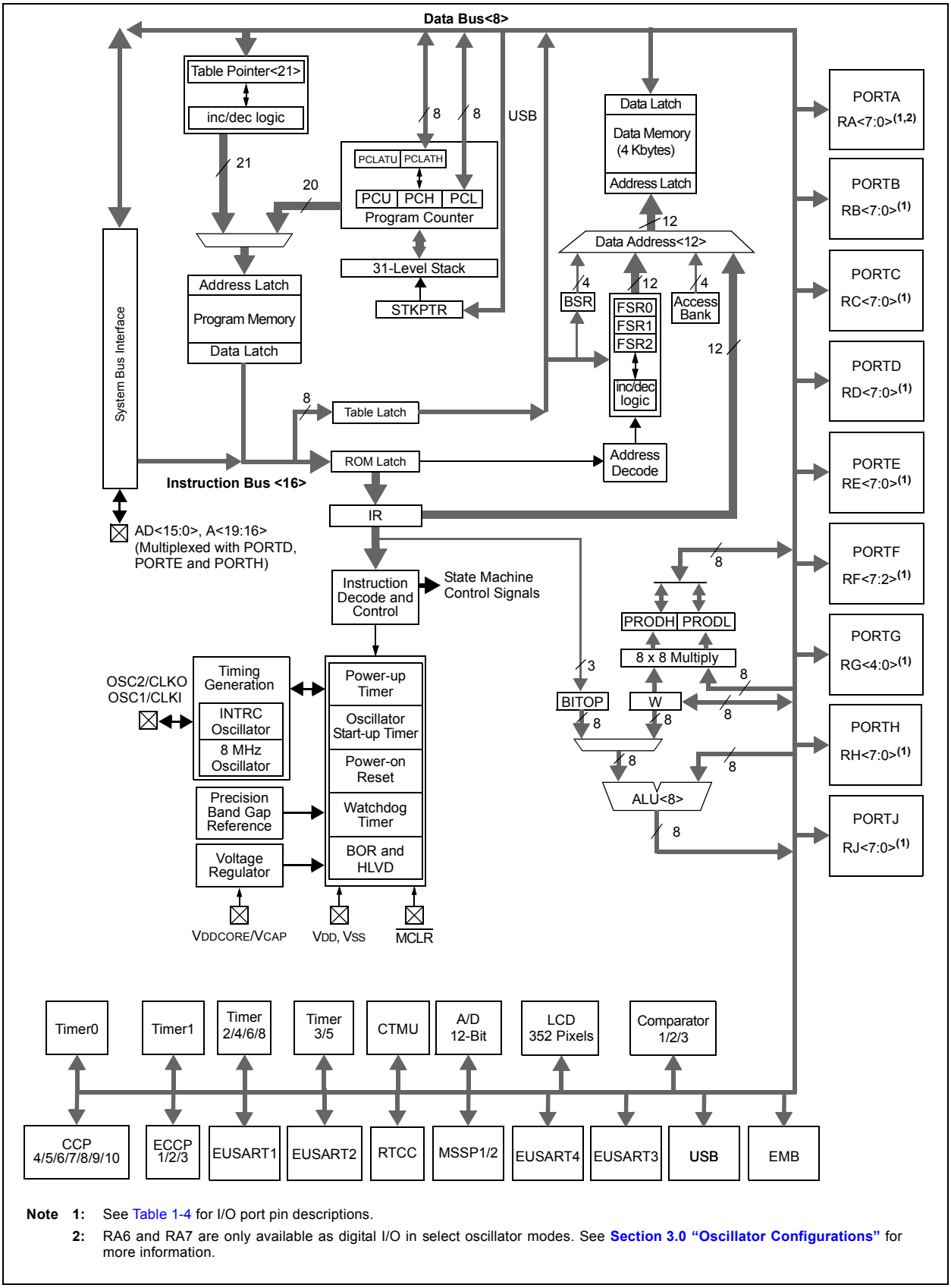
FIGURE 1-1: 64-PIN DEVICE BLOCK DIAGRAM



Note 1: See Table 1-4 for I/O port pin descriptions.

2: RA6 and RA7 are only available as digital I/O in select oscillator modes. For more information, see **Section 3.0 "Oscillator Configurations"**.

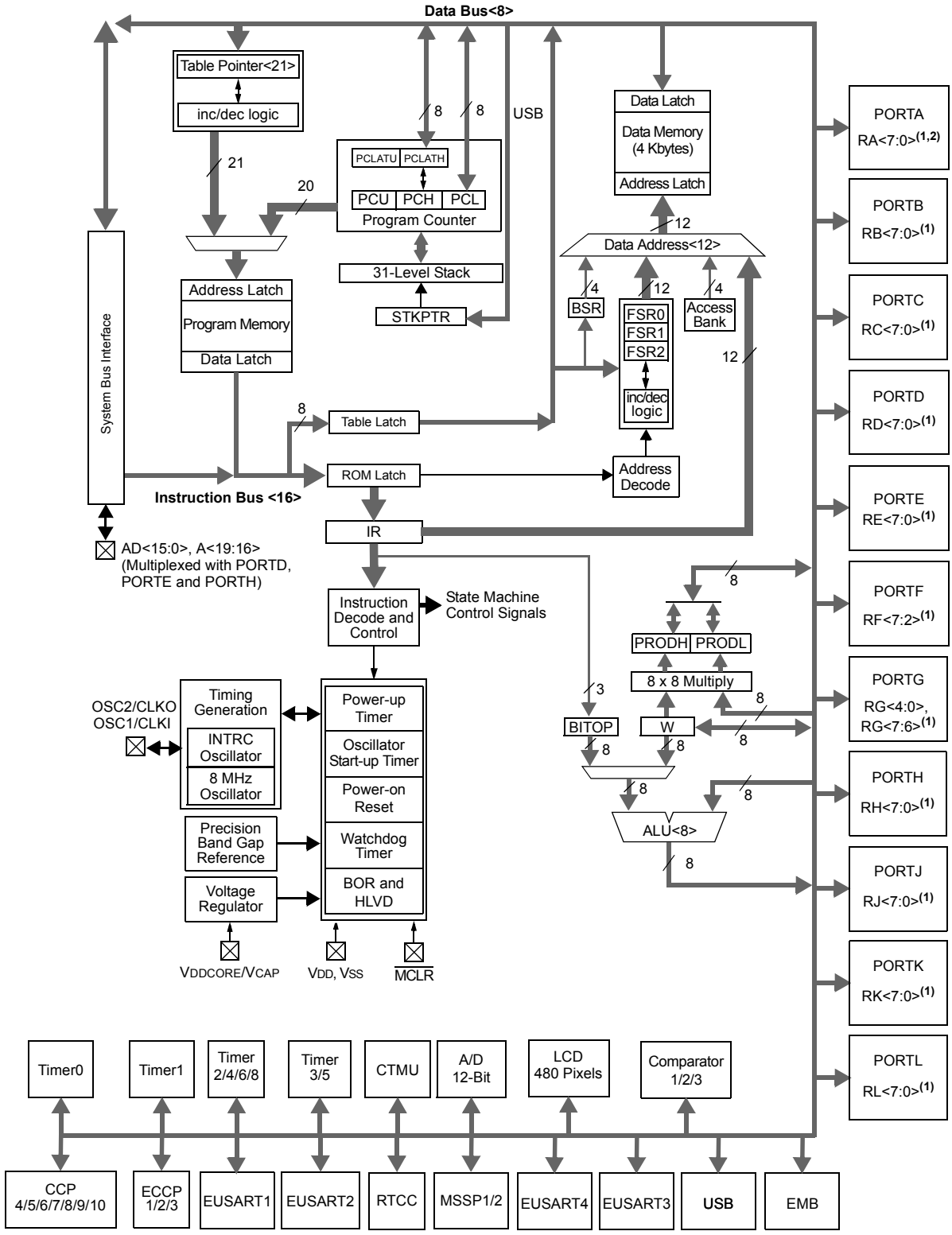
FIGURE 1-2: 80-PIN DEVICE BLOCK DIAGRAM



Note 1: See Table 1-4 for I/O port pin descriptions.

2: RA6 and RA7 are only available as digital I/O in select oscillator modes. See **Section 3.0 "Oscillator Configurations"** for more information.

FIGURE 1-3: 100-PIN DEVICE BLOCK DIAGRAM



Note 1: See Table 1-4 for I/O port pin descriptions.

2: RA6 and RA7 are only available as digital I/O in select oscillator modes. See **Section 3.0 "Oscillator Configurations"** for more information.

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
$\overline{MCLR}$	11	9	7	I	ST	Master Clear (input) or programming voltage (input). This pin is an active-low Reset to the device.
OSC1/CLKI/RP10/RA7	61	49	39			Oscillator crystal or external clock input.
OSC1				I	ST	Oscillator crystal input.
CLKI				I	CMOS	External clock source input. Always associated with pin function, OSC1. (See related OSC1/CLKI,OSC2/CLKO pins.)
RP10				I/O	ST/DIG	Remappable Peripheral Pin 10 input/output.
RA7				I/O	ST/DIG	General purpose I/O pin.
OSC2/CLKO/RP6/RA6	62	50	40			Oscillator crystal or clock output.
OSC2				O	—	Oscillator crystal output. Connects to crystal or resonator in Crystal Oscillator mode.
CLKO				O	DIG	In certain oscillator modes, OSC2 pin outputs CLKO, which has 1/4 the frequency of OSC1 and denotes the instruction cycle rate.
RP6				I/O	ST/DIG	Remappable Peripheral Pin 6 input/output.
RA6				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
SEG19/AN0/AN1-/RP0/RA0	37	30	24			
SEG19				O	Analog	SEG19 output for LCD.
AN0				I	Analog	Analog Input 0.
AN1-				I	Analog	A/D negative input channel.
RP0				I/O	ST/DIG	Remappable Peripheral Pin 0 input/output.
RA0				I/O	ST/DIG	General purpose I/O pin.
SEG18/AN1/RP1/RA1	36	29	23			
SEG18				O	Analog	SEG18 output for LCD.
AN1				I	Analog	Analog Input 1.
RP1				I/O	ST/DIG	Remappable Peripheral Pin 1 input/output.
RA1				I/O	ST/DIG	General purpose I/O pin.
SEG21/VREF-/AN2/RP2/RA2	34	28	22			
SEG21				O	Analog	SEG21 output for LCD.
VREF-				I	Analog	A/D reference voltage (low) input.
AN2				I	Analog	Analog Input 2.
RP2				I/O	ST/DIG	Remappable Peripheral Pin 2 input/output.
RA2				I/O	ST/DIG	General purpose I/O pin.
VREF+/AN3/RP3/RA3	33	27	21			
VREF+				I	Analog	A/D reference voltage (high) input.
AN3				I	Analog	Analog Input 3.
RP3				I/O	ST/DIG	Remappable Peripheral Pin 3 input/output.
RA3				I/O	ST/DIG	General purpose I/O pin.
SEG14/AN6/RP4/RA4	43	34	28			
SEG14				O	Analog	SEG14 output for LCD.
AN6				I	Analog	Analog Input 6.
RP4				I/O	ST/DIG	Remappable Peripheral Pin 4 input/output.
RA4				I/O	ST/DIG	General purpose I/O pin.
SEG15/AN4/LVDIN/C1INA/ C2INA/C3INA/RP5/RA5	42	33	27			
SEG15				O	Analog	SEG15 output for LCD.
AN4				I	Analog	Analog Input 4.
LVDIN				I	Analog	High/Low-Voltage Detect (HLVD) input.
C1INA				I	Analog	Comparator 1 Input A.
C2INA				I	Analog	Comparator 2 Input A.
C3INA				I	Analog	Comparator 3 Input A.
RP5				I/O	ST/DIG	Remappable Peripheral Pin 5 input/output.
RA5				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to VDD)

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
VLCAP1/RP8/CTED13/INT0/RB0	73	58	48			
VLCAP1				I	Analog	LCD Drive Charge Pump Capacitor Input 1.
RP8				I/O	ST/DIG	Remappable Peripheral Pin 8 input/output.
CTED13				I	ST	CTMU Edge 13 input.
INT0				I	ST	External Interrupt 0.
RB0				I/O	ST/DIG	General purpose I/O pin.
VLCAP2/RP9/RB1	72	57	47			
VLCAP2				I	Analog	LCD Drive Charge Pump Capacitor Input 2.
RP9				I/O	ST/DIG	Remappable Peripheral Pin 9 input/output.
RB1				I/O	ST/DIG	General purpose I/O pin.
SEG9/RP14/CTED1/RB2	70	56	46			
SEG9				O	Analog	SEG9 output for LCD.
RP14				I/O	ST/DIG	Remappable Peripheral Pin 14 input/output.
CTED1				I	ST	CTMU Edge 1 input.
RB2				I/O	ST/DIG	General purpose I/O pin.
SEG10/RP7/CTED2/RB3	69	55	45			
SEG10				O	Analog	SEG10 output for LCD.
RP7				I/O	ST/DIG	Remappable Peripheral Pin 7 input/output.
CTED2				I	ST	CTMU Edge 2 input.
RB3				I/O	ST/DIG	General purpose I/O pin.
SEG11/RP12/CTED3/RB4	68	54	44			
SEG11				O	Analog	SEG11 output for LCD.
RP12				I/O	ST/DIG	Remappable Peripheral Pin 12 input/output.
CTED3				I	ST	CTMU Edge 3 input.
RB4				I/O	ST/DIG	General purpose I/O pin.
SEG8/RP13/CTED4/RB5	67	53	43			
SEG8				O	Analog	SEG8 output for LCD.
RP13				I/O	ST/DIG	Remappable Peripheral Pin 13 input/output.
CTED4				I	ST	CTMU Edge 4 input.
RB5				I/O	ST/DIG	General purpose I/O pin.
PGC/CTED5/RB6	65	52	42			
PGC				I/O	ST/DIG	In-Circuit Debugger and ICSP™ programming clock pin.
CTED5				I	ST	CTMU Edge Input.
RB6				I/O	ST/DIG	General purpose I/O pin.
PGD/CTED6/RB7	58	47	37			
PGD				I/O	ST/DIG	In-Circuit Debugger and ICSP™ programming data pin.
CTED6				I	ST	CTMU Edge 6 input.
RB7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to VDD)

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
SOSCO/SCLKI/PWRLCLK/RC0	45	36	30			
SOSCO				O	—	SOSC oscillator output.
SCLKI				I	ST	Digital SOSC input.
PWRLCLK				I	ST	SOSC input at 50 Hz or 60 Hz only (RTCCLKSEL<1:0> = 11 or 10).
RC0				I/O	ST	General purpose Input pin.
SOSCI/RC1	44	35	29			
SOSCI				I	Analog	Timer1 oscillator input.
RC1				I/O	ST	General purpose Input pin.
SEG13/AN9/RP11/CTED7/RC2	53	43	33			
SEG13				O	Analog	SEG13 output for LCD.
AN9				I	Analog	Analog Input 9.
RP11				I/O	ST/DIG	Remappable Peripheral Pin 11 input/output.
CTED7				I	ST	CTMU Edge 7 input.
RC2				I/O	ST/DIG	General purpose I/O pin.
SEG17/SCL1/RP15/CTED8/RC3	54	44	34			
SEG17				O	Analog	SEG17 output for LCD.
SCL1				I/O	I^2C	I^2C clock input/output.
RP15				I/O	ST/DIG	Remappable Peripheral Pin 15 input/output.
CTED8				I	ST	CTMU Edge 8 input.
RC3				I/O	ST/DIG	General purpose I/O pin.
SEG16/SDA1/RP17/CTED9/RC4	56	45	35			
SEG16				O	Analog	SEG16 output for LCD.
SDA1				I/O	I^2C	I^2C data input/output.
RP17				I/O	ST/DIG	Remappable Peripheral Pin 17 input/output.
CTED9				I	ST	CTMU Edge 9 input.
RC4				I/O	ST/DIG	General purpose I/O pin.
SEG12/RP16/CTED10/RC5	57	46	36			
SEG12				O	Analog	SEG12 output for LCD.
RP16				I/O	ST/DIG	Remappable Peripheral Pin 16 input/output.
CTED10				I	ST	CTMU Edge 10 input.
RC5				I/O	ST/DIG	General purpose I/O pin.
SEG27/RP18/$\overline{UOE}$/CTED11/RC6	47	37	31			
SEG27				O	Analog	SEG27 output for LCD.
RP18				I/O	ST/DIG	Remappable Peripheral Pin 18 input/output.
$\overline{UOE}$/				O	DIG	External USB transceiver NOE output.
CTED11				I	ST	CTMU Edge 11 input.
RC6				I/O	ST/DIG	General purpose I/O pin.
SEG22/RP19/CTED12/RC7	48	38	32			
SEG22				O	Analog	SEG22 output for LCD.
RP19				I/O	ST/DIG	Remappable Peripheral Pin 19 input/output.
CTED12				I	ST	CTMU Edge 12 input.
RC7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
AD0/SEG0/RP20/PSP0/RD0	90	72	58			
AD0				I/O	TTL/DIG	External Memory Address/Data 0.
SEG0				O	Analog	SEG0 output for LCD.
RP20				I/O	ST/DIG	Remappable Peripheral Pin 20 input/output.
PSP0				I/O	ST/DIG	Parallel Slave Port data.
RD0				I/O	ST/DIG	General purpose I/O pin.
AD1/SEG1/RP21/PSP1/RD1	86	69	55			
AD1				I/O	TTL/DIG	External Memory Address/Data 1.
SEG1				O	Analog	SEG1 output for LCD.
RP21				I/O	ST/DIG	Remappable Peripheral Pin 21 input/output.
PSP1				I/O	ST/DIG	Parallel Slave Port data.
RD1				I/O	ST/DIG	General purpose I/O pin.
AD2/SEG2/RP22/PSP2/RD2	84	68	54			
AD2				I/O	TTL/DIG	External Memory Address/Data 2.
SEG2				O	Analog	SEG2 output for LCD.
RP22				I/O	ST/DIG	Remappable Peripheral Pin 22 input/output.
PSP2				I/O	ST/DIG	Parallel Slave Port data.
RD2				I/O	ST/DIG	General purpose I/O pin.
AD3/SEG3/RP23/PSP3/RD3	83	67	53			
AD3				I/O	TTL/DIG	External Memory Address/Data 3.
SEG3				O	Analog	SEG3 output for LCD.
RP23				I/O	ST/DIG	Remappable Peripheral Pin 3 input/output.
PSP3				I/O	ST/DIG	Parallel Slave Port data.
RD3				I/O	ST/DIG	General purpose I/O pin.
AD4/SEG4/RP24/PSP4/RD4	82	66	52			
AD4				I/O	TTL/DIG	External Memory Address/Data 4.
SEG4				O	Analog	SEG4 output for LCD.
RP24				I/O	ST/DIG	Remappable Peripheral Pin 24 input/output.
PSP4				I/O	ST/DIG	Parallel Slave Port data.
RD4				I/O	ST/DIG	General purpose I/O pin.
AD5/SEG5/SDA2/RP25/PSP5/RD5	81	65	51			
AD5				I/O	TTL/DIG	External Memory Address/Data 5.
SEG5				O	Analog	SEG5 output for LCD.
SDA2				I/O	I^2C	I^2C data input/output.
RP25				I/O	ST/DIG	Remappable Peripheral Pin 25 input/output.
PSP5				I/O	ST/DIG	Parallel Slave Port data.
RD5				I/O	ST/DIG	General purpose I/O pin.
AD6/SEG6/SCL2/RP26/PSP6/RD6	79	64	50			
AD6				I/O	TTL/DIG	External Memory Address/Data 6.
SEG6				O	Analog	SEG6 output for LCD.
SCL2				I/O	I^2C	I^2C clock input/output.
RP26				I/O	ST/DIG	Remappable Peripheral Pin 26 input/output.
PSP6				I/O	ST/DIG	Parallel Slave Port data.
RD6				I/O	ST/DIG	General purpose I/O pin.
AD7/SEG7/RP27/REFO2/ PSP7/RD7	78	63	49			
AD7				I/O	TTL/DIG	External Memory Address/Data 7.
SEG7				O	Analog	SEG7 output for LCD.
RP27				I/O	ST/DIG	Remappable Peripheral Pin 27 input/output.
REFO2				O	DIG	Reference output clock.
PSP7				I/O	ST/DIG	Parallel Slave Port data
RD7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
AD8/LCDBIAS1/RP28/$\overline{RD}$/RE0	4	4	2			
AD8				I/O	TTL/DIG	External Memory Address/Data 8.
LCDBIAS1				I	Analog	BIAS1 input for LCD.
RP28				I/O	ST/DIG	Remappable Peripheral Pin 28 input/output.
$\overline{RD}$				I	TTL	Parallel Slave Port read strobe.
RE0				I/O	ST/DIG	General purpose I/O pin.
AD9/LCDBIAS2/RP29/$\overline{WR}$/RE1	3	3	1			
AD9				I/O	TTL/DIG	External Memory Address/Data 9.
LCDBIAS2				I	Analog	BIAS2 input for LCD.
RP29				I/O	ST/DIG	Remappable Peripheral Pin 29 input/output.
$\overline{WR}$				I	TTL	Parallel Slave Port write strobe.
RE1				I/O	ST/DIG	General purpose I/O pin.
AD10/LCDBIAS3/RP30/$\overline{CS}$/RE2	98	78	64			
AD10				I/O	TTL/DIG	External Memory Address/Data 10.
LCDBIAS3				I	Analog	BIAS3 input for LCD.
RP30				I/O	ST/DIG	Remappable Peripheral Pin 30 input/output.
$\overline{CS}$				I	TTL	Parallel Slave Port chip select.
RE2				I/O	ST/DIG	General purpose I/O pin.
AD11/COM0/RP33/REFO1/RE3	97	77	63			
AD11				I/O	TTL/DIG	External Memory Address/Data 11.
COM0				O	Analog	COM0 output for LCD.
RP33				I/O	ST/DIG	Remappable Peripheral Pin 33 input/output.
REFO1				O	DIG	Reference output clock.
RE3				I/O	ST/DIG	General purpose I/O pin.
AD12/COM1/RP32/RE4	95	76	62			
AD12				I/O	TTL/DIG	External Memory Address/Data 12.
COM1				O	Analog	COM1 output for LCD.
RP32				I/O	ST/DIG	Remappable Peripheral Pin 32 input/output.
RE4				I/O	ST/DIG	General purpose I/O pin.
AD13/COM2/RP37/RE5	94	75	61			
AD13				I/O	TTL/DIG	External Memory Address/Data 13.
COM2				O	Analog	COM2 output for LCD.
RP37				I/O	ST/DIG	Remappable Peripheral Pin 37 input/output.
RE5				I/O	ST/DIG	General purpose I/O pin.
AD14/COM3/RP34/RE6	93	74	60			
AD14				I/O	TTL/DIG	External Memory Address/Data 14.
COM3				O	Analog	COM3 output for LCD.
RP34				I/O	ST/DIG	Remappable Peripheral Pin 34 input/output.
RE6				I/O	ST/DIG	General purpose I/O pin.
AD15/LCDBIAS0/RP31/RE7	92	73	59			
AD15				I/O	TTL/DIG	External Memory Address/Data 15.
LCDBIAS0				I	Analog	BIAS0 input for LCD.
RP31				I/O	ST/DIG	Remappable Peripheral Pin 31 input/output.
RE7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
SEG20/AN7/CTMUI/C2INB/RP36/RF2	23	18	16			
SEG20				O	Analog	SEG20 output for LCD.
AN7				I	Analog	Analog Input 7.
CTMUI				O	—	CTMU pulse generator charger for the C2INB comparator input.
C2INB				I	Analog	Comparator 2 Input B.
RP36				I/O	ST/DIG	Remappable Peripheral Pin 36 input/output.
RF2				I/O	ST/DIG	General purpose I/O pin.
D-/RF3	22	17	15			
D-				I/O	—	USB bus minus line input/output.
RF3				I	ST	General purpose input pin.
D+/RF4	20	16	14			
D+				I/O	—	USB bus plus line input/output.
RF4				I	ST	General purpose input pin.
SEG23/CVREF/AN10/C1INB/RP35/RF5	19	15	13			
SEG23				O	Analog	SEG23 output for LCD.
CVREF				O	Analog	Comparator reference voltage output.
AN10				I	Analog	Analog Input 10.
C1INB				I	Analog	Comparator 1 Input B.
RP35				I/O	ST/DIG	Remappable Peripheral Pin 35 input/output.
RF5				I/O	ST/DIG	General purpose I/O pin.
SEG24/AN11/C1INA/RP40/RF6	18	14	12			
SEG24				O	Analog	SEG24 output for LCD.
AN11				I	Analog	Analog Input 11.
C1INA				I	Analog	Comparator 1 Input A.
RP40				I/O	ST/DIG	Remappable Peripheral Pin 40 input/output.
RF6				I/O	ST/DIG	General purpose I/O pin.
SEG25/AN5/RP38/RF7	17	13	11			
SEG25				O	Analog	SEG25 output for LCD.
AN5				I	Analog	Analog Input 5.
RP38				I/O	ST/DIG	Remappable Peripheral Pin 38 input/output.
RF7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to VDD)

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
COM4/SEG28/AN8/RP46/RG0	6	5	3			
COM4				O	Analog	COM4 output for LCD.
SEG28				O	Analog	SEG28 output for LCD.
AN8				I	Analog	Analog Input 8.
RP46				I/O	ST/DIG	Remappable Peripheral Pin 46 input/output.
RG0				I/O	ST/DIG	General purpose I/O pin.
COM5/SEG29/AN19/RP39/RG1	7	6	4			
COM5				O	Analog	COM5 output for LCD.
SEG29				O	Analog	SEG29 output for LCD.
AN19				I	Analog	Analog Input 19.
RP39				I/O	ST/DIG	Remappable Peripheral Pin 39 input/output.
RG1				I/O	ST/DIG	General purpose I/O pin.
COM6/SEG30/AN18/C3INA/RP42/RG2	8	7	5			
COM6				O	Analog	COM6 output for LCD.
SEG30				O	Analog	SEG30 output for LCD.
AN18				I	Analog	Analog Input 18.
C3INA				I	Analog	Comparator 3 Input A.
RP42				I/O	ST/DIG	Remappable Peripheral Pin 42 input/output.
RG2				I/O	ST/DIG	General purpose I/O pin.
COM7/SEG31/AN17/C3INB/RP43/RG3	9	8	6			
COM7				O	Analog	COM7 output for LCD.
SEG31				O	Analog	SEG31 output for LCD.
AN17				I	Analog	Analog Input 17.
C3INB				I	Analog	Comparator 3 Input B.
RP43				I/O	ST/DIG	Remappable Peripheral Pin 43 input/output.
RG3				I/O	ST/DIG	General purpose I/O pin.
SEG26/AN16/C3INC/RP44/RTCC/RG4	12	10	8			
SEG26				O	Analog	SEG26 output for LCD.
AN16				I	Analog	Analog Input 16.
C3INC				I	Analog	Comparator 3 Input C.
RP44				I/O	ST/DIG	Remappable Peripheral Pin 44 input/output.
RTCC				O	—	RTCC output.
RG4				I/O	ST/DIG	General purpose I/O pin.
RG6	89			I/O	ST/DIG	General purpose I/O pin.
RG7	96			I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
A16/SEG47/AN23/RH0	99	79				
A16				O	DIG	External Memory Address 16.
SEG47				O	Analog	SEG47 output for LCD.
AN23				I	Analog	Analog Input 23.
RH0				I/O	ST/DIG	General purpose I/O pin.
A17/SEG46/AN22/RH1	100	80				
A17				O	DIG	External Memory Address 17.
SEG46				O	Analog	SEG46 output for LCD.
AN22				I	Analog	Analog Input 22.
RH1				I/O	ST/DIG	General purpose I/O pin.
A18/SEG45/AN21/RH2	1	1				
A18				O	DIG	External Memory Address 18.
SEG45				O	Analog	SEG45 output for LCD.
AN21				I	Analog	Analog Input 21.
RH2				I/O	ST/DIG	General purpose I/O pin.
A19/SEG44/AN20/RH3	2	2				
A19				O	DIG	External Memory Address 19.
SEG44				O	Analog	SEG44 output for LCD.
AN20				I	Analog	Analog Input 20.
RH3				I/O	ST/DIG	General purpose I/O pin.
SEG40/AN12/C2INC/RH4	27	22				
SEG40				O	Analog	SEG40 output for LCD.
AN12				I	Analog	Analog Input12.
C2INC				I	Analog	Comparator 2 Input C.
RH4				I/O	ST/DIG	General purpose I/O pin.
SEG41/AN13/C2IND/RH5	26	21				
SEG41				O	Analog	SEG41 output for LCD.
AN13				I	Analog	Analog Input 13.
C2IND				I	Analog	Comparator 2 Input D.
RH5				I/O	ST/DIG	General purpose I/O pin.
SEG42/AN14/C1INC/RH6	25	20				
SEG42				O	Analog	SEG42 output for LCD.
AN14				I	Analog	Analog Input 14.
C1INC				I	Analog	Comparator 1 Input C.
RH6				I/O	ST/DIG	General purpose I/O pin.
SEG43/AN15/RH7	24	19				
SEG43				O	Analog	SEG43 output for LCD.
AN15				I	Analog	Analog Input 15.
RH7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
ALE/SEG32/RJ0	77	62				
ALE				O	DIG	External memory address latch enable.
SEG32				O	Analog	SEG32 output for LCD.
RJ0				I/O	ST/DIG	General purpose I/O pin.
$\overline{OE}$/SEG33/RJ1	76	61				
$\overline{OE}$				O	DIG	External memory output enable.
SEG33				O	Analog	SEG33 output for LCD.
RJ1				I/O	ST/DIG	General purpose I/O pin.
$\overline{WRL}$/SEG34/RJ2	75	60				
$\overline{WRL}$				O	DIG	External memory write low control.
SEG34				O	Analog	SEG34 output for LCD.
RJ2				I/O	ST/DIG	General purpose I/O pin.
$\overline{WRH}$/SEG35/RJ3	74	59				
$\overline{WRH}$				O	DIG	External memory write high control.
SEG35				O	Analog	SEG35 output for LCD.
RJ3				I/O	ST/DIG	General purpose I/O pin.
BA0/SEG39/RJ4	49	39				
BA0				O	DIG	External Memory Byte Address 0 control
SEG39				O	Analog	SEG39 output for LCD.
RJ4				I/O	ST/DIG	General purpose I/O pin.
$\overline{CE}$/SEG38/RJ5	50	40				
$\overline{CE}$				O	DIG	External memory chip enable control.
SEG38				O	Analog	SEG38 output for LCD.
RJ5				I/O	ST/DIG	General purpose I/O pin.
$\overline{LB}$/SEG37/RJ6	51	41				
$\overline{LB}$				O	DIG	External memory low byte control.
SEG37				O	Analog	SEG37 output for LCD.
RJ6				I/O	ST/DIG	General purpose I/O pin.
$\overline{UB}$/SEG36/RJ7	52	42				
$\overline{UB}$				O	DIG	External memory high byte control.
SEG36				O	Analog	SEG36 output for LCD.
RJ7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
SEG56/RK0	46					
SEG56				O	Analog	SEG56 output for LCD.
RK0				I/O	ST/DIG	General purpose I/O pin.
SEG57/RK1	55					
SEG57				O	Analog	SEG57 output for LCD.
RK1				I/O	ST/DIG	General purpose I/O pin.
SEG58/RK2	60					
SEG58				O	Analog	SEG58 output for LCD.
RK2				I/O	ST/DIG	General purpose I/O pin.
SEG59/RK3	63					
SEG59				O	Analog	SEG59 output for LCD.
RK3				I/O	ST/DIG	General purpose I/O pin.
SEG60/RK4	66					
SEG60				O	Analog	SEG60 output for LCD.
RK4				I/O	ST/DIG	General purpose I/O pin.
SEG61/RK5	71					
SEG61				O	Analog	SEG61 output for LCD.
RK5				I/O	ST/DIG	General purpose I/O pin.
SEG62/RK6	80					
SEG62				O	Analog	SEG62 output for LCD.
RK6				I/O	ST/DIG	General purpose I/O pin.
SEG63/RK7	85					
SEG63				O	Analog	SEG63 output for LCD.
RK7				I/O	ST/DIG	General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
SEG48/RL0 SEG48 RL0	91			 O I/O	 Analog ST/DIG	 SEG48 output for LCD. General purpose I/O pin.
SEG49/RL1 SEG49 RL1	10			 O I/O	 Analog ST/DIG	 SEG49 output for LCD. General purpose I/O pin.
SEG50/RL2 SEG50 RL2	13			 O I/O	 Analog ST/DIG	 SEG50 output for LCD. General purpose I/O pin.
SEG51/RL3 SEG51 RL3	16			 O I/O	 Analog ST/DIG	 SEG51 output for LCD. General purpose I/O pin.
SEG52/RL4 SEG52 RL4	21			 O I/O	 Analog ST/DIG	 SEG52 output for LCD. General purpose I/O pin.
SEG53/RL5 SEG53 RL5	30			 O I/O	 Analog ST/DIG	 SEG53 output for LCD. General purpose I/O pin.
SEG54/RL6 SEG54 RL6	38			 O I/O	 Analog ST/DIG	 SEG54 output for LCD. General purpose I/O pin.
SEG55/RL7 SEG55 RL7	41			 O I/O	 Analog ST/DIG	 SEG55 output for LCD. General purpose I/O pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to V_{DD})

TABLE 1-4: PIC18FXXJ94 PINOUT I/O DESCRIPTIONS (CONTINUED)

Pin Name	Pin Number			Pin Type	Buffer Type	Description
	100	80	64			
VDD	5 40 59 88	32 48 71	26 38 57	P	—	Positive supply for logic and I/O pins.
VSS	14 35 39 64 87	11 31 51 70	9 25 41 56	P	—	Ground reference for logic and I/O pins.
AVDD	31	25	19	P	—	Positive supply for analog modules.
AVSS	32	26	20	P	—	Ground reference for analog modules.
VDDCORE/VCAP VDDCORE VCAP	15	12	10	 P P	 — —	 Core logic power or external filter capacitor connection. External filter capacitor connection (regulator enabled/disabled).
VBAT	29	24	18	P	—	
VUSB3V3	28	23	17	P	—	USB voltage input pin.

Legend: TTL = TTL compatible input
ST = Schmitt Trigger input with CMOS levels
I = Input
P = Power
I^2C = I^2C/SMBus
CMOS = CMOS compatible input or output
Analog = Analog input
O = Output
OD = Open-Drain (no P diode to VDD)

2.0 GUIDELINES FOR GETTING STARTED WITH PIC18FJ MICROCONTROLLERS

2.1 Basic Connection Requirements

Getting started with the PIC18FXXJ94 of 8-bit microcontrollers requires attention to a minimal set of device pin connection before proceeding with development.

The following pins must always be connected:

- All VDD and VSS pins (see **Section 2.2 "Power Supply Pins"**)
- All AVDD and AVSS pins, regardless of whether or not the analog device features are used (see **Section 2.2 "Power Supply Pins"**)
- $\overline{\text{MCLR}}$ pin (see **Section 2.3 "Master Clear (MCLR) Pin"**)

These pins must also be connected if they are being used in the end application:

- PGC/PGD pins used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes (see **Section 2.5 "ICSP Pins"**)
- OSC1 and OSC2 pins when an external oscillator source is used (see **Section 2.6 "External Oscillator Pins"**)

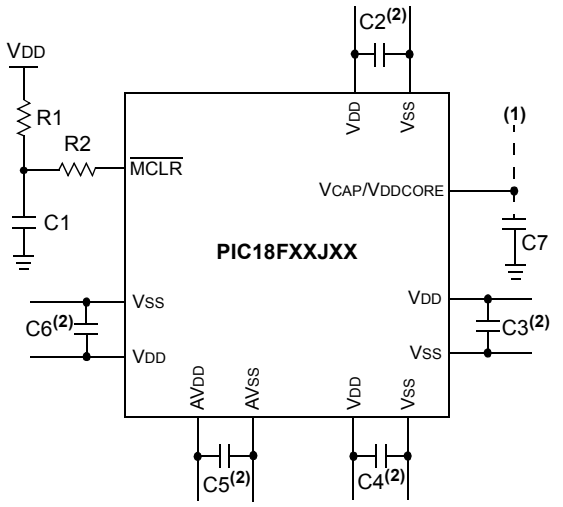
Additionally, the following pins may be required:

- VREF+/VREF- pins are used when external voltage reference for analog modules is implemented

Note:	The AVDD and AVSS pins must always be connected, regardless of whether any of the analog modules are being used.

The minimum mandatory connections are shown in Figure 2-1.

FIGURE 2-1: RECOMMENDED MINIMUM CONNECTIONS



Key (all values are recommendations):

C1 through C6: 0.1 μF, 20V ceramic

C7: 10 μF, 6.3V or greater, tantalum or ceramic

R1: 10 kΩ

R2: 100Ω to 470Ω

Note 1: See **Section 2.4 "Core Voltage Regulator (VCAP/VDDCORE)"** for explanation of VCAP/VDDCORE connections.

2: The example shown is for a PIC18F device with five VDD/VSS and AVDD/AVSS pairs. Other devices may have more or less pairs; adjust the number of decoupling capacitors appropriately.

2.2 Power Supply Pins

2.2.1 DECOUPLING CAPACITORS

The use of decoupling capacitors on every pair of power supply pins, such as VDD, VSS, AVDD and AVSS, is required.

Consider the following criteria when using decoupling capacitors:

- **Value and type of capacitor:** A 0.1 μF (100 nF), 10-20V capacitor is recommended. The capacitor should be a low-ESR device, with a resonance frequency in the range of 200 MHz and higher. Ceramic capacitors are recommended.
- **Placement on the printed circuit board:** The decoupling capacitors should be placed as close to the pins as possible. It is recommended to place the capacitors on the same side of the board as the device. If space is constricted, the capacitor can be placed on another layer on the PCB using a via; however, ensure that the trace length from the pin to the capacitor is no greater than 0.25 inch (6 mm).
- **Handling high-frequency noise:** If the board is experiencing high-frequency noise (upward of tens of MHz), add a second ceramic type capacitor in parallel to the above described decoupling capacitor. The value of the second capacitor can be in the range of 0.01 μF to 0.001 μF. Place this second capacitor next to each primary decoupling capacitor. In high-speed circuit designs, consider implementing a decade pair of capacitances as close to the power and ground pins as possible (e.g., 0.1 μF in parallel with 0.001 μF).
- **Maximizing performance:** On the board layout from the power supply circuit, run the power and return traces to the decoupling capacitors first, and then to the device pins. This ensures that the decoupling capacitors are first in the power chain. Equally important is to keep the trace length between the capacitor and the power pins to a minimum, thereby reducing PCB trace inductance.

2.2.2 TANK CAPACITORS

On boards with power traces running longer than six inches in length, it is suggested to use a tank capacitor for integrated circuits, including microcontrollers, to supply a local power source. The value of the tank capacitor should be determined based on the trace resistance that connects the power supply source to the device, and the maximum current drawn by the device in the application. In other words, select the tank capacitor so that it meets the acceptable voltage sag at the device. Typical values range from 4.7 μF to 47 μF.

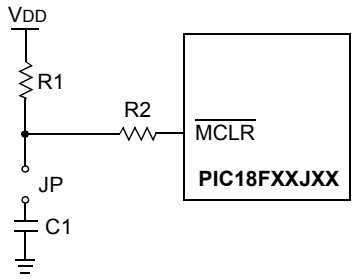
2.3 Master Clear ($\overline{\text{MCLR}}$) Pin

The $\overline{\text{MCLR}}$ pin provides two specific device functions: Device Reset, and Device Programming and Debugging. If programming and debugging are not required in the end application, a direct connection to VDD may be all that is required. The addition of other components, to help increase the application's resistance to spurious Resets from voltage sags, may be beneficial. A typical configuration is shown in Figure 2-1. Other circuit designs may be implemented, depending on the application's requirements.

During programming and debugging, the resistance and capacitance that can be added to the pin must be considered. Device programmers and debuggers drive the $\overline{\text{MCLR}}$ pin. Consequently, specific voltage levels (VIH and VIL) and fast signal transitions must not be adversely affected. Therefore, specific values of R1 and C1 will need to be adjusted based on the application and PCB requirements. For example, it is recommended that the capacitor, C1, be isolated from the $\overline{\text{MCLR}}$ pin during programming and debugging operations by using a jumper (Figure 2-2). The jumper is replaced for normal run-time operations.

Any components associated with the $\overline{\text{MCLR}}$ pin should be placed within 0.25 inch (6 mm) of the pin.

FIGURE 2-2: EXAMPLE OF $\overline{\text{MCLR}}$ PIN CONNECTIONS



Note 1: R1 ≤ 10 kΩ is recommended. A suggested starting value is 10 kΩ. Ensure that the $\overline{\text{MCLR}}$ pin VIH and VIL specifications are met.

2: R2 ≤ 470Ω will limit any current flowing into $\overline{\text{MCLR}}$ from the external capacitor, C, in the event of $\overline{\text{MCLR}}$ pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS). Ensure that the $\overline{\text{MCLR}}$ pin VIH and VIL specifications are met.

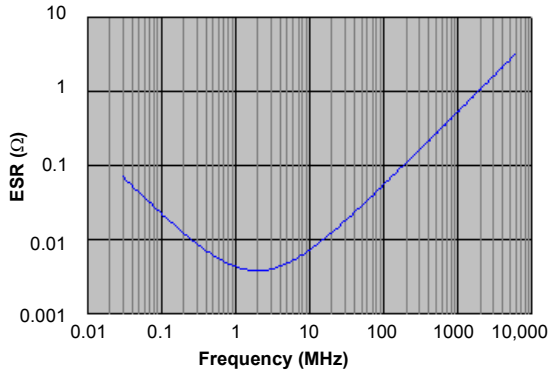
2.4 Core Voltage Regulator (VCAP/ VDDCORE)

A low-ESR (< 5Ω) capacitor is required on the VCAP pin to stabilize the output voltage of the on-chip voltage regulator. The VCAP pin must not be connected to VDD and must use a capacitor of 10 µF connected to ground. The type can be ceramic or tantalum. Suitable examples of capacitors are shown in Table 2-1. Capacitors with equivalent specification can be used.

Designers may use Figure 2-3 to evaluate ESR equivalence of candidate devices.

It is recommended that the trace length not exceed 0.25 inch (6 mm). Refer to **Section 30.0 "Electrical Specifications"** for additional information.

FIGURE 2-3: FREQUENCY vs. ESR PERFORMANCE FOR SUGGESTED VCAP



Note: Typical data measurement at 25°C, 0V DC bias.

TABLE 2-1: SUITABLE CAPACITOR EQUIVALENTS

Make	Part #	Nominal Capacitance	Base Tolerance	Rated Voltage	Temp. Range
TDK	C3216X7R1C106K	10 µF	±10%	16V	-55 to 125ºC
TDK	C3216X5R1C106K	10 µF	±10%	16V	-55 to 85ºC
Panasonic	ECJ-3YX1C106K	10 µF	±10%	16V	-55 to 125ºC
Panasonic	ECJ-4YB1C106K	10 µF	±10%	16V	-55 to 85ºC
Murata	GRM32DR71C106KA01L	10 µF	±10%	16V	-55 to 125ºC
Murata	GRM31CR61C106KC31L	10 µF	±10%	16V	-55 to 85ºC

2.4.1 CONSIDERATIONS FOR CERAMIC CAPACITORS

In recent years, large value, low-voltage, surface-mount ceramic capacitors have become very cost effective in sizes up to a few tens of microfarad. The low-ESR, small physical size and other properties make ceramic capacitors very attractive in many types of applications.

Ceramic capacitors are suitable for use with the VDDCORE voltage regulator of this microcontroller. However, some care is needed in selecting the capacitor to ensure that it maintains sufficient capacitance over the intended operating range of the application.

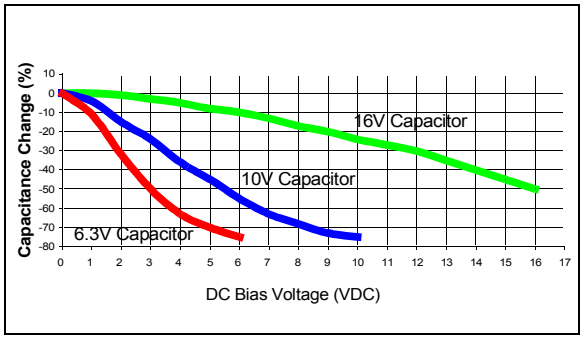
Typical low-cost, 10 µF ceramic capacitors are available in X5R, X7R and Y5V dielectric ratings (other types are also available, but are less common). The initial tolerance specifications for these types of capacitors are often specified as ±10% to ±20% (X5R and X7R), or -20%/+80% (Y5V). However, the effective capacitance that these capacitors provide in an application circuit will also vary based on additional factors, such as the applied DC bias voltage and the temperature. The total in-circuit tolerance is, therefore, much wider than the initial tolerance specification.

The X5R and X7R capacitors typically exhibit satisfactory temperature stability (ex: ±15% over a wide temperature range, but consult the manufacturer's data sheets for exact specifications). However, Y5V capacitors typically have extreme temperature tolerance specifications of +22%/-82%. Due to the extreme temperature tolerance, a 10 µF nominal rated Y5V type capacitor may not deliver enough total capacitance to meet minimum VDDCORE voltage regulator stability and transient response requirements. Therefore, Y5V capacitors are not recommended for use with the VDDCORE regulator if the application must operate over a wide temperature range.

In addition to temperature tolerance, the effective capacitance of large value ceramic capacitors can vary substantially, based on the amount of DC voltage applied to the capacitor. This effect can be very significant, but is often overlooked or is not always documented.

A typical DC bias voltage vs. capacitance graph for X7R type and Y5V type capacitors is shown in Figure 2-4.

FIGURE 2-4: DC BIAS VOLTAGE vs. CAPACITANCE CHARACTERISTICS



When selecting a ceramic capacitor to be used with the VDDCORE voltage regulator, it is suggested to select a high-voltage rating, so that the operating voltage is a small percentage of the maximum rated capacitor voltage. For example, choose a ceramic capacitor rated at 16V for the 2.5V VDDCORE voltage. Suggested capacitors are shown in Table 2-1.

2.5 ICSP Pins

The PGC and PGD pins are used for In-Circuit Serial Programming™ (ICSP™) and debugging purposes. It is recommended to keep the trace length between the ICSP connector and the ICSP pins on the device as short as possible. If the ICSP connector is expected to experience an ESD event, a series resistor is recommended, with the value in the range of a few tens of ohms, not to exceed 100Ω.

Pull-up resistors, series diodes, and capacitors on the PGC and PGD pins are not recommended as they will interfere with the programmer/debugger communications to the device. If such discrete components are an application requirement, they should be removed from the circuit during programming and debugging. Alternatively, refer to the AC/DC characteristics and timing requirements information in the respective device Flash programming specification for information on capacitive loading limits, and pin input voltage high (VIH) and input low (VIL) requirements.

For device emulation, ensure that the "Communication Channel Select" (i.e., PGCx/PGDx pins), programmed into the device, matches the physical connections for the ICSP to the Microchip debugger/emulator tool.

For more information on available Microchip development tools connection requirements, refer to **Section 31.0 "Development Support"**.

2.6 External Oscillator Pins

Many microcontrollers have options for at least two oscillators: a high-frequency primary oscillator and a low-frequency secondary oscillator (refer to **Section 3.0 "Oscillator Configurations"** for details).

The oscillator circuit should be placed on the same side of the board as the device. Place the oscillator circuit close to the respective oscillator pins with no more than 0.5 inch (12 mm) between the circuit components and the pins. The load capacitors should be placed next to the oscillator itself, on the same side of the board.

Use a grounded copper pour around the oscillator circuit to isolate it from surrounding circuits. The grounded copper pour should be routed directly to the MCU ground. Do not run any signal traces or power traces inside the ground pour. Also, if using a two-sided board, avoid any traces on the other side of the board where the crystal is placed.

Layout suggestions are shown in Figure 2-5. In-line packages may be handled with a single-sided layout that completely encompasses the oscillator pins. With fine-pitch packages, it is not always possible to completely surround the pins and components. A suitable solution is to tie the broken guard sections to a mirrored ground layer. In all cases, the guard trace(s) must be returned to ground.

In planning the application's routing and I/O assignments, ensure that adjacent port pins, and other signals in close proximity to the oscillator, are benign (i.e., free of high frequencies, short rise and fall times, and other similar noise).

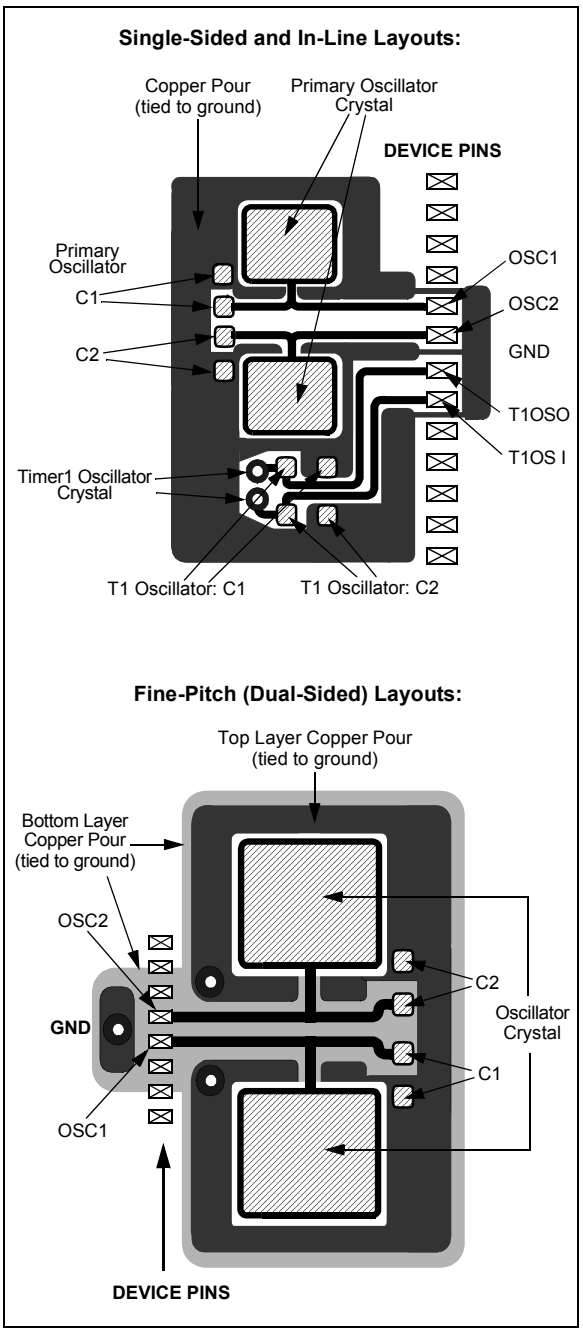
For additional information and design guidance on oscillator circuits, refer to these Microchip Application Notes, available at the corporate website (www.microchip.com):

- *AN826, "Crystal Oscillator Basics and Crystal Selection for rfPIC™ and PICmicro® Devices"*
- *AN849, "Basic PICmicro® Oscillator Design"*
- *AN943, "Practical PICmicro® Oscillator Analysis and Design"*
- *AN949, "Making Your Oscillator Work"*

2.7 Unused I/Os

Unused I/O pins should be configured as outputs and driven to a logic low state. Alternatively, connect a 1 kΩ to 10 kΩ resistor to VSS on unused pins and drive the output to logic low.

FIGURE 2-5: SUGGESTED PLACEMENT OF THE OSCILLATOR CIRCUIT



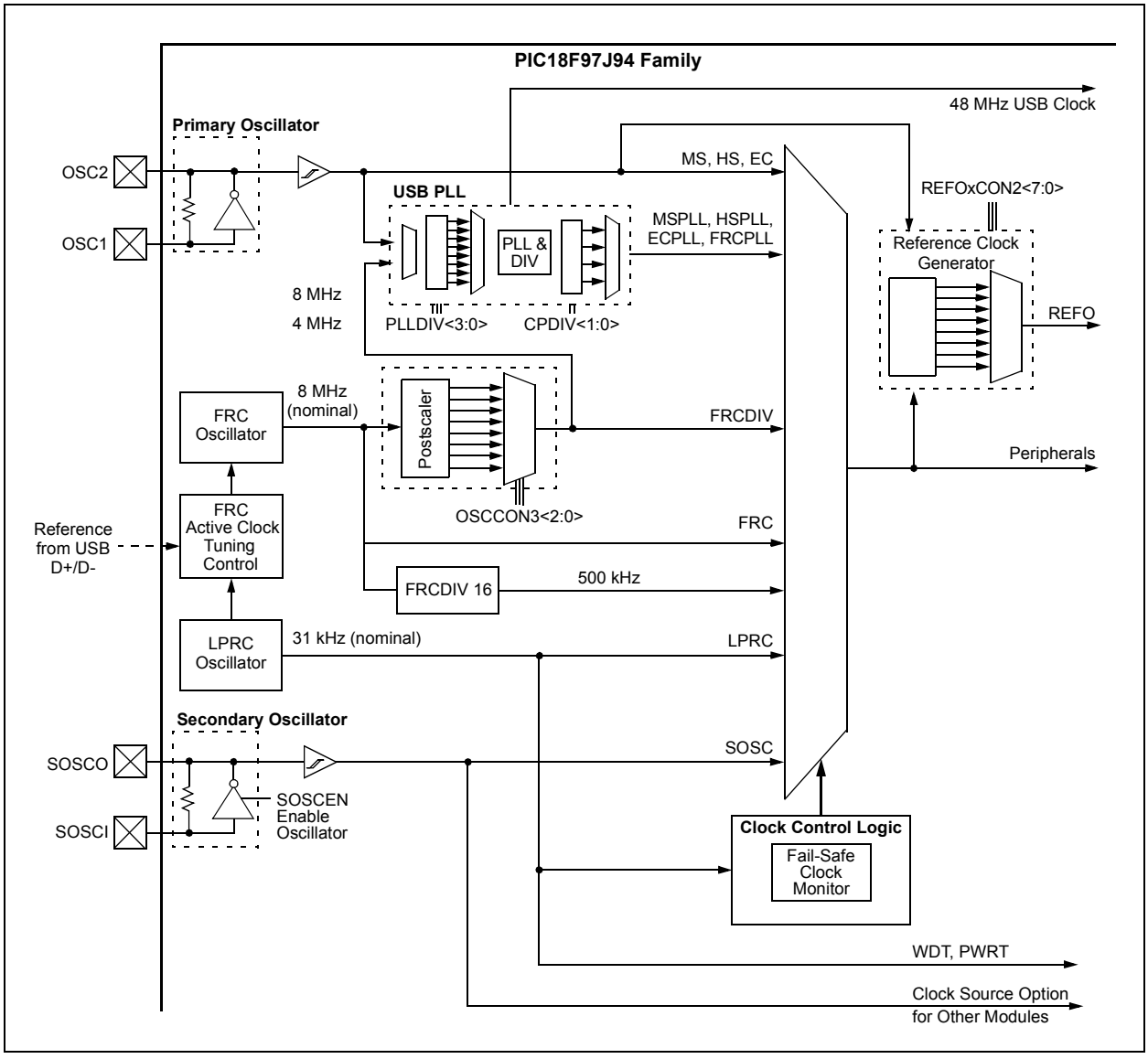
3.0 OSCILLATOR CONFIGURATIONS

This section describes the PIC18F oscillator system and its operation. The PIC18F oscillator system has the following modules and features:

- A total of four external and internal oscillator options as clock sources, providing up to 11 different clock modes
- An on-chip USB PLL block to provide a stable 48 MHz clock for the USB module, as well as a range of frequency options for the system clock
- Software-controllable switching between various clock sources
- Software-controllable postscaler for selective clocking of CPU for system power savings
- A Fail-Safe Clock Monitor (FSCM) that detects clock failure and permits safe application recovery or shutdown
- A separate and independently configurable system clock output for synchronizing external hardware

A simplified diagram of the oscillator system is shown in Figure 3-1.

FIGURE 3-1: PIC18F GENERAL SYSTEM CLOCK DIAGRAM



3.1 CPU Clocking Scheme

The system clock source can be provided by one of four sources:

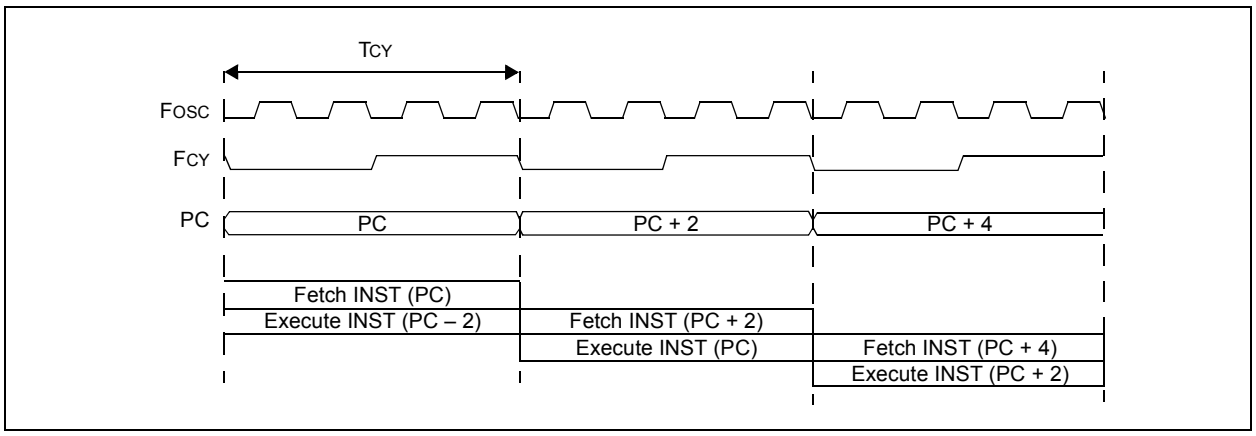
- Primary Oscillator (POSC) on the OSC1 and OSC2 pins
- Secondary Oscillator (SOSC) on the SOSCI and SOSCO pins
- Fast Internal RC (FRC) Oscillator
- Low-Power Internal RC (LPRC) Oscillator

The Primary Oscillator and FRC sources have the option of using the internal USB PLL block, which generates both the USB module clock and a separate system clock from the 96 MHz PLL. Refer to **Section 3.8.1 "Oscillator Modes and USB Operation"** for additional information.

The internal FRC provides an 8 MHz clock source. It can optionally be reduced by the programmable clock divider to provide a range of system clock frequencies.

The selected clock source generates the processor and peripheral clock sources. The processor clock source is divided by four to produce the internal instruction cycle clock, FCY. In this document, the instruction cycle clock is also denoted by FOSC/4. The internal instruction cycle clock, FOSC/4, can be provided on the OSC2 I/O pin for some operating modes of the Primary Oscillator. The timing diagram in Figure 3-2 shows the relationship between the processor clock source and instruction execution.

FIGURE 3-2: CLOCK OR INSTRUCTION CYCLE TIMING



3.2 Oscillator Configuration

The oscillator source (and operating mode) that is used at a device Power-on Reset (POR) event is selected using Configuration bit settings. The Oscillator Configuration bit settings are in the Configuration registers located in the program memory (refer to **Section 28.1 "Configuration Bits"** for more information). The Primary Oscillator Configuration bits, POSCMD<1:0> (CONFIG3L<1:0>), and Oscillator Configuration bits, FOSC<2:0> (CONFIG2L<2:0>), select the oscillator source that is used at a POR. The FRC Oscillator with Postscaler (FRCDIV) is the default (unprogrammed) selection. The Secondary Oscillator, or one of the internal oscillators, may be chosen by programming these bit locations.

The Configuration bits allow users to choose between 11 different clock modes, as shown in Table 3-1.

TABLE 3-1: CONFIGURATION BIT VALUES FOR CLOCK SELECTION

Oscillator Mode	Oscillator Source	POSCMD<1:0>	FOSC<2:0>	Notes
Fast RC Oscillator with Postscaler (FRCDIV)	Internal	11	111	1, 2
Fast RC Oscillator divided by 16 (FRC500kHz)	Internal	11	110	1
Low-Power RC Oscillator (LPRC)	Internal	11	101	1
Secondary (Timer1) Oscillator (SOSC)	Secondary	11	100	1
Primary Oscillator (HS) with PLL Module (HSPLL)	Primary	10	011	
Primary Oscillator (MS) with PLL Module (MSPLL)	Primary	01	011	
Primary Oscillator (EC) with PLL Module (ECPLL)	Primary	00	011	
Primary Oscillator (HS)	Primary	10	010	
Primary Oscillator (MS)	Primary	01	010	
Primary Oscillator (EC)	Primary	00	010	
Fast RC Oscillator with PLL Module (FRCPLL)	Internal	11	001	1
Fast RC Oscillator (FRC)	Internal	11	000	1

Note 1: OSC2 pin function is determined by the CLKOEN Configuration bit.

2: Default oscillator mode for an unprogrammed (erased) device.

3.2.1 CLOCK SWITCHING MODE CONFIGURATION BITS

The FSCMx Configuration bits (CONFIG3L<5:4>) are used to jointly configure device clock switching and the Fail-Safe Clock Monitor (FSCM). Clock switching is enabled only when FSCM1 is programmed ('0'). The FSCM is enabled only when FSCM<1:0> are both programmed ('00').

3.2.2 OSC1 AND OSC2 PIN FUNCTIONS IN NON-CRYSTAL MODES

When the Primary Oscillator on OSC1 and OSC2 is not configured as the clock source (POSCMD<1:0> = 11), the OSC1 pin is automatically reconfigured as a digital I/O. In this configuration, as well as when the Primary Oscillator is configured for EC mode (POSCMD<1:0> = 00), the OSC2 pin can also be configured as a digital I/O by programming the CLKOEN Configuration bit (CONFIG2L<5>).

When CLKOEN is unprogrammed ('1'), a FOSC/4 clock output is available on OSC2 for testing or synchronization purposes. With CLKOEN programmed ('0'), the OSC2 pin becomes a general purpose I/O pin. In both of these configurations, the feedback device between OSC1 and OSC2 is turned off to save current.

3.3 Control Registers

The operation of the oscillator is controlled by six Special Function Registers (SFRs):

- OSCCON
- OSCCON2
- OSCCON3
- OSCCON4
- ACTCON
- OSCTUNE

3.3.1 OSCILLATOR CONTROL REGISTER (OSCCON)

The OSCCON register (Register 3-1) is the main control register for the oscillator. It controls clock source switching and allows the monitoring of clock sources.

The COSCx (OSCCON<6:4>) Status bits are read-only bits that indicate the current oscillator source the device is operating from. The COSCx bits default to the Internal Fast RC Oscillator with Postscaler (FRCDIV), configured for 4 MHz, on a Power-on Reset (POR) and Master Clear Reset ($\overline{MCLR}$). A clock switch will automatically be performed to the new oscillator source selected by the FOSCx Configuration bits (CONFIG2L<2:0>). The COSCx bits will change to indicate the new oscillator source at the end of a clock switch operation.

The NOSCx Status bits select the clock source for the next clock switch operation. On POR and $\overline{MCLR}$s, these bits automatically select the oscillator source defined by the FOSCx Configuration bits. These bits can be modified by software.

Setting the CLKLOCK bit (OSCCON2<7>) prevents clock switching if the FSCM1 Configuration bit is set. If the FSCM1 bit is clear, the CLKLOCK bit state is ignored and clock switching can occur.

The IOLOCK bit (OSCCON2<6>) is used to unlock the Peripheral Pin Select (PPS) feature; it has no function in the system clock's operation.

The LOCK Status bit (OSCCON2<5>) is read-only and indicates the status of the PLL circuit. It is set when the PLL achieves a frequency lock and is reset when a valid clock switching sequence is initiated. It reads as '0' whenever the PLL is not used as part of the current clock source.

The CF Status bit (OSCCON2<3>) is a readable/clearable Status bit that indicates a clock failure; it is reset whenever a valid clock switch occurs.

The POSCEN bit (OSCCON2<2>) is used to control the operation of the Primary Oscillator in Sleep mode. Setting this bit bypasses the normal automatic shutdown of the oscillator whenever Sleep mode is invoked.

The Secondary Oscillator can be turned on by a variety of options:

- SOSCGO – OSCCON2<1>
- SOSCSEL – CONFIG2L<3>
- FOSC<2:0> – CONFIG2L<2:0>
- DSWDTOSC – CONFIG8H<1>
- RTCEN – RTCCON1<7>
- SOSCEN – T1CON<3>, T3CON<3> or T5CON<3>

The ACTCON register (Register 3-10) controls the Active Clock Tuning features.

REGISTER 3-1: OSCCON: OSCILLATOR CONTROL REGISTER

R/W-0	R-x	R-x	R-x	U-0	R/W-x	R/W-x	R/W-x
IDLEN	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **IDLEN:** Idle Enable bit
1 = SLEEP instruction invokes Idle mode
0 = SLEEP instruction invokes Sleep mode

bit 6-4 **COSC<2:0>:** Current Oscillator Selection bits (read-only)
000 = Fast RC Oscillator (FRC)
001 = Fast RC Oscillator (FRC), divided by N, with PLL module
010 = Primary Oscillator (MS, HS, EC)
011 = Primary Oscillator (MS, HS, EC) with PLL module
100 = Secondary Oscillator (SOSC)
101 = Low-Power RC Oscillator (LPRC)
110 = Fast RC Oscillator (FRC) divided by 16 (500 kHz)
111 = Fast RC Oscillator (FRC) divided by N

bit 3 **Unimplemented:** Read as ‘0’

bit 2-0 **NOSC<2:0>:** New Oscillator Selection bits
000 = Fast RC Oscillator (FRC)
001 = Fast RC Oscillator (FRC), divided by N, with PLL module
010 = Primary Oscillator (MS, HS, EC)
011 = Primary Oscillator (MS, HS, EC) with PLL module
100 = Secondary Oscillator (SOSC)
101 = Low-Power RC Oscillator (LPRC)
110 = Fast RC Oscillator (FRC) divided by 16 (500 kHz)
111 = Fast RC Oscillator (FRC) divided by N

REGISTER 3-2: OSCCON2: OSCILLATOR CONTROL REGISTER 2

R/W-0	R/W-0	R-0	U-0	R/C-0	R/W-0	R/W-0	U-0
CLKLOCK(2)	IOLOCK(1)	LOCK	—	CF	POSCEN	SOSCGO	—
bit 7							bit 0

Legend:	C = Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CLKLOCK:** Clock Lock Enabled bit(2)
- 1 = Clock and PLL selection are locked and may not be modified
- 0 = Clock and PLL selection are not locked, configurations may be modified

bit 6 **IOLOCK:** I/O Lock Enable bit(1)
- 1 = I/O lock is active (If IOL1WAY (CONFIG5H<0> = 1), the bit cannot be cleared, once it is set, except on a device Reset.)
- 0 = I/O lock is not active

bit 5 **LOCK:** PLL Lock Status bit (read-only)
- 1 = Indicates that PLL module is in lock or PLL start-up timer is satisfied
- 0 = Indicates that PLL module is out of lock, PLL start-up timer is in progress or PLL is disabled

bit 4 **Unimplemented:** Read as '0'

bit 3 **CF:** Clock Fail Detect bit (readable/clearable by application)
- 1 = FSCM has detected a clock failure
- 0 = FSCM has not detected A clock failure

bit 2 **POSCEN:** Primary Oscillator (POSC) Enable bit
- 1 = Enables Primary Oscillator in Sleep mode
- 0 = Disables Primary Oscillator in Sleep mode

bit 1 **SOSCGO:** 32 kHz Secondary (LP) Oscillator Enable bit
- 1 = Enables Secondary Oscillator independent of other SOSC enable requests; provides a way to keep the SOSC running even when not actively used by the system
- 0 = Disables Secondary Oscillator; the SOSC will be enabled if directly requested by the system. Reset on POR or BOR only.

bit 0 **Unimplemented:** Read as '0'

Note 1: The IOLOCK bit cannot be cleared once it has been set, provided that the IOL1WAY (CONFIG5H<0>) = 1.

2: If the user wants to change the clock source, ensure that the FSCM<1:0> bits (CONFIG3L<5:4>) are set appropriately.

3.3.2 OSCCON3 – CLOCK DIVIDER REGISTER (IRCF<2:0> BITS)

The IRCFx bits (OSCCON3<2:0>) select the postscaler option for the FRC Oscillator output, allowing users to choose a lower clock frequency than the nominal 8 MHz.

This option is described in more detail in **Section 3.10.2 "FRC Postscaler Mode (FRCDIV)"** and **Section 3.10.3 "FRC Oscillator with PLL Mode (FRCPLL)"**.

REGISTER 3-3: OSCCON3: OSCILLATOR CONTROL REGISTER 3

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-1
—	—	—	—	—	IRCF2(1)	IRCF1(1)	IRCF0(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **IRCF<2:0>:** Reference Clock Divider bits(1)

000 = FRC divide-by-1
001 = FRC divide-by-2 **(default)**
010 = FRC divide-by-4
011 = FRC divide-by-8
100 = FRC divide-by-16
101 = FRC divide-by-32
110 = FRC divide-by-64
111 = FRC divide-by-256

Note 1: The default FRC divide-by setting on an 8-bit device corresponds to 1 MIPS operation.

REGISTER 3-4: OSCCON4: OSCILLATOR CONTROL REGISTER 4

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
CPDIV1	CPDIV0	PLLEN	—	—	—	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **CPDIV<1:0>:** USB System Clock Select bits (postscaler select from 64 MHz clock branch)

00 = Input clock/1
01 = Input clock/2
10 = Input clock/4
11 = Input clock/8

bit 5 **PLLEN:** PLL Enable bit

1 = PLL is enabled even though it is not requested by the CPU; provides ability to "warm-up" the PLL and keep it running to avoid the PLL start-up time. This setting will force the PLL and associated clock source to stay active in Sleep.
0 = PLL is disabled; PLL will be automatically turned on when SRC1 is selected, or when REFO1 or REFO2 is enabled and using the PLL clock as its source. In either case, the PLL will require a start-up time.

bit 4-0 **Unimplemented:** Read as '0'

3.3.3 OSCILLATOR TUNING REGISTER (OSCTUNE)

The FRC Oscillator Tuning register (Register 3-5) allows the user to fine-tune the FRC Oscillator. Refer to the data sheet of the specific device for further information regarding the FRC Oscillator tuning.

The tuning response of the FRC Oscillator may not be monotonic or linear; the next closest frequency may be offset by a number of steps. It is recommended that users try multiple values of OSCTUNE to find the closest value to the desired frequency.

REGISTER 3-5: OSCTUNE: FRC OSCILLATOR TUNING REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at all Resets	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as ‘0’

bit 5-0 **TUN<5:0>:** FRC Oscillator Tuning bits

011111 = Maximum frequency deviation
011110 =
.
.
.
000001 =
000000 = Center frequency; oscillator is running at factory calibrated frequency
111111 =
.
.
.
100001 =
100000 = Minimum frequency deviation

3.4 Reference Clock Output Control Module

The PIC18F97J94 family has two Reference Clock Output (REFO) modules. Each of the Reference Clock Output modules provides the user with the ability to send out a programmed output clock onto the REFO1or REFO2 pins.

3.4.1 REFERENCE CLOCK SOURCE

The module provides the ability to select one of the following clock sources:

- Primary Crystal Oscillator (POSC)
- Secondary Crystal Oscillator (SOSC)
- 32.768 kHz Internal Oscillator (INTOSC)
- Fast Internal Oscillator (FRC)

It includes a programmable clock divider with ratios ranging from 1:1 to 1:65534.

When the clock source is a crystal or internal oscillator, the RSLP bit can be set to continue REFO operation while the device is in Sleep Mode.

3.4.2 CLOCK SYNCHRONIZATION

The Reference Clock Output is enabled only once (ON = 1). Note that the source of the clock and the divider values should be chosen prior to the bit being set to avoid glitches on the REFO output.

Once the ON bit is set, its value is synchronized to the Reference Clock Output domain to enable the output. This ensures that no glitches will be seen on the output. Similarly, when the ON bit is cleared, the output and the associated output enable signals will be synchronized and disabled on the falling edge of the Reference Clock Output. Note that with large divider values, this will cause the REFO to be enabled for some period after ON is cleared.

3.4.3 OPERATION IN SLEEP MODE

If any clock source, other than the peripheral clock, is used as a base reference (i.e., ROSEL<3:0> ≠ 0001), the user has the option to configure the behavior of the oscillator in Sleep mode. The RSLP Configuration bit determines if the oscillator will continue to run in Sleep. If RSLP = 0, the oscillator will be shut down in Sleep (assuming no other consumers are requesting it). If RSLP = 1, the oscillator will continue to run in Sleep.

The Reference Clock Output is synchronized with the Sleep signal to avoid any glitches on its output.

3.4.3.1 Module Enable Signal

The REFOx module may be enabled or disabled using the REFOxMD register bit, which holds the REFOx module in Reset, or the ON register bit, which does not.

3.4.3.2 Registers and Bits

This module provides the following device registers and/or bits:

- REFOxCON – Reference Clock Output Control Register
- REFOxCON1 – Reference Clock Output Control 1 Register
- REFOxCON2 – Reference Clock Output Control 2 Register
- REFOxCON3 – Reference Clock Output Control 3 Register

In addition, the REFOxCON1 module needs to be enabled by clearing the REFOxMD disable bit (PMD3<1>).

3.4.3.3 Interrupts

This module does not generate any interrupts.

Note:	Throughout this section, references to register and bit names that may be associated with specific Reference Clock Output modules are referred to generically by the use of ‘x’ in place of the specific module number. Thus, “REFOxCON” might refer to the control register for either REFO1 or REFO2.

REGISTER 3-6: REFOxCON: REFERENCE CLOCK OUTPUT CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	R/W-0	U-0	HC/R/W-0	HS/HC/R-0
ON	—	SIDL	OE	RSLP(1)	—	DIVSW_EN	ACTIVE
bit 7							bit 0

Legend:	HC = Hardware Clearable bit	HS = Hardware Settable bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at all Resets	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ON:** Reference Clock Output Enable bit
1 = Reference clock module is enabled
0 = Reference clock module is disabled

bit 6 **Unimplemented:** Read as '0'

bit 5 **SIDL:** Peripheral Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 4 **OE:** Reference Clock Output Enable bit
1 = Reference clock is driven out on REFOx pin
0 = Reference clock is NOT driven out on REFOx pin

bit 3 **RSLP:** Reference Clock Output Run in Sleep bit(1)
1 = Reference Clock Output continues to run in Sleep
0 = Reference Clock Output is disabled in Sleep

bit 2 **Unimplemented:** Read as '0'

bit 1 **DIVSW_EN:** Clock RODIV Switch Enabled Status bit
1 = Clock Divider Switching currently in progress
0 = Clock Divider Switching has completed

bit 0 **ACTIVE:** Reference Clock Output Request Status bit
1 = Reference clock request is active (user should not update the ROSEL and RODIV register fields)
0 = Reference clock request is not active (user may update the ROSEL and RODIV register fields)

Note 1: This bit has no effect when ROSEL<3:0> = 0000/0001, as the system clock and peripheral clock are always disabled in Sleep mode on PIC18 devices.

REGISTER 3-7: REFOxCON1: REFERENCE CLOCK OUTPUT CONTROL REGISTER 1

U-0	U-0	U-0	U-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)
—	—	—	—	ROSEL3	ROSEL2	ROSEL1	ROSEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at all Resets	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Read as '0'
(Reserved for additional ROSEL bits.)

bit 3-0 **ROSEL<3:0>:** Reference Clock Output Source Select bits(1)
Select one of the various clock sources to be used as the reference clock.
0111-1111 = Reserved
0110 = PLL (4/6/8x or 96 MHz)
0101 = SOSC
0100 = LPRC
0011 = FRC
0010 = POSC
0001 = Peripheral clock (reference clock reflects any peripheral clock switching)
0000 = System clock (reference clock reflects any device clock switching)
When PLLDIV<3:0> (CONFIG2H<3:0>) = 1111, ROSEL<3:0> should not be set to '0110'.

Note 1: The ROSEL register field should not be written while the ACTIVE (REFOxCON<0>) bit is '1'; undefined behavior will result.

REGISTER 3-8: REFOxCON2: REFERENCE CLOCK OUTPUT CONTROL REGISTER 2

R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W -0(1)	R/W -0(1)
RODIV7	RODIV6	RODIV5	RODIV4	RODIV3	RODIV2	RODIV1	RODIV0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at all Resets	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **RODIV<7:0>:** Reference Clock Output Divider bits(1)
Reserved for expansion of RODIV<15>.

Note 1: The RODIV register field should not be written while the ACTIVE (REFOxCON<0>) bit is '1'; Undefined behavior will result.

REGISTER 3-9: REFOxCON3: REFERENCE CLOCK OUTPUT CONTROL REGISTER 3

U-0	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)	R/W-0(1)
—	RODIV14	RODIV13	RODIV12	RODIV11	RODIV10	RODIV9	RODIV8
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at all Resets	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-0 **RODIV<14:8>:** Reference Clock Output Divider bits(1)
Used in conjunction with RODIV<7:0> to specify clock divider frequency.
111111111111111 = REFO clock is base clock frequency divided by 65,534 (32,767 * 2)
111111111111110 = REFO clock is base clock frequency divided by 65,532 (32,766 * 2)
•
•
•
000000000000011 = REFO clock is base clock frequency divided by 6 (3 * 2)
000000000000010 = REFO clock is base clock frequency divided by 4 (2 * 2)
000000000000001 = REFO clock is base clock frequency divided by 2 (1 * 2)
000000000000000 = REFO clock is the same frequency as the base clock (no divider)

Note 1: The RODIV register field should not be written while the ACTIVE (REFOxCON<0>) bit is '1'; undefined behavior will result.

3.5 Primary Oscillator (POSC)

The Primary Oscillator is available on the OSC1 and OSC2 pins of the PIC18F family. In general, the Primary Oscillator can be configured for an external clock input or an external crystal. Further details of the Primary Oscillator operating modes are described in subsequent sections. The Primary Oscillator has up to 6 operating modes, summarized in Table 3-2.

TABLE 3-2: PRIMARY OSCILLATOR OPERATING MODES

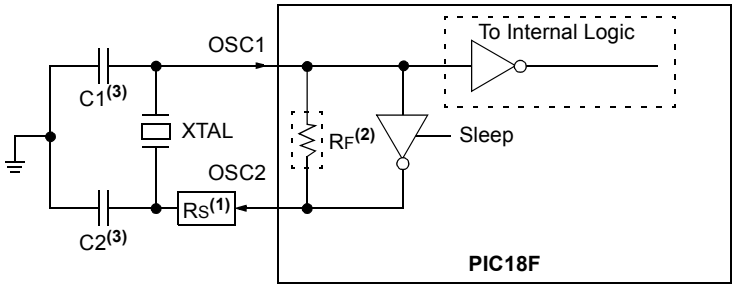
Oscillator Mode	Description	OSC2 Pin Function
EC	External clock input (0-64 MHz)	FOSC/4
ECPLL	External clock input (4-48 MHz), PLL enabled	FOSC/4, **Note 2**
HS	10 MHz-32 MHz crystal	**Note 1**
HSPLL	10 MHz-32 MHz crystal, PLL enabled	**Note 2**
MS	3.5 MHz-10 MHz crystal	**Note 1**
MSPLL	3.5 MHz-8 MHz crystal, PLL enabled	**Note 1**

Note 1: External crystal is connected to OSC1 and OSC2 in these modes.
2: Available only in devices with special PLL blocks (such as the 96 MHz PLL); the basic 4x PLL block generates clock frequencies beyond the device's operating range.

The POSCMDx and FOSCx Configuration bits (CONFIG3L<1:0> and CONFIG2L<2:0>, respectively) select the operating mode of the Primary Oscillator. The POSCMD<1:0> bits select the particular submode to be used (MS, HS or EC), while the FOSC<2:0> bits determine if the oscillator will be used by itself or with the internal PLL. The PIC18F operates from the Primary Oscillator whenever the COSCx bits (OSCCON<6:4>) are set to '010' or '011'.

Refer to the **"Electrical Characteristics"** section in the specific device data sheet for further information regarding frequency range for each crystal mode.

FIGURE 3-3: CRYSTAL OR CERAMIC RESONATOR OPERATION (MS OR HS OSCILLATOR MODE)



Note 1: A series resistor, Rs, may be required for AT strip cut crystals.
2: The internal feedback resistor, RF, is typically in the range of 2 to 10 MΩ.
3: See **Section 3.6.5 "Determining the Best Values for Oscillator Components"**.

3.5.1 SELECTING A PRIMARY OSCILLATOR MODE

The main difference between the MS and HS modes is the gain of the internal inverter of the oscillator circuit, which allows the different frequency ranges. The MS mode is a medium power, medium frequency mode. HS mode provides the highest oscillator frequencies with a crystal. OSC2 provides crystal feedback in both HS and MS Oscillator modes.

The EC and HS modes that use the PLL circuit provide the highest device operating frequencies. The oscillator circuit will consume the most current in these modes because the PLL is enabled to multiply the frequency of the oscillator.

In general, users should select the oscillator option with the lowest possible gain that still meets their specifications. This will result in lower dynamic currents (I_{DD}). The frequency range of each oscillator mode is the recommended frequency cutoff, but the selection of a different gain mode is acceptable as long as a thorough validation is performed (voltage, temperature and component variations, such as resistor, capacitor and internal oscillator circuitry).

The oscillator feedback circuit is disabled in all EC modes. The OSC1 pin is a high-impedance input and can be driven by a CMOS driver.

If the Primary Oscillator is configured for an external clock input, the OSC2 pin is not required to support the oscillator function. For these modes, the OSC2 pin can be used as an additional device I/O pin or a clock output pin. When the OSC2 pin is used as a clock output pin, the output frequency is $F_{OSC}/4$.

3.6 Crystal Oscillators and Ceramic Resonators

In MS and HS modes, a crystal or ceramic resonator is connected to the OSC1 and OSC2 pins to establish oscillation (Figure 3-3). The PIC18F oscillator design requires the use of a parallel cut crystal. Using a series cut crystal may give a frequency out of the crystal manufacturer's specifications.

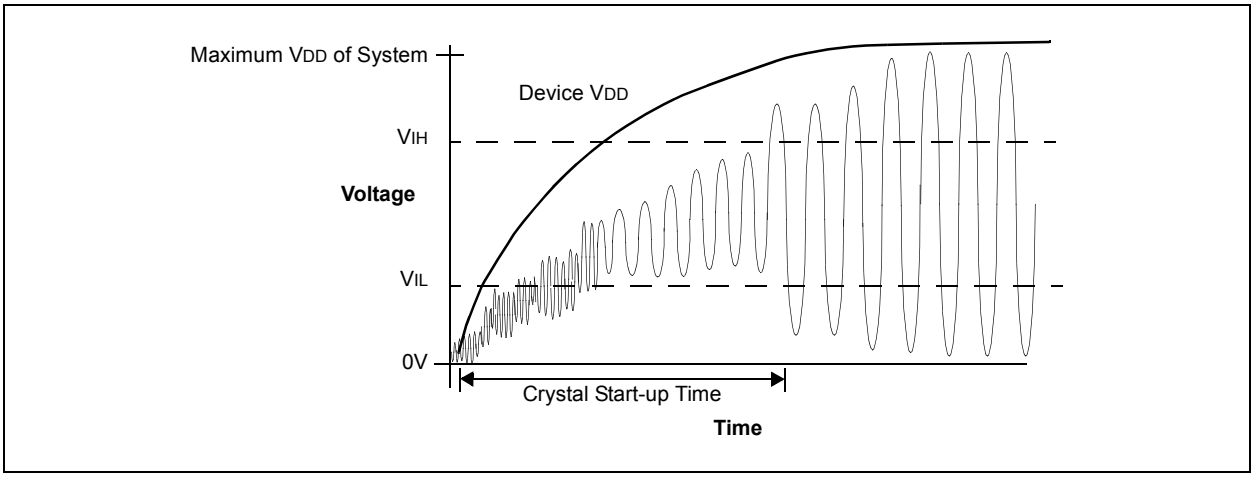
3.6.1 OSCILLATOR/RESONATOR START-UP

As the device voltage increases from V_{SS}, the oscillator will start its oscillations. The time required for the oscillator to start oscillating depends on many factors, including:

- Crystal/resonator frequency
- Capacitor values used
- Series resistor, if used, and its value and type
- Device V_{DD} rise time
- System temperature
- Oscillator mode selection of device (selects the gain of the internal oscillator inverter)
- Crystal quality
- Oscillator circuit layout
- System noise

The course of a typical crystal or resonator start-up is shown in Figure 3-4. Notice that the time to achieve stable oscillation is not instantaneous.

FIGURE 3-4: EXAMPLE OSCILLATOR/RESONATOR START-UP CHARACTERISTICS



3.6.2 PRIMARY OSCILLATOR START-UP FROM SLEEP MODE

The most difficult time for the oscillator to start-up is when waking up from Sleep mode. This is because the load capacitors have both partially charged to some quiescent value and phase differential at wake-up is minimal. Thus, more time is required to achieve stable oscillation. Also remember that low voltage, high temperatures and the lower frequency clock modes also impose limitations on loop gain, which in turn, affects start-up.

Each of the following factors increases the start-up time:

- Low-frequency design (with a Low Gain Clock mode)
- Quiet environment (such as a battery-operated device)
- Operating in a shielded box (away from the noisy RF area)
- Low voltage
- High temperature
- Wake-up from Sleep mode

Circuit noise, on the other hand, may actually help to "kick start" the oscillator and help to lower the oscillator start-up time.

3.6.3 OSCILLATOR START-UP TIMER

In order to ensure that a crystal oscillator (or ceramic resonator) has started and stabilized, an Oscillator Start-up Timer (OST) is provided. The OST is a simple, 10-bit counter that counts 1024 TOSC cycles before releasing the oscillator clock to the rest of the system. This time-out period is designated as TOST. The amplitude of the oscillator signal must reach the VIL and VIH thresholds for the oscillator pins before the OST can begin to count cycles.

The TOST interval is required every time the oscillator has to restart (i.e., on POR, BOR and wake-up from Sleep mode). The Oscillator Start-up Timer is applied to the MS and HS modes for the Primary Oscillator, as well as the Secondary Oscillator, SOSC (see **Section 3.9 "Secondary Oscillator (SOSC)"**).

3.6.4 TUNING THE OSCILLATOR CIRCUIT

Since Microchip devices have wide operating ranges (frequency, voltage and temperature, depending on the part and version ordered), and external components (crystals, capacitors, etc.) of varying quality and manufacture, validation of operation needs to be performed to ensure that the component selection will comply with the requirements of the application. There are many factors that go into the selection and arrangement of these external components. Depending on the application, these may include any of the following:

- Amplifier gain
- Desired frequency
- Resonant frequency(s) of the crystal
- Temperature of operation
- Supply voltage range
- Start-up time
- Stability
- Crystal life
- Power consumption
- Simplification of the circuit
- Use of standard components
- Component count

3.6.5 DETERMINING THE BEST VALUES FOR OSCILLATOR COMPONENTS

The best method for selecting components is to apply a little knowledge, and a lot of trial measurement and testing. Crystals are usually selected by their parallel resonant frequency only; however, other parameters may be important to your design, such as temperature or frequency tolerance. Microchip Application Note *AN588, "PICmicro® Microcontroller Oscillator Design Guide"* (DS00000588) is an excellent reference to learn more about crystal operation and ordering information.

The PIC18F internal oscillator circuit is a parallel oscillator circuit which requires that a parallel resonant crystal be selected. The load capacitance is usually specified in the 22 pF to 33 pF range. The crystal will oscillate closest to the desired frequency, with a load capacitance in this range. It may be necessary to alter these values, as described later, in order to achieve other benefits.

The clock mode is primarily chosen based on the desired frequency of the crystal oscillator. The main difference between the MS and HS Oscillator modes is the gain of the internal inverter of the oscillator circuit, which allows the different frequency ranges. In general, use the oscillator option with the lowest possible gain that still meets specifications. This will result in lower dynamic currents (IDD). The frequency range of each oscillator mode is the recommended frequency cutoff, but the selection of a different gain mode is acceptable as long as a thorough validation is performed (voltage, temperature and component variations, such as resistor, capacitor and internal oscillator circuitry). C1 and C2 should also be initially selected based on the load capacitance, as suggested by the crystal manufacturer, and the tables supplied in the device data sheet. The values given in the device data sheet can only be used as a starting point, since the crystal manufacturer, supply voltage, and other factors already mentioned, may cause your circuit to differ from the one used in the factory characterization process.

Ideally, the capacitance is chosen so that it will oscillate at the highest temperature and the lowest VDD that the circuit will be expected to perform under. High tempera-

ture and low VDD both have a limiting effect on the loop gain, such that if the circuit functions at these extremes, the designer can be more assured of proper operation at other temperatures and supply voltage combinations. The output sine wave should not be clipped in the highest gain environment (highest VDD and lowest temperature) and the sine output amplitude should be large enough in the lowest gain environment (lowest VDD and highest temperature) to cover the logic input requirements of the clock, as listed in the device data sheet. OSC1 may have specified VIL and VIH levels (refer to the specific product data sheet for more information).

A method for improving start-up is to use a value of C2 greater than C1. This causes a greater phase shift across the crystal at power-up, which speeds oscillator start-up. Besides loading the crystal for proper frequency response, these capacitors can have the effect of lowering loop gain if their value is increased. C2 can be selected to affect the overall gain of the circuit. A higher C2 can lower the gain if the crystal is being overdriven (also see discussion on Rs). Capacitance values that are too high can store and dump too much current through the crystal, so C1 and C2 should not become excessively large. Unfortunately, measuring the wattage through a crystal is difficult, but if you do not stray too far from the suggested values, you should not have to be concerned with this.

A series resistor, Rs, is added to the circuit if after all other external components are selected to satisfaction, and the crystal is still being overdriven. This can be determined by looking at the OSC2 pin, which is the driven pin, with an oscilloscope. Connecting the probe to the OSC1 pin will load the pin too much and negatively affect performance. Remember that a scope probe adds its own capacitance to the circuit, so this may have to be accounted for in your design (i.e., if the circuit worked best with a C2 of 22 pF and the scope probe was 10 pF, a 33 pF capacitor may actually be called for). The output signal should not be clipping or flattened. Overdriving the crystal can also lead to the circuit jumping to a higher harmonic level, or even, crystal damage.

The OSC2 signal should be a clean sine wave that easily spans the input minimum and maximum of the clock input pin. An easy way to set this is to again test the circuit at the minimum temperature and maximum VDD that the design will be expected to perform in; then, look at the output. This should be the maximum amplitude of the clock output. If there is clipping, or the sine wave is distorted near VDD and VSS, increasing load capacitors may cause too much current to flow through the crystal, or push the value too far from the manufacturer's load specification. To adjust the crystal current, add a trimmer potentiometer between the crystal inverter output pin and C2, and adjust it until the sine wave is clean. The crystal will experience the highest drive currents at the low temperature and high VDD extremes.

The trimmer potentiometer should be adjusted at these limits to prevent overdriving. A series resistor, Rs, of the closest standard value can now be inserted in place of the trimmer. If Rs is too high, perhaps more than 20 kΩ, the input will be too isolated from the output, making the clock more susceptible to noise. If you find a value this high is needed to prevent overdriving the crystal, try increasing C2 to compensate or changing the oscillator operating mode. Try to get a combination where Rs is around 10 kΩ or less, and load capacitance is not too far from the manufacturer's specification.

3.7 External Clock Input

In EC mode, the OSC1 pin is in a high-impedance state and can be driven by CMOS drivers. The OSC2 pin can be configured as either an I/O or the clock output (FOSC 4) by selecting the CLKOEN bit (CONFIG2L<5>). With CLKOEN set (Figure 3-5), the clock output is available for testing or synchronization purposes. With CLKOEN clear (Figure 3-6), the OSC2 pin becomes a general purpose I/O pin. The feedback device between OSC1 and OSC2 is turned off to save current.

FIGURE 3-5: EXTERNAL CLOCK INPUT OPERATION (CLKOEN = 1)

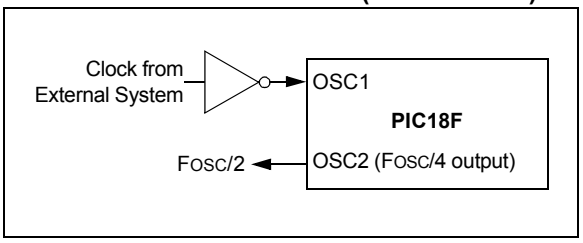
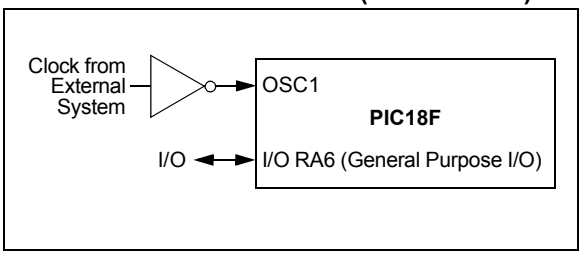


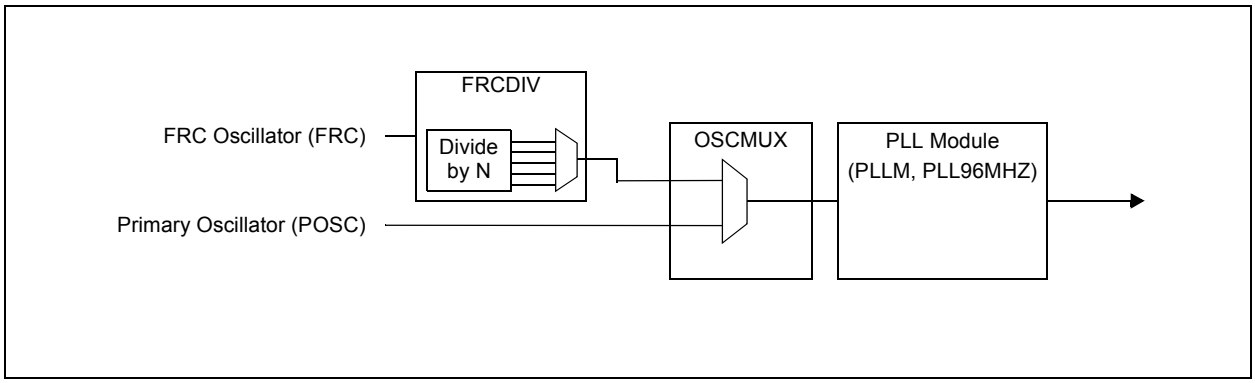
FIGURE 3-6: EXTERNAL CLOCK INPUT OPERATION (CLKOEN = 0)



3.8 Phase Lock Loop (PLL) Branch

The PLL module contains two separate PLL submodules: PLLM and PLL96MHZ. The PLLM submodule is configurable as a 4x, 6x or 8x PLL. The PLL96MHZ submodule runs at 96 MHz and requires an input clock between 4 MHz and 48 MHz (a multiple of 4 MHz). These are selected through the PLLDIV<3:0> bits.

FIGURE 3-7: BASIC OSCILLATOR BLOCK DIAGRAM



3.8.1 OSCILLATOR MODES AND USB OPERATION

Because of the timing requirements imposed by USB, an internal clock of 48 MHz is required at all times while the USB module is enabled and not in a suspended operating state. A method is provided to internally generate both the USB and system clocks from a single oscillator source. PIC18F97J94 family devices use the same clock structure as most other PIC18 devices, but include a two-branch PLL system to generate the two clock signals.

The USB PLL block is shown in Figure 3-8. In this system, the input from the Primary Oscillator is divided down by a PLL prescaler to generate a 4 MHz output. This is used to drive an on-chip 96 MHz PLL frequency multiplier to drive the two clock branches. One branch uses a fixed, divide-by-2 frequency divider to generate the 48 MHz USB clock. The other branch uses a fixed, divide-by-1.5 frequency divider and configurable PLL prescaler/divider to generate a range of system clock frequencies. The CPDIVx bits select the system clock speed; available clock options are listed in Table 3-3.

The USB PLL prescaler does not automatically sense the incoming oscillator frequency. The user must manually configure the PLL divider to generate the required 4 MHz output, using the PLLDIV<3:0> Configuration bits. This limits the choices for Primary Oscillator frequency to a total of 8 possibilities, shown in Table 3-4.

FIGURE 3-8: 96 MHz PLL BLOCK

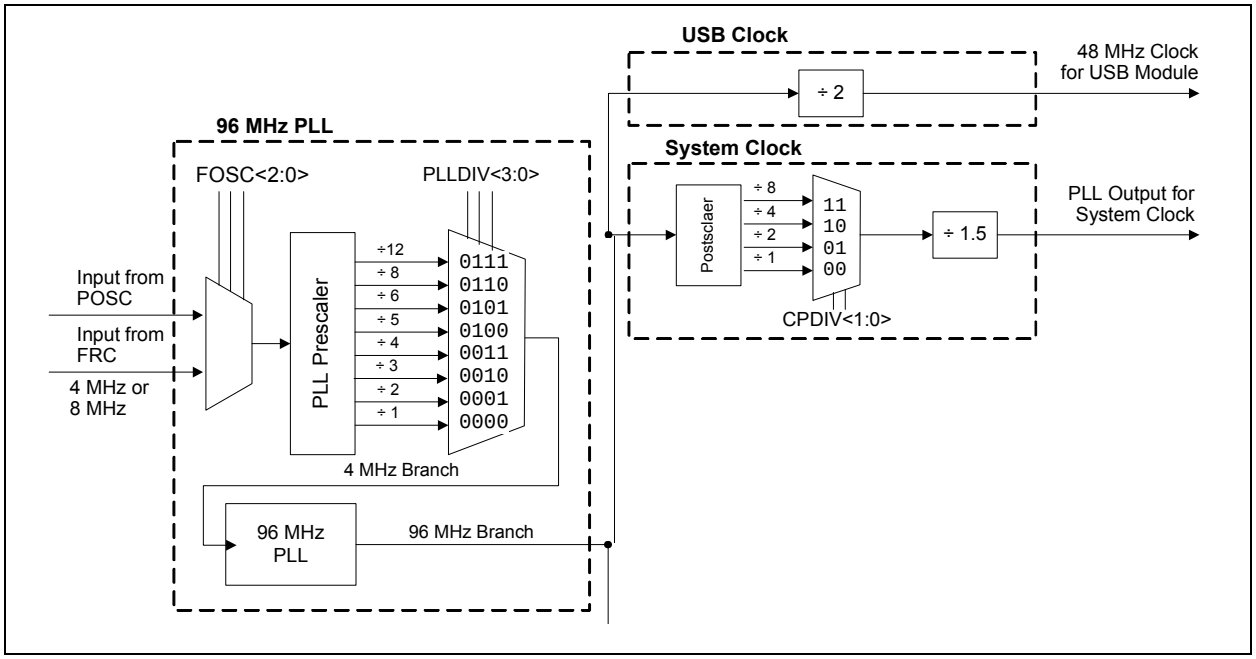


TABLE 3-3: SYSTEM CLOCK OPTIONS DURING USB OPERATION

MCU Clock Division (CPDIV<1:0>)	System Clock Frequency (Instruction Rate in MIPS®)
None (00)	64 MHz (16)
÷2 (01)	32 MHz (8)
÷4 (10)	16 MHz (4)
÷8 (11)	8 MHz (2)(1)

Note 1: These options are not compatible with USB operation. They may be used whenever the PLL branch is selected and the USB module is disabled.

TABLE 3-4: VALID PRIMARY OSCILLATOR CONFIGURATIONS FOR USB OPERATIONS

Input Oscillator Frequency	Clock Mode	PLL Division (PLLDIV<2:0>)
48 MHz	ECPLL	÷12 (111)
32 MHz	ECPLL	÷ 8 (110)
24 MHz	HSPLL, ECPLL	÷6 (101)
20 MHz	HSPLL, ECPLL	÷5 (100)
16 MHz	HSPLL, ECPLL	÷4 (011)
12 MHz	HSPLL, ECPLL	÷3 (010)
8 MHz	ECPLL, MSPLL, FRCPLL(1)	÷2 (001)
4 MHz	ECPLL, MSPLL, FRCPLL(1)	÷1 (000)

Note 1: FRCPLL with ±0.25% accuracy can be used for USB operation.

Note:	Because of USB clocking accuracy requirements (±0.25%), not all PIC18F devices support the use of the FRCPLL system clock configuration for USB operation. Refer to the specific device data sheet for details on the FRC Oscillator module.

3.8.2 CONSIDERATIONS FOR USING THE PLL BLOCK

All PLL blocks use the LOCK bit (OSCCON2<5>) as a read-only Status bit to indicate the lock status of the PLL. It is automatically set after the typical time delay for the PLL to achieve lock, designated as TLOCK. It is cleared at a POR and on clock switches when the PLL is selected as a clock source. It remains clear when any clock source not using the PLL is selected.

If the PLL does not stabilize properly during start-up, the LOCK bit may not reflect the actual status of the PLL lock, nor does it detect when the PLL loses lock during normal operation. Refer to the **"Electrical Characteristics"** section in the specific device data sheet for further information on the PLL lock interval.

Using any PLL block with the FRC Oscillator provides a stable system clock for microcontroller operations. USB operation is only possible with FRC Oscillators that are implemented with ±1/4% frequency accuracy. Serial communications using USART are only possible when FRC Oscillators are implemented with ±2% frequency accuracy. The PIC18F97J94 family is able to meet the required oscillator accuracy for both USB and USART providing stable communication by use of its active clock tuning feature. Refer to **Section 3.13.3 "Active Clock Tuning (ACT) Module"** for more information.

If an application is being migrated between PIC18F platforms with different PLL blocks, the differences in PLL and clock options may require the reconfiguration of peripherals that use the system clock. This is particularly true with serial communication peripherals, such as the USARTs.

3.9 Secondary Oscillator (SOSC)

In most PIC18F devices, the low-power Secondary Oscillator (SOSC) is implemented to run with a 32.768 kHz crystal. The oscillator is located on the SOSCO and SOSCI device pins, and serves as a secondary crystal clock source for low-power operation. It is used to drive Timer1, Real-Time Clock and Calendar (RTCC) and other modules requiring a clock signal while in low-power operation.

3.9.1 ENABLING THE SECONDARY OSCILLATOR

The operation of the SOSC is selected by the FOSCx Configuration bits or by selection of the NOSCx bits (OSCCON<2:0>). The SOSC can also be enabled by setting the SOSCEN bit in Timer1, Timer3 or Timer5. The SOSC has a long start-up time; therefore, to avoid delays for peripheral start-up, the SOSC can be manually started using one of the SOSCEN bits.

3.9.2 SECONDARY OSCILLATOR OPERATION

3.9.2.1 Continuous Operation

The SOSC is always running when any of the SOSCEN bits are set. Leaving the oscillator running at all times allows a fast switch to the 32 kHz system clock for lower power operation. Returning to the faster main oscillator still requires an oscillator start-up time if it is a crystal-type source. This start-up time can be avoided on PLL clock sources by setting the PLLEN bit (OSCCON4<5>) in advance of switching the clock source.

In addition, the oscillator will need to remain running at all times for Real-Time Clock (RTC) application using Timer1 or the RTCC module. Refer to **Section 14. "Timers"** and **Section 29. "Real-Time Clock and Calendar (RTCC)"** in the *"PIC18F Family Reference Manual"* for further details.

3.9.2.2 Intermittent Operation

When all SOSCEN bits are cleared, the oscillator will only operate when it is selected as the current device clock source (COSC<2:0> = 100). It will be disabled automatically if it is the current device clock source and the device enters Sleep mode.

3.9.3 OPERATING MODES

3.9.3.1 Digital Mode

The SOSCO pin can also be configured to operate as a digital clock input. The SOSCO pin is configured as a digital input by setting SOSCSEL (CONFIG2L<3>) = 10.

When running in this mode, the SOSCO/SCLKI pin will operate as a digital input to the oscillator section, while the SOSCI pin will function as a port pin. The crystal driving circuit is disabled. The Oscillator Configuration Fuse bits (FOSC<2:0>) and New Oscillator Selection bits (NOSC<2:0>) have no effect.

3.9.4 SOSC CRYSTAL SELECTION

A typical 50K ESR and 12.5 pF CL (capacitive loading) rated crystal is recommended for reliable operation of the SOSC. The duty cycle of the SOSC output can be measured on the REFO pin, and is recommended to be within +/-15% from a 50% duty cycle.

3.10 Internal Fast RC Oscillator (FRC)

The FRC Oscillator is a fast (8 MHz nominal), internal RC Oscillator. This oscillator is intended to be a precise internal RC Oscillator accurate enough to provide the clock frequency necessary to maintain baud rate tolerance for serial data transmissions, without the use of an external crystal or ceramic resonator. The PIC18F device operates from the FRC Oscillator whenever the COSCx bits are '111', '110', '001' or '000'.

3.10.1 ENABLING THE FRC OSCILLATOR

Since it serves as the system clock during device initialization, the FRC Oscillator is always enabled at a POR. After the device is configured and PWRT expires, FRC remains active only if it is selected as the device clock source.

3.10.2 FRC POSTSCALER MODE (FRCDIV)

Users are not limited to the nominal 8 MHz FRC output if they wish to use the Fast Internal Oscillator as a clock source. An additional FRC mode, FRCDIV, implements a selectable postscaler that allows the choice of a lower clock frequency, from 7 different options, plus the direct 8 MHz output. The postscaler is configured using the IRCF<2:0> bits (OSCCON3<2:0>). Assuming a nominal 8 MHz output, available lower frequency options range from 4 MHz (divide-by-2) to 31 kHz (divide-by-256). The range of frequencies allows users the ability to save power at any time in an application by simply changing the IRCFx bits.

The FRCDIV mode is selected whenever the COSCx bits are '111'.

3.10.3 FRC OSCILLATOR WITH PLL MODE (FRCPLL)

The FRCPLL mode is selected whenever the COSCx bits are '001'. In addition, this mode only functions when the direct or divide-by-2 FRC postscaler options are selected (IRCF<2:0> = 000 or 001).

When using the 4x or 8x PLL option, the output of the FRC postscaler may also be combined with the PLL to produce a nominal system clock of 16 MHz, 32 MHz or 64 MHz. Although somewhat less precise in frequency than using the Primary Oscillator with a crystal or resonator, it allows high-speed operation of the device without the use of external oscillator components.

For devices with the basic 4x PLL block, the output of the FRC postscaler block may also be combined with the PLL to produce a nominal system clock of either 16 MHz or 32 MHz. Although somewhat less precise in frequency than using the Primary Oscillator with a crystal or resonator, it still allows high-speed operation of the device without the use of external oscillator components.

When using the 96 MHz PLL block, the output of the FRC postscaler block may also be combined with the PLL to produce a nominal system clock of either 4 MHz, 8 MHz, 16 MHz or 32 MHz. It also produces a 48 MHz USB clock; however, this USB clock must be generated with the FRC Oscillator meeting the frequency accuracy requirement of USB for proper operation. Refer to the specific device data sheet for details on the FRC Oscillator electrical characteristics.

In cases where the frequency accuracy is not met for USB operation, the FRCPLL mode should not be used when USB is active.

Note:	Using FRC postscaler values, other than '000'or '001', will cause the clock input to the PLL to be below the operating frequency input range and may cause undesirable operation.

3.11 Internal Low-Power RC Oscillator (LPRC)

The LPRC Oscillator is separate from the FRC and oscillates at a nominal frequency of 31 kHz. LPRC is the clock source for the Power-up Timer (PWRT), Watchdog Timer (WDT) and FSCM circuits. It may also be used to provide a low-frequency clock source option for the device, in those applications where power consumption is critical and timing accuracy is not required.

3.11.1 ENABLING THE LPRC OSCILLATOR

Since it serves the Power-up Timer (PWRT) clock source, the LPRC Oscillator is enabled at POR events whenever the on-board voltage regulator is disabled. After the PWRT expires, the LPRC Oscillator will remain on if any one of the following is true:

- The FSCM is enabled.
- The WDT is enabled.
- The LPRC Oscillator is selected as the system clock (COSC<2:0> = 101).

If none of the above is true, the LPRC will shut off after the PWRT expires.

3.12 Fail-Safe Clock Monitor (FSCM)

The Fail-Safe Clock Monitor (FSCM) allows the device to continue to operate, even in the event of an oscillator failure. The FSCM function is enabled by programming the FSCMx (Clock Switch and Monitor) bits in CONFIG3L<5:4>. FSCM is only enabled when the FSCM<1:0> bits (CONFIG3L<5:4>) = 00. When FSCM is enabled, the internal LPRC Oscillator will run at all times (except during Sleep mode).

In the event of an oscillator failure, the FSCM will generate a clock failure trap and will switch the system clock to the FRC Oscillator. The user will then have the option to either attempt to restart the oscillator or execute a controlled shutdown. FSCM will monitor the system clock source regardless of its source or oscillator mode. This includes the Primary Oscillator for all oscillator modes and the Secondary Oscillator, SOSC, when configured as the system clock.

The FSCM module takes the following actions when switching to the FRC Oscillator:

1. The COSCx bits are loaded with '000'.
2. The CF Status bit is set to indicate the clock failure.

Note:	For more information about the oscillator failure trap, refer to **Section 10.0 "Interrupts"**.

3.12.1 FSCM DELAY

On a POR, BOR or wake from Sleep mode event, a nominal delay (T_{FSCM}) may be inserted before the FSCM begins to monitor the system clock source. The purpose of the FSCM delay is to provide time for the oscillator and/or PLL to stabilize when the PWRT is not utilized. The FSCM delay will be generated after the internal System Reset signal, $\overline{SYSRST}$, has been released. Refer to **Section 28.4 "Fail-Safe Clock Monitor"** for FSCM delay timing information.

The T_{FSCM} interval is applied whenever the FSCM is enabled and the EC, HS or SOSC Oscillator modes are selected as the system clock.

Note:	Refer to the "**Electrical Characteristics**" section of the specific device data sheet for T_{FSCM} specification values.

3.12.2 FSCM AND SLOW OSCILLATOR START-UP

If the chosen device oscillator has a slow start-up time coming out of POR, BOR or Sleep mode, it is possible that the FSCM delay will expire before the oscillator has started. In this case, the FSCM will initiate a clock failure trap. As this happens, the COSCx bits are loaded with the FRC Oscillator selection. This will effectively shut off the original oscillator that was trying to start. The user can detect this situation and initiate a clock switch back to the desired oscillator in the Trap Service Routine (TSR).

3.12.3 FSCM AND WDT

The FSCM and the WDT both use the LPRC Oscillator as their time base. In the event of a clock failure, the WDT is unaffected and continues to run on the LPRC.

3.13 Clock Switching Operation

With few limitations, applications are free to switch between any of the four clock sources (Primary, SOSC, FRC and LPRC) under software control and at any time. To limit the possible side effects that could result from this flexibility, PIC18F devices have a safeguard lock built into the switch process.

Note:	Primary Oscillator mode has three different submodes (MS, HS and EC), which are determined by the POSCMDx Configuration bits. While an application can switch to and from Primary Oscillator mode, in software, it cannot switch between the different primary submodes without reprogramming the device.

3.13.1 ENABLING CLOCK SWITCHING

To enable clock switching, the FCKSM1 Configuration bit must be programmed to '0'. If the FCKSM1 Configuration bit is unprogrammed ('1'), the clock switching function and Fail-Safe Clock Monitor function are disabled; this is the default setting.

The NOSCx control bits (OSCCON<2:0>) do not control the clock selection when clock switching is disabled. However, the COSCx bits (OSCCON<6:4>) will reflect the clock source selected by the FOSC Configuration bits.

3.13.2 OSCILLATOR SWITCHING SEQUENCE

At a minimum, performing a clock switch requires this basic sequence:

1. If desired, read the COSCx bits (OSCCON<6:4>) to determine the current oscillator source.
2. Clear the CLKLOCK bit (OSCCON2<7>) to enable writes to the NOSCx bits (OSCCON<2:0>).
3. Write the appropriate value to the NOSCx control bits (OSCCON<2:0>) for the new oscillator source.

Once the basic sequence is completed, the system clock hardware responds automatically as follows:

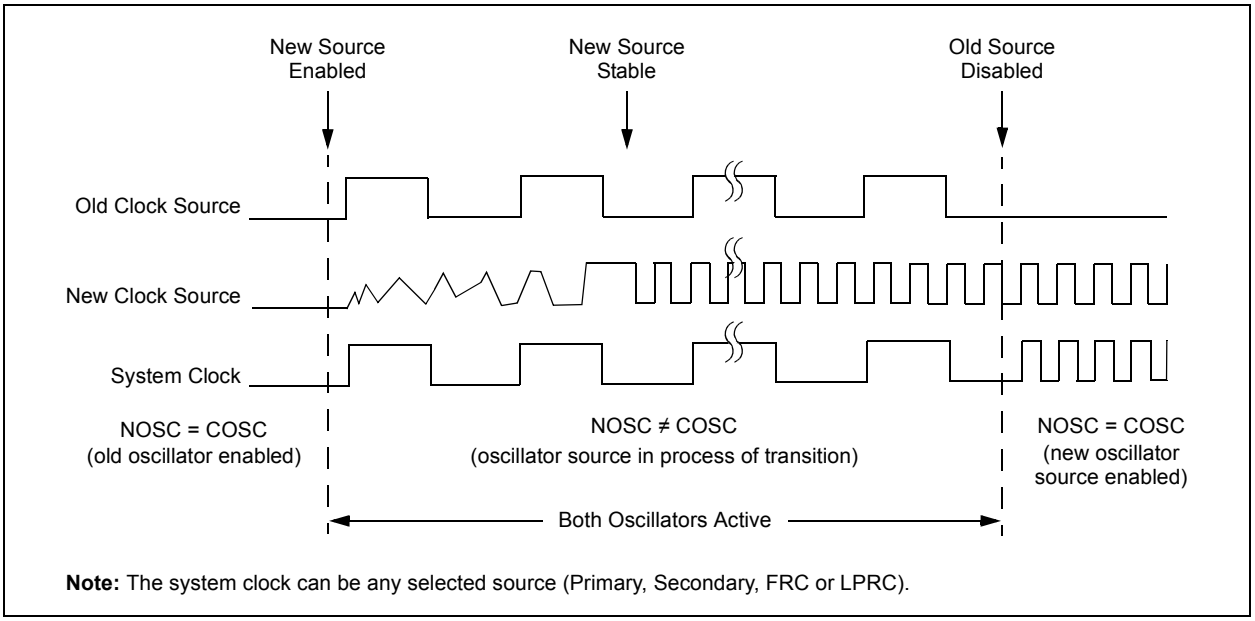
1. The clock switching hardware compares the COSC Status bits with the new value of the NOSC control bits. If they are the same, then the clock switch is a redundant operation. If they are different, then a valid clock switch has been initiated.
2. The new oscillator is turned on by the hardware if it is not currently running. If a crystal oscillator must be turned on, the hardware will wait until the OST expires. If the new source is using the PLL, then the hardware waits until a PLL lock is detected (LOCK = 1).
3. The hardware waits for the new clock source to stabilize and then performs the clock switch.
4. The NOSCx bit values are transferred to the COSCx Status bits.
5. The old clock source is turned off at this time, with the exception of LPRC (if WDT or FSCM is enabled) or SOSC (if it is enabled by one of the timer sources).

The timing of the transition between clock sources is shown in Figure 3-9.

Note 1: The processor will continue to execute code throughout the clock switching sequence. Timing-sensitive code should not be executed during this time.

2: Direct clock switches between any Primary Oscillator mode with PLL and FRCPLL mode are not permitted. This applies to clock switches in either direction. In these instances, the application must switch to FRC mode as a transition clock source between the two PLL modes.

FIGURE 3-9: CLOCK TRANSITION TIMING DIAGRAM



Note: The system clock can be any selected source (Primary, Secondary, FRC or LPRC).

A recommended code sequence for a clock switch includes the following:

1. Disable interrupts during the OSCCON register unlock and write sequence.
2. Clear the CLKLOCK bit (OSCCON2<7>) to enable writes to the NOSCx bits (OSCCON<2:0>).
3. Write new oscillator source to NOSCx control bits.
4. Continue to execute code that is not clock-sensitive (optional).
5. Invoke an appropriate amount of software delay (cycle counting) to allow the selected oscillator and/or PLL to start and stabilize.
6. Check to see if COSC contains the new oscillator values that were requested in Step 3.

3.13.2.1 Clock Switching Considerations

When incorporating clock switching into an application, users should keep certain things in mind when designing their code.

- If the new clock source is a crystal oscillator, the clock switch time will be dominated by the oscillator start-up time.
- If the new clock source does not start, or is not present, the clock switching hardware will wait indefinitely for the new clock source. The user can detect this situation because the COSCx bits will not change to reflect the new desired oscillator settings.
- Switching to a low-frequency clock source, such as the Secondary Oscillator, will result in very slow device operation.

Note:	The application should not attempt to switch to a clock with a frequency lower than 100 kHz when the FSCM is enabled. Clock switching in these instances may generate a false oscillator fail trap and result in a switch to the Internal Fast RC Oscillator.

3.13.3 ACTIVE CLOCK TUNING (ACT) MODULE

The Active Clock Tuning (ACT) module continuously adjusts the 8 MHz internal oscillator, using an available external reference, to achieve ± 0.20% accuracy. This eliminates the need for a high-speed, high-accuracy external crystal when the system has an available lower speed, lower power, high-accuracy clock source available. Systems implementing a Real-Time Clock Calendar (RTCC) or a full-speed USB application can take full advantage of the ACT module.

3.13.3.1 Active Clock Tuning Operation

The ACT module defaults to the disabled state after any Reset. When the ACT module is disabled, the user can write to the TUN<6:0> bits in the OSCTUNE register to manually adjust the 8 MHz internal oscillator.

The module is enabled by setting the ACTEN bit of the ACTCON register. When enabled, the ACT module takes control of the OSCTUNE register. The ACT module uses the selected ACT reference clock to tune the 8 MHz internal oscillator to an accuracy of 8 MHz ± 0.2%. The tuning automatically adjusts the OSCTUNE register every reference clock cycle.

3.13.3.2 Active Clock Tuning Source Selection

The ACT reference clock is selected with the ACTSRC bit of the ACTCON register. The reference clock sources are provided by the:

- USB module in full-speed operation (ACT_clk)
- Secondary clock at 32.768 kHz (SOSC_clk)

3.13.3.3 ACT Lock Status

The ACTLOCK bit will be set to ‘1’, when the 8 MHz internal oscillator is successfully tuned.

The bit will be cleared by the following conditions:

- Out of Lock condition
- Device Reset
- Module is disabled

3.13.3.4 ACT Out-of-Range Status

If the ACT module requires an OSCTUNE value outside the range to achieve ± 0.20% accuracy, then the ACT Out-of-Range (ACTORS) Status bit will be set to ‘1’.

An out-of-range status can occur:

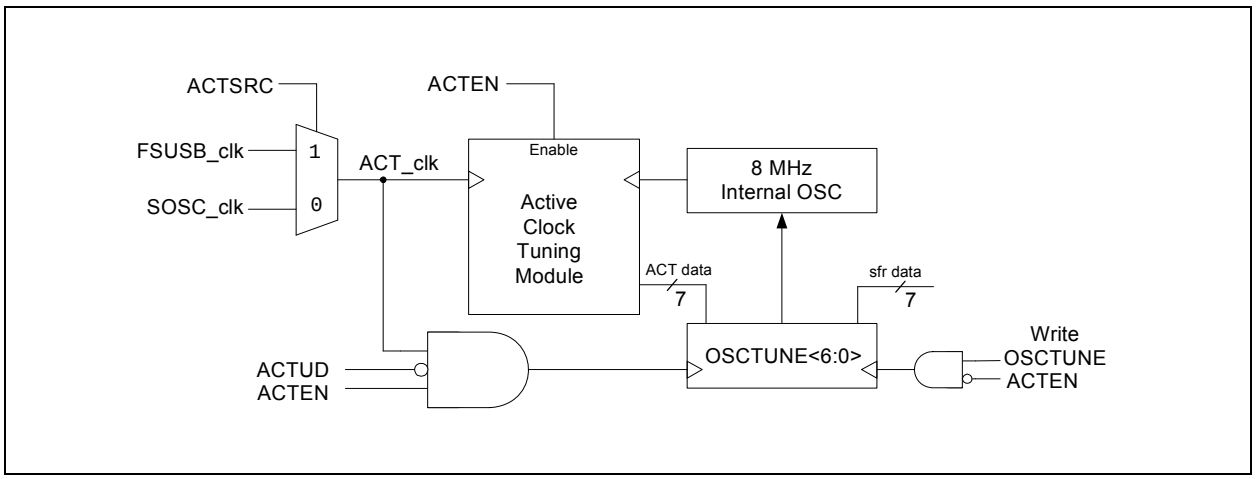
- When the 8 MHZ internal oscillator is tuned to its lowest frequency and the next ACT_clk event requests a lower frequency.
- When the 8 MHZ internal oscillator is tuned to its highest frequency and the next ACT_clk event requests a higher frequency.

When the ACT out-of-range event occurs, the 8 MHz internal oscillator will continue to use the last written OSCTUNE value. When the OSCTUNE value moves back within the tunable range and ACTLOCK is established, the ACTORS bit is cleared to ‘0’.

Note 1: When the ACT module is enabled, the OSCTUNE register is only updated by the module. Writes to the OSCTUNE register by the user are inhibited, but reading the register is permitted.

2: After disabling the ACT module, the user should wait three instructions before writing to the OSCTUNE register.

FIGURE 3-10: ACTIVE CLOCK TUNING BLOCK DIAGRAM



REGISTER 3-10: ACTCON: ACTIVE CLOCK TUNING (ACT) CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	R-0	R/W-0	R-0	R/W-0
ACTEN	—	ACTSIDL	ACTSRC(1)	ACTLOCK	ACTLOCK-POL	ACTORS	ACTOR-SPOL
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ACTEN:** Active Clock Tuning Selection bit
1 = ACT module is enabled, updates to OSCTUNE are exclusive to the ACT module
0 = ACT module is disabled

bit 6 **Unimplemented:** Reads as '0'

bit 5 **ACTSIDL:** Active Clock Tuning Stop in Idle bit
1 = Active clock tuning stops during Idle mode
0 = Active clock tuning continues during Idle mode

bit 4 **ACTSRC:** Active Clock Tuning Source Selection bit
1 = The FRC oscillator is tuned to approximately match the USB host clock tolerance
0 = The FRC oscillator is tuned to approximately match the 32.768 kHz SOSC tolerance

bit 3 **ACTLOCK:** Active Clock Tuning Lock Status bit
1 = Locked; internal oscillator is within ± 0.20%
0 = Not locked; internal oscillator tuning has not stabilized within ± 0.20%

bit 2 **ACTLOCKPOL:** Active Clock Tuning Lock Interrupt Polarity bit
1 = ACT lock interrupt is generated when ACTLOCK is '0'
0 = ACT lock interrupt is generated when ACTLOCK is '1'

bit 1 **ACTORS:** Active Clock Tuning Out-of-Range Status bit
1 = Out-of-range; oscillator frequency is outside of the OSCTUNE range
0 = In-range; oscillator frequency is within the OSCTUNE range

bit 0 **ACTORSPOL:** Active Clock Tuning Out of Range Interrupt Polarity bit
1 = ACT out of range interrupt is generated when ACTORS is '0'
0 = ACT out of range interrupt is generated when ACTORS is '1'

Note 1: The ACTSRC bit should only be changed when ACTEN = 0.

3.13.4 ABANDONING A CLOCK SWITCH

In the event the clock switch does not complete, it can be abandoned by setting the NOSCx bits to their previous values. This abandons the clock switch process, stops and resets the OST (if applicable), and stops the PLL (if applicable).

A clock switch procedure can be aborted at any time. A clock switch that is already in progress can also be aborted by performing a second clock switch.

3.13.5 ENTERING SLEEP MODE DURING A CLOCK SWITCH

If the device enters Sleep mode during a clock switch operation, the operation is abandoned. The processor keeps the old clock selection and the NOSCx bits return to their previous values (the same as COSC). The SLEEP instruction is then executed normally.

3.14 Two-Speed Start-Up

Two-Speed Start-up is an automatic clock switching feature that is independent of the manually controlled clock switching previously described. It helps to minimize the latency period, from oscillator start-up to code execution, by allowing the microcontroller to use the FRC Oscillator as a clock source until the primary clock source is available. This feature is controlled by the IESO Configuration bit (CONFIG2L<7>) and operates independently of the state of the FSCM Configuration bits.

Two-Speed Start-up is particularly useful when an external oscillator is selected by the FOSCx Configuration bits, and a crystal-based oscillator (either a Primary or Secondary Oscillator) may have a longer start-up time. As an internal RC Oscillator, the FRC clock source is available almost immediately following a POR or device wake-up.

With Two-Speed Start-up, the device starts executing code on POR in its default oscillator configuration (FRC). It continues to operate in this mode until the external oscillator source, specified by the FOSCx Configuration bits, becomes stable; at which time, it automatically switches to that source.

Two-Speed Start-up is used on wake-up from the power-saving Sleep mode. The device uses the FRC clock source until the selected primary clock is ready. It is not used in Idle mode, as the device will be clocked by the currently selected clock source until the primary clock source becomes available.

3.14.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the FRC Oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-saving modes, including SLEEP and IDLE instructions. In practice, this means that user code can change the NOSC<2:0> bit settings or issue #SLEEP instructions before the OST times out. This would allow an application to briefly wake-up, perform routine “housekeeping” tasks and return to Sleep before the device starts to operate from the external oscillator.

User code can also check which clock source is currently providing the device clocking by checking the status of the COSC<2:0> bits against the NOSC<2:0> bits. If these two sets of bits match, the clock switch has been completed successfully and the device is running from the intended clock source; the Primary Oscillator is providing the clock. Otherwise, FRC is providing the clock during wake-up from Reset or Sleep mode.

3.15 Reference Clock Output Module (REFO1 and REFO2)

3.15.1 APPLICATIONS

The PIC18F97J94 family has two Reference Clock Output modules. Each of the Reference Clock Output modules provides the user with the ability to send out a programmed output clock onto the REFO1or REFO2 pins.

3.15.2 REFERENCE CLOCK SOURCE

The module provides the ability to select one of the following clock sources:

- Primary Crystal Oscillator (POSC)
- Secondary Crystal Oscillator (SOSC)
- 32.768 kHz Internal Oscillator (INTOSC)
- Fast Internal Oscillator (FRC)
- Raw System Clock (sys_clk)
- Peripheral Clock (p1_clk)

It includes a programmable clock divider with ratios ranging from 1:1 to 1:65534.

When the clock source is a crystal or internal oscillator, the RSLP bit (REFOxCON<3> can be set to continue REFOx operation while the device is in Sleep Mode.

3.15.3 CLOCK SYNCHRONIZATION

The Reference Clock Output is enabled only once (ON = 1). Note that the source of the clock and the divider values should be chosen prior to the bit being set to avoid glitches on the REFO output.

Once the ON bit is set, its value is synchronized to the reference clock domain to enable the output. This ensures that no glitches will be seen on the output. Similarly, when the ON bit is cleared, the output and the associated output enable signals will be synchronized, and disabled on the falling edge of the reference clock. Note that with large divider values, this will cause the REFO to be enabled for some period after ON is cleared.

3.15.4 OPERATION IN SLEEP MODE

If any clock source, other than the peripheral clock, is used as a base reference (i.e., ROSEL<3:0> ≠ 0001), the user has the option to configure the behavior of the oscillator in Sleep mode. The RSLP Configuration bit determines if the oscillator will continue to run in Sleep. If RSLP = 0, the oscillator will be shut down in Sleep (assuming no other consumers are requesting it). If RSLP = 1, the oscillator will continue to run in Sleep.

The Reference Clock Output is synchronized with the Sleep signal to avoid any glitches on its output.

3.15.5 MODULE ENABLE SIGNAL

The REFOx module may be enabled or disabled using the REFOxMD register bit (PMD3, bit 1 or 0). The module also needs to be turned on using the ON bit (REFO1CON<7>).

3.15.5.1 Registers and Bits

This module provides the following device registers and/or bits:

- REFOxCON – Reference Clock Output Control Register
- REFOxCON1 – Reference Clock Output Control 1 Register
- REFOxCON2 – Reference Clock Output Control 2 Register
- REFOxCON3 – Reference Clock Output Control 3 Register

4.0 POWER-MANAGED MODES

All PIC18F97J94 Family devices offer a number of built-in strategies for reducing power consumption. These strategies can be particularly useful in applications, which are both power-constrained (such as battery operation), yet require periods of full-power operation for timing-sensitive routines (such as serial communications).

Aside from their low-power architecture, these devices include an expanded range of dedicated hardware features that allow the microcontroller to reduce power consumption to even lower levels when long-term hibernation is required, and still be able to resume operation on short notice.

The device has four power-saving features:

- Instruction-Based Power-Saving Modes
- Hardware-Based Power Reduction Features
- Microcontroller Clock Manipulation
- Selective Peripheral Control

Combinations of these methods can be used to selectively tailor an application's power consumption, while still maintaining critical or timing-sensitive application features. However, it is more convenient to discuss the strategies separately.

4.1 Overview of Power-Saving Modes

In addition to full-power operation, otherwise known as Run mode, PIC18F97J94 Family devices offer three instruction-based, power-saving modes and one hardware-based mode. In descending order of power consumption, they are:

- Idle
- Sleep (including retention Sleep)
- Deep Sleep (with and without retention)
- V_{BAT} (with and without RTCC)

By powering down all four modes, different functional areas of the microcontroller allow progressive reductions of operating and Idle power consumption. In addition, three of the modes can be tailored for more power reduction at a trade-off of some operating features. Table 4-1 lists all of the operating modes (including Run mode, for comparison) in order of increasing power savings and summarizes how the microcontroller exits the different modes.

TABLE 4-1: SUMMARY OF OPERATING MODES FOR PIC18F97J94 FAMILY DEVICES WITH VBAT POWER-SAVING FEATURES

Mode	Entry	Active Systems					Exit Conditions								
							Interrupts		Resets						
		Core	Peripherals	Data RAM Retention	RTCC(1)	DSGPRx(2)	All	INT0 Only	All	POR	$\overline{MCLR}$	RTCC Alarm	(DS)WDT(3)	VDD Restore	Code Execution Resumes
Run (default)	N/A	Y	Y	Y	Y	Y	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Idle	Instruction	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Next Instruction
Sleep modes:															
Sleep	Instruction	N	N(4)	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A	Next Instruction
Retention Sleep	Instruction + $\overline{RETEN}$ bit	N	N(4)	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N/A	
Deep Sleep modes:															
Retention Deep Sleep	Instruction + DSEN bit + $\overline{RETEN}$ bit	N	N	Y	Y	Y	N	Y	N	Y	Y	Y	Y	N/A	Next Instruction
Deep Sleep	Instruction + DSEN bit	N	N	N	Y	Y	N	Y	N	Y	Y	Y	Y	N/A	Reset Vector
VBAT:															
with RTCC	Hardware	N	N	N	Y	Y	N	N	N	N	N	N	N	Y	Reset Vector
w/o RTCC	Hardware + by disabling the RTCC PMD bit	N	N	N	N	Y	N	N	N	N	N	N	N	Y	

Note 1: If RTCC is otherwise enabled in firmware.

2: Data retention in the DSGPR0, DSGPR1, DSGPR2 and DSGPR3 registers.

3: Deep Sleep WDT in Deep Sleep modes; WDT in all other modes.

4: Some select peripherals may continue to operate in this mode, using either the LPRC or an external clock source.

4.2 Instruction-Based Power-Saving Modes

PIC18F97J94 Family devices have three instruction-based power-saving modes; two of these have additional features that allow for additional tailoring of power consumption. All three modes are entered through the execution of the SLEEP instruction. In descending order of power consumption, they are:

- Idle Mode: The CPU is disabled, but the system clock source continues to operate. Peripherals continue to operate, but can optionally be disabled.
- Sleep Modes: The CPU, system clock source and any peripherals that operate on the system clock source are disabled.
- Deep Sleep Modes: The CPU system clock source, and all the peripherals except RTCC and DSWDT are disabled. This is the lowest power mode for the device. The power to RAM and Flash is also disabled. Deep Sleep modes represent the lowest power modes available without removing power from the application.

Idle and Sleep modes are entered directly with the SLEEP statement. Having IDLEN (OSCCON<7>) set prior to the SLEEP statement will put the device into Idle mode. For Deep Sleep mode, it is necessary to set the DSEN bit (DSCONH<7>). To prevent inadvertent entry into Deep Sleep mode, and possible loss of data, the DSEN bit must be written to twice. The write need not be consecutive instructions; however, it is a better practice to write both, one after the other. It is also recommended to clear the DSCON1 register before setting the DSEN bit (Example 4-1).

Note:	SLEEP_MODE and IDLE_MODE are constants defined in the Assembler Include file for the selected device.

EXAMPLE 4-1: SLEEP ASSEMBLY SYNTAX

```
clrf   DSCON1
clrf   DSCON1
bsf    DSCON1,7
bsf    DSCON1,7
sleep

or

movlw  0x80
movwf  DSCON1
movwf  DSCON1
sleep
```

The instruction-based power-saving modes are exited as a result of several different hardware triggers. When the device exits one of these three operating modes, it is said to 'wake-up'. The characteristics of the power-saving modes are described in the subsequent sections.

4.2.1 INTERRUPTS COINCIDENT WITH POWER SAVE INSTRUCTIONS

Any interrupt that coincides with the execution of a SLEEP instruction will be held off until entry into Sleep, Idle or Deep Sleep mode is completed. The device will then wake-up from the power-managed mode.

Interrupts that occur during the Deep Sleep unlock sequence will interrupt the mandatory unlock sequence and cause a failure to enter Deep Sleep. For this reason, it is recommended to disable all interrupts during the Deep Sleep unlock sequence.

4.2.2 RETENTION REGULATOR

A second on-chip voltage regulator is used for power management in Sleep and Deep Sleep modes. This regulator, also known as the retention regulator, supplies core logic and other circuits with power at a lower VCORE level, about 1.2V nominal. Running these circuits at a lower voltage allows for an additional incremental power saving over the normal minimum VCORE level.

In Retention Sleep modes, using the regulator maintains the entire data RAM and its contents, instead of just a few protected registers. This allows the device to exit a power-saving mode and resume code execution as its previous state.

The retention regulator is controlled by the Configuration bit, $\overline{\text{RETEN}}$ (CONFIG7L<0>), and the SRETEN bit (RCON4<4>). The $\overline{\text{RETEN}}$ bit makes the retention regulator available for software control. By default ($\overline{\text{RETEN}}$ = 1), the regulator is disabled and the SRETEN bit has no effect. Programming $\overline{\text{RETEN}}$ (= 0) allows the SRETEN bit to control the regulator's operation, leaving its use in power-saving modes at the user's discretion.

Setting the SRETEN bit prior to executing the SLEEP instruction puts the device into Retention Sleep mode. If the DSEN bit was also unlocked and set prior to the instruction, the device will enter Retention Deep Sleep mode.

The retention regulator is not available outside of Sleep, Deep Sleep or VBAT modes. Enabling it while the device is operating in Run or Idle modes does not allow the device to operate at a lower level of VCORE.

4.2.3 IDLE MODE

When the device enters Idle mode, the following events occur:

- The CPU will stop executing instructions.
- The WDT is automatically cleared.
- The system clock source will remain active and the peripheral modules, by default, will continue to operate normally from the system clock source. Peripherals can optionally be shut down in Idle mode using their 'Stop in Idle' control bit. (See peripheral descriptions for further details.)
- If the WDT or FSCM is enabled, the LPRC will also remain active.

The processor will wake-up from Idle mode on the following events:

- On any interrupt that is individually enabled.
- On any source of device Reset.
- On a WDT time-out.

Upon wake-up from Idle mode, the clock is reapplied to the CPU and instruction execution begins immediately, starting with the instruction following the SLEEP instruction, or the first instruction in the Interrupt Service Routine (ISR).

4.2.3.1 Time Delays on Wake-up from Idle Mode

Unlike a wake-up from Sleep mode, there are no additional time delays associated with wake-up from Idle mode. The system clock is running during Idle mode, therefore, no start-up times are required at wake-up.

4.2.3.2 Wake-up from Idle on Interrupt

Any source of interrupt that is individually enabled using the corresponding control bit in the PIEx register, will be able to wake-up the processor from Idle mode. When the device wakes from Idle mode, one of two options may occur:

- If the GIE bit is set, the processor will wake and the Program Counter will begin execution at the interrupt vector.
- If the GIE bit is not set, the processor will wake and the Program Counter will continue execution following the SLEEP instruction.

The $\overline{PD}$ Status bit (RCON<2>) is set upon wake-up.

4.2.3.3 Wake-up from Idle on Reset

Any Reset, other than a Power-on Reset (POR), will wake-up the CPU from Idle mode on any device Reset, except a POR.

4.2.3.4 Wake-up from Idle on WDT Time-out

If the WDT is enabled, then the processor will wake-up from Idle mode on a WDT time-out and continue code execution with the instruction following the SLEEP instruction that initiated Idle mode. Note that the WDT time-out does not reset the device in this case. The $\overline{TO}$ bit (RCON<3>) will be set.

4.2.4 SLEEP MODES

Most 08KA101 family devices that incorporate power-saving features and VBAT, offer two distinct Sleep modes: Sleep mode and Retention Sleep mode. The characteristics of both Sleep modes are:

- The system clock source is shut down. If an on-chip oscillator is used, it is turned off.
- The device current consumption will be optimum, provided no I/O pin is sourcing the current.
- The Fail-Safe Clock Monitor (FSCM) does not operate during Sleep mode since the system clock source is disabled.
- The LPRC clock will continue to run in Sleep mode if the WDT is enabled.
- If Brown-out Reset (BOR) is enabled, the Brown-out Reset (BOR) circuit remains operational during Sleep mode.
- The WDT, if enabled, is automatically cleared prior to entering Sleep mode.
- Some peripherals may continue to operate in Sleep mode. These peripherals include I/O pins that detect a change in the input signal or peripherals that use an external clock input. Any peripheral that operates from the system clock source will be disabled in Sleep mode.

The processor will exit, or 'wake-up' from Sleep on one of the following events:

- On any interrupt source that is individually enabled
- On any form of device Reset
- On a WDT time-out

4.2.4.1 Retention Sleep Mode

Retention Sleep mode allows for additional power savings over Sleep mode by maintaining key systems from the lower power retention regulator. When the retention regulator is used, the normal on-chip voltage regulator (operating at 1.8V nominal) is turned off and will enable a low-power (1.2V typical) regulator. By using a lower voltage, a lower total power consumption is achieved.

Retention Sleep also offers the advantage of maintaining the contents of the data RAM. As a trade-off, the wake-up time is longer than that for Sleep mode.

Retention Sleep mode is controlled by the SRETEN bit (RCON4<4>) and the $\overline{\text{RETEN}}$ Configuration bit, as described in Section 4.2.2, Retention Regulator.

4.3 Clock Source Considerations

When the device wakes up from either of the Sleep modes, it will restart the same clock source that was active when Sleep mode was entered. Wake-up delays for the different oscillator modes are shown in Table 4-3 and Table 4-4, respectively.

If the system clock source is derived from a crystal oscillator and/or the PLL, the Oscillator Start-up Timer (OST) and/or PLL lock times must be applied before the system clock source is made available to the device. As an exception to this rule, no oscillator delays are necessary if the system clock source is the Secondary Oscillator and it was running while in Sleep mode.

4.3.1 SLOW OSCILLATOR START-UP

The OST and PLL lock times may not have expired when the power-up delays have expired.

To avoid this condition, one can enable Two-Speed Start-up by the device that will run on FRC until the clock source is stable. Once the clock source is stable, the device will switch to the selected clock source.

4.3.2 WAKE-UP FROM SLEEP ON INTERRUPT

Any source of interrupt that is individually enabled, using its corresponding control bit in the PIEx registers, can wake-up the processor from Sleep mode. When the device wakes from Sleep mode, one of two following actions may occur:

- If the GIE bit is set, the processor will wake and the Program Counter will begin execution at the interrupt vector.
- If the GIE bit is not set, the processor will wake and the Program Counter will continue execution following the SLEEP instruction that initiated Sleep mode.

4.3.3 WAKE-UP FROM SLEEP ON RESET

All sources of device Reset will wake-up the processor from Sleep mode.

4.3.4 WAKE-UP FROM SLEEP ON WATCHDOG TIME-OUT

If the Watchdog Timer (WDT) is enabled and expires while the device is in Sleep mode, the processor will wake-up. The SWDTEN Status bit (RCON2<5>) is set to indicate that the device resumed operation due to the WDT expiration. Note that this event does not reset the device. Operation continues from the instruction following the SLEEP instruction that initiated Sleep mode.

4.3.5 CONTROL BIT SUMMARY FOR SLEEP MODES

Table 4-2 shows the settings for the bits relevant to Sleep modes.

TABLE 4-2: BIT SETTINGS FOR ALL SLEEP MODES

Mode	DSEN DSCONH<7>	Retention Regulator		
		$\overline{\text{RETEN}}$ CONFIG7L<0>	SRETEN RCON4<4>	State
Sleep	x	1	x	Disabled
	x	0	0	Disabled
Retention Sleep	x	0	1	Enabled

4.3.6 WAKE-UP DELAYS

The restart delay, associated with waking up from Sleep and Retention Sleep modes, parallel each other in terms of clock start-up times. They differ in the time it takes to switch over from their respective regulators. The delays for the different oscillator modes are shown in Table 4-3 and Table 4-4, respectively.

TABLE 4-3: DELAY TIMES FOR EXITING FROM SLEEP MODE

Clock Source		Exit Delay	Oscillator Delay	Notes
EC		TPM	—	1
ECPLL		TPM	TLOCK	1, 3
MS, HS		TPM	TOST	1, 2
MSPLL, HSPLL		TPM	TOST + TLOCK	1, 2, 3
SOSC	(Off during Sleep)	TPM	TOST	1, 2
	(On during Sleep)	TPM	—	1
FRC, FRCDIV		TPM	TFRC	1, 4
LPRC	(Off during Sleep)	TPM	TLPRC	1, 4
	(On during Sleep)	TPM	—	1
FRCPLL		TPM	TLOCK	1, 3

Note 1: TPM = Start-up delay for program memory stabilization.

2: TOST = Oscillator Start-up Timer (OST); a delay of 1024 oscillator periods before the oscillator clock is released to the system.

3: TLOCK = PLL lock time.

4: TFRC and TLPRC are RC Oscillator start-up times.

TABLE 4-4: DELAY TIMES FOR EXITING FROM RETENTION SLEEP MODE

Clock Source		Exit Delay	Oscillator Delay	Notes
EC		TRETR + TPM	—	1, 2
ECPLL		TRETR + TPM	TLOCK	1, 2, 4
MS, HS		TRETR + TPM	TOST	1, 2, 3
MSPLL, HSPLL		TRETR + TPM	TOST + TLOCK	1, 2, 3, 4
SOSC	(Off during Sleep)	TRETR + TPM	TOST	1, 2, 3
	(On during Sleep)	TRETR + TPM	—	1, 2
FRC, FRCDIV		TRETR + TPM	TFRC	1, 2, 5
LPRC	(Off during Sleep)	TRETR + TPM	TLPRC	1, 2, 5
	(On during Sleep)	TRETR + TPM	—	1, 2
FRCPLL		TRETR + TPM	TLOCK	1, 2, 4

Note 1: TRETR = Retention regulator start-up delay.

2: TPM = Start-up delay for program memory stabilization; applicable only when IPEN (RCON<7>) = 0.

3: TOST = Oscillator Start-up Timer; a delay of 1024 oscillator periods before the oscillator clock is released to the system.

4: TLOCK = PLL lock time.

5: TFRC and TLPRC are RC Oscillator start-up times.

4.4 Deep Sleep Modes

The Deep Sleep modes puts the device into its lowest power consumption states without requiring the use of external switches to remove power from the device. There are two modes available: Deep Sleep mode and Retention Deep Sleep mode.

During both Deep Sleep modes, the power to the microcontroller core is removed to reduce leakage current. Therefore, most peripherals and functions of the microcontroller become unavailable during Deep Sleep. However, a few specific peripherals and functions are powered directly from the VDD supply rail of the microcontroller, and therefore, can continue to function in Deep Sleep. In addition, four data memory locations, DSGPR0, DSGPR1, DSGPR2 and DSGPR3, are preserved for context information after an exit from Deep Sleep.

Deep Sleep has a dedicated Deep Sleep Brown-out Reset (DSBOR) and a Deep Sleep Watchdog Timer Reset (DSWDT) for monitoring voltage and time-out events in Deep Sleep mode. The DSBOR and DSWDT are independent of the standard BOR and WDT used with other power-managed modes (Run, Idle and Sleep).

Entering Deep Sleep mode clears the Deep Sleep Wake-up Source Registers (DSWAKEL and DSWAKEH). If enabled, the Real-Time Clock and Calendar (RTCC) continues to operate uninterrupted.

When a wake-up event occurs in Deep Sleep mode (by Reset, RTCC alarm, External Interrupt (INT0) or DSWDT), the device will exit Deep Sleep mode and re-arm a Power-on Reset (POR). When the device is released from Reset, code execution will resume at the Reset vector.

4.4.1 RETENTION DEEP SLEEP MODE

In Retention Deep Sleep, the retention regulator is enabled, which allows the data RAM to retain data while all other systems are powered down. This also allows the device to return to code execution where it left off, instead of going through a POR-like Reset.

As a trade-off, Retention Deep Sleep mode has greater power consumption than Deep Sleep. However, it offers the lowest level of power consumption of the power-saving modes that still allows a direct return to code execution.

Retention Deep Sleep is controlled by the SRETEN bit (RCON4<4>) and the $\overline{\text{RETEN}}$ Configuration bit, as described in **Section 4.2.2 "Retention Regulator"**.

4.4.2 ENTERING DEEP SLEEP MODES

Deep Sleep modes are entered by:

- Setting the DSEN bit (DSCONH<7>)
- Executing the SLEEP instruction

To enter Retention Deep Sleep, the SRETEN bit must also be set prior to setting the DSEN bit (Example 4-1).

In order to minimize the possibility of inadvertently entering Deep Sleep, the DSEN bit must be set by two separate write operations. To enter Deep Sleep, the SLEEP instruction must be executed after setting the DSEN bit (i.e., the next instruction). If DSEN is not set when Sleep is executed, the device will enter a Sleep mode instead.

4.4.3 DEEP SLEEP WAKE-UP SOURCES

The device can be awakened from Deep Sleep modes by any of the following:

- $\overline{\text{MCLR}}$
- POR
- RTCC Alarm
- INT0 Interrupt
- DSWDT Event

After waking from Deep Sleep mode, the device performs a POR. When the device is released from Reset, code execution will begin at the device's Reset vector.

The software can determine if the wake-up was caused from an exit from Deep Sleep mode by reading the DPSLP bit (RCON4<2>). If this bit is set, the POR was caused by a Deep Sleep exit. The DPSLP bit must be manually cleared by the software.

The software can determine the wake-up event source by reading the DSWAKE registers. These registers are cleared automatically when entering Deep Sleep mode, so software should read these registers after exiting Deep Sleep mode or before re-enabling this mode.

4.4.4 CLOCK SELECTION ON WAKE-UP FROM DEEP SLEEP MODE

For Deep Sleep mode, the processor will restart with the default oscillator source, selected with the FOSCx Configuration bits. On wake-up from Deep Sleep, a POR is generated internally, hence, the system resets to its POR state with the exception of the RCONx, DSCONH/L and DSGPRx registers.

For Retention Deep Sleep, the processor restarts with the same clock source that was selected before entering Retention Deep Sleep mode. Wake-up is similar to that of Sleep and Retention Sleep modes.

4.4.5 SAVING CONTEXT DATA WITH THE DSGPRx REGISTERS

As exiting Deep Sleep mode causes a POR, most Special Function Registers (SFRs) reset to their default POR values. In addition, because the core power is not supplied in Deep Sleep mode, information in data RAM may be lost when exiting this mode. Applications which require critical data to be saved prior to Deep Sleep may use the Deep Sleep General Purpose registers, DSGPR0, DSGPR1, DSGPR2 and DSGPR3. Unlike other SFRs, the contents of these registers are preserved while the device is in Deep Sleep mode. After exiting Deep Sleep, software can restore the data by reading the registers and clearing the RELEASE bit (DSCONL<0>).

Any data stored in the DSGPRx registers must be written twice. Like other Deep Sleep control features, the write operations do not need to be sequential. However, back-to-back writes are a recommended programming practice.

Since the contents of data RAM are maintained in Retention Deep Sleep, the use of the DSGPRx registers to store critical data is not necessary in this mode.

4.4.6 I/O PINS DURING DEEP SLEEP

During Deep Sleep, general purpose I/O pins retain their previous states. Pins that are configured as inputs (TRIS bit is set), prior to entry into Deep Sleep, remain high-impedance during Deep Sleep.

Pins that are configured as outputs (TRIS bit is clear), prior to entry into Deep Sleep, will remain as output pins during Deep Sleep. While in this mode, they will drive the output level determined by their corresponding LAT bit at the time of entry into Deep Sleep.

Once the device wakes back up, all I/O pins will continue to maintain their previous states, even after the device has finished the POR sequence and is executing application code again. Pins configured as inputs during Deep Sleep will remain high-impedance and pins configured as outputs will continue to drive their previous value. After waking up, the TRIS and LAT registers will be reset. If firmware modifies the TRIS and LAT values for the I/O pins, they will not immediately go to the newly configured states. Once the firmware clears the RELEASE bit (DSCONL<0>), the I/O pins are “released”. This causes the I/O pins to take the states configured by their respective TRIS and LAT bit values.

If the Deep Sleep BOR (DSBOR) is enabled, and a DSBOR event occurs during Deep Sleep (or V_{DD} is hard-cycled to V_{SS}), the I/O pins will be immediately released, similar to clearing the RELEASE bit. All previous state information will be lost, including the general purpose DSGPR0, DSGPR1, DSGPR2 and DSGPR3 contents. DSGPRx register contents will be maintained if the V_{BAT} pin is powered.

If a $\overline{MCLR}$ Reset event occurs during Deep Sleep, the I/O pins will also be released automatically, but in this case, the DSGPR0, DSGPR1, DSGPR2 and DSGPR3 contents will remain valid.

In case of $\overline{MCLR}$ Reset and all other Deep Sleep wake-up cases, application firmware needs to clear the RELEASE bit (DSCONL<0>) in order to reconfigure the I/O pins.

4.4.7 DEEP SLEEP WATCHDOG TIMER (DSWDT)

Deep Sleep has its dedicated WDT (DSWDT). It is enabled through the DSWDTEN Configuration bit. The DSWDT is equipped with a postscaler for time-outs of 2.1 ms to 25.7 days, configurable through the Configuration bits, DSWDTPS<4:0>. Entering Deep Sleep mode automatically clears the DSWDT.

The DSWDT also has a configurable reference clock source for selecting the LPRC or SOSC. The reference clock source is configured through the DSWDTOSC Configuration bit.

Under certain circumstances, it is possible for the DSWDT clock source to be off when entering Deep Sleep mode. In this case, the clock source is turned on automatically (if DSWDT is enabled), without the need for software intervention. However, this can cause a delay in the start of the DSWDT counters. In order to avoid this delay, when using SOSC as a clock source, the application can activate SOSC prior to entering Deep Sleep mode.

4.4.8 DEEP SLEEP LOW-POWER BROWN-OUT RESET

Devices with a Deep Sleep Power-Saving mode also have a dedicated BOR for Deep Sleep modes (DSBOR). It has a trip point range of 1.7V-2.3V nominal and is enabled through the DSBOREN (CONFIG7L<3>) Configuration bit.

When the device enters a Deep Sleep mode and receives a DSBOR event, the device will not wake-up and will remain in the Deep Sleep mode. When a valid wake-up event occurs and causes the device to exit Deep Sleep mode, software can determine if a DSBOR event occurred during Deep Sleep mode by reading the BOR (DSWAKEL<6>) Status bit.

4.4.9 RTCC AND DEEP SLEEP

The RTCC can operate uninterrupted during Deep Sleep modes. It can wake-up the device from Deep Sleep by configuring an alarm. The RTCC clock source is configured with the RTCC Clock Select bits, RTCCLKSEL<1:0>. The available reference clock sources are the LPRC and SOSC. If the LPRC is used, the RTCC accuracy will directly depend on the LPRC tolerance.

If the RTCC is not required, Deep Sleep mode with the RTCC disabled, affords the lowest power consumption of any of the instruction-based power-saving modes.

4.4.10 CONTROL BIT SUMMARY FOR SLEEP MODES

Table 4-5 shows the settings for the bits relevant to Deep Sleep modes.

TABLE 4-5: BIT SETTINGS FOR ALL DEEP SLEEP MODES

Instruction-Based Mode	DSEN (DSCONH<7>)	Retention Regulator			DSWDTEN (CONFIG8H<0>)
		$\overline{\text{RETEN}}$ (CONFIG7L<0>)	SRETEN (RCON4<4>)	State	
Retention Deep Sleep	1	0	1	Enabled	0
Deep Sleep	1	1	x	Disabled	x

4.4.11 WAKE-UP DELAYS

The Reset delays associated with wake-up from Deep Sleep and Retention Deep Sleep modes, in different oscillator modes, are provided in Table 4-6 and Table 4-7, respectively.

Note: The PMSLP bit (RCON4<0>) allows the voltage regulator to be maintained during Sleep modes.

TABLE 4-6: DELAY TIMES FOR EXITING FROM DEEP SLEEP MODE

Clock Source		Exit Delay	Oscillator Delay	Notes
EC		TDSWU	—	
ECPLL		TDSWU	TLOCK	1, 3
MS, HS		TDSWU	TOST	1, 2
MSPLL, HSPLL		TDSWU	TOST + TLOCK	1, 2, 3
SOSC	(Off during Sleep)	TDSWU	TOST	1, 2
	(On during Sleep)	TDSWU	—	1
FRC, FRCDIV		TDSWU	TFRC	1, 4
LPRC	(Off during Sleep)	TDSWU	TLPRC	1, 4
	(On during Sleep)	TDSWU	—	1
FRCPLL		TDSWU	TFRC + TLOCK	1, 3, 4

Note 1: TDSWU = Deep Sleep wake-up delay.

2: TOST = Oscillator Start-up Timer; a delay of 1024 oscillator periods before the oscillator clock is released to the system.

3: TLOCK = PLL lock time.

4: TFRC and TLPRC are RC Oscillator start-up times.

TABLE 4-7: DELAY TIMES FOR EXITING RETENTION DEEP SLEEP MODE

Clock Source		Exit Delay	Oscillator Delay	Notes
EC		TRETR + TPM	—	1, 2, 6
ECPLL		TRETR + TPM	TLOCK	1, 2, 4, 6
MS, HS		TRETR + TPM	TOST	1, 2, 3, 6
MSPLL, HSPLL		TRETR + TPM	TOST + TLOCK	1, 2, 3, 4, 6
SOSC	Off during Sleep	TRETR + TPM	TOST	1, 2, 3, 6
	On during Sleep	TRETR + TPM	—	1, 2, 6
FRC, FRCDIV		TRETR + TPM	TFRC	1, 2, 5, 6
LPRC:	Off during Sleep	TRETR + TPM	TLPRC	1, 2, 5, 6
	On during Sleep	TRETR + TPM	—	1, 2, 6
FRCPLL		TRETR + TPM	TLOCK	1, 2, 3, 6

Note 1: TPM = Start-up delay for program memory stabilization; applicable only when IPEN (RCON<7>) = 0.

2: TRETR = Retention regulator start-up delay.

3: TOST = Oscillator Start-up Timer (OST); a delay of 1024 oscillator periods before the oscillator clock is released to the system.

4: TLOCK = PLL lock time.

5: TFRC and TLPRC = RC Oscillator start-up times.

6: TFLASH = Flash program memory ready delay. Setting the PMSLP bit will provide a faster wake-up.

4.5 VBAT Mode

VBAT mode is a hardware-based power mode that maintains only the most critical operations when a power loss occurs on VDD. The mode does this by powering these systems from a back-up power source connected to the VBAT pin. In this mode, the RTCC can run even when there is no power on VDD.

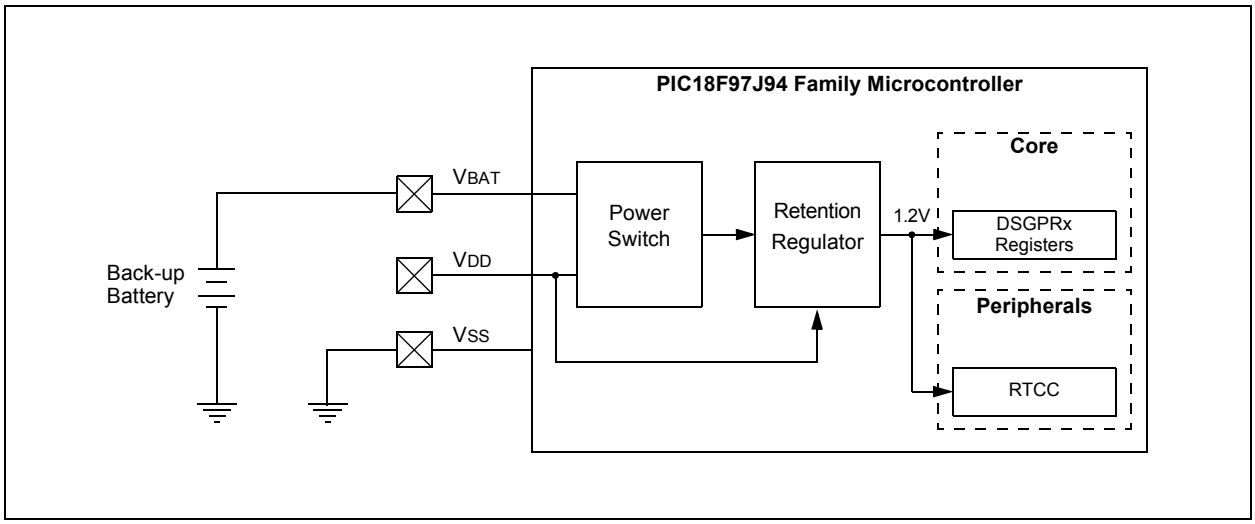
VBAT mode is entered whenever power is removed from VDD. An on-chip power switch detects the power loss from the VDD and connects the VBAT pin to the retention regulator. This provides power at 1.2V to maintain the retention regulator, as well as the RTCC, with its clock source (if enabled) and the Deep Sleep General Purpose (DSGPRx) registers (Figure 4-1).

Entering VBAT mode requires that a power source, distinct from the main VDD power source, be available on VBAT and that VDD be completely removed from the VDD pin(s). Removing VDD can be either unintentional, as in a power failure, or as part of a deliberate power reduction strategy.

As with Deep Sleep modes, the contents of the Deep Sleep General Purpose (DSGPRx) registers are maintained by the retention regulator. Since the power loss on VDD may be unforeseen, it is recommended to load any data to be saved in these registers in advance.

Any data stored in the DSGPRx registers must be written twice. The write operations do not need to be sequential; however, back-to-back writes are a recommended programming practice.

FIGURE 4-1: VBAT POWER TOPOLOGY



4.5.1 WAKE-UP FROM VBAT MODES

When VDD is restored to a device in VBAT mode, it automatically wakes. Wake-up occurs with a POR, after which the device starts executing code from the Reset vector. All SFRs, except the Deep Sleep semaphores and RTCC registers are reset to their POR values. If the RTCC was not configured to run during VBAT mode, it will remain disabled and RTCC will not run. Wake-up timing is similar to that for a normal POR.

Wake-up from VBAT mode is identified by checking the state of the VBAT bit (RCON3<0>). If this bit is set when the device is awake and starting to execute the code from the Reset vector, it indicates that the exit was from VBAT mode. To identify future VBAT wake-up events, the bit must be cleared in software.

When a POR event occurs with no battery connected to the VBAT pin, the VBPOR bit (RCON3<1>) becomes set. On the device, if there is no battery connected to the VBAT pin, VBPOR will indicate that the battery needs to be connected to the VBAT pin.

In addition, if the VBAT power source falls below the level needed for Deep Sleep semaphore operation while in VBAT mode (e.g., the battery has been drained), the VBPOR bit will be set. VBPOR is also set when the microcontroller is powered up the very first time, even if power is supplied to VBAT.

4.6 Saving Context Data with the DSGPRx Registers

As exiting VBAT causes a POR, most Special Function Registers reset to their default POR values. In addition, because the core power is not supplied in VBAT mode, information in data RAM will be lost when exiting this mode. Applications which require critical data to be saved, should be saved in DSGPR0, DSGPR1, DSGPR2 and DSGPR3.

Any data stored to the DSGPRx registers must be written twice. The write operations do not need to be sequential. However, back-to-back writes are a recommended programming practice.

After exiting VBAT mode, software can restore the data by reading the registers.

4.6.1 I/O PINS DURING VBAT MODE

All I/O pins should be maintained at VSS level; no I/O pins should be given VDD (refer to **"Absolute Maximum Ratings(†)"** in **Section 30.0 "Electrical Specifications"**) during VBAT mode. The only exceptions are the SOSCI and SOSCO pins, which maintain their states if the Secondary Oscillator is being used as the RTCC clock source. It is the user's responsibility to restore the I/O pins to their proper states, using the TRIS and LAT bits, once VDD has been restored.

REGISTER 4-1: DSCONL: DEEP SLEEP CONTROL REGISTER LOW

U-0	U-0	U-0	U-0	U-0	R-0	R/W-0, HSC	R/W-0, HS
—	—	—	—	—	r	DSBOR(1)	RELEASE(1)
bit 7							bit 0

Legend:	r = Reserved bit	HSC = Hardware Settable/Clearable bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown
HS = Hardware Settable bit			

bit 7-3 **Unimplemented:** Read as '0'

bit 2 **Reserved:** Maintained as '0'

bit 1 **DSBOR:** Deep Sleep BOR Event Status bit(1)
1 = DSBOR was enabled and VDD dropped below the DSBOR threshold during Deep Sleep(2)
0 = DSBOR disabled while device is in Deep Sleep mode

bit 0 **RELEASE:** I/O Pin State Release bit(1)
Upon waking from Deep Sleep, the I/O pins maintain their previous states. Clearing this bit will release the I/O pins and allow their respective TRIS and LAT bits to control their states.

Note 1: This is the value when VDD is initially applied.
2: Unlike all other events, a Deep Sleep BOR event will not cause a wake-up from Deep Sleep; this bit is present only as a Status bit.

REGISTER 4-2: DSCONH: DEEP SLEEP CONTROL REGISTER HIGH

R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0, HS(2)
DSEN(1)	—	—	—	—	—	—	RTCCWDIS
bit 7							bit 0

Legend:	HS = Hardware Settable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **DSEN:** Deep Sleep Mode Enable bit(1)
1 = Deep Sleep mode is enabled and device will enter Deep Sleep mode when the SLEEP instruction is executed
0 = Deep Sleep mode is not enabled

bit 6-1 **Unimplemented:** Read as '0'

bit 0 **RTCCWDIS:** RTCC Wake-up Disable bit(2)
1 = Wake-up from RTCC is disabled
0 = Wake-up from RTCC is enabled

Note 1: In order to enter Deep Sleep, DSEN must be written to in two separate operations. The write operations do not need to be consecutive. Before writing DSEN, the DSCON1 register should be cleared twice.
2: This is the value when VDD is initially applied.

REGISTER 4-3: DSWAKEL: DEEP SLEEP WAKE-UP SOURCE REGISTER LOW(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DSFLT	BOR	EXT	DSWDT	DSRTC	$\overline{\text{MCLR}}$	ICD	DSPOR
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **DSFLT:** Deep Sleep Fault Detect bit
1 = A Deep Sleep Fault was detected during Deep Sleep
0 = A Deep Sleep Fault was not detected during Deep Sleep

bit 6 **BOR:** BOR Deep-Sleep Wake-up Source Enable bit
1 = DSBOR event will wake device from Deep Sleep
0 = DSBOR event will not wake device from Deep Sleep

bit 5 **EXT:** External Interrupt Wake-up Source Enable bit
1 = External interrupt will wake device from Deep Sleep
0 = External interrupt will not wake device from Deep Sleep

bit 4 **DSWDT:** DSWDT Deep-Sleep Wake-up Source Enable bit
1 = DSWDT roll-over event will wake device from Deep Sleep
0 = DSWDT roll-over event will not wake device from Deep Sleep

bit 3 **DSRTC:** Real-Time Clock and Calendar Alarm bit
1 = The Real-Time Clock/Calendar triggered an alarm during Deep Sleep
0 = The Real-Time Clock /Calendar did not trigger an alarm during Deep Sleep

bit 2 **$\overline{\text{MCLR}}$:** $\overline{\text{MCLR}}$ Deep-Sleep Wake-up Source Enable bit
1 = The $\overline{\text{MCLR}}$ Reset will wake device from Deep Sleep
0 = The $\overline{\text{MCLR}}$ Reset will not wake device from Deep Sleep

bit 1 **ICD:** In-Circuit Debugger Deep-Sleep Wake-up Source Enable bit
1 = In-Circuit Debugger will wake device from Deep Sleep
0 = In-Circuit Debugger will not wake device from Deep Sleep

bit 0 **DSPOR:** Power-on Reset Event bit
1 = The V_{DD} supply POR circuit was active and a POR event was detected
0 = The V_{DD} supply POR circuit was not active, or was active but did not detect a POR event

Note 1: To be set in software, all bits in DSWAKE must be written to twice. The write operations do not need to be consecutive.

REGISTER 4-4: DSWAKEH: DEEP SLEEP WAKE-UP SOURCE REGISTER HIGH

U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
—	—	—	—	—	—	—	INT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-1 **Unimplemented:** Read as '0'

bit 0 **INT0:** Deep Sleep Wake-up Source Enable bit
1 = INT0 interrupt will wake device from Deep Sleep
0 = INT0 interrupt will not wake device from Deep Sleep

4.7 Selective Peripheral Power Control

Sleep and Idle modes allow users to substantially reduce power consumption by slowing or stopping the CPU clock. Even so, peripheral modules still remain clocked, and thus, consume some amount of power. There may be cases where the application needs what these modes do not provide: the ability to allocate limited power resources to the CPU while eliminating power consumption from the peripherals. The 08KA101 family addresses this requirement by allowing peripheral modules to be selectively enabled or disabled, reducing or eliminating their power consumption.

4.7.1 DISABLING PERIPHERAL MODULES

Most of the peripheral modules in the 08KA101 family architecture can be selectively disabled, reducing, or essentially eliminating, their power consumption during all operating modes. Two different options are available to users, each with a slightly different effect.

4.7.2 MODULE ENABLE BIT (XXXEN)

Many peripheral modules have a Module Enable bit, generically named, “XXXEN”, usually located in Bit Position 7 of their control registers (or Primary Control registers for more complex modules). Here, “XXX” represents the mnemonic form for the module of the module name. For example, the enable bit for an MSSPx module is “SSPEN”, and so on. The bit is provided for all serial and parallel communication modules and the Real-Time Clock (RTC). Clearing this bit disables the module’s operation; however, it continues to receive clock signals and draw a minimal amount of current.

As with all earlier PIC® MCU devices, timers continue to be under selective operation and are controlled by their own TON bit, also located in Position 7. The A/D Converter also has a legacy enable bit, ADON, that has the same function as the XXXEN bits. I/O ports and features associated with them, such as input change notification and input capture, do not have their own module enable bits, since their operation is secondary to other modules.

Disabling modules not required for a particular application, in this manner, allows for the selective and dynamic adjusting power consumption, under software control, as the application is running.

4.7.3 PERIPHERAL MODULE DISABLE BIT (XXMD)

All peripheral modules (except for I/O ports) also have a second control bit that can disable their functionality. These bits, known as the Peripheral Module Disable (PMD) bits, are generically named, “XXMD” (using “XX” as the mnemonic version of the module’s name), as shown in **Section 4.7.2 “Module Enable Bit (XXXEN)”**). These bits are located in the PMDx SFRs. In contrast to the module enable bits, the XXMD bit must be set (= 1) to disable the module.

While the PMD and module enable bits both disable a peripheral’s functionality, the PMD bit completely shuts down the peripheral, effectively powering down all circuits and removing all clock sources. This has the additional effect of making any of the module’s control and buffer registers, mapped in the SFR space, unavailable for operations. In other words, when the PMD bit is used to disable a module, the peripheral ceases to exist until the PMD bit is cleared. This differs from using the module enable bit, which allows the peripheral to be reconfigured and buffer registers preloaded, even when the peripheral’s operations are disabled.

The PMD bit is most useful in highly power-sensitive applications, where even tiny savings in power consumption can determine the ability of an application to function. In these cases, the bits can be set before the main body of the application to remove those peripherals that will not be needed at all.

REGISTER 4-5: PMD0: PERIPHERAL MODULE DISABLE REGISTER 0

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CCP10MD	CCP9MD	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	ECCP3MD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CCP10MD:** CCP10 Module Disable bit
1 = The CCP10 module is disabled. All CCP10 registers are held in Reset and are not writable.
0 = The CCP10 module is enable

bit 6 **CCP9MD:** CCP9 Module Disable bit
1 = The CCP9 module is disabled. All CCP9 registers are held in Reset and are not writable.
0 = The CCP9 module is enabled

bit 5 **CCP8MD:** CCP8 Module Disable bit
1 = The CCP8 module is disabled. All CCP8 registers are held in Reset and are not writable.
0 = The CCP8 module is enabled

bit 4 **CCP7MD:** CCP7 Module Disable bit
1 = The CCP7 module is disabled. All CCP7 registers are held in Reset and are not writable.
0 = The CCP7 module is enabled

bit 3 **CCP6MD:** CCP6 Module Disable bit
1 = The CCP6 module is disabled. All CCP6 registers are held in Reset and are not writable.
0 = The CCP6 module is enabled

bit 2 **CCP5MD:** CCP5 Module Disable bit
1 = The CCP5 module is disabled. All CCP5 registers are held in Reset and are not writable.
0 = The CCP5 module is enabled

bit 1 **CCP4MD:** CCP4 Module Disable bit
1 = The CCP4 module is disabled. All CCP4 registers are held in Reset and are not writable.
0 = The CCP4 module is enabled

bit 0 **ECCP3MD:** ECCP3 Module Disable bit
1 = The ECCP3 module is disabled. All ECCP3 registers are held in Reset and are not writable.
0 = The ECCP3 module is enabled

REGISTER 4-6: PMD1: PERIPHERAL MODULE DISABLE REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCP2MD	ECCP1MD	UART4MD	UART3MD	UART2MD	UART1MD	SSP2MD	SSP1MD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ECCP2MD:** ECCP2 Module Disable bit
1 = The ECCP2 module is disabled. All ECCP2 registers are held in Reset and are not writable.
0 = The ECCP2 module is enabled

bit 6 **ECCP1MD:** ECCP1 Module Disable bit
1 = The ECCP1 module is disabled. All ECCP1 registers are held in Reset and are not writable.
0 = The ECCP1 module is enabled

bit 5 **UART4MD:** USART4 Module Disable bit
1 = The USART4 module is disabled. All USART4 registers are held in Reset and are not writable.
0 = The USART4 module is enabled

bit 4 **UART3MD:** USART3 Module Disable bit
1 = The USART3 module is disabled. All USART3 registers are held in Reset and are not writable.
0 = The USART3 module is enabled

bit 3 **UART2MD:** USART2 Module Disable bit
1 = The USART2 module is disabled. All USART2 registers are held in Reset and are not writable.
0 = The USART2 module is enabled

bit 2 **UART1MD:** USART1 Module Disable bit
1 = The USART1 module is disabled. All USART1 registers are held in Reset and are not writable.
0 = The USART1 module is enabled

bit 1 **SSP2MD:** SSP2 Module Disable bit
1 = The SSP2 module is disabled. All SSP2 registers are held in Reset and are not writable.
0 = The SSP2 module is enabled

bit 0 **SSP1MD:** SSP1 Module Disable bit
1 = The SSP1 module is disabled. All SSP1 registers are held in Reset and are not writable.
0 = The SSP1 module is enabled

REGISTER 4-7: PMD2: PERIPHERAL MODULE DISABLE REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR8MD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TMR8MD:** Timer8 Module Disable bit
1 = The Timer8 module is disabled. All Timer8 registers are held in Reset and are not writable.
0 = The Timer8 module is enabled

bit 6 **TMR6MD:** Timer6 Module Disable bit
1 = The Timer6 module is disabled. All Timer6 registers are held in Reset and are not writable.
0 = The Timer6 module is enabled

bit 5 **TMR5MD:** Timer5 Module Disable bit
1 = The Timer5 module is disabled. All Timer5 registers are held in Reset and are not writable.
0 = The Timer5 module is enabled

bit 4 **TMR4MD:** Timer4 Module Disable bit
1 = The Timer4 module is disabled. All Timer4 registers are held in Reset and are not writable.
0 = The Timer4 module is enabled

bit 3 **TMR3MD:** Timer3 Module Disable bit
1 = The Timer3 module is disabled. All Timer3 registers are held in Reset and are not writable.
0 = The Timer3 module is enabled

bit 2 **TMR2MD:** Timer2 Module Disable bit
1 = The Timer2 module is disabled. All Timer2 registers are held in Reset and are not writable.
0 = The Timer2 module is enabled

bit 1 **TMR1MD:** Timer1 Module Disable bit
1 = The Timer1 module is disabled. All Timer1 registers are held in Reset and are not writable.
0 = The Timer1 module is enabled

bit 0 **TMR0MD:** Timer0 Module Disable bit
1 = The Timer0 module is disabled. All Timer0 registers are held in Reset and are not writable.
0 = The Timer0 module is enabled

REGISTER 4-8: PMD3: PERIPHERAL MODULE DISABLE REGISTER 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DSMMD	CTMUMD	ADCMD	RTCCMD	LCDMD	PSPMD	REFO1MD	REFO2MD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **DSMMD:** Modulator Output Module Disable bit

1 = The Modulator Output module is disabled. All Modulator Output registers are held in Reset and are not writable.
0 = The Modulator Output module is enabled

bit 6 **CTMUMD:** CTMU Module Disable bit

1 =The CTMU module is disabled. All CTMU registers are held in Reset and are not writable.
0 =The CTMU module is enabled

bit 5 **ADCMD:** ADC Module Disable bit

1 =The ADC module is disabled. All ADC registers are held in Reset and are not writable.
0 =The ADC module is enabled

bit 4 **RTCCMD:** RTCC Module Disable bit

1 = The RTCC module is disabled. All RTCC registers are held in Reset and are not writable.
0 = The RTCC module is enabled

bit 3 **LCDMD:** LCD Module Disable bit

1 = The LCD module is disabled. All LCD registers are held in Reset and are not writable.
0 = The LCD module is enabled

bit 2 **PSPMD:** PSP Module Disable bit

1 = The PSP module is disabled. All PSP registers are held in Reset and not are writable.
0 = The PSP module is enabled

bit 1 **REFO1MD:** REFO1 Module Disable bit

1 = The REFO1 module is disabled. All REFO1 registers are held in Reset and are not writable.
0 = The REFO1 module is enabled

bit 0 **REFO2MD:** REFO2 Module Disable bit

1 = The REFO2 module is disabled. All REFO2 registers are held in Reset and are not writable.
0 = The REFO2 module is enabled

REGISTER 4-9: PMD4: PERIPHERAL MODULE DISABLE REGISTER 4

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
CMP1MD	CMP2MD	CMP3MD	USBMD	IOCMD	LVDMD	—	EMBMD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CMP1MD:** CMP1 Module Disable bit
1 = The CMP1 module is disabled; all CMP1 registers are held in Reset and are not writable
0 = The CMP1 module is enabled

bit 6 **CMP2MD:** CMP2 Module Disable bit
1 = The CMP2 module is disabled; all CMP2 registers are held in Reset and are not writable
0 = The CMP2 module is enabled

bit 5 **CMP3MD:** CMP3 Module Disable bit
1 = The CMP3 module is disabled; all CMP3 registers are held in Reset and are not writable
0 = The CMP3 module is enabled

bit 4 **USBMD:** USB Module Disable bit
1 = The USB module is disabled; all USB registers are held in Reset and are not writable
0 = The USB module is enabled

bit 3 **IOCMD:** Interrupt-on-Change Module Disable bit
1 = The IOC module is disabled; all IOC registers are held in Reset and are not writable
0 = The IOC module is enabled

bit 2 **LVDMD:** Low Voltage Detect Module Disable bit
1 = The LVD module is disabled; all LVD registers are held in Reset and are not writable
0 = The LVD module is enabled

bit 1 **Unimplemented:** Read as '0'

bit 0 **EMBMD:** EMB Module Disable bit
1 = The EMB module is disabled; all EMB registers are held in Reset and are not writable
0 = The EMB module is enabled

5.0 RESET

The 08KA101 family devices differentiate between various kinds of Reset:

a) Power-on Reset (POR)
b) $\overline{\text{MCLR}}$ Reset
c) Watchdog Timer (WDT) Reset
d) Configuration Mismatch (CM)
e) Brown-out Reset (BOR)
f) RESET Instruction
g) Stack Underflow/Overflow Reset

This section discusses Resets generated by $\overline{\text{MCLR}}$, POR and BOR, and covers the operation of the various start-up timers. For information on WDT Resets, see **Section 28.2 "Watchdog Timer (WDT)"**. For Stack Reset events, see **Section 6.1.4.4 "Stack Full and Underflow Resets"**. For Deep Sleep mode, see **Section 4.4 "Deep Sleep Modes"**.

A simplified block diagram of the On-Chip Reset Circuit is shown in Figure 5-1.

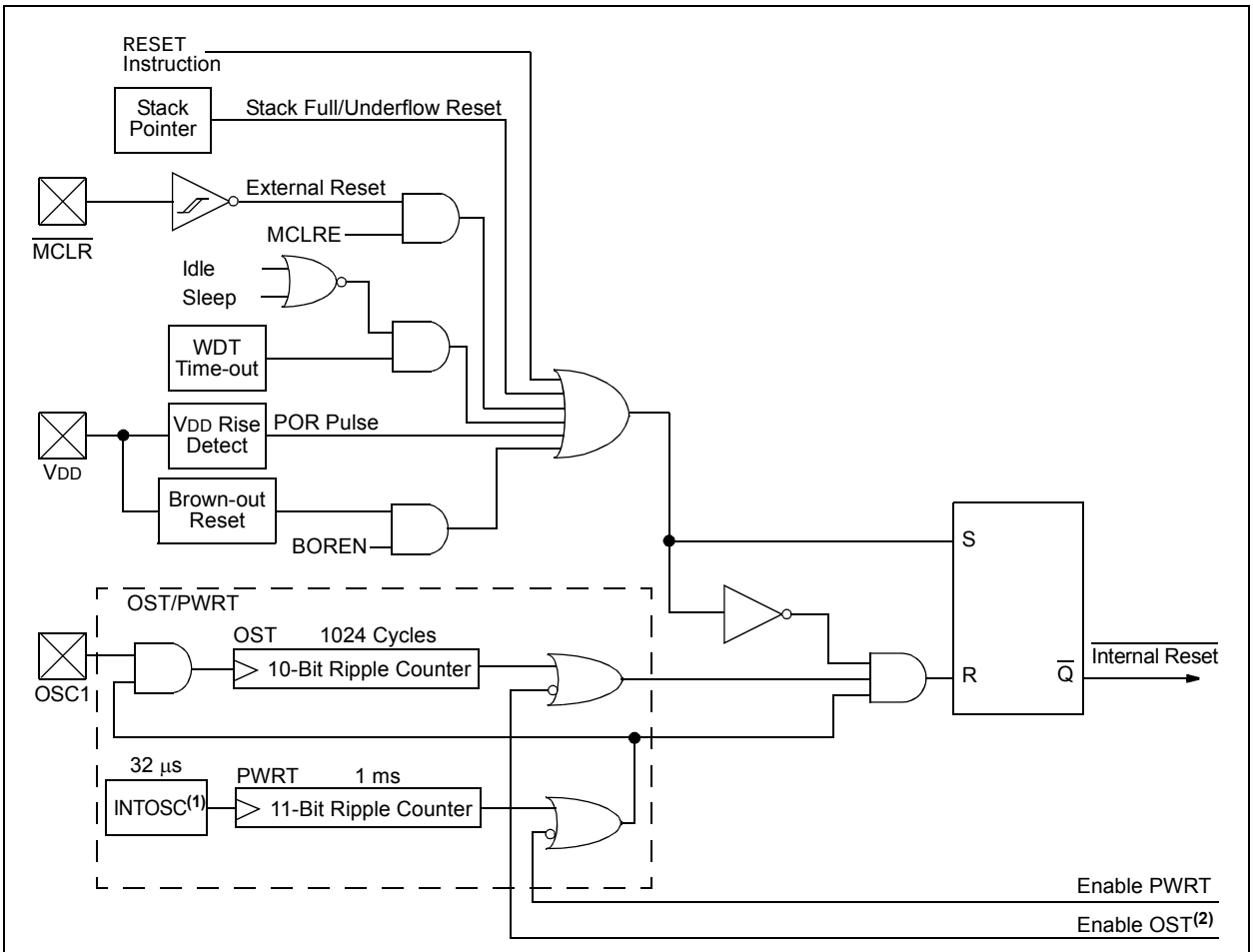
5.1 RCON Registers

Device Reset events are tracked through the RCON, RCON2, RCON3 and RCON4 registers (Register 5-1, Register 5-2, Register 5-3 and Register 5-4). The register bits indicate that a specific Reset event has occurred. Depending on the definition, Status bits may be set or cleared by the event, and re-initialized by the application, after the event to the opposite state. Setting or clearing Reset Status bits does not cause a Reset.

The state of these flag bits, taken together, can be read to indicate the type of Reset that just occurred.

The RCON register also has a control bit for setting interrupt priority (IPEN). Interrupt priority is discussed in **Section 10.0 "Interrupts"**.

FIGURE 5-1: SIMPLIFIED BLOCK DIAGRAM OF ON-CHIP RESET CIRCUIT



Note 1: This is the INTOSC source from the internal oscillator block and is separate from the RC Oscillator of the CLKI pin.
2: See Table 5-1 for time-out situations.

REGISTER 5-1: RCON: RESET CONTROL REGISTER

R/W-0	U-0	R/W-1	R/W-1	R-1	R-1	R/W-0(1)	R/W-0
IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

Legend:	HC = Hardware Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **IPEN:** Interrupt Priority Enable Register bit
1 = Prioritized interrupts are enabled
0 = Prioritized interrupts are disabled

bit 6 **Unimplemented**: Read as '0'

bit 5 **$\overline{CM}$:** Configuration Mismatch Flag bit
1 = A Configuration Mismatch Reset has not occurred
0 = A Configuration Mismatch Reset occurred; must be set in software once the Reset occurs

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed (set by firmware only)
0 = The RESET instruction was executed, causing a device Reset (must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{TO}$:** Watchdog Time-out Flag bit
1 = Set by power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{PD}$:** Power-down Detection Flag bit
1 = Set by power-up or by the CLRWDT instruction
0 = Set by execution of the SLEEP instruction

bit 1 **$\overline{POR}$:** Power-on Reset Status bit(1)
1 = A Power-on Reset has not occurred (set by firmware only)
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
1 = A Brown-out Reset has not occurred (set by firmware only)
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

Note 1: Brown-out Reset is said to have occurred when $\overline{BOR}$ is '0' and $\overline{POR}$ is '1' (assuming that $\overline{POR}$ was set to '1' by software immediately after a Power-on Reset).

REGISTER 5-2: RCON2: RESET CONTROL REGISTER 2

R/W-0, HS	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
EXTR(1)	—	SWDTEN(2)	—	—	—	—	—
bit 7							bit 0

Legend:	HS = Hardware Settable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **EXTR:** External Reset ($\overline{MCLR}$) Pin bit(1)
1 = A Master Clear (pin) Reset has occurred
0 = A Master Clear (pin) Reset has not occurred

bit 6 **Unimplemented**: Read as '0'

bit 5 **SWDTEN:** Software Controlled Watchdog Timer Enable bit(2)
1 = Watchdog Timer is on
0 = Watchdog Timer is off

bit 4-0 **Unimplemented**: Read as '0'

Note 1: This bit is set in hardware; it can be cleared in software.
2: This bit has no effect unless the Configuration bits, WDTEN<1:0> = 10.

REGISTER 5-3: RCON3: RESET CONTROL REGISTER 3

U-0	U-0	U-0	U-0	R/C-0	R/C-0	R/C-0	R/W-0
—	—	—	—	VDDBOR(1)	VDDPOR(1,2)	VBPOR(1,3)	VBAT
bit 7							bit 0

Legend:	C = Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented**: Read as '0'

bit 3 **VDDBOR:** VDD Brown-out Reset Flag bit(1)
- 1 = A VDD Brown-out Reset has occurred
- 0 = A VDD Brown-out Reset has not occurred

bit 2 **VDDPOR:** VDD Power-On Reset Flag bit(1,2)
- 1 = A VDD Power-up Reset has occurred
- 0 = A VDD Power-up Reset has not occurred

bit 1 **VBPOR:** VBPOR Flag bit(1,3)
- 1 = A VBAT POR has occurred
- 0 = A VBAT POR has not occurred

bit 0 **VBAT**: VBAT Flag bit(1)
- 1 = A POR exit has occurred while power was applied to VBAT pin
- 0 = A POR exit from VBAT has not occurred

Note 1: This bit is set in hardware only; it can only be cleared in software.

2: Indicates a VDD POR. Setting the $\overline{\text{POR}}$ bit (RCON<0>) indicates a VCORE POR.

3: This bit is set when the device is originally powered up, even if power is present on VBAT.

REGISTER 5-4: RCON4: RESET CONTROL REGISTER 4

U-0	U-0	U-0	R/W-0	U-0	R/C-0	U-0	R/W-0
—	—	—	SRETEN(1)	—	DPSLP(2)	—	PMSLP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented**: Read as '0'

bit 4 **SRETEN:** Retention Regulator Voltage Sleep Disable bit(1)
- 1 = If $\overline{\text{RETEN}}$ (CONFIG7L<0>) = 0 and the regulator is enabled, the device goes into Retention mode in Sleep
- 0 = The regulator is on when device's Sleep mode is enabled and the Low-Power mode is controlled by the PMSLP bit

bit 3 **Unimplemented**: Read as '0'

bit 2 **DPSLP:** Deep Sleep Wake-up Status bit (used in conjunction with the $\overline{\text{POR}}$ and $\overline{\text{BOR}}$ bits in RCON to determine the Reset source)(2)
- 1 = The last exit from Reset was caused by a normal wake-up from Deep Sleep
- 0 = The last exit from Reset was not due to a wake-up from Deep Sleep

bit 1 **Unimplemented**: Read as '0'

bit 0 **PMSLP:** Program Memory Power During Sleep bit
- 1 = Program memory bias voltage remains powered during Sleep
- 0 = Program memory bias voltage is powered down during Sleep

Note 1: This bit is available only when $\overline{\text{RETEN}}$ (CONFIG7L<0>) = 0.

2: This bit is set in hardware only; it can only be cleared in software.

5.2 Power-on Reset (POR)

The PIC18F97J94 family has two types of Power-on Resets:

- POR
- VBAT POR

POR is the legacy PIC18J series Power-on Reset which monitors core power supply. The second, VBAT POR, monitors voltage on the VBAT pin. These POR circuits use the same technique to enable and monitor their respective power source for adequate voltage levels to ensure proper chip operation. There are two threshold voltages associated with them. The first voltage is the device threshold voltage, VPOR. The device threshold voltage is the voltage at which the POR module becomes operable. The second voltage associated with a POR event is the POR circuit threshold voltage. Once the correct threshold voltage is detected, a power-on event occurs and the POR module hibernates to minimize current consumption.

A power-on event generates an internal POR pulse when a VDD rise is detected. The device supply voltage characteristics must meet the specified starting voltage, VPOR, and rise rate requirements, SVDD, to generate the POR pulse. In particular, VDD must fall below VPOR before a new POR is initiated. For more information on the VPOR and VDD rise rate specifications, refer to **Section 30.0 "Electrical Specifications"**.

5.2.1 POR CIRCUIT

The POR circuit behaves differently than VBAT POR once the POR state becomes active. The internal POR pulse resets the POR timer and places the device in the Reset state. The POR also selects the device clock source identified by the Oscillator Configuration bits. After the POR pulse is generated, the POR circuit inserts a small delay, TCSD, to ensure that internal device bias circuits are stable.

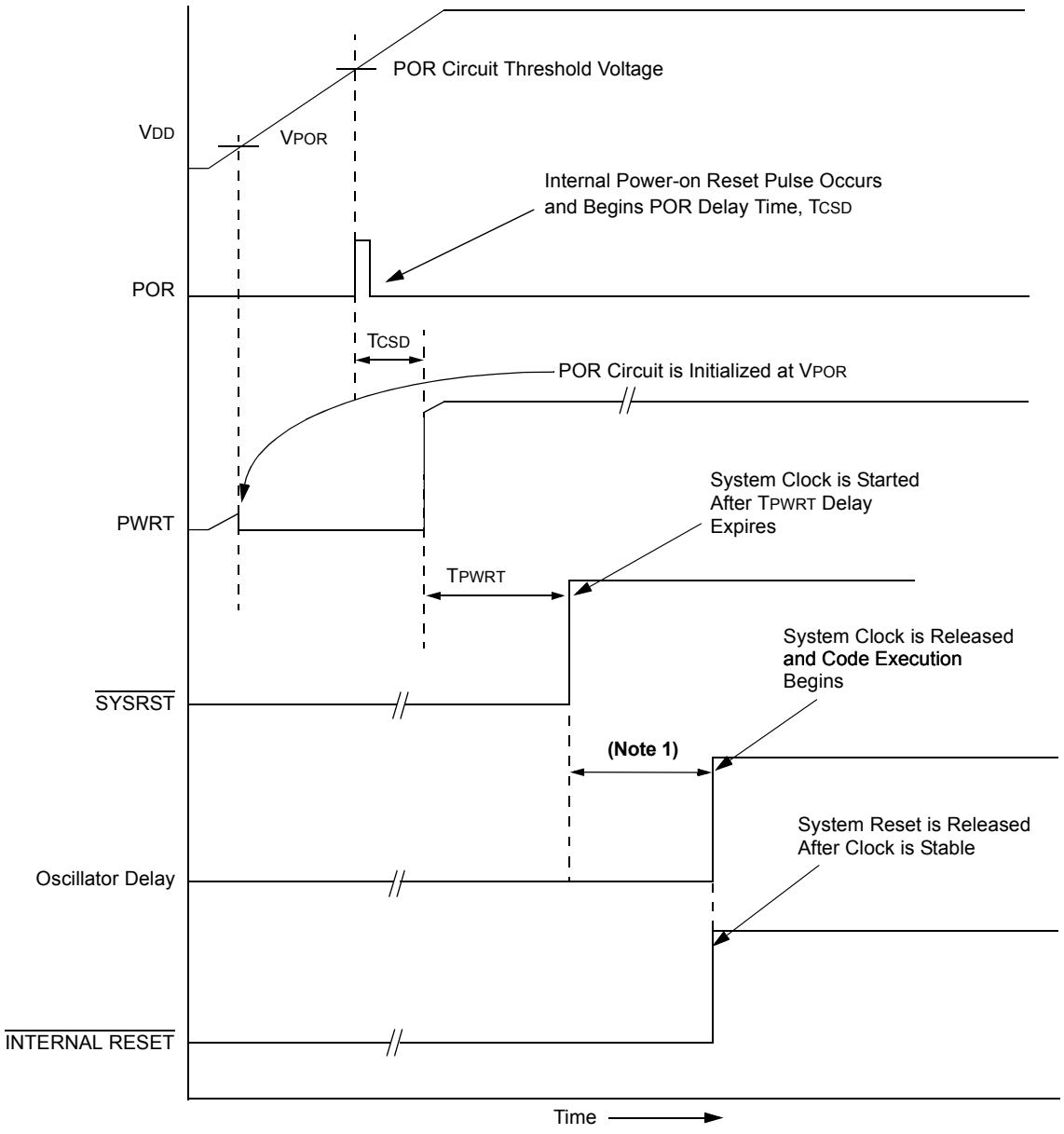
After the expiration of TCSD, a delay, TPWRT, is always inserted every time the device resumes operation after any power-down. During this time, code execution is disabled. The PWRT is used to extend the duration of a power-up sequence to permit the on-chip band gap and regulator to stabilize and to load the Configuration Word settings. The on-chip regulator is always enabled and its stabilization time is shorter than other concurrently running delays, and does not extend start-up time.

The power-on event clears the $\overline{\text{BOR}}$ and $\overline{\text{POR}}$ Status bits (RCON<1:0>); it does not change for any other Reset event. $\overline{\text{POR}}$ is not reset to '1' by any hardware event. To capture multiple events, the user manually resets the bit to '1' in software following any Power-on Reset. Alternatively, the VDDPOR (RCON3<2>) bit can be used; it is set on a VDD POR event. It must be cleared after any Power-on Reset to detect subsequent VDD POR events.

After TPWRT expires, an additional start-up time for the system clock (either TOST, TIOBST and TRC, depending on the source) occurs while the clock source becomes stable. Internal Reset is then released and the device is no longer held in Reset (Table 5-2). Once all of the delays have expired, the system clock is released and code execution can begin. Refer to **Section 30.0 "Electrical Specifications"** for more information on the values of the delay parameters.

Note:	When the device exits the Reset condition (begins normal operation), the device operating parameters (voltage, frequency, temperature, etc.) must be within their operating ranges; otherwise, the device will not function correctly. The user must ensure that the delay between the time power is first applied, and the time, INTERNAL RESET, becomes inactive, is long enough to get all operating parameters within specification.

FIGURE 5-2: POR MODULE TIMING SEQUENCE FOR RISING VDD



Note 1: Timer and interval are determined by the initial start-up oscillator configuration; TOSC is for external oscillator modes, TFRC is for the FRC Oscillator or TLPRC for the internal 31 kHz RC Oscillator.

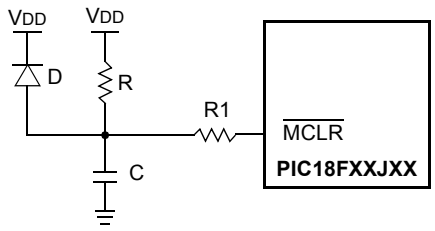
5.2.1.1 Using the POR Circuit

To take advantage of the POR circuit, tie the MCLR pin directly to VDD. This will eliminate external RC components usually needed to create a POR delay. A minimum rise time for VDD is required. Refer to the **"Electrical Characteristics"** section of the specific device data sheet for more information.

Depending on the application, a resistor may be required between the MCLR pin and VDD. This resistor can be used to decouple the MCLR pin from a noisy power supply rail.

Figure 5-3 displays a possible POR circuit for a slow power supply ramp up. The external POR circuit is only required if the device would exit Reset before the device VDD is in the valid operating range. The diode, D, helps discharge the capacitor quickly when VDD powers down.

FIGURE 5-3: EXTERNAL POWER-ON RESET CIRCUIT (FOR SLOW VDD POWER-UP)



Note 1: External Power-on Reset circuit is required only if the VDD power-up slope is too slow. The diode, D, helps discharge the capacitor quickly when VDD powers down.

2: R < 40 kΩ is recommended to make sure that the voltage drop across R does not violate the device's electrical specification.

3: R1 ≥ 1 kΩ will limit any current flowing into MCLR from external capacitor, C, in the event of MCLR/VPP pin breakdown, due to Electrostatic Discharge (ESD) or Electrical Overstress (EOS).

5.2.2 VBAT POWER-ON-RESET (VBPOR)

The device will remain in VBAT mode as long as no power is present on VDD. The VBPOR is active when the device is operating in VBAT mode and deriving power from the VBAT pin. Similar to the POR, the circuit monitors VBAT voltage and holds the device in Reset until adequate voltage is present to power up the device. After exiting the VBAT POR condition, the VBPOR (RCON3<1>) bit is set. All other registers will be in a POR state, including Deep Sleep semaphores. Minimum VBAT ramp time and rearm voltage requirements apply. Refer to Parameters D003 and D004 in **Section 30.0 "Electrical Specifications"** for details.

The device does not execute code in VBAT mode. Also, there is no Power-up Timer associated with VBPOR.

After VDD power is restored, the device exits VBAT mode and the VBAT (RCON3<0>) bit is set. All other registers, except those associated with RTCC, its clock source and the Deep Sleep semaphores (DSGPRx), will be in a POR state. For more information about VBAT mode, see **Section 4.5 "Vbat Mode"**.

5.3 Master Clear Reset (MCLR)

Whenever the MCLR pin is driven low, the device asynchronously asserts SYSRST, provided the input pulse on MCLR is longer than a certain minimum width, TMCL (see **Section 30.0 "Electrical Specifications"**). When the MCLR pin is released, SYSRST is also released. The Reset vector fetch starts from the SYSRST release. The processor continues to use the existing clock source that was in use before the MCLR Reset occurred. The EXTR Status bit (RCON2<7>) is set to indicate the MCLR Reset.

5.4 Watchdog Timer Reset (WDT)

Whenever a Watchdog Timer time-out occurs, the device asynchronously asserts SYSRST. The clock source remains unchanged. Note that a WDT time-out during Sleep or Idle mode will wake-up the processor, but NOT reset the processor. The TO bit (RCON<3>) is cleared when a WDT time-out occurs. Software must set this bit to initialize the flag. For more information, refer to **Section 28.2 "Watchdog Timer (WDT)"**.

Note: The WDT described here is not the same one used in Deep Sleep mode. For more information on Deep Sleep WDT, see **Section 28.2 "Watchdog Timer (WDT)"**.

5.5 Configuration Mismatch Reset (CM)

The Configuration Mismatch (CM) Reset is designed to detect, and attempt to recover from, random memory corrupting events. These include Electrostatic Discharge (ESD) events, which can cause widespread, single bit changes throughout the device and result in catastrophic failure.

In PIC18FXXJXX Flash devices, device Configuration registers (located in the configuration memory space) are continuously monitored during operation by comparing their values to complimentary shadow registers. If a mismatch is detected between the two sets of registers, a CM Reset automatically occurs. These events are captured by the $\overline{CM}$ bit (RCON<5>) being set to '0'.

This bit does not change for any other Reset event. A CM Reset behaves similarly to a Master Clear Reset, RESET instruction, WDT Time-out Reset or Stack Event Reset. As with all hard and power Reset events, the device's Configuration Words are reloaded from the Flash Configuration Words in program memory as the device restarts.

5.6 Brown-out Reset (BOR) Features

The PIC97J94 family has four different types of BOR circuits:

- Brown-out Reset (BOR)
- V_{DDCORE} Brown-out Reset (VDDBOR)
- V_{BAT} Brown-out Reset (VBATBOR)
- Deep Sleep Brown-out Reset (DSBOR)

All four BOR circuits monitor a voltage and put the device in a Reset condition while the voltage is in a specified region. SFRs will reset to the BOR state, including the Deep Sleep semaphore holding registers, DSGPR0 and DSGPR1. Upon BOR exit, the device remains in Reset until the associated trip point voltage is exceeded. Any I/O pins configured as outputs will be tri-stated. BOR, VDDBOR and DSBOR exit into Run mode; VBATBOR remains in V_{BAT} mode.

These features differ by their power mode, monitored voltage source, trip points, control and status. Refer to Table 5-1 for the PIC18F97J94 BOR differences.

TABLE 5-1: BOR FEATURE SUMMARY(1)

Feature	Mode	Source	Trip Points	Enable
BOR	Run, Idle, Sleep	V_{DDCORE}	1.6V (typ)	Always Enabled
VDDBOR	Run, Idle, Sleep	V_{DD}	V_{VDDBOR}	BOREN (CONFIG1H<0>)
VBATBOR	V_{BAT}	V_{BAT}	$V_{VBATBOR}$	VBTBOR (CONFIG7L<2>)
DSBOR	Deep Sleep	V_{DD}	V_{DSBOR}	DSBOREN (CONFIG7L<3>)

Note 1: Refer to Table for details.

5.6.1 BROWN-OUT RESET (BOR)

Brown-out Reset is the legacy PIC18 “J” feature that monitors the core voltage, VDDCORE. Since the regulator on the PIC18F97J94 family is always enabled, this feature is always active. Its trip point is non-configurable. A Brown-out Reset will occur as the regulator output voltage drops below, approximately 1.6V. After proper operating voltage recovers, the Brown-out Reset condition is exited and execution begins after the Power-up Timer has expired. The BOR (RCON<0>) bit is also cleared. This bit must be set after each Brown-out and Power-on Reset event to detect subsequent Brown-out Reset events.

Note:	Brown-out Reset (BOR) has been provided to support legacy devices that can disable their internal regulator. The PIC18F97J94 family’s regulator is always enabled. Therefore, it’s recommended that new designs use VDDBOR to detect Brown-out conditions.

5.6.2 VDD BOR (VDDBOR)

VDDBOR is enabled by setting the BOREN (CONFIG1H<0>) Configuration bit. The low-power BOR trip level is configurable to either 1.8V or 2.0V, (typ) depending on the BORV (CONFIG1H<1>) Configuration bit setting. When in normal Run mode, Idle or normal Sleep modes, the BOR circuit that monitors VDD is active and will cause the device to be held in BOR if VDD drops below VBOR. Once VDD rises back above VVDDBOR, the device will be held in Reset until the expiration of the Power-up Timer, with period, TPWRT. This event is captured by the VDDBOR flag bit (RCON3<3>).

5.6.3 DETECTING VDD BOR

When the BOR module is enabled, the VDDBOR (RCON3<3>) bit is set on a Brown-out Reset event. This makes it difficult to determine if a Brown-out Reset event has occurred just by reading the state of VDDBOR alone. A more reliable method is to simultaneously check the state of both VDDPOR and VDDBOR. This assumes that the VDDPOR bit is reset to ‘1’ in software immediately after any Power-on Reset event. If VDDBOR is ‘0’ while VDDPOR is ‘1’, it can be reliably assumed that a Brown-out Reset event has occurred. Legacy PIC18 software can use the respective POR (RCON<1>) and BOR (RCON<0>) bits. This technique monitors the regulator output voltage, VDDCORE. To take advantage of the configuration features, it is recommended to use VDDBOR instead of BOR.

5.6.4 VBAT BROWN-OUT RESET (VBATBOR)

The VBAT BOR can be enabled/disabled using the VBTBOR bit in the Configuration register (CONFIG7L<2>). If the VBTBOR enable bit is cleared, the VBATBOR is always disabled and there will be no indication of a VBAT BOR. If the VBTBOR bit is set, the VBAT POR will reset the device when the battery voltage drops below VVBATBOR. After power is restored to the VBAT pin, the device exits Reset and returns to VBAT mode. The device remains in VBAT mode until power returns to the VDD pin. For more information on using the VBAT feature, refer to **Section 4.5 “Vbat Mode”**.

5.6.5 DEEP SLEEP BROWN-OUT RESET (DSBOR)

The PIC18F97J94 has its dedicated BOR for Deep Sleep mode (DSBOR). It is enabled through the DSBOREN (CONFIG7L<3>) Configuration bit. When the device enters Deep Sleep mode and receives a DSBOR event, the device will not wake-up and will remain in Deep Sleep mode. When a valid wake-up event occurs and causes the device to exit Deep Sleep mode, software can determine if a DSBOR event occurred during Deep Sleep mode by reading the DSBOR (DSCONL<1>) Status bit.

5.7 RESET Instruction

Whenever the RESET instruction is executed, the device asserts SYSRST. This Reset state does not re-initialize the clock. The clock source that is in effect prior to the RESET instruction remains in effect. Configuration settings are updated and the SYSRST is released at the next instruction cycle. A noise filter in the MCLR Reset path detects and ignores small pulses. The RI bit (RCON<4>) is cleared when a RESET instruction is executed. Software must set this bit to initialize the flag.

5.8 Stack Underflow/Overflow Reset

A Reset can be enabled on stack error conditions by setting the STVREN (CONFIG1L<5>) Configuration bit. See **Section 6.1.4.4 “Stack Full and Underflow Resets”**section for additional information.

5.9 Device Reset Timers

PIC18F97J94 family devices incorporate three separate on-chip timers that help regulate the Power-on Reset process. Their main function is to ensure that the device clock is stable before code is executed. These timers are:

- Power-up Timer (PWRT)
- Oscillator Start-up Timer (OST)
- PLL Lock Time-out

5.9.1 POWER-UP TIMER (PWRT)

The Power-up Timer (PWRT) of the PIC18F97J94 family devices is a counter which uses the INTOSC source as the clock input. While the PWRT is counting, the device is held in Reset. The power-up time delay depends on the INTOSC clock and varies slightly from chip-to-chip due to temperature and process variation. See the TPWRT specification for details. The PWRT is always enabled and active after Brown-out and Power-on Reset events.

5.9.2 OSCILLATOR START-UP TIMER (OST)

The Oscillator Start-up Timer (OST) provides a 1024 oscillator cycle (from OSC1 input) delay after the PWRT delay is over. This ensures that the crystal oscillator or resonator has started and stabilized.

The OST time-out is invoked only for LP, MS, HS and HSPLL modes, and only on Power-on Reset or on exit from most power-managed modes.

5.9.3 PLL LOCK TIME-OUT

The PLL is enabled by programming FOSC<2:0> = 011 (CONFIG2L<2:0>. With the PLL enabled, the time-out sequence, following a Power-on Reset, is slightly different from other oscillator modes. A separate timer is used to provide a fixed time-out that is sufficient for the PLL to lock to the main oscillator frequency. This PLL lock time-out (TRC) follows the oscillator start-up time-out.

5.9.4 RESET STATE OF REGISTERS

Most registers are unaffected by a Reset. Their status is unknown on a Power-on Reset and unchanged by all other Resets. The other registers are forced to a “Reset state” depending on the type of Reset that occurred.

Most registers are not affected by a WDT wake-up, since this is viewed as the resumption of normal operation. Status bits from the RCONx registers are set or cleared differently in different Reset situations, as indicated in Table 5-2. These bits are used in software to determine the nature of the Reset.

Table 5-2 describes the Reset states for all of the Special Function Registers. These are categorized by Power-on and Brown-out Resets, Master Clear and WDT Resets, and WDT wake-ups.

TABLE 5-2: RCONx BIT OPERATION ON VARIOUS RESETS AND WAKE-UPS

Conditions	PC	DPSLP	EXTR	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	IDLE	$\overline{CM}$	$\overline{BOR}$	$\overline{POR}$	VDDBOR	VDDPOR	VBPOR(4,6)	VBAT(4)
DSPOR:(4) Loss of VDDBAT	000000	0	0	0	0	1	0	0	1	1	1	1	1	0
VBAT:(4) Loss of VDD While VBAT is Established	000000	1	0	0	0	1	0	0	1	1	1	1	u	1
VDD POR: Loss of VDD	000000	0	0	0	0	1	0	0	1	1	1	1	u	u
VDD BOR: Brown-out of VDD	000000	u	u	0	0	1	0	0	u	u	1	u	u	u
POR: Loss of VDDCORE	000000	0	0	0	0	1	0	0	1	1	u	u	u	u
BOR Brown-out of VDDCORE	000000	u	u	0	0	1	0	0	1	u	u	u	u	u
Deep Sleep Exit	000000	1	0	0	0	1	0	0	1	1	u	u	u	u
Retention Deep Sleep Exit	000000	1	0	0	0	1	0	0	0	0	u	u	u	u
$\overline{MCLR}$ Reset Operational Mode	000000	u	1	u	u	u	u	u	u	u	u	u	u	u
$\overline{MCLR}$ Reset in Idle Mode	000000	u	1	u	0(1)	0(2)	1(2)	u	u	u	u	u	u	u
$\overline{MCLR}$ Reset in Sleep Mode	000000	u	1	u	0(1)	0(2)	0(2)	u	u	u	u	u	u	u
RESET Instruction Reset	000000	u	u	1	u	u	u	u	u	u	u	u	u	u
Configuration Mismatch Reset	000000	u	u	u	u	u	u	1	u	u	u	u	u	u
WDT Reset	000000	u	u	u	1	u	u	u	u	u	u	u	u	u
WDT Reset in Idle Mode	PC + 2	u	u	u	1	1(2)	1(2)	u	u	u	u	u	u	u
WDT Reset in Sleep Mode	PC + 2	u	u	u	1	0(2)	0(2)	u	u	u	u	u	u	u
Interrupt in Idle Mode with GIE = 0	PC + 2	u	u	u	0(1)	1(2)	1(2)	u	u	u	u	u	u	u
Interrupt in Idle Mode with GIE = 1	Vector	u	u	u	0(1)	1(2)	1(2)	u	u	u	u	u	u	u
Interrupt in Sleep Mode With GIE = 0	PC + 2	u	u	u	0(1)	0(2)	0(2)	u	u	u	u	u	u	u
Interrupt in Sleep Mode with GIE = 1	Vector	u	u	u	0(1)	0(2)	0(2)	u	u	u	u	u	u	u
CLRWDT Instruction	PC + 2	u	u	u	0(3)	1	u	u	u	u	u	u	u	u
IDLE Instruction	PC + 2	u	u	u	0	1	1	u	u	u	u	u	u	u
SLEEP Instruction	PC + 2	u	u	u	0	0	0	u	u	u	u	u	u	u
User Instruction Writes '1'	PC + 2	u	1	1	1	0	1	1	1	1	1	1	1	1
User Instruction Writes '0'	PC + 2	0	0	0	0	1	0	0	0	0	0	0	0	0

Note 1: The SLEEP instruction clears the WDTO bit.

2: The CLRWDT clears the WDTO bit only when the WDT window feature is disabled or the WDT is in the safe window.

3: This bit is also set, flagging the loss of state retention even though the true POR condition has not occurred.

4: This bit is set in hardware only; it can only be cleared in software.

5: Indicates a VDD POR. Setting the $\overline{POR}$ bit (RCON<0>) indicates a VCORE POR.

6: This bit is set when the device is originally powered up, even if power is present on VBAT.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
TOSU	64-pin	80-pin	100-pin	---0 0000	---0 0000	---0 uuuu(1)
TOSH	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu(1)
TOSL	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu(1)
STKPTR	64-pin	80-pin	100-pin	00-0 0000	uu-0 0000	uu-u uuuu(1)
PCLATU	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
PCLATH	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PCL	64-pin	80-pin	100-pin	0000 0000	0000 0000	PC + 2(2)
TBLPTRU	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
TBLPTRH	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TBLPTRL	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TABLAT	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PRODH	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
PRODL	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
INTCON	64-pin	80-pin	100-pin	0000 000x	0000 000x	uuuu uuuu(3)
INTCON2	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu(3)
INTCON3	64-pin	80-pin	100-pin	1100 0000	1100 0000	uuuu uuuu(3)
INDF0	64-pin	80-pin	100-pin	N/A	N/A	N/A
POSTINC0	64-pin	80-pin	100-pin	N/A	N/A	N/A
POSTDEC0	64-pin	80-pin	100-pin	N/A	N/A	N/A
PREINC0	64-pin	80-pin	100-pin	N/A	N/A	N/A
PLUSW0	64-pin	80-pin	100-pin	N/A	N/A	N/A
FSR0H	64-pin	80-pin	100-pin	---- xxxx	---- uuuu	---- uuuu
FSR0L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
WREG	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
INDF1	64-pin	80-pin	100-pin	N/A	N/A	N/A
POSTINC1	64-pin	80-pin	100-pin	N/A	N/A	N/A
POSTDEC1	64-pin	80-pin	100-pin	N/A	N/A	N/A
PREINC1	64-pin	80-pin	100-pin	N/A	N/A	N/A
PLUSW1	64-pin	80-pin	100-pin	N/A	N/A	N/A
FSR1H	64-pin	80-pin	100-pin	---- xxxx	---- uuuu	---- uuuu
FSR1L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
BSR	64-pin	80-pin	100-pin	---- 0000	---- 0000	---- uuuu
INDF2	64-pin	80-pin	100-pin	N/A	N/A	N/A

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
POSTINC2	64-pin	80-pin	100-pin	N/A	N/A	N/A
POSTDEC2	64-pin	80-pin	100-pin	N/A	N/A	N/A
PREINC2	64-pin	80-pin	100-pin	N/A	N/A	N/A
PLUSW2	64-pin	80-pin	100-pin	N/A	N/A	N/A
FSR2H	64-pin	80-pin	100-pin	---- xxxx	---- uuuu	---- uuuu
FSR2L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
STATUS	64-pin	80-pin	100-pin	---x xxxx	---u uuuu	---u uuuu
TMR0H	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
TMR0L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
T0CON	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RESERVED	64-pin	80-pin	100-pin	---- ----	---- ----	---- ----
OSCCON	64-pin	80-pin	100-pin	0qqq -qqq	uuuu -uuu	uuuu -uuu
IPR5	64-pin	80-pin	100-pin	-111 -111	-uuu -uuu	-uuu -uuu
IOCF	64-pin	80-pin	100-pin	0000 0000	0000 0000	qqqq qqqq
RCON(4)	64-pin	80-pin	100-pin	0-11 11qq	0-qq qquu	u-qq qquu
TMR1H	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR1L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
T1CON	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
TMR2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PR2	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
T2CON	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu
SSP1BUF	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
SSP1ADD	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SSP1STAT	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SSP1CON1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SSP1CON2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CMSTAT	64-pin	80-pin	100-pin	---- -xxx	---- -uuu	---- -uuu
ADCBUF0H	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCBUF0L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
ADCON1H	64-pin	80-pin	100-pin	0--- -000	u--- -uuu	u--- -uuu
ADCON1L	64-pin	80-pin	100-pin	0000 -000	uuuu -uuu	uuuu -uuu
CVRCONH	64-pin	80-pin	100-pin	---0 0000	---u uuuu	---u uuuu
CVRCONL	64-pin	80-pin	100-pin	0000 ---0	uuuu ---u	uuuu ---u

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the VBAT is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
ECCP1AS	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ECCP1DEL	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
CCPR1H	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCPR1L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
CCP1CON	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PIR5	64-pin	80-pin	100-pin	-000 -0000	-000 -000	-uuu -uuu(3)
PIE5	64-pin	80-pin	100-pin	-000 -000	-000 -000	-uuu -uuu
IPR4	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
PIR4	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu(3)
PIE4	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TMR3H	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR3L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
T3CON	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
T3GCON	64-pin	80-pin	100-pin	0000 0x00	0000 00x0	uuuu uuuu
SPBRG1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RCREG1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TXREG1	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXSTA1	64-pin	80-pin	100-pin	0000 0010	0000 0010	uuuu uuuu
RCSTA1	64-pin	80-pin	100-pin	0000 000x	0000 000x	uuuu uuuu
T1GCON	64-pin	80-pin	100-pin	0000 0x00	0000 0x00	uuuu uuuu
IPR6	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
HLVDCON	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PSPCON	64-pin	80-pin	100-pin	0000 ----	0000 ----	uuuu ----
PIR6	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu(3)
IPR3	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
PIR3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu(3)
PIE3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
IPR2	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
PIR2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu(3)
PIE2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
IPR1	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
PIR1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu(3)
PIE1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
PSTR1CON	64-pin	80-pin	100-pin	00-0 0001	00-0 0001	uu-u uuuu
OSCTUNE	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
TRISJ	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TRISH	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TRISG(5)	64-pin	80-pin	100-pin	11-1 1111	11-1 1111	uu-u uuuu
TRISF	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TRISE	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TRISD	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TRISC	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TRISB	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TRISA	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
LATJ	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATH	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATG(5)	64-pin	80-pin	100-pin	xx-x xxxx	uu-u uuuu	uu-u uuuu
LATF	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATE	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATD	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATC	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATB	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
LATA	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTJ	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTH	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTG(5)	64-pin	80-pin	100-pin	xx-x x-xx	xx-x x-xx	uu-u u-uu
PORTF	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTE	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTD	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTC	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTB	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTA	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
EECON1	64-pin	80-pin	100-pin	xx-0 x000	uu-0 u000	uu-u uuuu
EECON2	64-pin	80-pin	100-pin	---- ----	---- ----	---- ----
RCON2	64-pin	80-pin	100-pin	0-0- 0---	q-u- 0---	0-u- 1---
RCON3	64-pin	80-pin	100-pin	---0 q000	---u 0000	---u 0000

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the VBAT is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
RCON4	64-pin	80-pin	100-pin	00-0 -0-0	00-u -0-u	00-u -0-u
UFRML	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
UFRMH	64-pin	80-pin	100-pin	---- -xxx	---- -xxx	---- -uuu
UIR	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu
UEIR	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu
USTAT	64-pin	80-pin	100-pin	0--0 0000	0--0 0000	u--u uuuu
UCON	64-pin	80-pin	100-pin	-0x0 000-	-0x0 000-	-uuu uuu-
UADDR	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu
TRISVP	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
LATVP	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
PORTVP	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXADDRL	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TXADDRH	64-pin	80-pin	100-pin	---- 0000	---- 0000	---- uuuu
RXADDRL	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RXADDRH	64-pin	80-pin	100-pin	---- 0000	---- 0000	---- uuuu
DMABCL	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
DMABCH	64-pin	80-pin	100-pin	---- --00	---- --00	---- --uu
TXBUF	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SSP1CON3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SSP1MSK	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
BAUDCON1	64-pin	80-pin	100-pin	0100 0000	0100 0000	uuuu uuuu
OSCCON2	64-pin	80-pin	100-pin	000- 000-	00q- 000-	uuu- uuu-
OSCCON3	64-pin	80-pin	100-pin	---- -001	---- -uuu	---- -uuu
OSCCON4	64-pin	80-pin	100-pin	000- ----	uuu- ----	uuu- ----
OSCCON5	64-pin	80-pin	100-pin	0-00 0000	u-uu uuuu	u-uu uuuu
WPUB	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
PIE6	64-pin	80-pin	100-pin	0000 -000	0000 -000	uuuu -uuu
DMACON1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RTCCON1	64-pin	80-pin	100-pin	0-00 0000	u-uu uuuu	u-uu uuuu
RTCCAL	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
RTCVALH	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
RTCVALL	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
ALRMCFG	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
ALRMRPT	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
ALRMVALH	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
ALRMVALL	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
RTCCON2	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
IOCP	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
IOCN	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PADCFG1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CM1CON	64-pin	80-pin	100-pin	0001 1111	0001 1111	uuuu uuuu
ECCP2AS	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ECCP2DEL	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CCPR2H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR2L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ECCP2CON	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ECCP3AS	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ECCP3DEL	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CCPR3H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR3L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ECCP3CON	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CCPR8H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR8L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP8CON	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
CCPR9H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR9L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP9CON	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
CCPR10H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR10L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP10CON	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
TMR6	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PR6	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
T6CON	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu
TMR8	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PR8	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
T8CON	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
SSP2CON3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CM2CON	64-pin	80-pin	100-pin	0001 1111	0001 1111	uuuu uuuu
CM3CON	64-pin	80-pin	100-pin	0001 1111	0001 1111	uuuu uuuu
CCPTMRS0	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CCPTMRS1	64-pin	80-pin	100-pin	00-0 -000	00-0 -000	uuuu uuuu
CCPTMRS2	64-pin	80-pin	100-pin	---0 -000	---0 -000	uuuu uuuu
RCSTA2	64-pin	80-pin	100-pin	0000 000x	0000 000x	uuuu uuuu
TXSTA2	64-pin	80-pin	100-pin	0000 0010	0000 0010	uuuu uuuu
BAUDCON2	64-pin	80-pin	100-pin	01x0 0000	01x0 0000	uuuu uuuu
SPBRGH1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RCSTA3	64-pin	80-pin	100-pin	0000 000x	0000 000x	uuuu uuuu
TXSTA3	64-pin	80-pin	100-pin	0000 0010	0000 0010	uuuu uuuu
BAUDCON3	64-pin	80-pin	100-pin	01x0 0000	01x0 0000	uuuu uuuu
SPBRGH3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SPBRG3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RCREG3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TXREG3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
DSCONL	64-pin	80-pin	100-pin	---- -000	---- -000	--- -uuu
DSCONH	64-pin	80-pin	100-pin	0-0- ---0	u-u- ---u	u-u- ---u
DSWAKEL	64-pin	80-pin	100-pin	0000 0001	uuuu uuuu	uuuu uuuu
DSWAKEH	64-pin	80-pin	100-pin	---- ---0	---- ---u	---- ---q
DSGPR0(6)	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
DSGPR1(6)	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
DSGPR2(6)	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
DSGPR3	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
SPBRGH2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SPBRG2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RCREG2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TXREG2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PSTR2CON	64-pin	80-pin	100-pin	00-0 0001	00-0 0001	uu-u uuuu
PSTR3CON	64-pin	80-pin	100-pin	00-0 0001	00-0 0001	uu-u uuuu
SSP2STAT	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SSP2CON1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
SSP2CON2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SSP2MSK	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
TMR5H	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
TMR5L	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
T5CON	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
T5GCON	64-pin	80-pin	100-pin	0000 0x00	0000 00x0	uuuu uuuu
CCPR4H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR4L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP4CON	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
CCPR5H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR5L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP5CON	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
CCPR6H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR6L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP6CON	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
CCPR7H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCPR7L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
CCP7CON	64-pin	80-pin	100-pin	--00 0000	--00 0000	--uu uuuu
TMR4	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
PR4	64-pin	80-pin	100-pin	1111 1111	uuuu uuuu	uuuu uuuu
T4CON	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu
SSP2BUF	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
SSP2ADD	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ANCFG	64-pin	80-pin	100-pin	---- -000	---- -000	---- -uuu
DMACON2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RCSTA4	64-pin	80-pin	100-pin	0000 000x	0000 000x	uuuu uuuu
TXSTA4	64-pin	80-pin	100-pin	0000 0010	0000 0010	uuuu uuuu
BAUDCON4	64-pin	80-pin	100-pin	01x0 0000	01x0 0000	uuuu uuuu
SPBRGH4	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
SPBRG4	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RCREG4	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TXREG4	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CTMUCON	64-pin	80-pin	100-pin	0-00 0000	0-00 0000	u-uu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the VBAT is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
CTMUCON1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
CTMUCON2	64-pin	80-pin	100-pin	0000 00--	0000 00--	uuuu uu--
CTMUCON3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PMD0	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PMD1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PMD2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PMD3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
PMD4	64-pin	80-pin	100-pin	0000 00--	0000 00--	uuuu uu--
MDCON	64-pin	80-pin	100-pin	0010 0--0	0010 0--0	uuuu u--u
MDSRC	64-pin	80-pin	100-pin	0--- xxxx	0--- uuuu	u--- uuuu
MDCARH	64-pin	80-pin	100-pin	0xx- xxxx	0uu- uuuu	uuu- uuuu
MDCARL	64-pin	80-pin	100-pin	0xx- xxxx	0uu- uuuu	uuu- uuuu
ODCON1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ODCON2	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
TRISK	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
LATK	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTK	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
TRISL	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
LATL	64-pin	80-pin	100-pin	xxxx xxxx	uuuu uuuu	uuuu uuuu
PORTL	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
MEMCON	64-pin	80-pin	100-pin	0-00 --00	0-00 --00	u-uu --uu
REFO1CON	64-pin	80-pin	100-pin	0-00 0-00	u-uu u-uu	u-uu u-uu
REFO1CON1	64-pin	80-pin	100-pin	---- 0000	---- uuuu	---- uuuu
REFO1CON2	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
REFO1CON3	64-pin	80-pin	100-pin	-000 0000	-uuu uuuu	-uuu uuuu
REFO2CON	64-pin	80-pin	100-pin	0-00 0-00	u-uu u-uu	u-uu u-uu
REFO2CON1	64-pin	80-pin	100-pin	---- 0000	---- uuuu	---- uuuu
REFO2CON2	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
REFO2CON3	64-pin	80-pin	100-pin	-000 0000	-uuu uuuu	-uuu uuuu
LCDPS	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDREG	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDCON	64-pin	80-pin	100-pin	0000 0000	0000 0000	u-uu uuuu
LCDREF	64-pin	80-pin	100-pin	0-00 0000	u-uu uuuu	u-uu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
LCDREFL	64-pin	80-pin	100-pin	0000 -000	uuuu -uuu	uuuu -uuu
LCDSE7	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDSE6	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDSE5	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDSE4	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDSE3	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDSE2	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDSE1	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDSE0	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA63	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA62	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA61	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA60	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA59	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA58	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA57	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA56	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA55	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA54	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA53	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA52	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA51	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA50	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA49	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA48	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA47	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA46	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA45	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA44	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA43	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA42	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA41	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA40	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the VBAT is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
LCDDATA39	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA38	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA37	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA36	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA35	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA34	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA33	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA32	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA31	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA30	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA29	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA28	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA27	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA26	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA25	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA24	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA23	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA22	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA21	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA20	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA19	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA18	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA17	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA16	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA15	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA14	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA13	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA12	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA11	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA10	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA9	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA8	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA7	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
LCDDATA6	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA5	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA4	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA3	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA2	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA1	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
LCDDATA0	64-pin	80-pin	100-pin	0000 0000	uuuu uuuu	uuuu uuuu
ADCON2H	64-pin	80-pin	100-pin	0000 00--	0000 00--	uuuu uu--
ADCON2L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCON3H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCON3L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCON5H	64-pin	80-pin	100-pin	000- --00	000- --00	uuu- --uu
ADCON5L	64-pin	80-pin	100-pin	---- 0000	---- 0000	---- uuuu
ADCHS0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHS0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCSS1H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCSS1L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCSS0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCSS0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHIT1H	64-pin	80-pin	100-pin	---- --00	---- --00	---- --uu
ADCHIT1L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHIT0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCHIT0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCTMUEN1H	64-pin	80-pin	100-pin	-000 0000	-000 0000	uuuu uuuu
ADCTMUEN1L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCTMUEN0H	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCTMUEN0L	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
ADCBUF25H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF25L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF24H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF24L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF23H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF23L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the VBAT is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
ADCBUF22H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF22L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF21H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF21L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF20H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF20L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF19H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF19L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF18H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF18L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF17H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF17L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF16H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF16L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF15H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF15L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF14H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF14L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF13H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF13L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF12H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF12L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF11H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF11L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF10H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF10L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF9H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF9L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF8H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF8L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF7H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF7L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF6H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
ADCBUF6L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF5H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF5L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF4H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF4L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF3H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF3L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF2H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF2L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF1H	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ADCBUF1L	64-pin	80-pin	100-pin	xxxx xxxx	xxxx xxxx	uuuu uuuu
ANCON1	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
ANCON2	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
ANCON3	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR52_53	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR50_51	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR48_49	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR46_47	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR44_45	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR42_43	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR40_41	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR38_39	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR36_37	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR34_35	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR32_33	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR30_31	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR28_29	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR26_27	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR24_25	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR22_23	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR20_21	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR18_19	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR16_17	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the VBAT is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	MCLR Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
RPINR14_15	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR12_13	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR10_11	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR8_9	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR6_7	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR4_5	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR2_3	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPINR0_1	64-pin	80-pin	100-pin	1111 1111	1111 1111	uuuu uuuu
RPOR46	64-pin	80-pin	100-pin	---- 0000	---- 0000	---- uuuu
RPOR44_45	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR42_43	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR40_41	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR38_39	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR36_37	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR34_35	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR32_33	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR30_31	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR28_29	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR26_27	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR24_25	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR22_23	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR20_21	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR18_19	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR16_17	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR14_15	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR12_13	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR10_11	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR8_9	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR6_7	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR4_5	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR2_3	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
RPOR0_1	64-pin	80-pin	100-pin	0000 0000	0000 0000	uuuu uuuu
UCFG	64-pin	80-pin	100-pin	00-0 -000	00-0 -000	uu-u -uuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

TABLE 5-3: INITIALIZATION CONDITIONS FOR ALL REGISTERS (CONTINUED)

Register	Applicable Devices			Power-on Reset, Brown-out Reset	$\overline{\text{MCLR}}$ Resets, WDT Reset, RESET Instruction, Stack Resets	Wake-up via WDT or Interrupt
UIE	64-pin	80-pin	100-pin	-000 0000	-000 0000	-uuu uuuu
UEIE	64-pin	80-pin	100-pin	0--0 0000	0--0 0000	u--u uuuu
UEP0	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP1	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP2	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP3	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP4	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP5	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP6	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP7	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP8	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP9	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP10	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP11	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP12	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP13	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP14	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu
UEP15	64-pin	80-pin	100-pin	---0 0000	---0 0000	---u uuuu

Legend: u = unchanged; x = unknown; - = unimplemented bit, read as '0'; q = value depends on condition. Shaded cells indicate that conditions do not apply for the designated device.

Note 1: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the TOSU, TOSH and TOSL are updated with the current value of the PC. The STKPTR is modified to point to the next location in the hardware stack.

2: When the wake-up is due to an interrupt and the GIEL or GIEH bit is set, the PC is loaded with the interrupt vector (0008h or 0018h).

3: One or more bits in the INTCONx or PIRx registers will be affected (to cause wake-up).

4: See Table 5-2 for Reset value for specific condition.

5: Bits 7,6 are unimplemented on 64 and 80-pin devices.

6: If the V_{BAT} is always powered, the DSGPx register values will remain unchanged after the first POR.

6.0 MEMORY ORGANIZATION

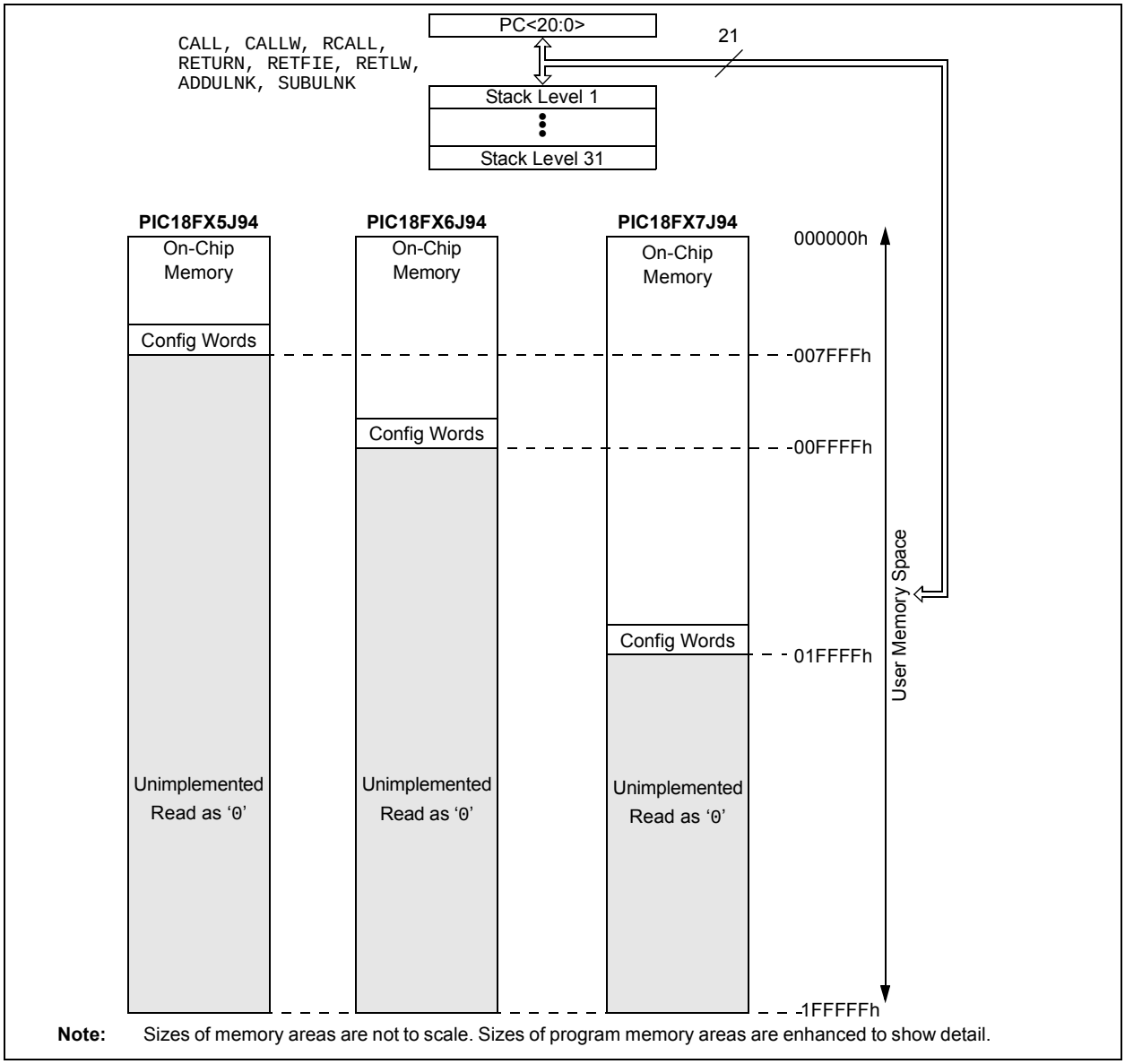
PIC18FXXJ94 devices have these types of memory:

- Program Memory
- Data RAM

As Harvard architecture devices, the data and program memories use separate buses. This enables concurrent access of the two memory spaces.

Additional detailed information on the operation of the Flash program memory is provided in **Section 7.0 "Flash Program Memory"**.

FIGURE 6-1: MEMORY MAPS FOR PIC18F97J94 FAMILY DEVICES



Note: Sizes of memory areas are not to scale. Sizes of program memory areas are enhanced to show detail.

6.1 Program Memory Organization

PIC18 microcontrollers implement a 21-bit Program Counter that is capable of addressing a 2-Mbyte program memory space. Accessing a location between the upper boundary of the physically implemented memory and the 2-Mbyte address will return all '0's (a NOP instruction).

The entire PIC18FXXJ94 offers a range of on-chip Flash program memory sizes, from 32 Kbytes (up to 16,384 single-word instructions) to 128 Kbytes (65,536 single-word instructions).

- PIC18F65J94, PIC18F85J94 and PIC18F95J94 – 32 Kbytes of Flash memory, storing up to 16,384 single-word instructions
- PIC18F66J94, PIC18F86J94 and PIC18F96J94 – 64 Kbytes of Flash memory, storing up to 32,768 single-word instructions
- PIC18F67J94, PIC18F87J94 and PIC18F97J94 – 128 Kbytes of Flash memory, storing up to 65,536 single-word instructions

The program memory maps for individual family members are shown in Figure 6-1.

6.1.1 HARD MEMORY VECTORS

All PIC18 devices have a total of three hard-coded return vectors in their program memory space. The Reset vector address is the default value to which the Program Counter returns on all device Resets; it is located at 0000h.

PIC18 devices also have two interrupt vector addresses for handling high-priority and low-priority interrupts. The high-priority interrupt vector is located at 0008h and the low-priority interrupt vector is at 0018h. The locations of these vectors are shown, in relation to the program memory map, in Figure 6-2.

6.1.2 FLASH CONFIGURATION WORDS

Because PIC18FXXJ94 devices do not have persistent configuration memory, the top eight words of on-chip program memory are reserved for configuration information. On Reset, the configuration information is copied into the Configuration registers.

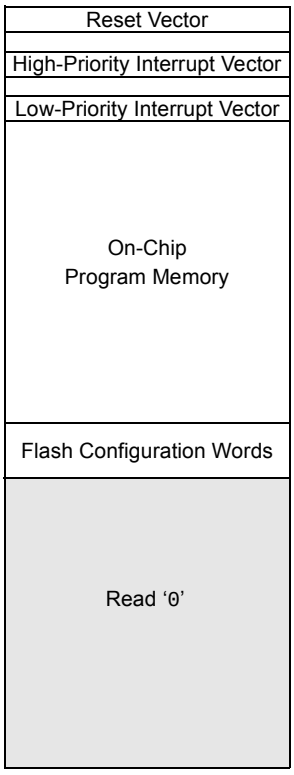
The Configuration Words are stored in their program memory location in numerical order, starting with the lower byte of CONFIG1 at the lowest address and ending with the upper byte of CONFIG8. The actual addresses of the Flash Configuration Word for devices in the PIC18FXXJ94 are shown in Table 6-1.

Their location in the memory map is shown with the other memory vectors in Figure 6-2. Additional details on the device Configuration Words are provided in **Section 28.1 "Configuration Bits"**.

TABLE 6-1: FLASH CONFIGURATION WORD FOR PIC18FXXJ94 FAMILY DEVICES

Device	Program Memory (Kbytes)	Configuration Word Addresses
PIC18F65J94 PIC18F85J94 PIC18F95J94	32	7FF0h to 7FFFh
PIC18F66J94 PIC18F86J94 PIC18F96J94	64	FFF0h to FFFFh
PIC18F67J94 PIC18F87J94 PIC18F97J94	128	1FFF0h to 1FFFFh

FIGURE 6-2: HARD VECTOR FOR PIC18F97J94 FAMILY DEVICES



Legend: (Top of Memory) represents upper boundary of on-chip program memory space (see Figure 6-1 for device-specific values). Shaded area represents unimplemented memory. Areas are not shown to scale.

6.1.3 PROGRAM COUNTER

The Program Counter (PC) specifies the address of the instruction to fetch for execution. The PC is 21 bits wide and contained in three separate 8-bit registers.

The low byte, known as the PCL register, is both readable and writable. The high byte, or PCH register, contains the PC<15:8> bits and is not directly readable or writable. Updates to the PCH register are performed through the PCLATH register. The upper byte is called PCU. This register contains the PC<20:16> bits; it is also not directly readable or writable. Updates to the PCU register are performed through the PCLATU register.

The contents of PCLATH and PCLATU are transferred to the Program Counter by any operation that writes PCL. Similarly, the upper two bytes of the Program Counter are transferred to PCLATH and PCLATU by an operation that reads PCL. This is useful for computed offsets to the PC (see **Section 6.1.6.1 "Computed GOTO"**).

The PC addresses bytes in the program memory. To prevent the PC from becoming misaligned with word instructions, the Least Significant bit of PCL is fixed to a value of '0'. The PC increments by two to address sequential instructions in the program memory.

The CALL, RCALL, GOTO and program branch instructions write to the Program Counter directly. For these instructions, the contents of PCLATH and PCLATU are not transferred to the Program Counter.

6.1.4 RETURN ADDRESS STACK

The return address stack enables execution of any combination of up to 31 program calls and interrupts. The PC is pushed onto the stack when a CALL or RCALL instruction is executed or an interrupt is Acknowledged. The PC value is pulled off the stack on a RETURN, RETLW or a RETFIE instruction. The value also is pulled off the stack on ADDULNK and SUBULNK instructions, if the extended instruction set is enabled. PCLATU and PCLATH are not affected by any of the RETURN or CALL instructions.

The stack operates as a 31-word by 21-bit RAM and a 5-bit Stack Pointer, STKPTR. The stack space is not part of either program or data space. The Stack Pointer is readable and writable and the address on the top of the stack is readable and writable through the Top-of-Stack Special Function Registers. Data can also be pushed to, or popped from, the stack using these registers.

A CALL type instruction causes a push onto the stack. The Stack Pointer is first incremented and the location pointed to by the Stack Pointer is written with the contents of the PC (already pointing to the instruction following the CALL). A RETURN type instruction causes a pop from the stack. The contents of the location pointed to by the STKPTR are transferred to the PC and then the Stack Pointer is decremented.

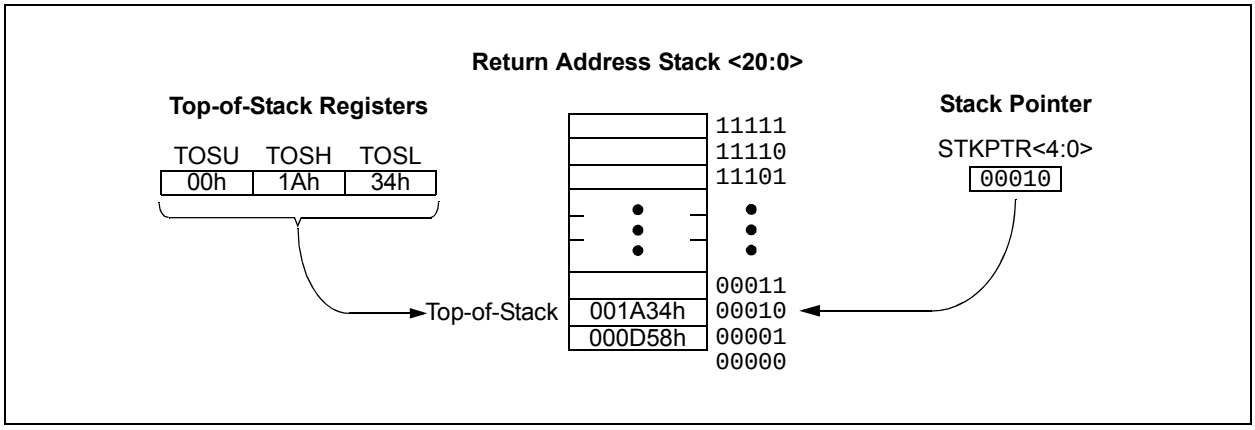
The Stack Pointer is initialized to '00000' after all Resets. There is no RAM associated with the location corresponding to a Stack Pointer value of '00000'; this is only a Reset value. Status bits indicate if the stack is full, has overflowed or has underflowed.

6.1.4.1 Top-of-Stack Access

Only the top of the return address stack (TOS) is readable and writable. A set of three registers, TOSU:TOSH:TOSL, holds the contents of the stack location pointed to by the STKPTR register (Figure 6-3). This allows users to implement a software stack, if necessary. After a CALL, RCALL or interrupt (or ADDULNK and SUBULNK instructions, if the extended instruction set is enabled), the software can read the pushed value by reading the TOSU:TOSH:TOSL registers. These values can be placed on a user-defined software stack. At return time, the software can return these values to TOSU:TOSH:TOSL and do a return.

While accessing the stack, users must disable the Global Interrupt Enable bits to prevent inadvertent stack corruption.

FIGURE 6-3: RETURN ADDRESS STACK AND ASSOCIATED REGISTERS



6.1.4.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 6-1) contains the Stack Pointer value, the STKFUL (Stack Full) Status bit and the STKUNF (Stack Underflow) Status bits. The value of the Stack Pointer can be 0 through 31. The Stack Pointer increments before values are pushed onto the stack and decrements after values are popped off the stack. On Reset, the Stack Pointer value will be zero.

The user may read and write the Stack Pointer value. This feature can be used by a Real-Time Operating System (RTOS) for return-stack maintenance.

After the PC is pushed onto the stack, 31 times (without popping any values off the stack), the STKFUL bit is set. The STKFUL bit is cleared by software or by a POR.

What happens when the stack becomes full depends on the state of the STVREN (Stack Overflow Reset Enable) Configuration bit. (For a description of the device Configuration bits, see **Section 28.1 "Configuration Bits"**.) If STVREN is set (default), the 31st push will push the (PC + 2) value onto the stack, set the STKFUL bit and reset the device. The STKFUL bit will remain set and the Stack Pointer will be set to zero.

If STVREN is cleared, the STKFUL bit will be set on the 31st push and the Stack Pointer will increment to 31. Any additional pushes will not overwrite the 31st push and the STKPTR will remain at 31.

When the stack has been popped enough times to unload the stack, the next pop will return a value of zero to the PC and set the STKUNF bit, while the Stack Pointer remains at zero. The STKUNF bit will remain set until cleared by software or until a POR occurs.

Note:	Returning a value of zero to the PC on an underflow has the effect of vectoring the program to the Reset vector, where the stack conditions can be verified and appropriate actions can be taken. This is not the same as a Reset, as the contents of the SFRs are not affected.

6.1.4.3 PUSH and POP Instructions

Since the Top-of-Stack is readable and writable, the ability to push values onto the stack and pull values off the stack, without disturbing normal program execution, is a desirable feature. The PIC18 instruction set includes two instructions, PUSH and POP, that permit the TOS to be manipulated under software control. TOSU, TOSH and TOSL can be modified to place data or a return address on the stack.

The PUSH instruction places the current PC value onto the stack. This increments the Stack Pointer and loads the current PC value onto the stack.

The POP instruction discards the current TOS by decrementing the Stack Pointer. The previous value pushed onto the stack then becomes the TOS value.

REGISTER 6-1: STKPTR: STACK POINTER REGISTER

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL(1)	STKUNF(1)	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

Legend:	C = Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **STKFUL:** Stack Full Flag bit(1)
1 = Stack has become full or overflowed
0 = Stack has not become full or overflowed

bit 6 **STKUNF:** Stack Underflow Flag bit(1)
1 = Stack underflow has occurred
0 = Stack underflow did not occur

bit 5 **Unimplemented:** Read as '0'

bit 4-0 **SP<4:0>:** Stack Pointer Location bits

Note 1: Bit 7 and bit 6 are cleared by user software or by a POR.

6.1.4.4 Stack Full and Underflow Resets

Device Resets on stack overflow and stack underflow conditions are enabled by setting the STVREN bit (CONFIG1L<5>). When STVREN is set, a full or underflow condition will set the appropriate STKFUL or STKUNF bit and then cause a device Reset. When STVREN is cleared, a full or underflow condition will set the appropriate STKFUL or STKUNF bit, but not cause a device Reset. The STKFUL or STKUNF bits are cleared by user software or a Power-on Reset.

6.1.5 FAST REGISTER STACK

A Fast Register Stack is provided for the STATUS, WREG and BSR registers to provide a “fast return” option for interrupts. This stack is only one level deep and is neither readable nor writable. It is loaded with the current value of the corresponding register when the processor vectors for an interrupt. All interrupt sources will push values into the Stack registers. The values in the registers are then loaded back into the working registers if the RETFIE, FAST instruction is used to return from the interrupt.

If both low and high-priority interrupts are enabled, the Stack registers cannot be used reliably to return from low-priority interrupts. If a high-priority interrupt occurs while servicing a low-priority interrupt, the Stack register values stored by the low-priority interrupt will be overwritten. In these cases, users must save the key registers in software during a low-priority interrupt.

If interrupt priority is not used, all interrupts may use the Fast Register Stack for returns from interrupt. If no interrupts are used, the Fast Register Stack can be used to restore the STATUS, WREG and BSR registers at the end of a subroutine call. To use the Fast Register Stack for a subroutine call, a CALL label, FAST instruction must be executed to save the STATUS, WREG and BSR registers to the Fast Register Stack. A RETURN, FAST instruction is then executed to restore these registers from the Fast Register Stack.

Example 6-1 shows a source code example that uses the Fast Register Stack during a subroutine call and return.

EXAMPLE 6-1: FAST REGISTER STACK CODE EXAMPLE

```
CALL SUB1, FAST    ;STATUS, WREG, BSR
                   ;SAVED IN FAST REGISTER
                   ;STACK
       •
       •

SUB1   •
       •
    RETURN FAST    ;RESTORE VALUES SAVED
                   ;IN FAST REGISTER STACK
```

6.1.6 LOOK-UP TABLES IN PROGRAM MEMORY

There may be programming situations that require the creation of data structures, or look-up tables, in program memory. For PIC18 devices, look-up tables can be implemented in two ways:

- Computed GOTO
- Table Reads

6.1.6.1 Computed GOTO

A computed GOTO is accomplished by adding an offset to the Program Counter. An example is shown in Example 6-2.

A look-up table can be formed with an ADDWF PCL instruction and a group of RETLW nn instructions. The W register is loaded with an offset into the table before executing a call to that table. The first instruction of the called routine is the ADDWF PCL instruction. The next instruction executed will be one of the RETLW nn instructions that returns the value, ‘nn’, to the calling function.

The offset value (in WREG) specifies the number of bytes that the Program Counter should advance and should be multiples of two (LSb = 0).

In this method, only one data byte may be stored in each instruction location and room on the return address stack is required.

EXAMPLE 6-2: COMPUTED GOTO USING AN OFFSET VALUE

```
        MOVF   OFFSET, W
        CALL   TABLE
ORG     nn00h
TABLE   ADDWF  PCL
        RETLW  nnh
        RETLW  nnh
        RETLW  nnh
        .
        .
        .
```

6.1.6.2 Table Reads

A better method of storing data in program memory allows two bytes of data to be stored in each instruction location.

Look-up table data may be stored, two bytes per program word, while programming. The Table Pointer (TBLPTR) specifies the byte address and the Table Latch (TABLAT) contains the data that is read from the program memory. Data is transferred from program memory one byte at a time.

The table read operation is discussed further in **Section 7.1 “Table Reads and Table Writes”**.

6.2 PIC18 Instruction Cycle

6.2.1 CLOCKING SCHEME

The microcontroller clock input, whether from an internal or external source, is internally divided by four to generate four non-overlapping quadrature clocks (Q1, Q2, Q3 and Q4). Internally, the Program Counter is incremented on every Q1, with the instruction fetched from the program memory and latched into the Instruction Register (IR) during Q4.

The instruction is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow are shown in Figure 6-4.

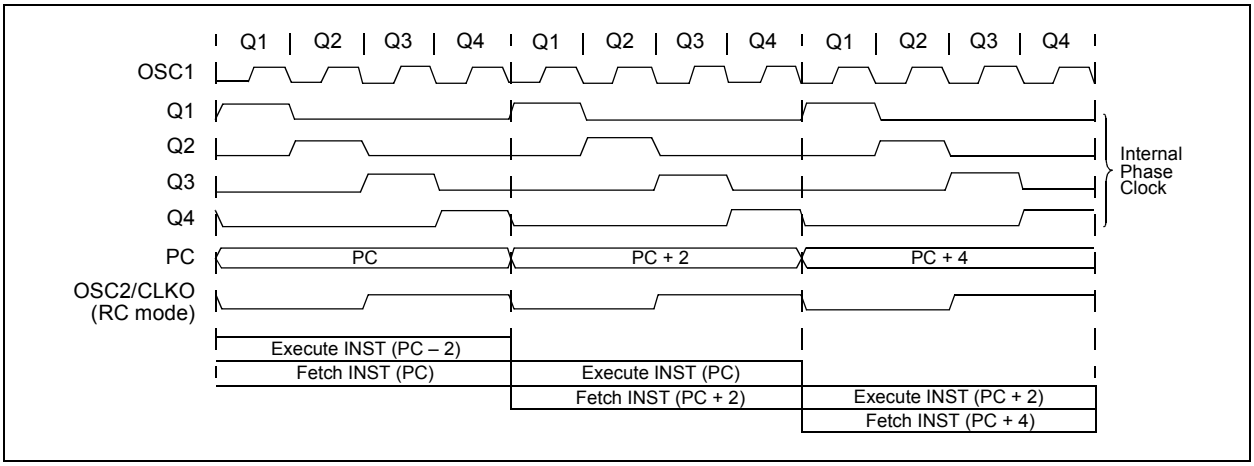
6.2.2 INSTRUCTION FLOW/PIPELINING

An “Instruction Cycle” consists of four Q cycles, Q1 through Q4. The instruction fetch and execute are pipelined in such a manner that a fetch takes one instruction cycle, while the decode and execute take another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction (such as `GOTO`) causes the Program Counter to change, two cycles are required to complete the instruction. (See Example 6-3.)

A fetch cycle begins with the Program Counter (PC) incrementing in Q1.

In the execution cycle, the fetched instruction is latched into the Instruction Register (IR) in cycle Q1. This instruction is then decoded and executed during the Q2, Q3 and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

FIGURE 6-4: CLOCK/INSTRUCTION CYCLE



EXAMPLE 6-3: INSTRUCTION PIPELINE FLOW

	TCY0	TCY1	TCY2	TCY3	TCY4	TCY5
1. MOVLW 55h	Fetch 1	Execute 1				
2. MOVWF PORTB		Fetch 2	Execute 2			
3. BRA SUB_1			Fetch 3	Execute 3		
4. BSF PORTA, BIT3 (Forced NOP)				Fetch 4	Flush (NOP)	
5. Instruction @ address SUB_1					Fetch SUB_1	Execute SUB_1

All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is “flushed” from the pipeline while the new instruction is being fetched and then executed.

6.2.3 INSTRUCTIONS IN PROGRAM MEMORY

The program memory is addressed in bytes. Instructions are stored as two or four bytes in program memory. The Least Significant Byte of an instruction word is always stored in a program memory location with an even address (LSB = 0). To maintain alignment with instruction boundaries, the PC increments in steps of two and the LSB will always read '0' (see **Section 6.1.3 "Program Counter"**).

Figure 6-5 shows an example of how instruction words are stored in the program memory.

The CALL and GOTO instructions have the absolute program memory address embedded into the instruction. Since instructions are always stored on word boundaries, the data contained in the instruction is a word address. The word address is written to PC<20:1> which accesses the desired byte address in program memory. Instruction #2 in Figure 6-5 shows how the instruction, GOTO 0006h, is encoded in the program memory. Program branch instructions, which encode a relative address offset, operate in the same manner. The offset value stored in a branch instruction represents the number of single-word instructions that the PC will be offset by. For more details on the instruction set, see **Section 29.0 "Instruction Set Summary"**.

FIGURE 6-5: INSTRUCTIONS IN PROGRAM MEMORY

			LSB = 1	LSB = 0	Word Address ↓
Program Memory Byte Locations →					000000h
					000002h
					000004h
					000006h
Instruction 1:	MOVLW	055h	0Fh	55h	000008h
Instruction 2:	GOTO	0006h	EFh	06h	00000Ah
			F0h	00h	00000Ch
Instruction 3:	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

6.2.4 TWO-WORD INSTRUCTIONS

The standard PIC18 instruction set has four, two-word instructions: CALL, MOVFF, GOTO and LSFR. In all cases, the second word of the instructions always has '1111' as its four Most Significant bits. The other 12 bits are literal data, usually a data memory address.

The use of '1111' in the 4 MSbs of an instruction specifies a special form of NOP. If the instruction is executed in proper sequence, immediately after the first word, the data in the second word is accessed and used by the instruction sequence. If the first word is skipped, for some reason, and the second word is executed by itself, a NOP is executed instead. This is necessary for cases when the two-word instruction is preceded by a conditional instruction that changes the PC. Example 6-4 shows how this works.

Note: For information on two-word instructions in the extended instruction set, see **Section 6.5 "Program Memory and the Extended Instruction Set"**.

EXAMPLE 6-4: TWO-WORD INSTRUCTIONS

CASE 1:			
Object Code	**Source Code**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; No, skip this word
1111 0100 0101 0110			; Execute this word as a NOP
0010 0100 0000 0000	ADDWF	REG3	; continue code
CASE 2:			
Object Code	**Source Code**		
0110 0110 0000 0000	TSTFSZ	REG1	; is RAM location 0?
1100 0001 0010 0011	MOVFF	REG1, REG2	; Yes, execute this word
1111 0100 0101 0110			; 2nd word of instruction
0010 0100 0000 0000	ADDWF	REG3	; continue code

6.3 Data Memory Organization

Note: The operation of some aspects of data memory are changed when the PIC18 extended instruction set is enabled. See **Section 6.6 "Data Memory and the Extended Instruction Set"** for more information.

The data memory in PIC18 devices is implemented as static RAM. Each register in the data memory has a 12-bit address, allowing up to 4,096 bytes of data memory. The memory space is divided into as many as 16 banks that contain 256 bytes each. PIC18FXXJ94 devices implement all 16 banks, for a total of 4 Kbytes.

Figure 6-6 and Figure 6-7 show the data memory organization for the devices.

The data memory contains Special Function Registers (SFRs) and General Purpose Registers (GPRs). The SFRs are used for control and status of the controller and peripheral functions, while GPRs are used for data storage and scratchpad operations in the user's application. Any read of an unimplemented location will read as '0's.

The instruction set and architecture allow operations across all banks. The entire data memory may be accessed by Direct, Indirect or Indexed Addressing modes. Addressing modes are discussed later in this section.

To ensure that commonly used registers (select SFRs and select GPRs) can be accessed in a single cycle, PIC18 devices implement an Access Bank. This is a 256-byte memory space that provides fast access to select SFRs and the lower portion of GPR Bank 0 without using the Bank Select Register. For details on the Access RAM, see **Section 6.3.2 "Access Bank"**.

6.3.1 BANK SELECT REGISTER

Large areas of data memory require an efficient addressing scheme to make it possible for rapid access to any address. Ideally, this means that an entire address does not need to be provided for each read or write operation. For PIC18 devices, this is accomplished with a RAM banking scheme. This divides the memory space into 16 contiguous banks of 256 bytes. Depending on the instruction, each location can be addressed directly by its full 12-bit address, or an 8-bit, low-order address and a four-bit Bank Pointer.

Most instructions in the PIC18 instruction set make use of the Bank Pointer, known as the Bank Select Register (BSR). This SFR holds the four Most Significant bits of a location's address. The instruction itself includes the eight Least Significant bits. Only the four lower bits of the BSR are implemented (BSR<3:0>). The upper four bits are unused, always read as '0' and cannot be written to. The BSR can be loaded directly by using the MOVLB instruction.

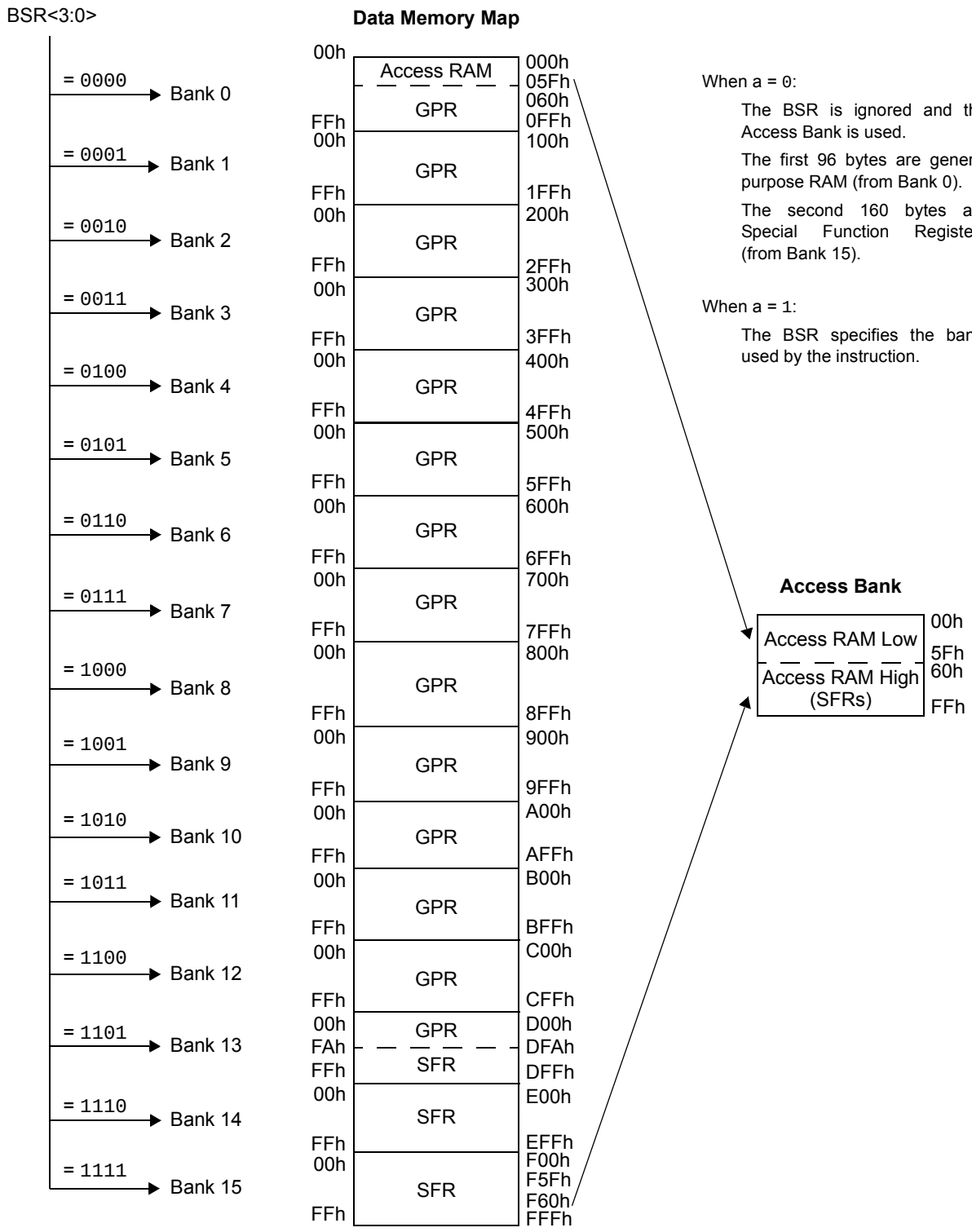
The value of the BSR indicates the bank in data memory. The eight bits in the instruction show the location in the bank and can be thought of as an offset from the bank's lower boundary. The relationship between the BSR's value and the bank division in data memory is shown in Figure 6-7.

Since up to 16 registers may share the same low-order address, the user must always be careful to ensure that the proper bank is selected before performing a data read or write. For example, writing what should be program data to an 8-bit address of F9h, while the BSR is 0Fh, will end up resetting the Program Counter.

While any bank can be selected, only those banks that are actually implemented can be read or written to. Writes to unimplemented banks are ignored, while reads from unimplemented banks will return '0's. Even so, the STATUS register will still be affected as if the operation was successful. The data memory map in Figure 6-6 indicates which banks are implemented.

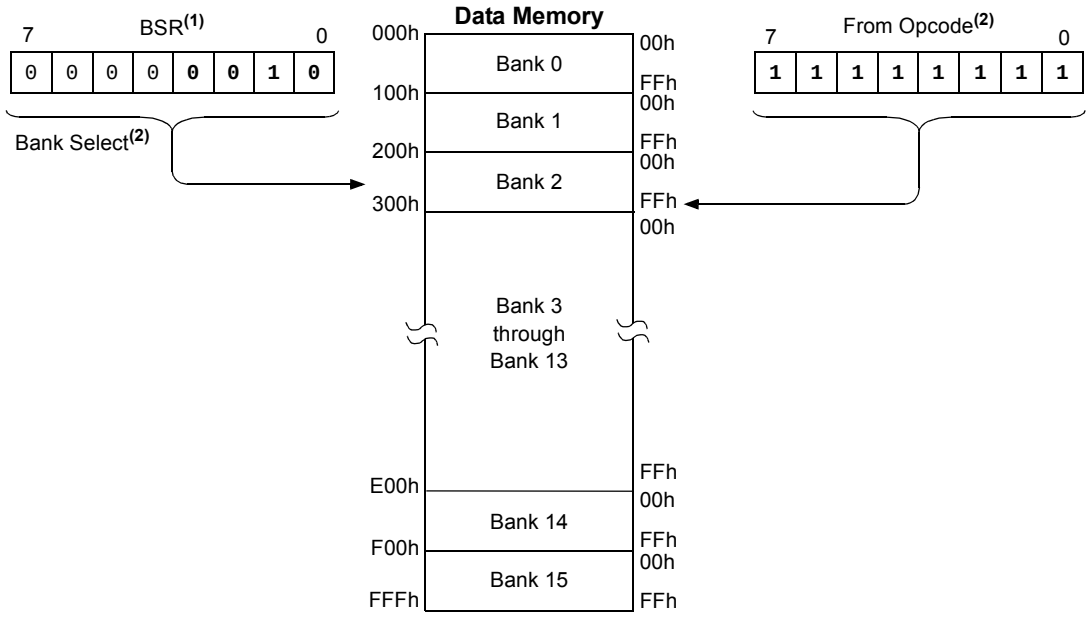
In the core PIC18 instruction set, only the MOVFF instruction fully specifies the 12-bit address of the source and target registers. When this instruction executes, it ignores the BSR completely. All other instructions include only the low-order address as an operand and must use either the BSR or the Access Bank to locate their target registers.

FIGURE 6-6: DATA MEMORY MAP FOR PIC18F97J94 FAMILY DEVICES



Note 1: Addresses, DFAh through F5Fh, are also SFRs, but are not part of the Access RAM. Users must always use the complete address, or load the proper BSR value, to access these registers.

FIGURE 6-7: USE OF THE BANK SELECT REGISTER (DIRECT ADDRESSING)



Note 1: The Access RAM bit of the instruction can be used to force an override of the selected bank (BSR<3:0>) to the registers of the Access Bank.

2: The `MOVFF` instruction embeds the entire 12-bit address in the instruction.

6.3.2 ACCESS BANK

While the use of the BSR, with an embedded 8-bit address, allows users to address the entire range of data memory, it also means that the user must ensure that the correct bank is selected. If not, data may be read from, or written to, the wrong location. This can be disastrous if a GPR is the intended target of an operation, but an SFR is written to instead. Verifying and/or changing the BSR for each read or write to data memory can become very inefficient.

To streamline access for the most commonly used data memory locations, the data memory is configured with an Access Bank, which allows users to access a mapped block of memory without specifying a BSR. The Access Bank consists of the first 96 bytes of memory (00h-5Fh) in Bank 0 and the last 160 bytes of memory (60h-FFh) in Bank 15. The lower half is known as the “Access RAM” and is composed of GPRs. The upper half is where the device’s SFRs are mapped. These two areas are mapped contiguously in the Access Bank and can be addressed in a linear fashion by an 8-bit address (Figure 6-6).

The Access Bank is used by core PIC18 instructions that include the Access RAM bit (the ‘a’ parameter in the instruction). When ‘a’ is equal to ‘1’, the instruction uses the BSR and the 8-bit address included in the opcode for the data memory address. When ‘a’ is ‘0’, however, the instruction is forced to use the Access Bank address map. In that case, the current value of the BSR is ignored entirely.

Using this “forced” addressing allows the instruction to operate on a data address in a single cycle without updating the BSR first. For 8-bit addresses of 60h and above, this means that users can evaluate and operate on SFRs more efficiently. The Access RAM below 60h is a good place for data values that the user might need to access rapidly, such as immediate computational results or common program variables.

Access RAM also allows for faster and more code efficient context saving and switching of variables.

The mapping of the Access Bank is slightly different when the extended instruction set is enabled (XINST Configuration bit = 1). This is discussed in more detail in **Section 6.6.3 “Mapping the Access Bank in Indexed Literal Offset Mode”**.

6.3.3 GENERAL PURPOSE REGISTER FILE

PIC18 devices may have banked memory in the GPR area. This is data RAM which is available for use by all instructions. GPRs start at the bottom of Bank 0 (address 000h) and grow upwards towards the bottom of the SFR area. GPRs are not initialized by a Power-on Reset and are unchanged on all other Resets.

6.3.4 SPECIAL FUNCTION REGISTERS

The Special Function Registers (SFRs) are registers used by the CPU and peripheral modules for controlling the desired operation of the device. These registers are implemented as static RAM. SFRs start at the top of data memory (FFFh) and extend downward to occupy all of Bank 15 (F00h to FFFh), Bank 14 (E00h to EFFh) and part of Bank 13 (DFAh to DFFh).

A list of these registers is given in Table 6-2.

The SFRs can be classified into two sets: those associated with the “core” device functionality (ALU, Resets and interrupts) and those related to the peripheral functions. The Reset and Interrupt registers are described in their respective chapters, while the ALU’s STATUS register is described later in this section. Registers related to the operation of the peripheral features are described in the chapter for that peripheral.

The SFRs are typically distributed among the peripherals whose functions they control. Unused SFR locations are unimplemented and read as ‘0’s.

TABLE 6-2: REGISTER FILE SUMMARY

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FFFh	TOSU	—	—	—	Top-of-Stack Upper Byte (TOS<20:16>)				
FFEh	TOSH	Top-of-Stack High Byte (TOS<15:8>)							
FFDh	TOSL	Top-of-Stack Low Byte (TOS<7:0>)							
FFCh	STKPTR	STKFUL	STKUNF	—	STKPTR				
FFBh	PCLATU	—	—	—	Holding Register for PC<20:16>				
FFAh	PCLATH	Holding Register for PC<15:8>							
FF9h	PCL	PC Low Byte (PC<7:0>)							
FF8h	TBLPTRU	—	—	ACSS	Program Memory Table Pointer Upper Byte (TBLPTR<20:16>)				
FF7h	TBLPTRH	Program Memory Table Pointer High Byte (TBLPTR<15:8>)							
FF6h	TBLPTRL	Program Memory Table Pointer Low Byte (TBLPTR<7:0>)							
FF5h	TABLAT	Program Memory Table Latch							
FF4h	PRODH	Product Register High Byte							
FF3h	PRODL	Product Register Low Byte							
FF2h	INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	IOCIE	TMR0IF	INT0IF	IOCIF
FF1h	INTCON2	$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	IOCIP
FF0h	INTCON3	INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
FEFh	INDF0	Uses contents of FSR0 to address data memory – value of FSR0 not changed (not a physical register)							
FEEh	POSTINC0	Uses contents of FSR0 to address data memory – value of FSR0 post-incremented (not a physical register)							
FEDh	POSTDEC0	Uses contents of FSR0 to address data memory – value of FSR0 post-decremented (not a physical register)							
FECh	PREINC0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register)							
FEBh	PLUSW0	Uses contents of FSR0 to address data memory – value of FSR0 pre-incremented (not a physical register) – value of FSR0 offset by W							
FEAh	FSR0H	—	—	—	—	Indirect Data Memory Address Pointer 0 High			
FE9h	FSR0L	Indirect Data Memory Address Pointer 0 Low Byte							
FE8h	WREG	Working Register							
FE7h	INDF1	Uses contents of FSR1 to address data memory – value of FSR1 not changed (not a physical register)							
FE6h	POSTINC1	Uses contents of FSR1 to address data memory – value of FSR1 post-incremented (not a physical register)							
FE5h	POSTDEC1	Uses contents of FSR1 to address data memory – value of FSR1 post-decremented (not a physical register)							
FE4h	PREINC1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register)							
FE3h	PLUSW1	Uses contents of FSR1 to address data memory – value of FSR1 pre-incremented (not a physical register) – value of FSR1 offset by W							
FE2h	FSR1H	—	—	—	—	Indirect Data Memory Address Pointer 1 High			
FE1h	FSR1L	Indirect Data Memory Address Pointer 1 Low Byte							
FE0h	BSR	—	—	—	—	Bank Select Register			
FDFh	INDF2	Uses contents of FSR2 to address data memory – value of FSR2 not changed (not a physical register)							
FDEh	POSTINC2	Uses contents of FSR2 to address data memory – value of FSR2 post-incremented (not a physical register)							
FDDh	POSTDEC2	Uses contents of FSR2 to address data memory – value of FSR2 post-decremented (not a physical register)							
FDCh	PREINC2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register)							
FDBh	PLUSW2	Uses contents of FSR2 to address data memory – value of FSR2 pre-incremented (not a physical register) – value of FSR2 offset by W							
FDAh	FSR2H	—	—	—	—	Indirect Data Memory Address Pointer 2 High			
FD9h	FSR2L	Indirect Data Memory Address Pointer 2 Low Byte							
FD8h	STATUS	—	—	—	N	OV	Z	DC	C
FD7h	TMR0H	Timer0 Register High Byte							
FD6h	TMR0L	Timer0 Register Low Byte							
FD5h	T0CON	TMR0ON	T08BIT	T0CS1	T0CS0	PSA	T0PS2	T0PS1	T0PS0
FD4h	Unimplemented	—	—	—	—	—	—	—	—
FD3h	OSCCON	IDLEN	COSC2	COSC1	COSC0	—	NOSC2	NOSC1	NOSC0
FD2h	IPR5	—	ACTORSIP	ACTLOCKIP	TMR8IP	—	TMR6IP	TMR5IP	TMR4IP
FD1h	IOCF	IOCF7	IOCF6	IOCF5	IOCF4	IOCF3	IOCF2	IOCF1	IOCF0
FD0h	RCON	IPEN	—	$\overline{\text{CM}}$	$\overline{\text{RI}}$	$\overline{\text{TO}}$	$\overline{\text{PD}}$	$\overline{\text{POR}}$	$\overline{\text{BOR}}$

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FCFh	TMR1H	Timer1 Register High Byte							
FCEh	TMR1L	Timer1 Register Low Byte							
FCDh	T1CON	TMR1CS1	TMR1CS0	T1CKPS1	T1CKPS0	SOSCEN	$\overline{\text{T1SYNC}}$	RD16	TMR1ON
FCCh	TMR2	Timer2 Register							
FCBh	PR2	Timer2 Period Register							
FCAh	T2CON	—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
FC9h	SSP1BUF	MSSP1 Receive Buffer/Transmit Register							
FC8h	SSP1ADD	MSSP1 Address Register in I^2C Slave Mode. MSSP1 Baud Rate Reload Register in I^2C Master Mode.							
FC7h	SSP1STAT	SMP	CKE	D/$\overline{\text{A}}$	P	S	R/$\overline{\text{W}}$	UA	BF
FC6h	SSP1CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
FC5h	SSP1CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
FC4h	CMSTAT	—	—	—	—	—	C3OUT	C2OUT	C1OUT
FC3h	ADCBUF0H	A/D Result Register 0 High Byte							
FC2h	ADCBUF0L	A/D Result Register 0 Low Byte							
FC1h	ADCON1H	ADON	—	—	—	—	MODE12	FORM1	FORM0
FC0h	ADCON1L	SSRC3	SSRC2	SSRC1	SSRC0	—	ASAM	SAMP	DONE
FBFh	CVRCONH	—	—	—	CVR4	CVR3	CVR2	CVR1	CVR0
FBEh	CVRCONL	CVREN	CVROE	CVRPSS1	CVRPSS0	—	—	—	CVRNSS
FBDh	ECCP1AS	ECCP1ASE	ECCP1AS2	ECCP1AS1	ECCP1AS0	PSS1AC1	PSS1AC0	PSS1BD1	PSS1BD0
FBCh	ECCP1DEL	P1RSEN	P1DC6	P1DC5	P1DC4	P1DC3	P1DC2	P1DC1	P1DC0
FBBh	CCPR1H	Capture/Compare/PWM Register1 High Byte							
FBAh	CCPR1L	Capture/Compare/PWM Register1 Low Byte							
FB9h	CCP1CON	P1M1	P1M0	CCP1X	CCP1Y	CCP1M3	CCP1M2	CCP1M1	CCP1M0
FB8h	PIR5	—	ACTORSIF	ACTLOCKIF	TMR8IF	—	TMR6IF	TMR5IF	TMR4IF
FB7h	PIE5	—	ACTORSIE	ACTLOCKIE	TMR8IE	—	TMR6IE	TMR5IE	TMR4IE
FB6h	IPR4	CCP10IP	CCP9IP	CCP8IP	CCP7IP	CCP6IP	CCP5IP	CCP4IP	ECCP3IP
FB5h	PIR4	CCP10IF	CCP9IF	CCP8IF	CCP7IF	CCP6IF	CCP5IF	CCP4IF	ECCP3IF
FB4h	PIE4	CCP10IE	CCP9IE	CCP8IE	CCP7IE	CCP6IE	CCP5IE	CCP4IE	ECCP3IE
FB3h	TMR3H	Timer3 Register High Byte							
FB2h	TMR3L	Timer3 Register Low Byte							
FB1h	T3CON	TMR3CS1	TMR3CS0	T3CKPS1	T3CKPS0	SOSCEN	$\overline{\text{T3SYNC}}$	RD16	TMR3ON
FB0h	T3GCON	TMR3GE	T3GPOL	T3GTM	T3GSPM	T3GGO/$\overline{\text{T3DONE}}$	T3GVAL	T3GSS1	T3GSS0
FAFh	SPBRG1	EUSART1 Baud Rate Generator							
FAEh	RCREG1	EUSART1 Receive Register							
FADh	TXREG1	EUSART1 Transmit Register							
FACh	TXSTA1	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
FABh	RCSTA1	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
FAAh	T1GCON	TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/$\overline{\text{T1DONE}}$	T1GVAL	T1GSS1	T1GSS0
FA9h	IPR6	RC4IP	TX4IP	RC3IP	TX3IP	—	CMP3IP	CMP2IP	CMP1IP
FA8h	HLVDCON	VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3	HLVDL2	HLVDL1	HLVDL0
FA7h	PSPCON	IBF	OBF	IBOV	PSPMODE	—	—	—	—
FA6h	PIR6	RC4IF	TX4IF	RC3IF	TX3IF	—	CMP3IF	CMP2IF	CMP1IF
FA5h	IPR3	TMR5GIP	LCDIP	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
FA4h	PIR3	TMR5GIF	LCDIF	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
FA3h	PIE3	TMR5GIE	LCDIE	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
FA2h	IPR2	OSCFIP	SSP2IP	BCL2IP	USBIP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
FA1h	PIR2	OSCFIF	SSP2IF	BCL2IF	USBIF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
FA0h	PIE2	OSCFIE	SSP2IE	BCL2IE	USBIE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
F9Fh	IPR1	PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
F9Eh	PIR1	PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
F9Dh	PIE1	PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
F9Ch	PSTR1CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA
F9Bh	OSCTUNE	—	—	TUN5	TUN4	TUN3	TUN2	TUN1	TUN0
F9Ah	TRISJ	TRISJ7	TRISJ6	TRISJ5	TRISJ4	TRISJ3	TRISJ2	TRISJ1	TRISJ0
F99h	TRISH	TRISH7	TRISH6	TRISH5	TRISH4	TRISH3	TRISH2	TRISH1	TRISH0
F98h	TRISG	TRISG7	TRISG6	—	TRISG4	TRISG3	TRISG2	TRISG1	TRISG0
F97h	TRISF	TRISF7	TRISF6	TRISF5	—	—	TRISF2	—	—
F96h	TRISE	TRISE7	TRISE6	TRISE5	TRISE4	TRISE3	TRISE2	TRISE1	TRISE0
F95h	TRISD	TRISD7	TRISD6	TRISD5	TRISD4	TRISD3	TRISD2	TRISD1	TRISD0
F94h	TRISC	TRISC7	TRISC6	TRISC5	TRISC4	TRISC3	TRISC2	—	—
F93h	TRISB	TRISB7	TRISB6	TRISB5	TRISB4	TRISB3	TRISB2	TRISB1	TRISB0
F92h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
F91h	LATJ	LATJ7	LATJ6	LATJ5	LATJ4	LATJ3	LATJ2	LATJ1	LATJ0
F90h	LATH	LATH7	LATH6	LATH5	LATH4	LATH3	LATH2	LATH1	LATH0
F8Fh	LATG	LATG7	LATG6	—	LATG4	LATG3	LATG2	LATG1	LATG0
F8Eh	LATF	LATF7	LATF6	LATF5	—	—	LATF2	—	—
F8Dh	LATE	LATE7	LATE6	LATE5	LATE4	LATE3	LATE2	LATE1	LATE0
F8Ch	LATD	LATD7	LATD6	LATD5	LATD4	LATD3	LATD2	LATD1	LATD0
F8Bh	LATC	LATC7	LATC6	LATC5	LATC4	LATC3	LATC2	—	—
F8Ah	LATB	LATB7	LATB6	LATB5	LATB4	LATB3	LATB2	LATB1	LATB0
F89h	LATA	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0
F88h	PORTJ	RJ7	RJ6	RJ5	RJ4	RJ3	RJ2	RJ1	RJ0
F87h	PORTH	RH7	RH6	RH5	RH4	RH3	RH2	RH1	RH0
F86h	PORTG	RG7	RG6	—	RG4	RG3	RG2	RG1	RG0
F85h	PORTF	RF7	RF6	RF5	RF4	RF3	RF2	—	—
F84h	PORTE	RE7	RE6	RE5	RE4	RE3	RE2	RE1	RE0
F83h	PORTD	RD7	RD6	RD5	RD4	RD3	RD2	RD1	RD0
F82h	PORTC	RC7	RC6	RC5	RC4	RC3	RC2	RC1	RC0
F81h	PORTB	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
F80h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
F7Fh	EECON1	—	—	WWPROG	FREE	WRERR	WREN	WR	—
F7Eh	EECON2	EEPROM Control Register 2 (not a physical register)							
F7Dh	RCON2	EXTR	—	SWDTEN	—	—	—	—	—
F7Ch	RCON3	STKERR	—	—	—	VDDBOR	VDDPOR	VBPOR	VBAT
F7Bh	RCON4	—	—	—	SRETEN	—	DPSLP	—	PMSLP
F7Ah	UFRML	FRM7	FRM6	FRM5	FRM4	FRM3	FRM2	FRM1	FRM0
F79h	UFRMH	—	—	—	—	—	FRM10	FRM9	FRM8
F78h	UIR	—	SOFIF	STALLIF	IDLEIF	TRNIF	ACTVIF	UERRIF	URSTIF
F77h	UEIR	BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF
F76H	USTAT	—	ENDP3	ENDP2	ENDP1	ENDP0	DIR	PPBI	—
F75h	UCON	—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—
F74h	UADDR	—	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0
F73h	TRISVP	TRISVP7	TRISVP6	TRISVP5	TRISVP4	TRISVP3	TRISVP2	TRISVP1	TRISVP0
F72h	LATVP	LATVP7	LATVP6	LATVP5	LATVP4	LATVP3	LATVP2	LATVP1	LATVP0
F71h	PORTVP	RVP7	RVP6	RVP5	RVP4	RVP3	RVP2	RVP1	RVP0
F70h	TXADDRL	SPI DMA Transmit Data Pointer Low Byte							
F6Fh	TXADDRH	—	—	—	—	SPI DMA Transmit Data Pointer High Byte			
F6Eh	RXADDRL	SPI DMA Receive Data Pointer Low Byte							
F6Dh	RXADDRH	—	—	—	—	SPI DMA Receive Data Pointer High Byte			
F6Ch	DMABCL	SPI DMA Byte Count Low Byte							
F6Bh	DMABCH	—	—	—	—	—	—	SPI DMA Byte Count High Byte	
F6Ah	TXBUF	TXBUF7	TXBUF6	TXBUF5	TXBUF4	TXBUF3	TXBUF2	TXBUF1	TXBUF0

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
F69h	SSP1CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN
F68h	SSP1MSK	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
F67h	BAUDCON1	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	IREN	WUE	ABDEN
F66h	OSCCON2	CLKLOCK	IOLOCK	LOCK	—	CF	POSCEN	SOSCGO	—
F65h	OSCCON3	—	—	—	—	—	IRCF2	IRCF1	IRCF0
F64h	OSCCON4	CPDIV1	CPDIV0	PLLEN	—	—	—	—	—
F63h	ACTCON	ACTEN	—	ACTSIDL	ACTSRC	ACTLOCK	ACTLOCKPOL	ACTORS	ACTORSPOL
F62h	WPUB	WPUB7	WPUB6	WPUB5	WPUB4	WPUB3	WPUB2	WPUB1	WPUB0
F61h	PIE6	RC4IE	TX4IE	RC3IE	TX3IE	—	CMP3IE	CMP2IE	CMP1IE
F60h	DMACON1	SSCON1	SSCON0	TXINC	RXINC	DUPLEX1	DUPLEX0	DLYINTEN	DMAEN
F5Fh	RTCCON1	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0
F5Eh	RTCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
F5Dh	RTCVALH	RTCC Value High Register Window Based on RTCPTR<1:0>							
F5Ch	RTCVALL	RTCC Value Low Register Window Based on RTCPTR<1:0>							
F5Bh	ALRMCFG	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
F5Ah	ALRMRPT	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
F59h	ALRMVALH	Alarm Value High Register Window Based on APTR<1:0>							
F58h	ALRMVALL	Alarm Value Low Register Window Based on APTR<1:0>							
F57h	RTCCON2	PWCEN	PWCPOL	PWCCPRE	PWCSPRE	RTCCLKSEL1	RTCCLKSEL0	RTCSECSEL1	RTCSECSEL0
F56h	IOCP	IOCP7	IOCP6	IOCP5	IOCP4	IOCP3	IOCP2	IOCP1	IOCP0
F55h	IOCN	IOCN7	IOCN6	IOCN5	IOCN4	IOCN3	IOCN2	IOCN1	IOCN0
F54h	PADCFG1	RDPU	REPU	RFPU	RGPU	RHPU	RJPU	RKPU	RLPU
F53h	CM1CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
F52h	ECCP2AS	ECCP2ASE	ECCP2AS2	ECCP2AS1	ECCP2AS0	PSS2AC1	PSS2AC0	PSS2BD1	PSS2BD0
F51h	ECCP2DEL	P2RSEN	P2DC6	P2DC5	P2DC4	P2DC3	P2DC2	P2DC1	P2DC0
F50h	CCPR2H	Capture/Compare/PWM Register 1 High Byte							
F4Fh	CCPR2L	Capture/Compare/PWM Register 1 Low Byte							
F4Eh	CCP2CON	P2M1	P2M0	CCP2X	CCP2Y	CCP2M3	CCP2M2	CCP2M1	CCP2M0
F4Dh	ECCP3AS	ECCP3ASE	ECCP3AS2	ECCP3AS1	ECCP3AS0	PSS3AC1	PSS3AC0	PSS3BD1	PSS3BD0
F4Ch	ECCP3DEL	P3RSEN	P3DC6	P3DC5	P3DC4	P3DC3	P3DC2	P3DC1	P3DC0
F4Bh	CCPR3H	Capture/Compare/PWM Register 1 High Byte							
F4Ah	CCPR3L	Capture/Compare/PWM Register 1 Low Byte							
F49H	CCP3CON	P3M1	P3M0	CCP3X	CCP3Y	CCP3M3	CCP3M2	CCP3M1	CCP3M0
F48h	CCPR8H	Capture/Compare/PWM Register 8 High Byte							
F47h	CCPR8L	Capture/Compare/PWM Register 8 Low Byte							
F46h	CCP8CON	—	—	CCP8X	CCP8Y	CCP8M3	CCP8M2	CCP8M1	CCP8M0
F45h	CCPR9H	Capture/Compare/PWM Register 9 High Byte							
F44h	CCPR9L	Capture/Compare/PWM Register 9 Low Byte							
F43h	CCP9CON	—	—	CCP9X	CCP9Y	CCP9M3	CCP9M2	CCP9M1	CCP9M0
F42h	CCPR10H	Capture/Compare/PWM Register 10 High Byte							
F41h	CCPR10L	Capture/Compare/PWM Register 10 Low Byte							
F40h	CCP10CON	—	—	CCP10X	CCP10Y	CCP10M3	CCP10M2	CCP10M1	CCP10M0
F3Fh	TMR6	Timer6 Register							
F3Eh	PR6	Timer6 Period Register							
F3Dh	T6CON	—	T6OUTPS3	T6OUTPS2	T6OUTPS1	T6OUTPS0	TMR6ON	T6CKPS1	T6CKPS0
F3Ch	TMR8	Timer8 Register							
F3Bh	PR8	Timer8 Period Register							
F3Ah	T8CON	—	T8OUTPS3	T8OUTPS2	T8OUTPS1	T8OUTPS0	TMR8ON	T8CKPS1	T8CKPS0
F39H	SSP2CON3	ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN
F38h	CM2CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
F37h	CM3CON	CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
F36h	CCPTMRS0	C3TSEL1	C3TSEL0	C2TSEL2	C2TSEL1	C2TSEL0	C1TSEL2	C1TSEL1	C1TSEL0
F35h	CCPTMRS1	C7TSEL1	C7TSEL0	—	C6TSEL0	—	C5TSEL0	C4TSEL1	C4TSEL0
F34h	CCPTMRS2	—	—	—	C10TSEL0	—	C9TSEL0	C8TSEL1	C8TSEL0
F33h	RCSTA2	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
F32h	TXSTA2	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
F31h	BAUDCON2	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	IREN	WUE	ABDEN
F30h	SPBRGH1	EUSART1 Baud Rate Generator High Byte							
F2Fh	RCSTA3	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
F2Eh	TXSTA3	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
F2Dh	BAUDCON3	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	IREN	WUE	ABDEN
F2Ch	SPBRGH3	EUSART3 Baud Rate Generator High Byte							
F2Bh	SPBRG3	EUSART3 Baud Rate Generator							
F2Ah	RCREG3	EUSART3 Receive Data FIFO							
F29H	TXREG3	EUSART3 Transmit Data FIFO							
F28h	DSCONL	—	—	—	—	—	ULPWDIS	DSBOR	RELEASE
F27h	DSCONH	DSEN	—	—	—	—	—	—	RTCWDIS
F26h	DSWAKEL	DSFLT	BOR	DSULP	DSWDT	DSRTC	DSMCLR	DSICD	DSPOR
F25h	DSWAKEH	—	—	—	—	—	—	—	DSINT0
F24h	DSGPR0	Deep Sleep General Purpose Register 0							
F23h	DSGPR1	Deep Sleep General Purpose Register 1							
F22h	DSGPR2	Deep Sleep General Purpose Register 2							
F21h	DSGPR3	Deep Sleep General Purpose Register 3							
F20h	SPBRGH2	EUSART2 Baud Rate Generator High Byte							
F1Fh	SPBRG2	EUSART2 Baud Rate Generator							
F1Eh	RCREG2	Receive Data FIFO							
F1Dh	TXREG2	Transmit Data FIFO							
F1Ch	PSTR2CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA
F1Bh	PSTR3CON	CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA
F1Ah	SSP2STAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
F19h	SSP2CON1	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
F18h	SSP2CON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
F17h	SSP2MSK	MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
F16h	TMR5H	Timer5 Register High Byte							
F15h	TMR5L	Timer5 Register Low Byte							
F14h	T5CON	TMR5CS1	TMR5CS0	T5CKPS1	T5CKPS0	SOSCEN	$\overline{T5SYNC}$	RD16	TMR5ON
F13h	T5GCON	TMR5GE	T5GPOL	T5GTM	T5GSPM	T5GGO/$\overline{T5DONE}$	T5GVAL	T5GSS1	T5GSS0
F12h	CCPR4H	Capture/Compare/PWM Register 4 High Byte							
F11h	CCPR4L	Capture/Compare/PWM Register 4 Low Byte							
F10h	CCP4CON	—	—	DC4B1	DC4B0	CCP4M3	CCP4M2	CCP4M1	CCP4M0
F0Fh	CCPR5H	Capture/Compare/PWM Register 5 High Byte							
F0Eh	CCPR5L	Capture/Compare/PWM Register 5 Low Byte							
F0Dh	CCP5CON	—	—	DC5B1	DC5B0	CCP5M3	CCP5M2	CCP5M1	CCP5M0
F0Ch	CCPR6H	Capture/Compare/PWM Register 6 High Byte							
F0Bh	CCPR6L	Capture/Compare/PWM Register 6 Low Byte							
F0Ah	CCP6CON	—	—	DC6B1	DC6B0	CCP6M3	CCP6M2	CCP6M1	CCP6M0
F09h	CCPR7H	Capture/Compare/PWM Register 7 High Byte							
F08h	CCPR7L	Capture/Compare/PWM Register 7 Low Byte							
F07h	CCP7CON	—	—	DC7B1	DC7B0	CCP7M3	CCP7M2	CCP7M1	CCP7M0
F06h	TMR4	Timer4 Register							
F05h	PR4	Timer4 Period Register							
F04h	T4CON	—	T4OUTPS3	T4OUTPS2	T4OUTPS1	T4OUTPS0	TMR4ON	T4CKPS1	T4CKPS0

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
F03h	SSP2BUF	MSSP2 Receive Buffer/Transmit Register							
F02h	SSP2ADD	MSSP2 Address Register in I^2C Slave Mode. MSSP1 Baud Rate Reload Register in I^2C Master Mode.							
F01h	ANCFG	—	—	—	—	—	VBG6EN	VBG2EN	VBGEN
F00h	DMACON2	DLYCYC3	DLYCYC2	DLYCYC1	DLYCYC0	INTLVL3	INTLVL2	INTLVL1	INTLVL0
EFFh	RCSTA4	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
EFEh	TXSTA4	CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
EFDh	BAUDCON4	ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	IREN	WUE	ABDEN
EFCh	SPBRGH4	EUSART4 Baud Rate Generator High Byte							
EFBh	SPBRG4	EUSART4 Baud Rate Generator							
EFAh	RCREG4	EUSART4 Receive Data FIFO							
EF9h	TXREG4	EUSART4 Transmit Data FIFO							
EF8h	CTMUCON1	CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	TRIGEN
EF7h	CTMUCON2	ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0
EF6h	CTMUCON3	EDG2EN	EDG2POL	EDG2SEL3	EDG2SEL2	EDG2SEL1	EDG2SEL0	—	—
EF5h	CTMUCON4	EDG1EN	EDG1POL	EDG1SEL3	EDG1SEL2	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
EF4h	PMD0	CCP10MD	CCP9MD	CCP8MD	CCP7MD	CCP6MD	CCP5MD	CCP4MD	ECCP3MD
EF3h	PMD1	ECCP2MD	ECCP1MD	UART4MD	UART3MD	UART2MD	UART1MD	SSP2MD	SSP1MD
EF2h	PMD2	TMR8MD	TMR6MD	TMR5MD	TMR4MD	TMR3MD	TMR2MD	TMR1MD	TMR0MD
EF1h	PMD3	DSMMD	CTMUMD	ADCMD	RTCCMD	LCDMD	PSPMD	REFO1MD	REFO2MD
EF0h	PMD4	CMP1MD	CMP2MD	CMP3MD	USBMD	IOCMD	LVDMD	—	EMBMD
EEFh	MDCON	MDEN	MDOE	MDSLR	MDOPOL	MDO	—	—	MDBIT
EEEh	MDSRC	MDSODIS	—	—	—	MDSRC3	MDSRC2	MDSRC1	MDSRC0
EEDh	MDCARH	MDCHODIS	MDCHPOL	MDCHSYNC	—	MDCH3	MDCH2	MDCH1	MDCH0
EECh	MDCARL	MDCLODIS	MDCLPOL	MDCLSYNC	—	MDCL3	MDCL2	MDCL1	MDCL0
EEBh	ODCON1	ECCP2OD	ECCP1OD	USART4OD	USART3OD	USART2OD	USART1OD	SSP2OD	SSP1OD
EEAh	ODCON2	CCP10OD	CCP9OD	CCP8OD	CCP7OD	CCP6OD	CCP5OD	CCP4OD	ECCP3OD
EE9h	TRISK	TRISK7	TRISK6	TRISK5	TRISK4	TRISK3	TRISK2	TRISK1	TRISK0
EE8h	LATK	LATK7	LATK6	LATK5	LATK4	LATK3	LATK2	LATK1	LATK0
EE7h	PORTK	RK7	RK6	RK5	RK4	RK3	RK2	RK1	RK0
EE6h	TRISL	TRISL7	TRISL6	TRISL5	TRISL4	TRISL3	TRISL2	TRISL1	TRISL0
EE5h	LATL	LATL7	LATL6	LATL5	LATL4	LATL3	LATL2	LATL1	LATL0
EE4h	PORTL	RL7	RL6	RL5	RL4	RL3	RL2	RL1	RL0
EE3h	MEMCON	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
EE2h	REFO1CON	ON	—	SIDL	OE	RSLP	—	DIVSWEN	ACTIVE
EE1h	REFO1CON1	—	—	—	—	ROSEL3	ROSEL2	ROSEL1	ROSEL0
EE0h	REFO1CON2	RODIV7	RODIV6	RODIV5	RODIV4	RODIV3	RODIV2	RODIV1	RODIV0
EDFh	REFO1CON3	—	RODIV14	RODIV13	RODIV12	RODIV11	RODIV10	RODIV9	RODIV8
EDEh	REFO2CON	ON	—	SIDL	OE	RSLP	—	DIVSWEN	ACTIVE
EDDh	REFO2CON1	—	—	—	—	ROSEL3	ROSEL2	ROSEL1	ROSEL0
EDCh	REFO2CON2	RODIV7	RODIV6	RODIV5	RODIV4	RODIV3	RODIV2	RODIV1	RODIV0
EDBh	REFO2CON3	—	RODIV14	RODIV13	RODIV12	RODIV11	RODIV10	RODIV9	RODIV8
EDAh	LCDPS	WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0
ED9h	LCDCON	LCDEN	SLPEN	WERR	CS1	CS0	LMUX2	LMUX1	LMUX0
ED8h	LCDREG	CPEN	—	BIAS2	BIAS1	BIAS0	MODE13	CLKSEL1	CLKSEL0
ED7h	LCDREF	LCDIRE	—	LCDCST2	LCDCST1	LCDCST0	VLCD3PE	VLCD2PE	VLCD1PE
ED6h	LCDRL	LRLAP1	LRLAP0	LRLBP1	LRLBP0	—	LRLAT2	LRLAT1	LRLAT0
ED5h	LCDSE7	SE63	SE62	SE61	SE60	SE59	SE58	SE57	SE56
ED4h	LCDSE6	SE55	SE54	SE53	SE52	SE51	SE50	SE49	SE48
ED3h	LCDSE5	SE47	SE46	SE45	SE44	SE43	SE42	SE41	SE40
ED2h	LCDSE4	SE39	SE38	S37	SE36	SE35	SE34	SE33	SE32
ED1h	LCDSE3	SE31	SE30	SE29	SE28	SE27	SE26	SE25	SE24
ED0h	LCDSE2	SE23	SE22	SE21	SE20	SE19	SE18	SE17	SE16

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ECFh	LCDSE1	SE15	SE14	SE13	SE12	SE11	SE10	SE09	SE08
ECEh	LCDSE0	SE07	SE06	SE05	SE04	SE03	SE02	SE01	SE00
ECDh	LCDDATA63	S63C7	S62C7	S61C7	S60C7	S59C7	S58C7	S57C7	S56C7
ECCh	LCDDATA62	S55C7	S54C7	S53C7	S52C7	S51C7	S50C7	S49C7	S48C7
ECBh	LCDDATA61	S47C7	S46C7	S45C7	S44C7	S43C7	S42C7	S41C7	S40C7
ECAh	LCDDATA60	S39C7	S38C7	S37C7	S36C7	S35C7	S34C7	S33C7	S32C7
EC9h	LCDDATA59	S31C7	S30C7	S29C7	S28C7	S27C7	S26C7	S25C7	S24C7
EC8h	LCDDATA58	S23C7	S22C7	S21C7	S20C7	S19C7	S18C7	S17C7	S16C7
EC7h	LCDDATA57	S15C7	S14C7	S13C7	S12C7	S11C7	S10C7	S09C7	S08C7
EC6h	LCDDATA56	S07C7	S06C7	S05C7	S04C7	S03C7	S02C7	S01C7	S00C7
EC5h	LCDDATA55	S63C6	S62C6	S61C6	S60C6	S59C6	S58C6	S57C6	S56C6
EC4h	LCDDATA54	S55C6	S54C6	S53C6	S52C6	S51C6	S50C6	S49C6	S48C6
EC3h	LCDDATA53	S47C6	S46C6	S45C6	S44C6	S43C6	S42C6	S41C6	S40C6
EC2h	LCDDATA52	S39C6	S38C6	S37C6	S36C6	S35C6	S34C6	S33C6	S32C6
EC1h	LCDDATA51	S31C6	S30C6	S29C6	S28C6	S27C6	S26C6	S25C6	S24C6
EC0h	LCDDATA50	S23C6	S22C6	S21C6	S20C6	S19C6	S18C6	S17C6	S16C6
EBFh	LCDDATA49	S15C6	S14C6	S13C6	S12C6	S11C6	S10C6	S09C6	S08C6
EBEh	LCDDATA48	S07C6	S06C6	S05C6	S04C6	S03C6	S02C6	S01C6	S00C6
EBDh	LCDDATA47	S63C5	S62C5	S61C5	S60C5	S59C5	S58C5	S57C5	S56C5
EBCh	LCDDATA46	S55C5	S54C5	S53C5	S52C5	S51C5	S50C5	S49C5	S48C5
EBBh	LCDDATA45	S47C5	S46C5	S45C5	S44C5	S43C5	S42C5	S41C5	S40C5
EBAh	LCDDATA44	S39C5	S38C5	S37C5	S36C5	S35C5	S34C5	S33C5	S32C5
EB9h	LCDDATA43	S31C5	S30C5	S29C5	S28C5	S27C5	S26C5	S25C5	S24C5
EB8h	LCDDATA42	S23C5	S22C5	S21C5	S20C5	S19C5	S18C5	S17C5	S16C5
EB7h	LCDDATA41	S15C5	S14C5	S13C5	S12C5	S11C5	S10C5	S09C5	S08C5
EB6h	LCDDATA40	S07C5	S06C5	S05C5	S04C5	S03C5	S02C5	S01C5	S00C5
EB5h	LCDDATA39	S63C4	S62C4	S61C4	S60C4	S59C4	S58C4	S57C4	S56C4
EB4h	LCDDATA38	S55C4	S54C4	S53C4	S52C4	S51C4	S50C4	S49C4	S48C4
EB3h	LCDDATA37	S47C4	S46C4	S45C4	S44C4	S43C4	S42C4	S41C4	S40C4
EB2h	LCDDATA36	S39C4	S38C4	S37C4	S36C4	S35C4	S34C4	S33C4	S32C4
EB1h	LCDDATA35	S31C4	S30C4	S29C4	S28C4	S27C4	S26C4	S25C4	S24C4
EB0h	LCDDATA34	S23C4	S22C4	S21C4	S20C4	S19C4	S18C4	S17C4	S16C4
EAFh	LCDDATA33	S15C4	S14C4	S13C4	S12C4	S11C4	S10C4	S09C4	S08C4
EAEh	LCDDATA32	S07C4	S06C4	S05C4	S04C4	S03C4	S02C4	S01C4	S00C4
EADh	LCDDATA31	S63C3	S62C3	S61C3	S60C3	S59C3	S58C3	S57C3	S56C3
EACh	LCDDATA30	S55C3	S54C3	S53C3	S52C3	S51C3	S50C3	S49C3	S48C3
EABh	LCDDATA29	S47C3	S46C3	S45C3	S44C3	S43C3	S42C3	S41C3	S40C3
EAAh	LCDDATA28	S39C3	S38C3	S37C3	S36C3	S35C3	S34C3	S33C3	S32C3
EA9h	LCDDATA27	S31C3	S30C3	S29C3	S28C3	S27C3	S26C3	S25C3	S24C3
EA8h	LCDDATA26	S23C3	S22C3	S21C3	S20C3	S19C3	S18C3	S17C3	S16C3
EA7h	LCDDATA25	S15C3	S14C3	S13C3	S12C3	S11C3	S10C3	S09C3	S08C3
EA6h	LCDDATA24	S07C3	S06C3	S05C3	S04C3	S03C3	S02C3	S01C3	S00C3
EA5h	LCDDATA23	S63C2	S62C2	S61C2	S60C2	S59C2	S58C2	S57C2	S56C2
EA4h	LCDDATA22	S55C2	S54C2	S53C2	S52C2	S51C2	S50C2	S49C2	S48C2
EA3h	LCDDATA21	S47C2	S46C2	S45C2	S44C2	S43C2	S42C2	S41C2	S40C2
EA2h	LCDDATA20	S39C2	S38C2	S37C2	S36C2	S35C2	S34C2	S33C2	S32C2
EA1h	LCDDATA19	S31C2	S30C2	S29C2	S28C2	S27C2	S26C2	S25C2	S24C2
EA0h	LCDDATA18	S23C2	S22C2	S21C2	S20C2	S19C2	S18C2	S17C2	S16C2
E9Fh	LCDDATA17	S15C2	S14C2	S13C2	S12C2	S11C2	S10C2	S09C2	S08C2
E9Eh	LCDDATA16	S07C2	S06C2	S05C2	S04C2	S03C2	S02C2	S01C2	S00C2
E9Dh	LCDDATA15	S63C1	S62C1	S61C1	S60C1	S59C1	S58C1	S57C1	S56C1

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
E9Ch	LCDDATA14	S55C1	S54C1	S53C1	S52C1	S51C1	S50C1	S49C1	S48C1
E9Bh	LCDDATA13	S47C1	S46C1	S45C1	S44C1	S43C1	S42C1	S41C1	S40C1
E9Ah	LCDDATA12	S39C1	S38C1	S37C1	S36C1	S35C1	S34C1	S33C1	S32C1
E99h	LCDDATA11	S31C1	S30C1	S29C1	S28C1	S27C1	S26C1	S25C1	S24C1
E98h	LCDDATA10	S23C1	S22C1	S21C1	S20C1	S19C1	S18C1	S17C1	S16C1
E97h	LCDDATA9	S15C1	S14C1	S13C1	S12C1	S11C1	S10C1	S09C1	S08C1
E96h	LCDDATA8	S07C1	S06C1	S05C1	S04C1	S03C1	S02C1	S01C1	S00C1
E95h	LCDDATA7	S63C0	S62C0	S61C0	S60C0	S59C0	S58C0	S57C0	S56C0
E94h	LCDDATA6	S55C0	S54C0	S53C0	S52C0	S51C0	S50C0	S49C0	S48C0
E93h	LCDDATA5	S47C0	S46C0	S45C0	S44C0	S43C0	S42C0	S41C0	S40C0
E92h	LCDDATA4	S39C0	S38C0	S37C0	S36C0	S35C0	S34C0	S33C0	S32C0
E91h	LCDDATA3	S31C0	S30C0	S29C0	S28C0	S27C0	S26C0	S25C0	S24C0
E90h	LCDDATA2	S23C0	S22C0	S21C0	S20C0	S19C0	S18C0	S17C0	S16C0
E8Fh	LCDDATA1	S15C0	S14C0	S13C0	S12C0	S11C0	S10C0	S09C0	S08C0
E8Eh	LCDDATA0	S07C0	S06C0	S05C0	S04C0	S03C0	S02C0	S01C0	S00C0
E8Dh	ADCON2H	PVCFG1	PVCFG0	NVCFG0	OFFCAL	BUFREGEN	CSCNA	—	—
E8Ch	ADCON2L	BUFS	SMPI4	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS
E8Bh	ADCON3H	ADRC	EXTSAM	PUMPEN	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0
E8Ah	ADCON3L	ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0
E89h	ADCON5H	ASENA	LPENA	CTMUREQ	—	—	—	ASINTMD1	ASINTMD0
E88h	ADCON5L	—	—	—	—	WM1	WM0	CM1	CM0
E87h	ADCHS0H	CH0NB2	CH0NB1	CH0NB0	CH0SB4	CH0SB3	CH0SB2	CH0SB1	CH0SB0
E86h	ADCHS0L	CH0NA2	CH0NA1	CH0NA0	CH0SA4	CH0SA3	CH0SA2	CH0SA1	CH0SA0
E85h	ADCSS1H	—	CSS30	CSS29	CSS28	CSS27	CSS26	CSS25	CSS24
E84h	ADCSS1L	CSS23	CSS22	CSS21	CSS20	CSS19	CSS18	CSS17	CSS16
E83h	ADCSS0H	CSS15	CSS14	CSS13	CSS12	CSS11	CSS10	CSS9	CSS8
E82h	ADCSS0L	CSS7	CSS6	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0
E81h	ADCHIT1H	—	CHH30	CHH29	CHH28	CHH27	CHH26	CHH25	CHH24
E80h	ADCHIT1L	CHH23	CHH22	CHH21	CHH20	CHH19	CHH18	CHH17	CHH16
E7Fh	ADCHIT0H	CHH15	CHH14	CHH13	CHH12	CHH11	CHH10	CHH9	CHH8
E7Eh	ADCHIT0L	CHH7	CHH6	CHH5	CHH4	CHH3	CHH2	CHH1	CHH0
E7Dh	ADCTMUEN1H	—	CTMUEN30	CTMUEN29	CTMUEN28	CTMUEN27	CTMUEN26	CTMUEN25	CTMUEN24
E7Ch	ADCTMUEN1L	CTMUEN23	CTMUEN22	CTMUEN21	CTMUEN20	CTMUEN19	CTMUEN18	CTMUEN17	CTMUEN16
E7Bh	ADCTMUEN0H	CTMUEN15	CTMUEN14	CTMUEN13	CTMUEN12	CTMUEN11	CTMUEN10	CTMUEN9	CTMUEN8
E7Ah	ADCTMUEN0L	CTMUEN7	CTMUEN6	CTMUEN5	CTMUEN4	CTMUEN3	CTMUEN2	CTMUEN1	CTMUEN0
E79h	ADCBUF25H	A/D Result Register 25 High Byte							
E78h	ADCBUF25L	A/D Result Register 25 Low Byte							
E77h	ADCBUF24H	A/D Result Register 24 High Byte							
E76h	ADCBUF24L	A/D Result Register 24 Low Byte							
E75h	ADCBUF23H	A/D Result Register 23 High Byte							
E74h	ADCBUF23L	A/D Result Register 23 Low Byte							
E73h	ADCBUF22H	A/D Result Register 22 High Byte							
E72h	ADCBUF22L	A/D Result Register 22 Low Byte							
E71h	ADCBUF21H	A/D Result Register 21 High Byte							
E70h	ADCBUF21L	A/D Result Register 21 Low Byte							
E6Fh	ADCBUF20H	A/D Result Register 20 High Byte							
E6Eh	ADCBUF20L	A/D Result Register 20 Low Byte							
E6Dh	ADCBUF19H	A/D Result Register 19 High Byte							
E6Ch	ADCBUF19L	A/D Result Register 19 Low Byte							
E6Bh	ADCBUF18H	A/D Result Register 18 High Byte							
E6Ah	ADCBUF18L	A/D Result Register 18 Low Byte							

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
E69h	ADCBUF17H	A/D Result Register 17 High Byte							
E68h	ADCBUF17L	A/D Result Register 17 Low Byte							
E67h	ADCBUF16H	A/D Result Register 16 High Byte							
E66h	ADCBUF16L	A/D Result Register 16 Low Byte							
E65h	ADCBUF15H	A/D Result Register 15 High Byte							
E64h	ADCBUF15L	A/D Result Register 15 Low Byte							
E63h	ADCBUF14H	A/D Result Register 14 High Byte							
E62h	ADCBUF14L	A/D Result Register 14 Low Byte							
E61h	ADCBUF13H	A/D Result Register 13 High Byte							
E60h	ADCBUF13L	A/D Result Register 13 Low Byte							
E5Fh	ADCBUF12H	A/D Result Register 12 High Byte							
E5Eh	ADCBUF12L	A/D Result Register 12 Low Byte							
E5Dh	ADCBUF11H	A/D Result Register 11 High Byte							
E5Ch	ADCBUF11L	A/D Result Register 11 Low Byte							
E5Bh	ADCBUF10H	A/D Result Register 10 High Byte							
E5Ah	ADCBUF10L	A/D Result Register 10 Low Byte							
E59h	ADCBUF9H	A/D Result Register 9 High Byte							
E58h	ADCBUF9L	A/D Result Register 9 Low Byte							
E57h	ADCBUF8H	A/D Result Register 8 High Byte							
E56h	ADCBUF8L	A/D Result Register 8 Low Byte							
E55h	ADCBUF7H	A/D Result Register 7 High Byte							
E54h	ADCBUF7L	A/D Result Register 7 Low Byte							
E53h	ADCBUF6H	A/D Result Register 6 High Byte							
E52h	ADCBUF6L	A/D Result Register 6 Low Byte							
E51h	ADCBUF5H	A/D Result Register 5 High Byte							
E50h	ADCBUF5L	A/D Result Register 5 Low Byte							
E4Fh	ADCBUF4H	A/D Result Register 4 High Byte							
E4Eh	ADCBUF4L	A/D Result Register 4 Low Byte							
E4Dh	ADCBUF3H	A/D Result Register 3 High Byte							
E4Ch	ADCBUF3L	A/D Result Register 3 Low Byte							
E4Bh	ADCBUF2H	A/D Result Register 2 High Byte							
E4Ah	ADCBUF2L	A/D Result Register 2 Low Byte							
E49h	ADCBUF1H	A/D Result Register 1 High Byte							
E48h	ADCBUF1L	A/D Result Register 1 Low Byte							
E47h	ANCON1	ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
E46h	ANCON2	ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
E45h	ANCON3	ANSEL23	ANSEL22	ANSEL21	ANSEL20	ANSEL19	ANSEL18	ANSEL17	ANSEL16
E44h	RPINR52_53	PBIO7R<3:0>				PBIO6R<3:0>			
E43h	RPINR50_51	PBIO5R<3:0>				PBIO4R<3:0>			
E42h	RPINR48_49	PBIO3R<3:0>				PBIO2R<3:0>			
E41h	RPINR46_47	PBIO1R<3:0>				PBIO0R<3:0>			
E40h	RPINR44_45	T5CKIR<3:0>				T5GR<3:0>			
E3Fh	RPINR42_43	T3CKIR<3:0>				T3GR<3:0>			
E3Eh	RPINR40_41	T1CKIR<3:0>				T1GR<3:0>			
E3Dh	RPINR38_39	T0CKIR<3:0>				CCP10R<3:0>			
E3Ch	RPINR36_37	CCP9R<3:0>				CCP8R<3:0>			
E3Bh	RPINR34_35	CCP7R<3:0>				CCP6R<3:0>			
E3Ah	RPINR32_33	CCP5R<3:0>				CCP4R<3:0>			
E39h	RPINR30_31	MDCIN2R<3:0>				MDCIN1R<3:0>			
E38h	RPINR28_29	MDMINR<3:0>				INT3R<3:0>			
E37h	RPINR26_27	INT2R<3:0>				INT1R<3:0>			

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
E36h	RPINR24_25	IOC7R<3:0>				IOC6R<3:0>			
E35h	RPINR22_23	IOC5R<3:0>				IOC4R<3:0>			
E34h	RPINR20_21	IOC3R<3:0>				IOC2R<3:0>			
E33h	RPINR18_19	IOC1R<3:0>				IOC0R<3:0>			
E32h	RPINR16_17	ECCP3R<3:0>				ECCP2R<3:0>			
E31h	RPINR14_15	ECCP1R<3:0>				FLT0R<3:0>			
E30h	RPINR12_13	SS2R<3:0>				SDI2R<3:0>			
E2Fh	RPINR10_11	SCK2R<3:0>				SS1R<3:0>			
E2Eh	RPINR8_9	SDI1R<3:0>				SCK1R<3:0>			
E2Dh	RPINR6_7	U4TXR<3:0>				U4RXR<3:0>			
E2Ch	RPINR4_5	U3TXR<3:0>				U3RXR<3:0>			
E2Bh	RPINR2_3	U2TXR<3:0>				U2RXR<3:0>			
E2Ah	RPINR0_1	U1TXR<3:0>				U1RXR<3:0>			
E29h	RPOR46	—	—	—	—	RPO46R<3:0>			
E28h	RPOR44_45	RPO45R<3:0>				RPO44R<3:0>			
E27h	RPOR42_43	RPO43R<3:0>				RPO42R<3:0>			
E26h	RPOR40_41	RPO41R<3:0>				RPO40R<3:0>			
E25h	RPOR38_39	RPO39R<3:0>				RPO38R<3:0>			
E24h	RPOR36_37	RPO37R<3:0>				RPO36R<3:0>			
E23h	RPOR34_35	RPO35R<3:0>				RPO34R<3:0>			
E22h	RPOR32_33	RPO33R<3:0>				RPO32R<3:0>			
E21h	RPOR30_31	RPO31R<3:0>				RPO30R<3:0>			
E20h	RPOR28_29	RPO29R<3:0>				RPO28R<3:0>			
E1Fh	RPOR26_27	RPO27R<3:0>				RPO26R<3:0>			
E1Eh	RPOR24_25	RPO25R<3:0>				RPO24R<3:0>			
E1Dh	RPOR22_23	RPP23R<3:0>				RPO22R<3:0>			
E1Ch	RPOR20_21	RPO21R<3:0>				RPO20R<3:0>			
E1Bh	RPOR18_19	RPO19R<3:0>				RPO18R<3:0>			
E1Ah	RPOR16_17	RPO17R<3:0>				RPO16R<3:0>			
E19h	RPOR14_15	RPO15R<3:0>				RPO14R<3:0>			
E18h	RPOR12_13	RPO13R<3:0>				RPO12R<3:0>			
E17h	RPOR10_11	RPO11R<3:0>				RPO10R<3:0>			
E16h	RPOR8_9	RPO9R<3:0>				RPO8R<3:0>			
E15h	RPOR6_7	RPO7R<3:0>				RPO6R<3:0>			
E14h	RPOR4_5	RPO5R<3:0>				RPO4R<3:0>			
E13h	RPOR2_3	RPO3R<3:0>				RPO2R<3:0>			
E12h	RPOR0_1	RPO1R<3:0>				RPO0R<3:0>			
E11h	UCFG	UTEYE	UOEMON	—	UPUEN	UTRDIS	FSEN	PPB1	PPB0
E10h	UIE	—	SOFIE	STALLIE	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE
E0Fh	UEIE	BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE
E0Eh	UEP15	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Dh	UEP14	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Ch	UEP13	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Bh	UEP12	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E0Ah	UEP11	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E09h	UEP10	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E08h	UEP9	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E07h	UEP8	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E06h	UEP7	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E05h	UEP6	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E04h	UEP5	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL

Legend: — = unimplemented, read as '0'.

TABLE 6-2: REGISTER FILE SUMMARY (CONTINUED)

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
E03h	UEP4	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E02h	UEP3	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E01h	UEP2	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
E00h	UEP1	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
DFFh	UEP0	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
DFEh	Unimplemented	—	—	—	—	—	—	—	—
DFDh	Unimplemented	—	—	—	—	—	—	—	—
DFCh	Unimplemented	—	—	—	—	—	—	—	—
DFBh	Unimplemented	—	—	—	—	—	—	—	—
DFAh	Unimplemented	—	—	—	—	—	—	—	—

Legend: — = unimplemented, read as '0'.

6.3.5 STATUS REGISTER

The STATUS register, shown in Register 6-2, contains the arithmetic status of the ALU. The STATUS register can be the operand for any instruction, as with any other register. If the STATUS register is the destination for an instruction that affects the Z, DC, C, OV or N bits, the write to these five bits is disabled.

These bits are set or cleared according to the device logic. Therefore, the result of an instruction with the STATUS register as destination may be different than intended. For example, `CLRF STATUS` will set the Z bit but leave the other bits unchanged. The STATUS register then reads back as '`000u u1uu`'.

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF`, `MOVFF` and `MOVWF` instructions be used to alter the STATUS register because these instructions do not affect the Z, C, DC, OV or N bits in the STATUS register.

For other instructions not affecting any Status bits, see the instruction set summaries in Table 29-2 and Table 29-3.

Note: The C and DC bits operate, in subtraction, as borrow and digit borrow bits, respectively.

REGISTER 6-2: STATUS REGISTER

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	N	OV	Z	DC(1)	C(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **N:** Negative bit

This bit is used for signed arithmetic (2's complement). It indicates whether the result was negative (ALU MSB = 1).

1 = Result was negative
0 = Result was positive

bit 3 **OV:** Overflow bit

This bit is used for signed arithmetic (2's complement). It indicates an overflow of the seven-bit magnitude which causes the sign bit (bit 7) to change state.

1 = Overflow occurred for signed arithmetic (in this arithmetic operation)
0 = No overflow occurred

bit 2 **Z:** Zero bit

1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC:** Digit Carry/$\overline{\text{Borrow}}$ bit(1)

For `ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions:

1 = A carry-out from the 4th low-order bit of the result occurred
0 = No carry-out from the 4th low-order bit of the result

bit 0 **C:** Carry/$\overline{\text{Borrow}}$ bit(2)

For `ADDWF`, `ADDLW`, `SUBLW` and `SUBWF` instructions:

1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

Note 1: For $\overline{\text{borrow}}$, the polarity is reversed. A subtraction is executed by adding the 2's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either bit 4 or bit 3 of the source register.

2: For $\overline{\text{borrow}}$, the polarity is reversed. A subtraction is executed by adding the 2's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low-order bit of the source register.

6.4 Data Addressing Modes

Note: The execution of some instructions in the core PIC18 instruction set are changed when the PIC18 extended instruction set is enabled. For more information, see **Section 6.6 "Data Memory and the Extended Instruction Set"**.

While the program memory can be addressed in only one way, through the Program Counter, information in the data memory space can be addressed in several ways. For most instructions, the addressing mode is fixed. Other instructions may use up to three modes, depending on which operands are used and whether or not the extended instruction set is enabled.

The addressing modes are:

- Inherent
- Literal
- Direct
- Indirect

An additional addressing mode, Indexed Literal Offset, is available when the extended instruction set is enabled (XINST Configuration bit = 1). For details on this mode's operation, see **Section 6.6.1 "Indexed Addressing with Literal Offset"**.

6.4.1 INHERENT AND LITERAL ADDRESSING

Many PIC18 control instructions do not need any argument at all. They either perform an operation that globally affects the device or they operate implicitly on one register. This addressing mode is known as Inherent Addressing. Examples of this mode include SLEEP, RESET and DAW.

Other instructions work in a similar way, but require an additional explicit argument in the opcode. This method is known as the Literal Addressing mode because the instructions require some literal value as an argument. Examples of this include ADDLW and MOVLW which, respectively, add or move a literal value to the W register. Other examples include CALL and GOTO, which include a 20-bit program memory address.

6.4.2 DIRECT ADDRESSING

Direct Addressing specifies all or part of the source and/or destination address of the operation within the opcode itself. The options are specified by the arguments accompanying the instruction.

In the core PIC18 instruction set, bit-oriented and byte-oriented instructions use some version of Direct Addressing by default. All of these instructions include some 8-bit literal address as their Least Significant Byte. This address specifies the instruction's data source as either a register address in one of the banks of data RAM (see **Section 6.3.3 "General Purpose Register File"**) or a location in the Access Bank (see **Section 6.3.2 "Access Bank"**).

The Access RAM bit, 'a', determines how the address is interpreted. When 'a' is '1', the contents of the BSR (**Section 6.3.1 "Bank Select Register"**) are used with the address to determine the complete 12-bit address of the register. When 'a' is '0', the address is interpreted as being a register in the Access Bank. Addressing that uses the Access RAM is sometimes also known as Direct Forced Addressing mode.

A few instructions, such as MOVFF, include the entire 12-bit address (either source or destination) in their opcodes. In these cases, the BSR is ignored entirely.

The destination of the operation's results is determined by the destination bit, 'd'. When 'd' is '1', the results are stored back in the source register, overwriting its original contents. When 'd' is '0', the results are stored in the W register. Instructions without the 'd' argument have a destination that is implicit in the instruction, either the target register being operated on or the W register.

6.4.3 INDIRECT ADDRESSING

Indirect Addressing allows the user to access a location in data memory without giving a fixed address in the instruction. This is done by using File Select Registers (FSRs) as pointers to the locations to be read or written to. Since the FSRs are themselves located in RAM as Special Function Registers, they can also be directly manipulated under program control. This makes FSRs very useful in implementing data structures such as tables and arrays in data memory.

The registers for Indirect Addressing are also implemented with Indirect File Operands (INDFs) that permit automatic manipulation of the pointer value with auto-incrementing, auto-decrementing or offsetting with another value. This allows for efficient code using loops, such as the example of clearing an entire RAM bank in Example 6-5. It also enables users to perform Indexed Addressing and other Stack Pointer operations for program memory in data memory.

EXAMPLE 6-5: HOW TO CLEAR RAM (BANK 1) USING INDIRECT ADDRESSING

```
        LFSR    FSR0, 100h ;
NEXT    CLRF    POSTINC0   ; Clear INDF
                           ; register then
                           ; inc pointer
        BTFSS   FSR0H, 1   ; All done with
                           ; Bank1?
        BRA     NEXT       ; NO, clear next
CONTINUE                   ; YES, continue
```

6.4.3.1 FSR Registers and the INDF Operand

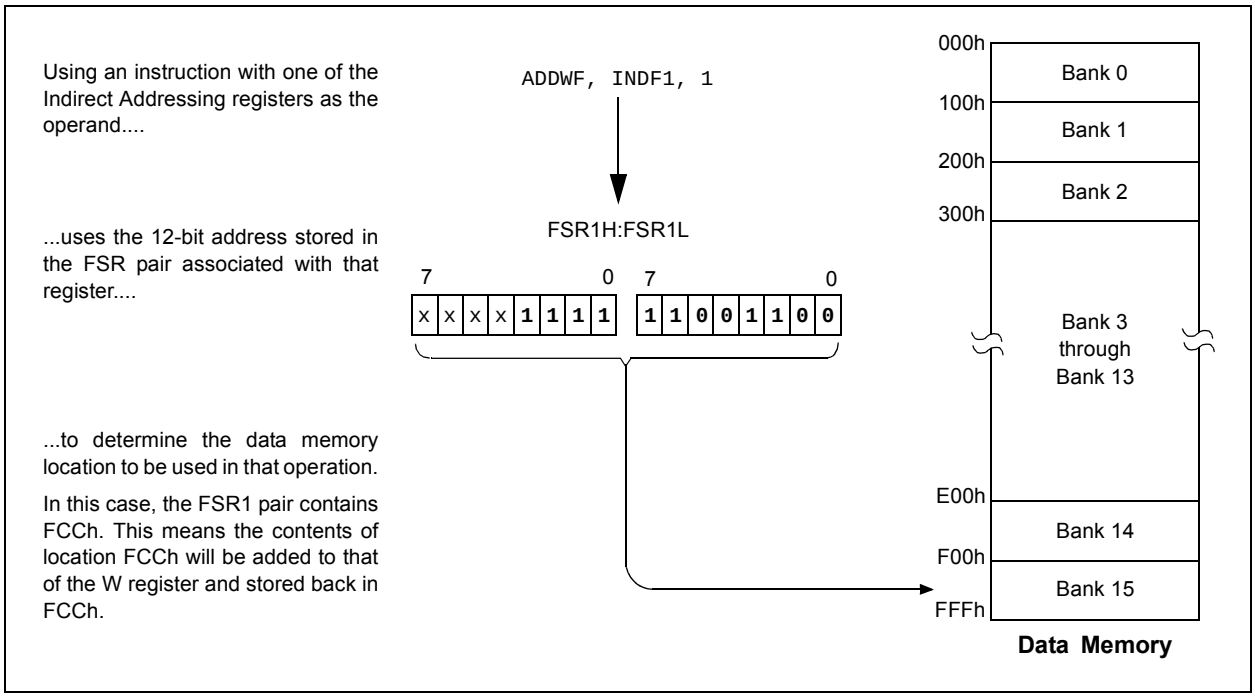
At the core of Indirect Addressing are three sets of registers: FSR0, FSR1 and FSR2. Each represents a pair of 8-bit registers: FSRnH and FSRnL. The four upper bits of the FSRnH register are not used, so each FSR pair holds a 12-bit value. This represents a value that can address the entire range of the data memory in a linear fashion. The FSR register pairs, then, serve as pointers to data memory locations.

Indirect Addressing is accomplished with a set of Indirect File Operands, INDF0 through INDF2. These can be thought of as "virtual" registers. The operands are mapped in the SFR space, but are not physically implemented. Reading or writing to a particular INDF register actually accesses its corresponding FSR register pair. A read from INDF1, for example, reads the data at the address indicated by FSR1H:FSR1L.

Instructions that use the INDF registers as operands actually use the contents of their corresponding FSR as a pointer to the instruction's target. The INDF operand is just a convenient way of using the pointer.

Because Indirect Addressing uses a full 12-bit address, data RAM banking is not necessary. Thus, the current contents of the BSR and the Access RAM bit have no effect on determining the target address.

FIGURE 6-8: INDIRECT ADDRESSING



6.4.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are “virtual” registers that cannot be indirectly read or written to. Accessing these registers actually accesses the associated FSR register pair, but also performs a specific action on its stored value.

These operands are:

- POSTDEC – Accesses the FSR value, then automatically decrements it by ‘1’ afterwards
- POSTINC – Accesses the FSR value, then automatically increments it by ‘1’ afterwards
- PREINC – Increments the FSR value by ‘1’, then uses it in the operation
- PLUSW – Adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the new value in the operation

In this context, accessing an INDF register uses the value in the FSR registers without changing them. Similarly, accessing a PLUSW register gives the FSR value, offset by the value in the W register, with neither value actually changed in the operation. Accessing the other virtual registers changes the value of the FSR registers.

Operations on the FSRs with POSTDEC, POSTINC and PREINC affect the entire register pair. Rollovers of the FSRnL register, from FFh to 00h, carry over to the FSRnH register. On the other hand, results of these operations do not change the value of any flags in the STATUS register (for example, Z, N and OV bits).

The PLUSW register can be used to implement a form of Indexed Addressing in the data memory space. By manipulating the value in the W register, users can reach addresses that are fixed offsets from pointer addresses. In some applications, this can be used to implement some powerful program control structure, such as software stacks, inside of data memory.

6.4.3.3 Operations by FSRs on FSRs

Indirect Addressing operations that target other FSRs or virtual registers represent special cases. For example, using an FSR to point to one of the virtual registers will not result in successful operations.

As a specific case, assume that the FSR0H:FSR0L registers contain FE7h, the address of INDF1. Attempts to read the value of the INDF1, using INDF0 as an operand, will return 00h. Attempts to write to INDF1, using INDF0 as the operand, will result in a NOP.

On the other hand, using the virtual registers to write to an FSR pair may not occur as planned. In these cases, the value will be written to the FSR pair, but without any incrementing or decrementing. Thus, writing to INDF2 or POSTDEC2 will write the same value to the FSR2H:FSR2L.

Since the FSRs are physical registers mapped in the SFR space, they can be manipulated through all direct operations. Users should proceed cautiously when working on these registers, however, particularly if their code uses Indirect Addressing.

Similarly, operations by Indirect Addressing are generally permitted on all other SFRs. Users should exercise the appropriate caution, so that they do not inadvertently change settings that might affect the operation of the device.

6.5 Program Memory and the Extended Instruction Set

The operation of program memory is unaffected by the use of the extended instruction set.

Enabling the extended instruction set adds five additional two-word commands to the existing PIC18 instruction set: ADDFSR, CALLW, MOVSF, MOVSS and SUBFSR. These instructions are executed as described in **Section 6.2.4 “Two-Word Instructions”**.

6.6 Data Memory and the Extended Instruction Set

Enabling the PIC18 extended instruction set (XINST Configuration bit = 1) significantly changes certain aspects of data memory and its addressing. Using the Access Bank for many of the core PIC18 instructions introduces a new addressing mode for the data memory space. This mode also alters the behavior of Indirect Addressing using FSR2 and its associated operands.

What does not change is just as important. The size of the data memory space is unchanged, as well as its linear addressing. The SFR map remains the same. Core PIC18 instructions can still operate in both Direct and Indirect Addressing mode. Inherent and literal instructions do not change at all. Indirect Addressing with FSR0 and FSR1 also remains unchanged.

6.6.1 INDEXED ADDRESSING WITH LITERAL OFFSET

Enabling the PIC18 extended instruction set changes the behavior of Indirect Addressing using the FSR2 register pair and its associated file operands. Under the proper conditions, instructions that use the Access Bank – that is, most bit-oriented and byte-oriented instructions – can invoke a form of Indexed Addressing using an offset specified in the instruction. This special addressing mode is known as Indexed Addressing with Literal Offset or the Indexed Literal Offset mode.

When using the extended instruction set, this addressing mode requires the following:

- Use of the Access Bank (‘a’ = 0)
- A file address argument that is less than or equal to 5Fh

Under these conditions, the file address of the instruction is not interpreted as the lower byte of an address (used with the BSR in Direct Addressing) or as an 8-bit address in the Access Bank. Instead, the value is interpreted as an offset value to an Address Pointer specified by FSR2. The offset and the contents of FSR2 are added to obtain the target address of the operation.

6.6.2 INSTRUCTIONS AFFECTED BY INDEXED LITERAL OFFSET MODE

Any of the core PIC18 instructions that can use Direct Addressing are potentially affected by the Indexed Literal Offset Addressing mode. This includes all byte-oriented and bit-oriented instructions, or almost one-half of the standard PIC18 instruction set. Instructions that only use Inherent or Literal Addressing modes are unaffected.

Additionally, byte-oriented and bit-oriented instructions are not affected if they do not use the Access Bank (Access RAM bit = 1), or include a file address of 60h or above. Instructions meeting these criteria will continue to execute as before. A comparison of the different possible addressing modes when the extended instruction set is enabled is shown in Figure 6-9.

Those who desire to use byte-oriented or bit-oriented instructions in the Indexed Literal Offset mode should note the changes to assembler syntax for this mode. This is described in more detail in **Section 29.2.1 “Extended Instruction Syntax”**.

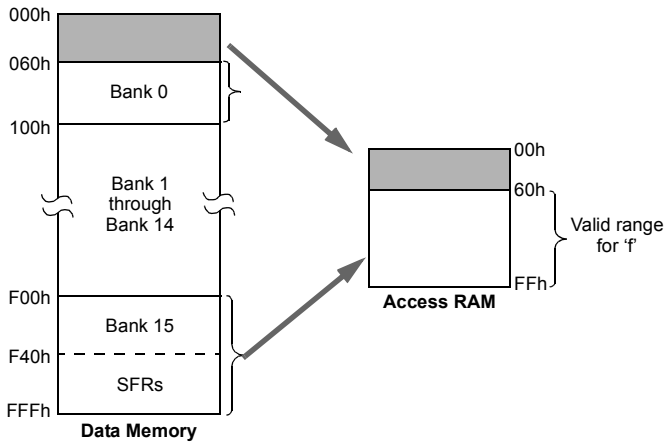
FIGURE 6-9: COMPARING ADDRESSING OPTIONS FOR BIT-ORIENTED AND BYTE-ORIENTED INSTRUCTIONS (EXTENDED INSTRUCTION SET ENABLED)

EXAMPLE INSTRUCTION: `ADDWF, f, d, a` (Opcode: `0010 01da ffff ffff`)

When a = `0` and f ≥ 60h:

The instruction executes in Direct Forced mode. 'f' is interpreted as a location in the Access RAM between 060h and FFFh. This is the same as locations, F60h to FFFh (Bank 15), of data memory.

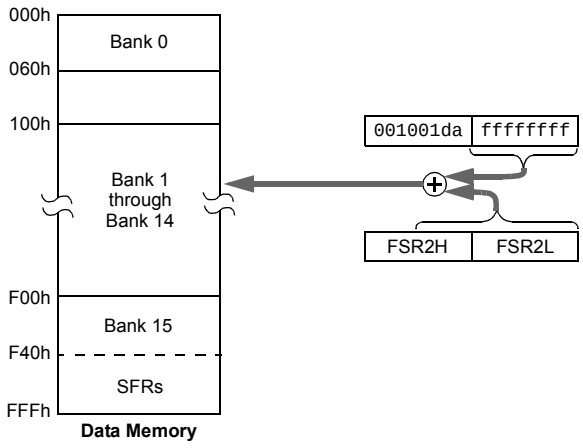
Locations below 060h are not available in this addressing mode.



When a = `0` and f ≤ 5Fh:

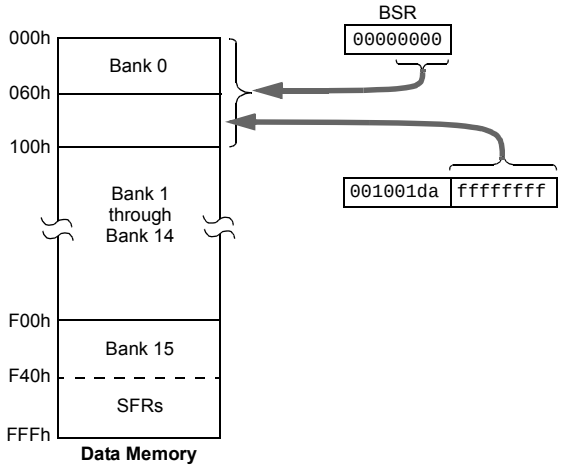
The instruction executes in Indexed Literal Offset mode. 'f' is interpreted as an offset to the address value in FSR2. The two are added together to obtain the address of the target register for the instruction. The address can be anywhere in the data memory space.

Note that in this mode, the correct syntax is now:
`ADDWF [k], d`
where 'k' is the same as 'f'.



When a = `1` (all values of f):

The instruction executes in Direct mode (also known as Direct Long mode). 'f' is interpreted as a location in one of the 16 banks of the data memory space. The bank is designated by the Bank Select Register (BSR). The address can be in any implemented bank in the data memory space.



6.6.3 MAPPING THE ACCESS BANK IN INDEXED LITERAL OFFSET MODE

The use of Indexed Literal Offset Addressing mode effectively changes how the lower part of Access RAM (00h to 5Fh) is mapped. Rather than containing just the contents of the bottom part of Bank 0, this mode maps the contents from Bank 0 and a user-defined "window" that can be located anywhere in the data memory space.

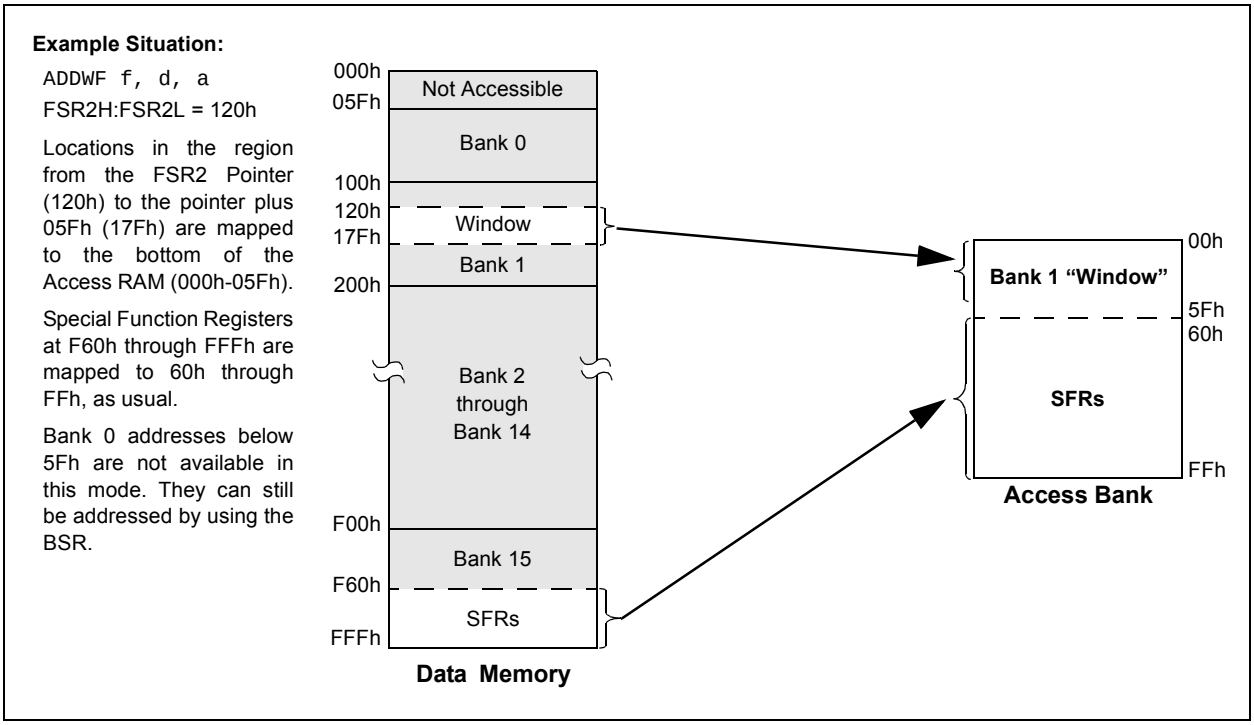
The value of FSR2 establishes the lower boundary of the addresses mapped into the window, while the upper boundary is defined by FSR2 plus 95 (5Fh). Addresses in the Access RAM above 5Fh are mapped as previously described. (See **Section 6.3.2 "Access Bank"**.) An example of Access Bank remapping in this addressing mode is shown in Figure 6-10.

Remapping the Access Bank applies *only* to operations using the Indexed Literal Offset mode. Operations that use the BSR (Access RAM bit = 1) will continue to use Direct Addressing as before. Any Indirect or Indexed Addressing operation that explicitly uses any of the indirect file operands (including FSR2) will continue to operate as standard Indirect Addressing. Any instruction that uses the Access Bank, but includes a register address of greater than 05Fh, will use Direct Addressing and the normal Access Bank map.

6.6.4 BSR IN INDEXED LITERAL OFFSET MODE

Although the Access Bank is remapped when the extended instruction set is enabled, the operation of the BSR remains unchanged. Direct Addressing, using the BSR to select the data memory bank, operates in the same manner as previously described.

FIGURE 6-10: REMAPPING THE ACCESS BANK WITH INDEXED LITERAL OFFSET ADDRESSING



7.0 FLASH PROGRAM MEMORY

The Flash program memory is readable, writable and erasable during normal operation over the entire VDD range.

A read from program memory is executed on 1 byte at a time. A write to program memory is executed on blocks of 64 bytes at a time or 2 bytes at a time. Program memory is erased in blocks of 512 bytes at a time. A bulk erase operation may not be issued from user code.

Writing or erasing program memory will cease instruction fetches until the operation is complete. The program memory cannot be accessed during the write or erase, therefore, code cannot execute. An internal programming timer terminates program memory writes and erases.

A value written to program memory does not need to be a valid instruction. Executing a program memory location that forms an invalid instruction results in a NOP.

7.1 Table Reads and Table Writes

In order to read and write program memory, there are two operations that allow the processor to move bytes between the program memory space and the data RAM:

- Table Read (TBLRD)
- Table Write (TBLWT)

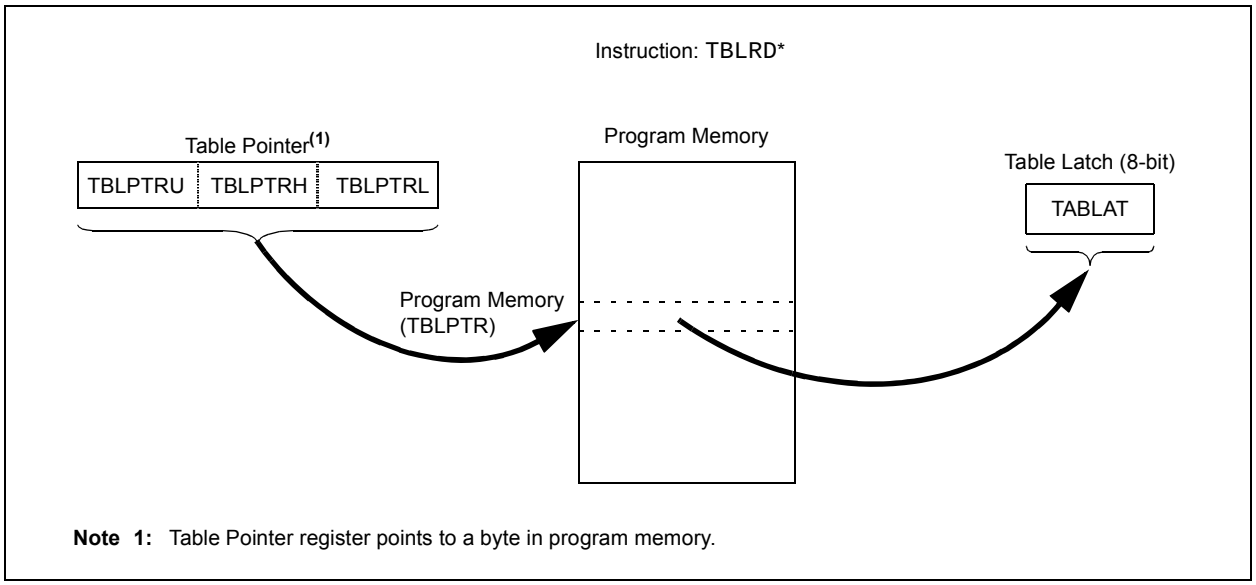
The program memory space is 16 bits wide, while the data RAM space is 8 bits wide. Table reads and table writes move data between these two memory spaces through an 8-bit register (TABLAT).

Table read operations retrieve data from program memory and place it into the data RAM space. Figure 7-1 shows the operation of a table read with program memory and data RAM.

Table write operations store data from the data memory space into holding registers in program memory. The procedure to write the contents of the holding registers into program memory is detailed in **Section 7.5 "Writing to Flash Program Memory"**. Figure 7-2 shows the operation of a table write with program memory and data RAM.

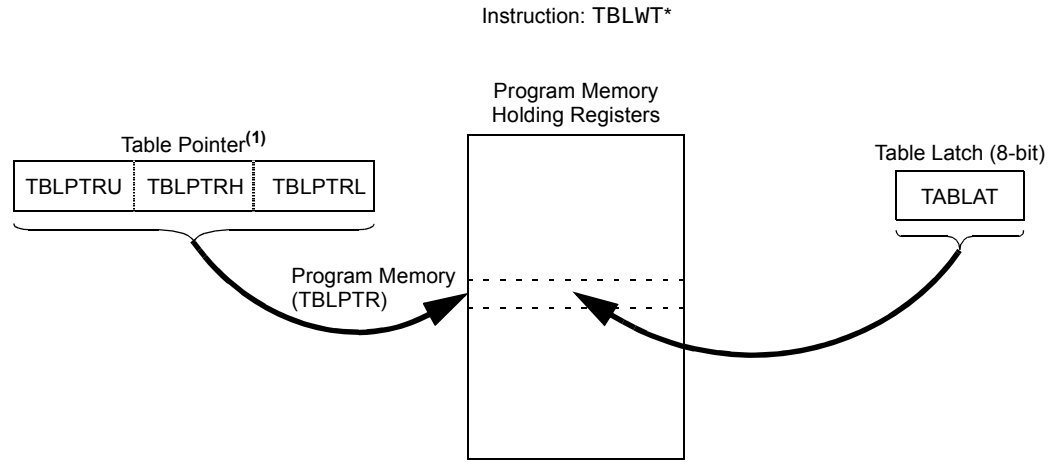
Table operations work with byte entities. A table block containing data, rather than program instructions, is not required to be word-aligned. Therefore, a table block can start and end at any byte address. If a table write is being used to write executable code into program memory, program instructions will need to be word-aligned.

FIGURE 7-1: TABLE READ OPERATION



Note 1: Table Pointer register points to a byte in program memory.

FIGURE 7-2: TABLE WRITE OPERATION



Note 1: The Table Pointer actually points to one of 64 holding registers; the address of which is determined by TBLPTRL<5:0>. The process for physically writing data to the program memory array is discussed in **Section 7.5 "Writing to Flash Program Memory"**.

7.2 Control Registers

Several control registers are used in conjunction with the TBLRD and TBLWT instructions. These include:

- EECON1 register
- EECON2 register
- TABLAT register
- TBLPTR registers

7.2.1 EECON1 AND EECON2 REGISTERS

The EECON1 register (Register 7-1) is the control register for memory accesses. The EECON2 register is not a physical register; it is used exclusively in the memory write and erase sequences. Reading EECON2 will read all '0's.

The WWPROG bit, when set, will allow programming two bytes per word on the execution of the WR command. If this bit is cleared, the WR command will result in programming on a block of 64 bytes.

The FREE bit, when set, will allow a program memory erase operation. When FREE is set, the erase operation is initiated on the next WR command. When FREE is clear, only writes are enabled.

The WREN bit, when set, will allow a write operation. On power-up, the WREN bit is clear. The WRERR bit is set in hardware when the WR bit is set, and cleared when the internal programming timer expires and the write operation is complete.

Note:	During normal operation, the WRERR is read as '1'. This can indicate that a write operation was prematurely terminated by a Reset or a write operation was attempted improperly.

Register 7-1: EECON1: EEPROM CONTROL REGISTER 1 (ACCESS FA6h)

U-0	U-0	R/W-0	R/W-0	R/W-x	R/W-0	R/S-0	U-0
—	—	WWPROG	FREE	WRERR(1)	WREN	WR	—
bit 7							bit 0

Legend:	S = Settable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'

bit 5 **WWPROG:** One Word-Wide Program bit
1 = Programs 2 bytes on the next WR command
0 = Programs 64 bytes on the next WR command

bit 4 **FREE:** Flash Erase Enable bit
1 = Performs an erase operation on the next WR command (cleared by hardware after completion of erase)
0 = Performs write-only

bit 3 **WRERR:** Flash Program Error Flag bit(1)
1 = A write operation is prematurely terminated (any Reset during self-timed programming in normal operation or an improper write attempt)
0 = The write operation completed

bit 2 **WREN:** Flash Program Write Enable bit
1 = Allows write cycles to Flash program memory
0 = Inhibits write cycles to Flash program memory

bit 1 **WR:** Write Control bit
1 = Initiates a program memory erase cycle or write cycle (the operation is self-timed and the bit is cleared by hardware once the write is complete)
The WR bit can only be set (not cleared) in software.
0 = Write cycle is complete

bit 0 **Unimplemented:** Read as '0'

Note 1: When a WRERR error occurs, the EEPGD and CFGS bits are not cleared. This allows tracing of the error condition.

7.2.2 TABLE LATCH REGISTER (TABLAT)

The Table Latch (TABLAT) is an 8-bit register mapped into the Special Function Register (SFR) space. The Table Latch register is used to hold 8-bit data during data transfers between program memory and data RAM.

7.2.3 TABLE POINTER REGISTER (TBLPTR)

The Table Pointer (TBLPTR) register addresses a byte within the program memory. The TBLPTR is comprised of three SFR registers: Table Pointer Upper Byte, Table Pointer High Byte and Table Pointer Low Byte (TBLPTRU:TBLPTRH:TBLPTRL). These three registers join to form a 22-bit wide pointer. The low-order 21 bits allow the device to address up to 2 Mbytes of program memory space. The 22nd bit allows access to the Device ID, the User ID and the Configuration bits.

The Table Pointer register, TBLPTR, is used by the TBLRD and TBLWT instructions. These instructions can update the TBLPTR in one of four ways, based on the table operation. These operations are shown in Table 7-1 and only affect the low-order 21 bits.

7.2.4 TABLE POINTER BOUNDARIES

TBLPTR is used in reads, writes and erases of the Flash program memory.

When a TBLRD is executed, all 22 bits of the TBLPTR determine which byte is read from program memory into TABLAT.

When a TBLWT is executed, the seven Least Significant bits (LSbs) of the Table Pointer register (TBLPTR<6:0>) determine which of the 64 program memory holding registers is written to. When the timed write to program memory begins (via the WR bit), the 12 Most Significant bits (MSbs) of the TBLPTR (TBLPTR<21:10>) determine which program memory block of 1024 bytes is written to. For more detail, see **Section 7.5 "Writing to Flash Program Memory"**.

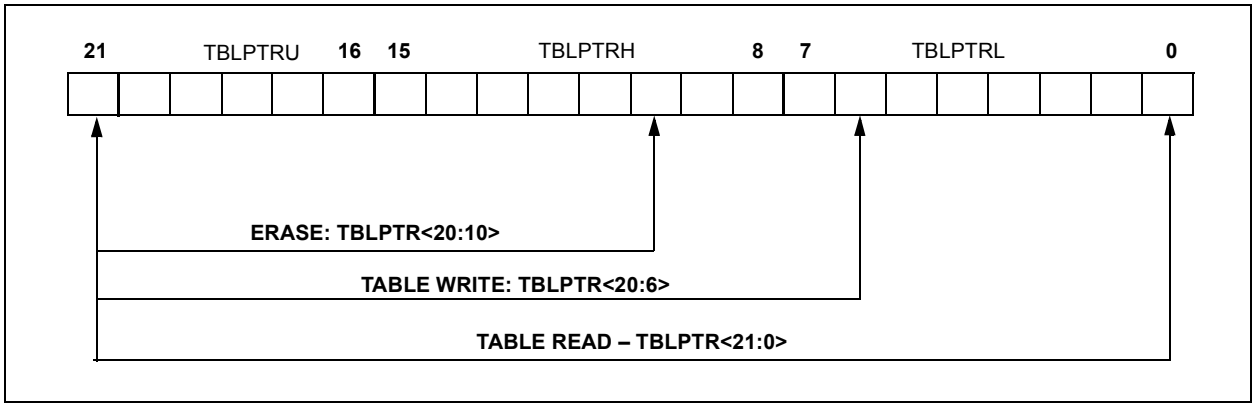
When an erase of program memory is executed, the 12 MSbs of the Table Pointer register point to the 1024-byte block that will be erased. The LSbs are ignored.

Figure 7-3 describes the relevant boundaries of the TBLPTR based on Flash program memory operations.

TABLE 7-1: TABLE POINTER OPERATIONS WITH TBLRD AND TBLWT INSTRUCTIONS

Example	Operation on Table Pointer
TBLRD* TBLWT*	TBLPTR is not modified
TBLRD*+ TBLWT*+	TBLPTR is incremented after the read/write
TBLRD*- TBLWT*-	TBLPTR is decremented after the read/write
TBLRD+* TBLWT+*	TBLPTR is incremented before the read/write

FIGURE 7-3: TABLE POINTER BOUNDARIES BASED ON OPERATION



7.3 Reading the Flash Program Memory

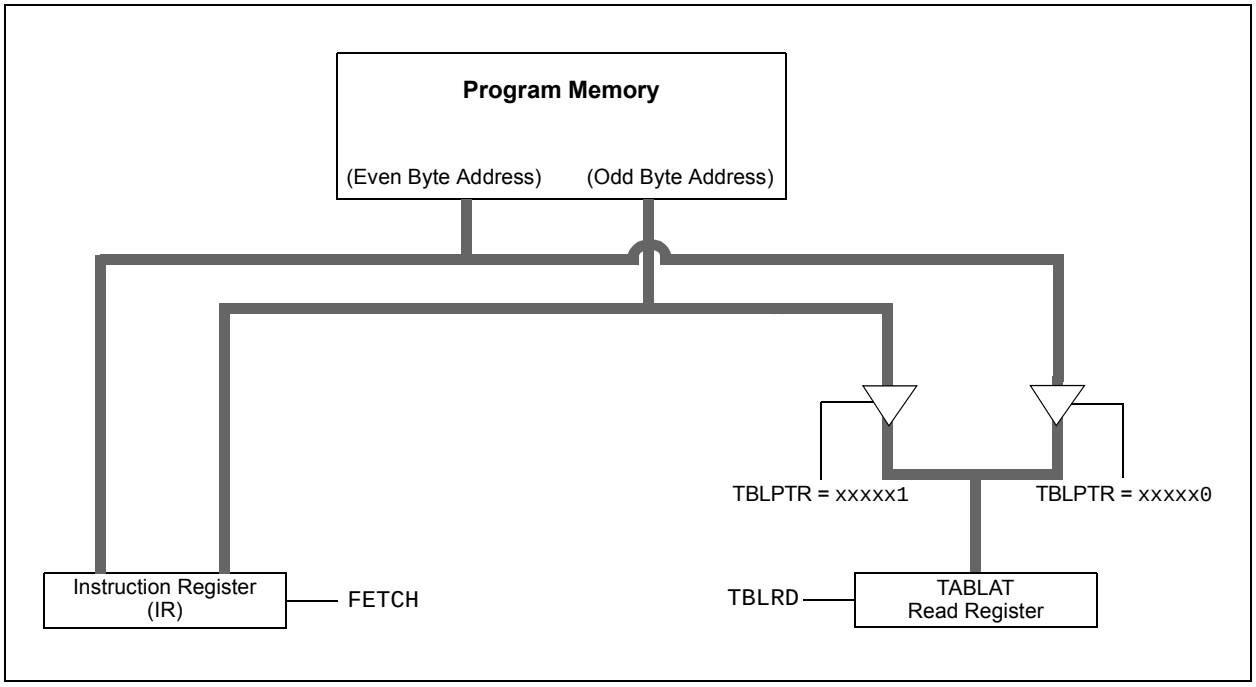
The TBLRD instruction is used to retrieve data from program memory and places it into data RAM. Table reads from program memory are performed one byte at a time.

The TBLPTR points to a byte address in program space. Executing TBLRD places the byte pointed to into TABLAT. In addition, the TBLPTR can be modified automatically for the next table read operation.

The internal program memory is typically organized by words. The Least Significant bit of the address selects between the high and low bytes of the word.

Figure 7-4 shows the interface between the internal program memory and the TABLAT.

FIGURE 7-4: READS FROM FLASH PROGRAM MEMORY



EXAMPLE 7-1: READING A FLASH PROGRAM MEMORY WORD

```
            MOVLW   CODE_ADDR_UPPER        ; Load TBLPTR with the base
            MOVWF   TBLPTRU                ; address of the word
            MOVLW   CODE_ADDR_HIGH
            MOVWF   TBLPTRH
            MOVLW   CODE_ADDR_LOW
            MOVWF   TBLPTRL
READ_WORD
            TBLRD*+                        ; read into TABLAT and increment
            MOVF    TABLAT, W              ; get data
            MOVWF   WORD_EVEN
            TBLRD*+                        ; read into TABLAT and increment
            MOVF    TABLAT, W              ; get data
            MOVWF   WORD_ODD
```

7.4 Erasing Flash Program Memory

The minimum erase block is 256 words or 512 bytes. Only through the use of an external programmer, or through ICSP control, can larger blocks of program memory be bulk erased. Word erase in the Flash array is not supported.

When initiating an erase sequence from the microcontroller itself, a block of 512 bytes of program memory is erased. The Most Significant 12 bits of the TBLPTR<21:10> point to the block being erased; TBLPTR<9:0> are ignored.

The EECON1 register commands the erase operation. The WREN bit must be set to enable write operations. The FREE bit is set to select an erase operation. For protection, the write initiate sequence for EECON2 must be used.

A long write is necessary for erasing the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

7.4.1 FLASH PROGRAM MEMORY ERASE SEQUENCE

The sequence of events for erasing a block of internal program memory location is:

1. Load Table Pointer register with address of row being erased.
2. Set the WREN and FREE bits (EECON1<2,4>) to enable the erase operation.
3. Disable interrupts.
4. Write 55h to EECON2.
5. Write 0AAh to EECON2.
6. Set the WR bit; this will begin the erase cycle.
7. The CPU will stall for the duration of the erase for TIE (see Parameter D133B).
8. Re-enable interrupts.

EXAMPLE 7-2: ERASING A FLASH PROGRAM MEMORY ROW

```
                    MOVLW   CODE_ADDR_UPPER         ; load TBLPTR with the base
                    MOVWF   TBLPTRU                 ; address of the memory block
                    MOVLW   CODE_ADDR_HIGH
                    MOVWF   TBLPTRH
                    MOVLW   CODE_ADDR_LOW
                    MOVWF   TBLPTRL
        ERASE_ROW
                    BSF     EECON1, WREN            ; enable write to memory
                    BSF     EECON1, FREE            ; enable Row Erase operation
                    BCF     INTCON, GIE             ; disable interrupts
    Required        MOVLW   0x55
    Sequence        MOVWF   EECON2                  ; write 55h
                    MOVLW   0xAA
                    MOVWF   EECON2                  ; write 0AAh
                    BSF     EECON1, WR              ; start erase (CPU stall)
                    BSF     INTCON, GIE             ; re-enable interrupts
```

7.5 Writing to Flash Program Memory

The programming block is 32 words or 64 bytes. Programming one word or 2 bytes at a time is also supported.

Table writes are used internally to load the holding registers needed to program the Flash memory. There are 64 holding registers used by the table writes for programming.

Since the Table Latch (TABLAT) is only a single byte, the TBLWT instruction may need to be executed 64 times for each programming operation (if WWPROG = 0). All of the table write operations will essentially be short writes because only the holding registers are written. At the end of updating the 64 holding registers, the EECON1 register must be written to in order to start the programming operation with a long write.

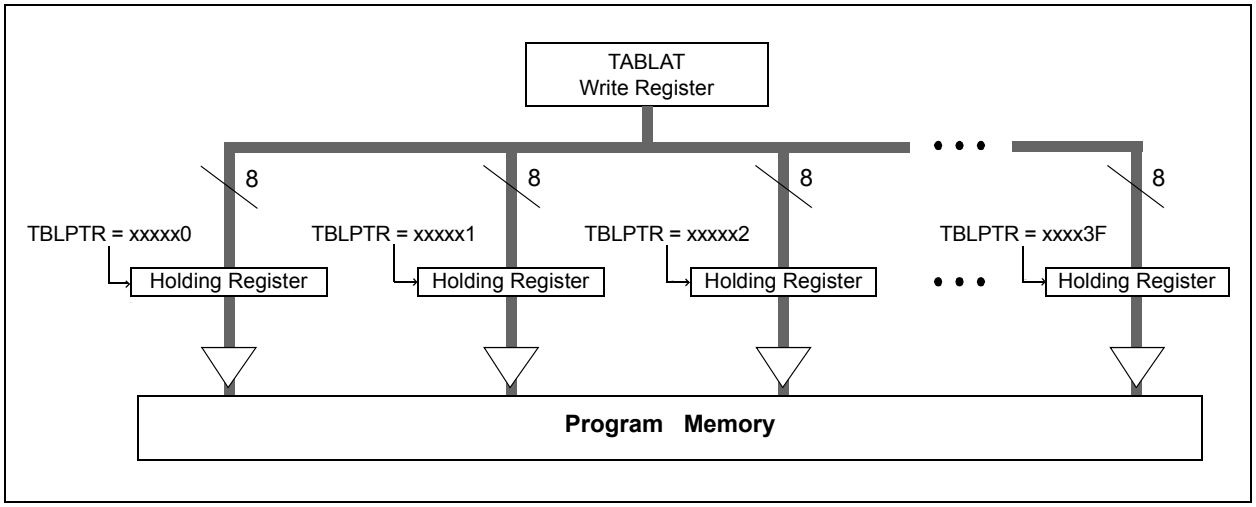
The long write is necessary for programming the internal Flash. Instruction execution is halted while in a long write cycle. The long write will be terminated by the internal programming timer.

The on-chip timer controls the write time. The write/erase voltages are generated by an on-chip charge pump, rated to operate over the voltage range of the device.

Note 1: Unlike previous PIC® MCUs, devices of the PIC18FXXJ94 do not reset the holding registers after a write occurs. The holding registers must be cleared or overwritten before a programming sequence.

2: To maintain the endurance of the program memory cells, each Flash byte should not be programmed more than once between erase operations. Before attempting to modify the contents of the target cell a second time, an erase of the target page, or a bulk erase of the entire memory, must be performed.

FIGURE 7-5: TABLE WRITES TO FLASH PROGRAM MEMORY



7.5.1 FLASH PROGRAM MEMORY WRITE SEQUENCE

The sequence of events for programming an internal program memory location should be:

1. Read the 512 bytes into RAM.
2. Update the data values in RAM as necessary.
3. Load the Table Pointer register with the address being erased.
4. Execute the erase procedure.
5. Load the Table Pointer register with the address of the first byte being written, minus 1.
6. Write the 64 bytes into the holding registers with auto-pre-increment.
7. Set the WREN bit (EECON1<2>) to enable byte writes.
8. Disable the interrupts.
9. Write 55h to EECON2.
10. Write 0xAAh to EECON2.
11. Set the WR bit. This will begin the write cycle. The CPU will stall for duration of the write for TIW (see Parameter D133A).
12. Re-enable the interrupts.
13. Verify the memory (table read).

An example of the required code is shown in Example 7-3 on the following Page 153.

Note: Before setting the WR bit, the Table Pointer address needs to be within the intended address range of the 64 bytes in the holding register.

EXAMPLE 7-3: WRITING TO FLASH PROGRAM MEMORY

```
                MOVLW   CODE_ADDR_UPPER         ; Load TBLPTR with the base
                MOVWF   TBLPTRU                 ; address of the memory block, minus 1
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW
                MOVWF   TBLPTRL
ERASE_BLOCK
                BSF     EECON1, WREN            ; enable write to memory
                BSF     EECON1, FREE            ; enable Erase operation
                BCF     INTCON, GIE             ; disable interrupts
                MOVLW   55h
                MOVWF   EECON2                  ; write 55h
                MOVLW   0AAh
                MOVWF   EECON2                  ; write 0AAh
                BSF     EECON1, WR              ; start erase (CPU stall)
                BSF     INTCON, GIE             ; re-enable interrupts
                MOVLW   D'8'
                MOVWF   WRITE_COUNTER           ; Need to write 8 blocks of 64 to write
                                                ; one erase block of 512

RESTART BUFFER
                MOVLW   D'64'
                MOVWF   COUNTER
                MOVLW   BUFFER_ADDR_HIGH        ; point to buffer
                MOVWF   FSR0H
                MOVLW   BUFFER_ADDR_LOW
                MOVWF   FSR0L
FILL_BUFFER
                ...                             ; read the new data from I2C, SPI,
                                                ; PSP, USART, etc.
WRITE_BUFFER
                MOVLW   D’64                    ; number of bytes in holding register
                MOVWF   COUNTER
WRITE_BYTE_TO_HREGS
                MOVFF   POSTINC0, WREG          ; get low byte of buffer data
                MOVWF   TABLAT                  ; present data to table latch
                TBLWT+*                         ; write data, perform a short write
                                                ; to internal TBLWT holding register.
                DECFSZ  COUNTER                 ; loop until buffers are full
                BRA     WRITE_BYTE_TO_HREGS
PROGRAM_MEMORY
                BSF     EECON1, WREN            ; enable write to memory
                BCF     INTCON, GIE             ; disable interrupts
                MOVLW   55h
   Required     MOVWF   EECON2                  ; write 55h
   Sequence     MOVLW   0AAh
                MOVWF   EECON2                  ; write 0AAh
                BSF     EECON1, WR              ; start program (CPU stall)
                BSF     INTCON, GIE             ; re-enable interrupts
                BCF     EECON1, WREN            ; disable write to memory

                DECFSZ  WRITE_COUNTER           ; done with one write cycle
                BRA     RESTART_BUFFER          ; if not done replacing the erase block
```

7.5.2 FLASH PROGRAM MEMORY WRITE SEQUENCE (WORD PROGRAMMING)

The PIC18FXXJ94 of devices has a feature that allows programming a single word (two bytes). This feature is enabled when the WWPROG bit is set. If the memory location is already erased, the following sequence is required to enable this feature:

1. Load the Table Pointer register with the address of the data to be written. (It must be an even address.)
2. Write the 2 bytes into the holding registers by performing table writes. (Do not post-increment on the second table write).
3. Set the WREN bit (EECON1<2>) to enable writes and the WWPROG bit (EECON1<5>) to select Word Write mode.
4. Disable interrupts.
5. Write 55h to EECON2.
6. Write 0AAh to EECON2.
7. Set the WR bit; this will begin the write cycle.
8. The CPU will stall for the duration of the write for TIW (see Parameter D133A).
9. Re-enable interrupts.

EXAMPLE 7-4: SINGLE-WORD WRITE TO FLASH PROGRAM MEMORY

```
                MOVLW   CODE_ADDR_UPPER    ; Load TBLPTR with the base address
                MOVWF   TBLPTRU
                MOVLW   CODE_ADDR_HIGH
                MOVWF   TBLPTRH
                MOVLW   CODE_ADDR_LOW      ; The table pointer must be loaded with an even address
                MOVWF   TBLPTRL
                MOVLW   DATA0              ; LSB of word to be written
                MOVWF   TABLAT
                TBLWT*+
                MOVLW   DATA1              ; MSB of word to be written
                MOVWF   TABLAT
                TBLWT*                     ; The last table write must not increment the
                                             table pointer! The table pointer needs to
                                             point to the MSB before starting the write operation.

PROGRAM_MEMORY
                BSF     EECON1, WWPROG     ; enable single word write
                BSF     EECON1, WREN       ; enable write to memory
                BCF     INTCON, GIE        ; disable interrupts
                MOVLW   55h
    Required    MOVWF   EECON2             ; write 55h
    Sequence    MOVLW   0AAh
                MOVWF   EECON2             ; write AAh
                BSF     EECON1, WR         ; start program (CPU stall)
                BSF     INTCON, GIE        ; re-enable interrupts
                BCF     EECON1, WWPROG     ; disable single word write
                BCF     EECON1, WREN       ; disable write to memory
```

7.5.3 WRITE VERIFY

Depending on the application, good programming practice may dictate that the value written to the memory should be verified against the original value. This should be used in applications where excessive writes can stress bits near the specification limit.

7.5.4 UNEXPECTED TERMINATION OF WRITE OPERATION

If a write is terminated by an unplanned event, such as loss of power or an unexpected Reset, the memory location just programmed should be verified and reprogrammed if needed. If the write operation is interrupted by a $\overline{\text{MCLR}}$ Reset, or a WDT time-out Reset during normal operation, the user can check the WRERR bit and rewrite the location(s) as needed

7.6 Flash Program Operation During Code Protection

See **Section 28.4.5 "Program Verification and Code Protection"** for details on code protection of Flash program memory.

8.0 EXTERNAL MEMORY BUS

Note: The External Memory Bus is not implemented on 64-pin devices.

The External Memory Bus (EMB) allows the device to access external memory devices (such as Flash, EPROM or SRAM) as program or data memory. It supports both 8 and 16-Bit Data Width modes, and three address widths of up to 20 bits.

The bus is implemented with 28 pins, multiplexed across four I/O ports. Three ports (PORTD, PORTE and PORTH) are multiplexed with the address/data bus for a total of 20 available lines, while PORTJ is multiplexed with the bus control signals.

A list of the pins and their functions is provided in Table 8-1.

TABLE 8-1: PIC18F97J94 FAMILY EXTERNAL BUS – I/O PORT FUNCTIONS

Name	Port	Bit	External Memory Bus Function
RD0/AD0	PORTD	0	Address Bit 0 or Data Bit 0
RD1/AD1	PORTD	1	Address Bit 1 or Data Bit 1
RD2/AD2	PORTD	2	Address Bit 2 or Data Bit 2
RD3/AD3	PORTD	3	Address Bit 3 or Data Bit 3
RD4/AD4	PORTD	4	Address Bit 4 or Data Bit 4
RD5/AD5	PORTD	5	Address Bit 5 or Data Bit 5
RD6/AD6	PORTD	6	Address Bit 6 or Data Bit 6
RD7/AD7	PORTD	7	Address Bit 7 or Data Bit 7
RE0/AD8	PORTE	0	Address Bit 8 or Data Bit 8
RE1/AD9	PORTE	1	Address Bit 9 or Data Bit 9
RE2/AD10	PORTE	2	Address Bit 10 or Data Bit 10
RE3/AD11	PORTE	3	Address Bit 11 or Data Bit 11
RE4/AD12	PORTE	4	Address Bit 12 or Data Bit 12
RE5/AD13	PORTE	5	Address Bit 13 or Data Bit 13
RE6/AD14	PORTE	6	Address Bit 14 or Data Bit 14
RE7/AD15	PORTE	7	Address Bit 15 or Data Bit 15
RH0/A16	PORTH	0	Address Bit 16
RH1/A17	PORTH	1	Address Bit 17
RH2/A18	PORTH	2	Address Bit 18
RH3/A19	PORTH	3	Address Bit 19
RJ0/ALE	PORTJ	0	Address Latch Enable (ALE) Control Pin
RJ1/$\overline{OE}$	PORTJ	1	Output Enable ($\overline{OE}$) Control Pin
RJ2/$\overline{WRL}$	PORTJ	2	Write Low ($\overline{WRL}$) Control Pin
RJ3/$\overline{WRH}$	PORTJ	3	Write High ($\overline{WRH}$) Control Pin
RJ4/BA0	PORTJ	4	Byte Address Bit 0 (BA0)
RJ5/$\overline{CE}$	PORTJ	5	Chip Enable ($\overline{CE}$) Control Pin
RJ6/$\overline{LB}$	PORTJ	6	Lower Byte Enable ($\overline{LB}$) Control Pin
RJ7/$\overline{UB}$	PORTJ	7	Upper Byte Enable ($\overline{UB}$) Control Pin

Note: For the sake of clarity, only I/O port and external bus assignments are shown here. One or more additional multiplexed features may be available on some pins.

8.1 External Memory Bus Control

The operation of the interface is controlled by the MEMCON register (Register 8-1). This register is available in all program memory operating modes, except Microcontroller mode. In this mode, the register is disabled and cannot be written to.

The EBDIS bit (MEMCON<7>) controls the operation of the bus and related port functions. Clearing EBDIS enables the interface and disables the I/O functions of the ports, as well as any other functions multiplexed to those pins. Setting the bit enables the I/O ports and other functions, but allows the interface to override everything else on the pins when an external memory operation is required. By default, the external bus is always enabled and disables all other I/O.

The operation of the EBDIS bit is also influenced by the program memory mode being used. This is discussed in more detail in **Section 8.5 "Program Memory Modes and the External Memory Bus"**.

The WAITx bits allow for the addition of Wait states to external memory operations. The use of these bits is discussed in **Section 8.3 "Wait States"**.

The WMx bits select the particular operating mode used when the bus is operating in 16-Bit Data Width mode. This is discussed in more detail in **Section 8.6 "16-Bit Data Width Modes"**. These bits have no effect when an 8-Bit Data Width mode is selected.

REGISTER 8-1: MEMCON: EXTERNAL MEMORY BUS CONTROL REGISTER(1)

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **EBDIS**: External Bus Disable bit
- 1 = External bus is enabled when microcontroller accesses external memory; otherwise, all external bus drivers are mapped as I/O ports
- 0 = External bus is always enabled, I/O ports are disabled

bit 6 **Unimplemented**: Read as '0'

bit 5-4 **WAIT<1:0>:** Table Reads and Writes Bus Cycle Wait Count bits
- 11 = Table reads and writes will wait 0 TCY
- 10 = Table reads and writes will wait 1 TCY
- 01 = Table reads and writes will wait 2 TCY
- 00 = Table reads and writes will wait 3 TCY

bit 3-2 **Unimplemented**: Read as '0'

bit 1-0 **WM<1:0>:** TBLWT Operation with 16-Bit Data Bus Width Select bits
- 1x = Word Write mode: TABLAT word output, $\overline{WRH}$ is active when TABLAT is written
- 01 = Byte Select mode: TABLAT data is copied on both MSB and LSB, $\overline{WRH}$ and ($\overline{UB}$ or $\overline{LB}$) will activate
- 00 = Byte Write mode: TABLAT data is copied on both MSB and LSB, $\overline{WRH}$ or $\overline{WRL}$ will activate

Note 1: This register is unimplemented on 64-pin devices, read as '0'.

8.2 Address and Data Width

The PIC18FXXJ94 of devices can be independently configured for different address and data widths on the same memory bus. Both address and data width are set by Configuration bits in the CONFIG5L register. As Configuration bits, this means that these options can only be configured by programming the device and are not controllable in software.

The BW bit selects an 8-bit or 16-bit data bus width. Setting this bit (default) selects a data width of 16 bits.

The ABW<1:0> bits determine both the program memory operating mode and the address bus width. The available options are 20-bit, 16-bit and 12-bit, as well as Microcontroller mode (external bus disabled). Selecting a 16-bit or 12-bit width makes a corresponding number of high-order lines available for I/O functions. These pins are no longer affected by the setting of the EBDIS bit. For example, selecting a 16-Bit Addressing mode (ABW<1:0> = 01) disables A<19:16> and allows PORTH<3:0> to function without interruptions from the bus. Using the smaller address widths allows users to tailor the memory bus to the size of the external memory space for a particular design while freeing up pins for dedicated I/O operation.

Because the ABWx bits have the effect of disabling pins for memory bus operations, it is important to always select an address width at least equal to the data width. If a 12-bit address width is used with a 16-bit data width, the upper four bits of data will not be available on the bus.

All combinations of address and data widths require multiplexing of address and data information on the same lines. The address and data multiplexing, as well as I/O ports made available by the use of smaller address widths, are summarized in Table 8-2.

8.2.1 ADDRESS SHIFTING ON THE EXTERNAL BUS

By default, the address presented on the external bus is the value of the PC. In practical terms, this means that addresses in the external memory device, below the top of on-chip memory, are unavailable to the microcontroller. To access these physical locations, the glue logic between the microcontroller and the external memory must somehow translate addresses.

To simplify the interface, the external bus offers an extension of Extended Microcontroller mode that automatically performs address shifting. This feature is controlled by the EASHFT Configuration bit. Setting this bit offsets addresses on the bus by the size of the microcontroller's on-chip program memory and sets the bottom address at 0000h. This allows the device to use the entire range of physical addresses of the external memory.

8.2.2 21-BIT ADDRESSING

As an extension of 20-bit address width operation, the External Memory Bus can also fully address a 2-Mbyte memory space. This is done by using the Bus Address Bit 0 (BA0) control line as the Least Significant bit of the address. The $\overline{UB}$ and $\overline{LB}$ control signals may also be used with certain memory devices to select the upper and lower bytes within a 16-bit wide data word.

This addressing mode is available in both 8-Bit and certain 16-Bit Data Width modes. Additional details are provided in **Section 8.6.3 "16-Bit Byte Select Mode"** and **Section 8.7 "8-Bit Data Width Mode"**.

TABLE 8-2: ADDRESS AND DATA LINES FOR DIFFERENT ADDRESS AND DATA WIDTHS

Data Width	Address Width	Multiplexed Data and Address Lines (and Corresponding Ports)	Address Only Lines (and Corresponding Ports)	Ports Available for I/O
8-bit	12-bit	AD<7:0> (PORTD<7:0>)	AD<11:8> (PORTE<3:0>)	PORTE<7:4>, All of PORTH
	16-bit		AD<15:8> (PORTE<7:0>)	All of PORTH
	20-bit		A<19:16>, AD<15:8> (PORTH<3:0>, PORTE<7:0>)	—
16-bit	16-bit	AD<15:0> (PORTD<7:0>, PORTE<7:0>)	—	All of PORTH
	20-bit		A<19:16> (PORTH<3:0>)	—

8.3 Wait States

While it may be assumed that external memory devices will operate at the microcontroller clock rate, this is often not the case. In fact, many devices require longer times to write or retrieve data than the time allowed by the execution of table read or table write operations.

To compensate for this, the External Memory Bus can be configured to add a fixed delay to each table operation using the bus. Wait states are enabled by setting the WAIT Configuration bit. When enabled, the amount of delay is set by the WAIT<1:0> bits (MEMCON<5:4>). The delay is based on multiples of microcontroller instruction cycle time and is added following the instruction cycle when the table operation is executed. The range is from no delay to 3 TCY (default value).

8.4 Port Pin Weak Pull-ups

With the exception of the upper address lines, A<19:16>, the pins associated with the External Memory Bus are equipped with weak pull-ups. The pull-ups are controlled by the upper nibble of the PADCFG register (PADCFG<7:4>). They are named RDPU, REPU, RHPU and RJPU, and control pull-ups on PORTD, PORTE, PORTH and PORTJ, respectively. Setting one of these bits enables the corresponding pull-ups for that port. All pull-ups are disabled by default on all device Resets.

In Extended Microcontroller mode, the port pull-ups can be useful in preserving the memory state on the external bus while the bus is temporarily disabled (EBDIS = 1).

8.5 Program Memory Modes and the External Memory Bus

The PIC18FXXJ94 of devices is capable of operating in one of two program memory modes, using combinations of on-chip and external program memory. The functions of the multiplexed port pins depend on the program memory mode selected, as well as the setting of the EBDIS bit.

In **Microcontroller Mode,** the bus is not active and the pins have their port functions only. Writes to the MEMCOM register are not permitted. The Reset value of EBDIS ('0') is ignored and the ABWx pins behave as I/O ports.

In **Extended Microcontroller Mode,** the external program memory bus shares I/O port functions on the pins. When the device is fetching or doing table read/table write operations on the external program memory space, the pins will have the external bus function.

If the device is fetching and accessing internal program memory locations only, the EBDIS control bit will change the pins from external memory to I/O port functions. When EBDIS = 0, the pins function as the external bus. When EBDIS = 1, the pins function as I/O ports.

If the device fetches or accesses external memory while EBDIS = 1, the pins will switch to the external bus. If the EBDIS bit is set by a program executing from external memory, the action of setting the bit will be delayed until the program branches into the internal memory. At that time, the pins will change from external bus to I/O ports.

If the device is executing out of internal memory when EBDIS = 0, the memory bus address/data and control pins will not be active. They will go to a state where the active address/data pins are tri-state; the $\overline{CE}$, $\overline{OE}$, $\overline{WRH}$, $\overline{WRL}$, $\overline{UB}$ and $\overline{LB}$ signals are '1', and ALE and BA0 are '0'. Note that only those pins associated with the current address width are forced to tri-state; the other pins continue to function as I/O. In the case of 16-bit address width, for example, only AD<15:0> (PORTD and PORTE) are affected; A<19:16> (PORTH<3:0>) continue to function as I/O.

In all external memory modes, the bus takes priority over any other peripherals that may share pins with it. This includes the Parallel Master Port and serial communication modules which would otherwise take priority over the I/O port.

8.6 16-Bit Data Width Modes

In 16-Bit Data Width mode, the external memory interface can be connected to external memories in three different configurations:

- 16-Bit Byte Write
- 16-Bit Word Write
- 16-Bit Byte Select

The configuration to be used is determined by the WM<1:0> bits in the MEMCON register (MEMCON<1:0>). These three different configurations allow the designer maximum flexibility in using both 8-bit and 16-bit devices with 16-bit data.

For all 16-bit modes, the Address Latch Enable (ALE) pin indicates that the address bits, AD<15:0>, are available on the external memory interface bus. Following the address latch, the Output Enable ($\overline{OE}$) signal will enable both bytes of program memory at once to form a 16-bit instruction word. The Chip Enable ($\overline{CE}$ signal) is active at any time that the microcontroller accesses external memory, whether reading or writing; it is inactive (asserted high) whenever the device is in Sleep mode.

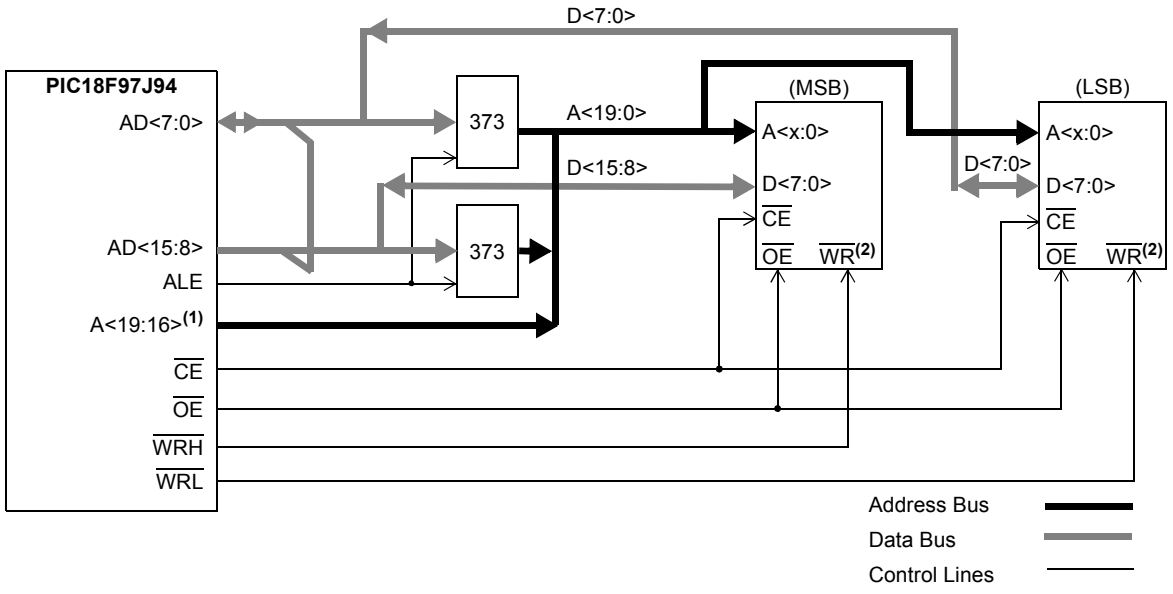
In Byte Select mode, JEDEC® standard Flash memories will require BA0 for the byte address line and one I/O line to select between Byte and Word mode. The other 16-bit modes do not need BA0. JEDEC standard static RAM memories will use the $\overline{UB}$ or $\overline{LB}$ signals for byte selection.

8.6.1 16-BIT BYTE WRITE MODE

Figure 8-1 shows an example of 16-Bit Byte Write mode for PIC18FXXJ94 devices. This mode is used for two separate 8-bit memories connected for 16-bit operation. This generally includes basic EPROM and Flash devices. It allows table writes to byte-wide external memories.

During a TBLWT instruction cycle, the TABLAT data is presented on the upper and lower bytes of the AD<15:0> bus. The appropriate $\overline{WRH}$ or $\overline{WRL}$ control line is strobed on the LSb of the TBLPTR.

FIGURE 8-1: 16-BIT BYTE WRITE MODE EXAMPLE



Note 1: Upper order address lines are used only for 20-bit address widths.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

8.6.2 16-BIT WORD WRITE MODE

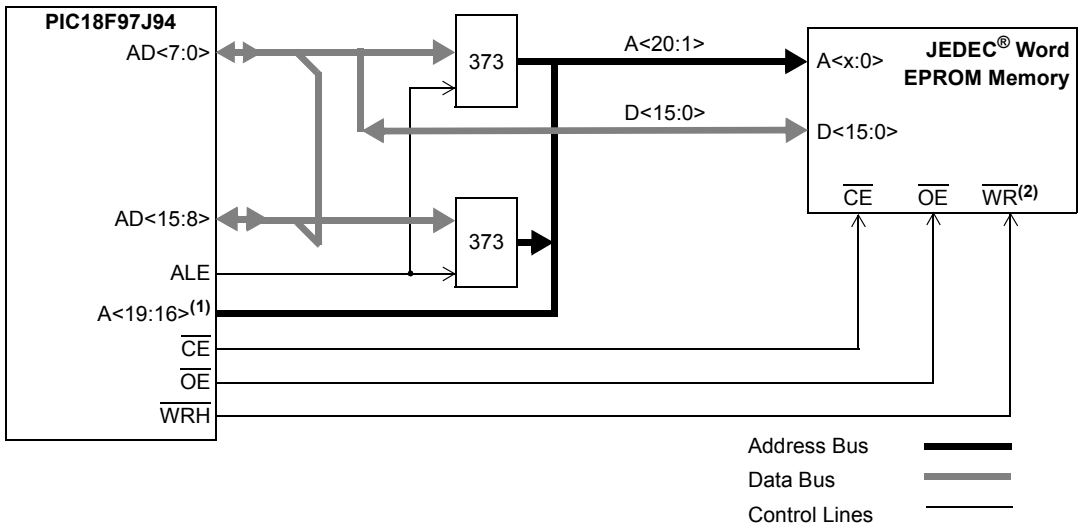
Figure 8-2 shows an example of 16-Bit Word Write mode for PIC18FXXJ94 devices. This mode is used for word-wide memories, which includes some of the EPROM and Flash-type memories. This mode allows opcode fetches and table reads from all forms of 16-bit memory, and table writes to any type of word-wide external memories. This method makes a distinction between TBLWT cycles to even or odd addresses.

During a TBLWT cycle to an even address (TBLPTR<0> = 0), the TABLAT data is transferred to a holding latch and the external address data bus is tri-stated for the data portion of the bus cycle. No write signals are activated.

During a TBLWT cycle to an odd address (TBLPTR<0> = 1), the TABLAT data is presented on the upper byte of the AD<15:0> bus. The contents of the holding latch are presented on the lower byte of the AD<15:0> bus.

The $\overline{WRH}$ signal is strobed for each write cycle; the $\overline{WRL}$ pin is unused. The signal on the BA0 pin indicates the LSb of the TBLPTR, but it is left unconnected. Instead, the $\overline{UB}$ and $\overline{LB}$ signals are active to select both bytes. The obvious limitation to this method is that the table write must be done in pairs on a specific word boundary to correctly write a word location.

FIGURE 8-2: 16-BIT WORD WRITE MODE EXAMPLE



Note 1: Upper order address lines are used only for 20-bit address widths.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

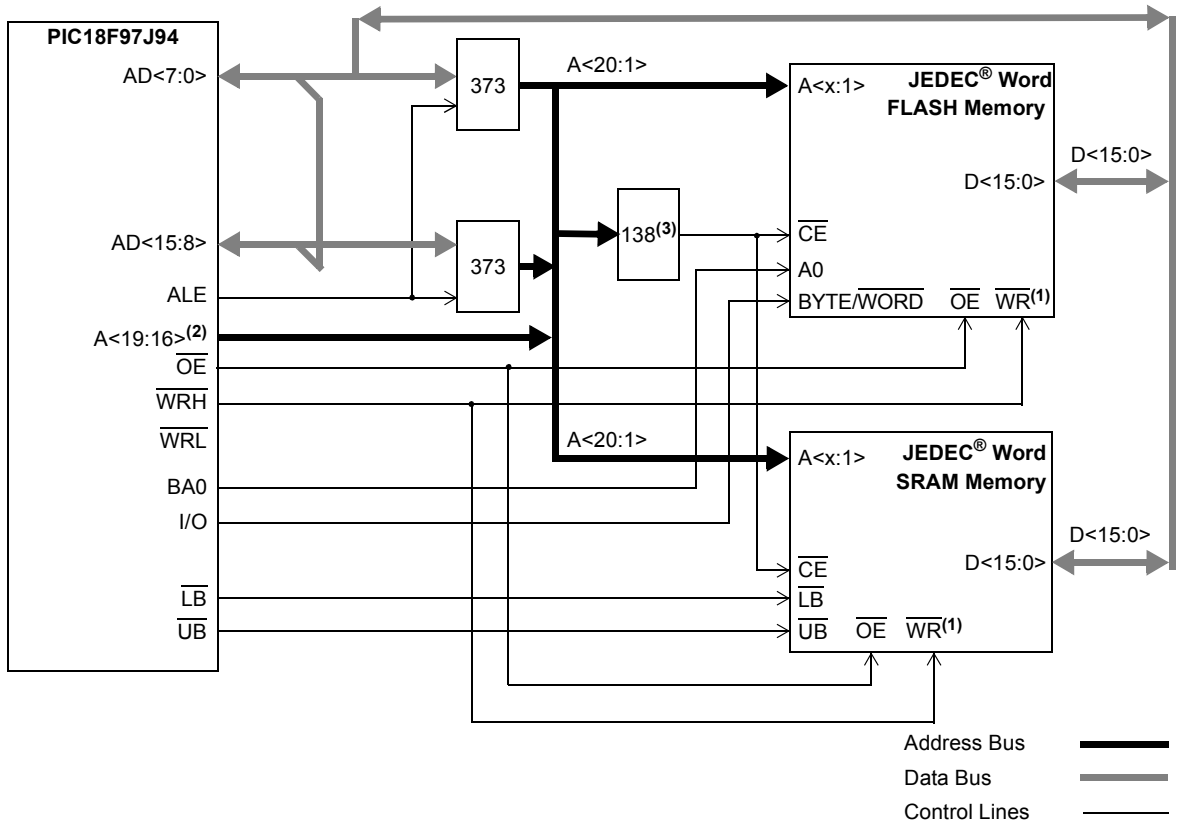
8.6.3 16-BIT BYTE SELECT MODE

Figure 8-3 shows an example of 16-Bit Byte Select mode. This mode allows table write operations to word-wide external memories with byte selection capability. This generally includes both word-wide Flash and SRAM devices.

During a TBLWT cycle, the TABLAT data is presented on the upper and lower byte of the AD<15:0> bus. The $\overline{WRH}$ signal is strobed for each write cycle; the $\overline{WRL}$ pin is not used. The BA0 or $\overline{UB}$/$\overline{LB}$ signals are used to select the byte to be written, based on the Least Significant bit of the TBLPTR register.

Flash and SRAM devices use different control signal combinations to implement Byte Select mode. JEDEC standard Flash memories require that a controller I/O port pin be connected to the memory's BYTE/$\overline{WORD}$ pin to provide the select signal. They also use the BA0 signal from the controller as a byte address. JEDEC standard static RAM memories, on the other hand, use the $\overline{UB}$ or $\overline{LB}$ signals to select the byte.

FIGURE 8-3: 16-BIT BYTE SELECT MODE EXAMPLE



Note 1: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

2: Upper order address lines are used only for 20-bit address width.

3: Demultiplexing is only required when multiple memory devices are accessed.

8.6.4 16-BIT MODE TIMING

The presentation of control signals on the External Memory Bus is different for the various operating modes. Typical signal timing diagrams are shown in Figure 8-4 and Figure 8-5.

FIGURE 8-4: EXTERNAL MEMORY BUS TIMING FOR TBLRD (EXTENDED MICROCONTROLLER MODE)

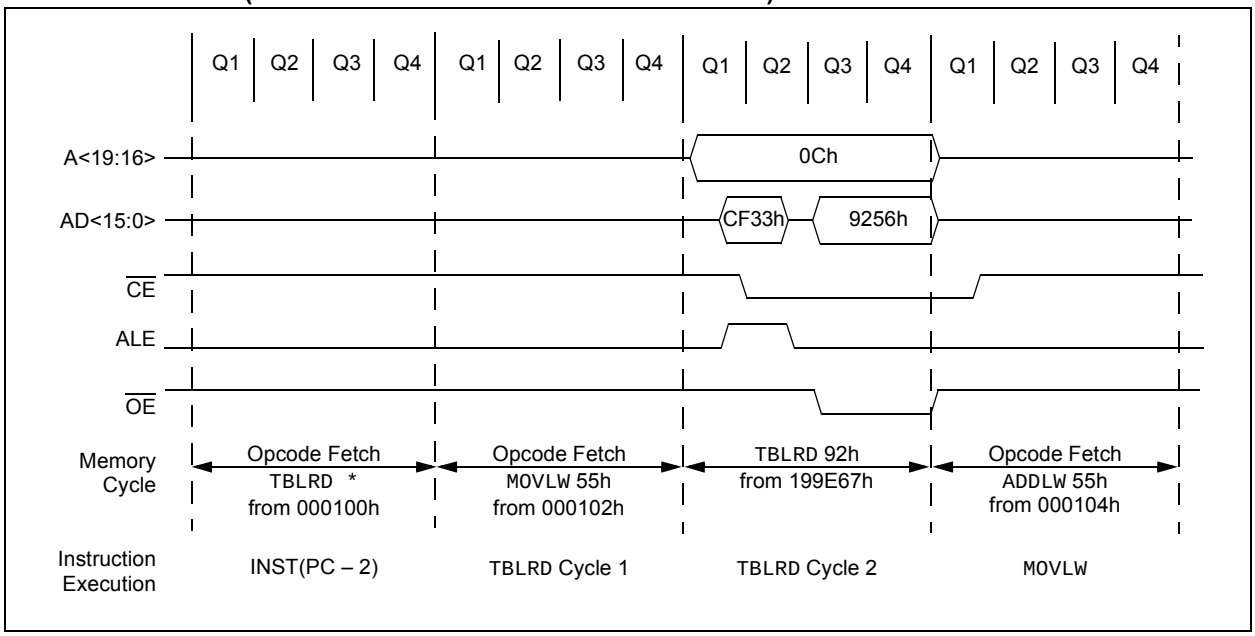
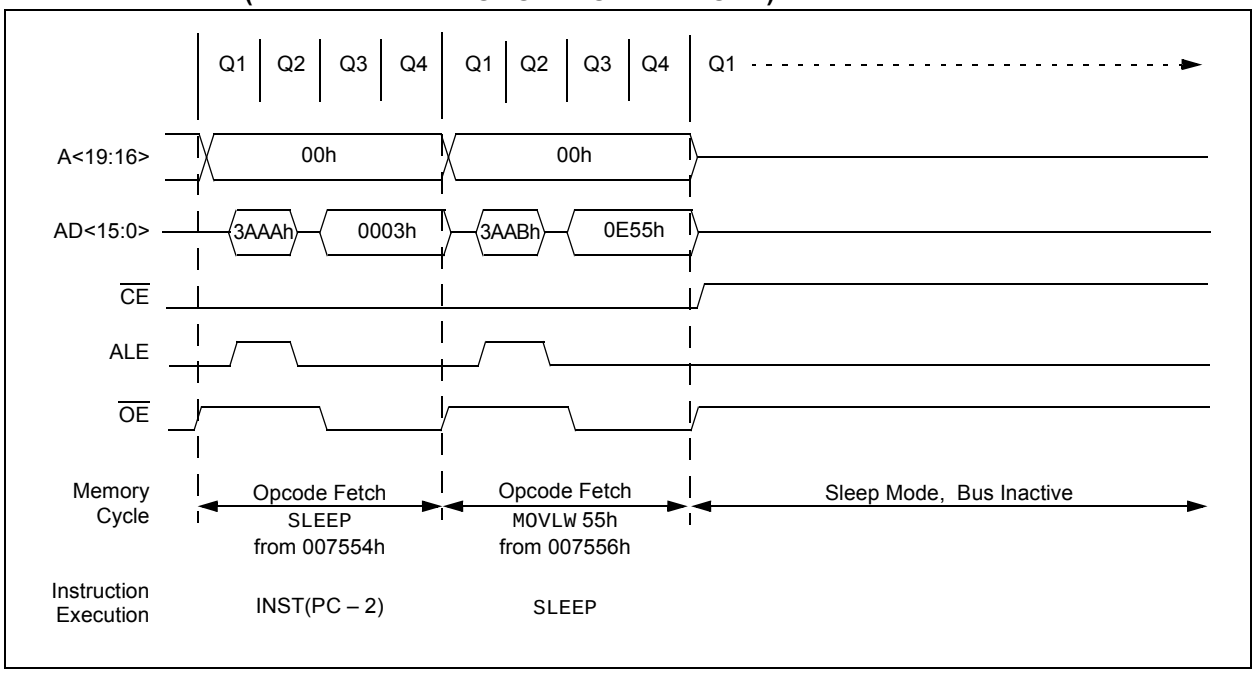


FIGURE 8-5: EXTERNAL MEMORY BUS TIMING FOR SLEEP (EXTENDED MICROCONTROLLER MODE)



8.7 8-Bit Data Width Mode

In 8-Bit Data Width mode, the External Memory Bus operates only in Multiplexed mode; that is, data shares the 8 Least Significant bits of the address bus.

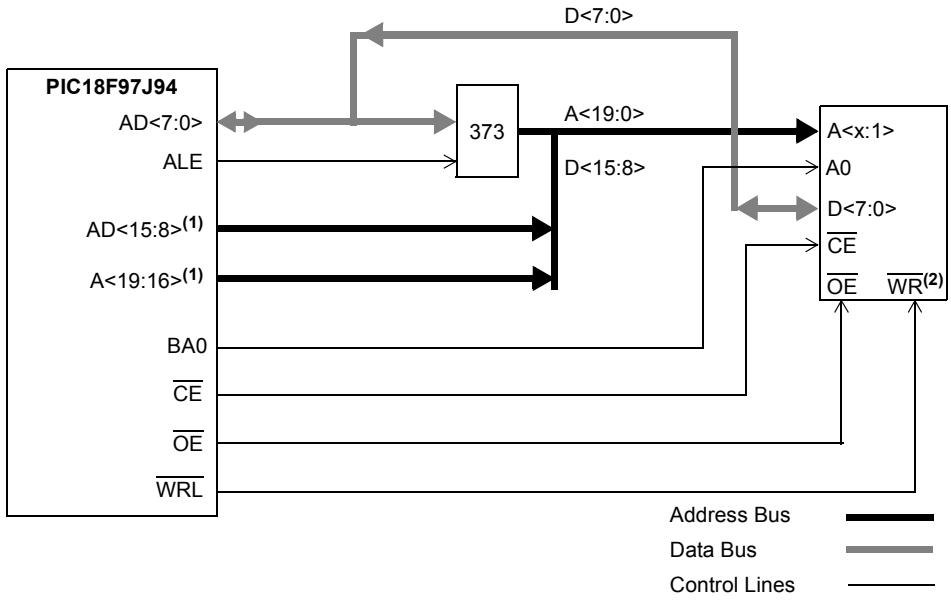
Figure 8-6 shows an example of 8-Bit Multiplexed mode for 100-pin devices. This mode is used for a single, 8-bit memory, connected for 16-bit operation. The instructions will be fetched as two 8-bit bytes on a shared data/address bus. The two bytes are sequentially fetched within one instruction cycle (TCY). Therefore, the designer must choose external memory devices, according to timing calculations based on 1/2 TCY (2 times the instruction rate). For proper memory speed selection, glue logic propagation delay times must be considered, along with setup and hold times.

The Address Latch Enable (ALE) pin indicates that the address bits, AD<15:0>, are available on the External Memory Bus interface. The Output Enable ($\overline{OE}$) signal will enable one byte of program memory for a portion of the instruction cycle, then BA0 will change and the second byte will be enabled to form the 16-bit instruction word. The Least Significant bit of the address, BA0, must be connected to the memory devices in this mode. The Chip Enable ($\overline{CE}$) signal is active at any time that the microcontroller accesses external memory, whether reading or writing. It is inactive (asserted high) whenever the device is in Sleep mode.

This generally includes basic EPROM and Flash devices. It allows table writes to byte-wide external memories.

During a TBLWT instruction cycle, the TABLAT data is presented on the upper and lower bytes of the AD<15:0> bus. The appropriate level of the BA0 control line is strobed on the LSb of the TBLPTR.

FIGURE 8-6: 8-BIT MULTIPLEXED MODE EXAMPLE



Note 1: Upper order address bits are only used for 20-bit address width. The upper AD byte is used for all address widths except 8-bit.

2: This signal only applies to table writes. See **Section 7.1 "Table Reads and Table Writes"**.

8.7.1 8-BIT MODE TIMING

The presentation of control signals on the External Memory Bus is different for the various operating modes. Typical signal timing diagrams are shown in Figure 8-7 and Figure 8-8.

FIGURE 8-7: EXTERNAL MEMORY BUS TIMING FOR TBLRD (EXTENDED MICROCONTROLLER MODE)

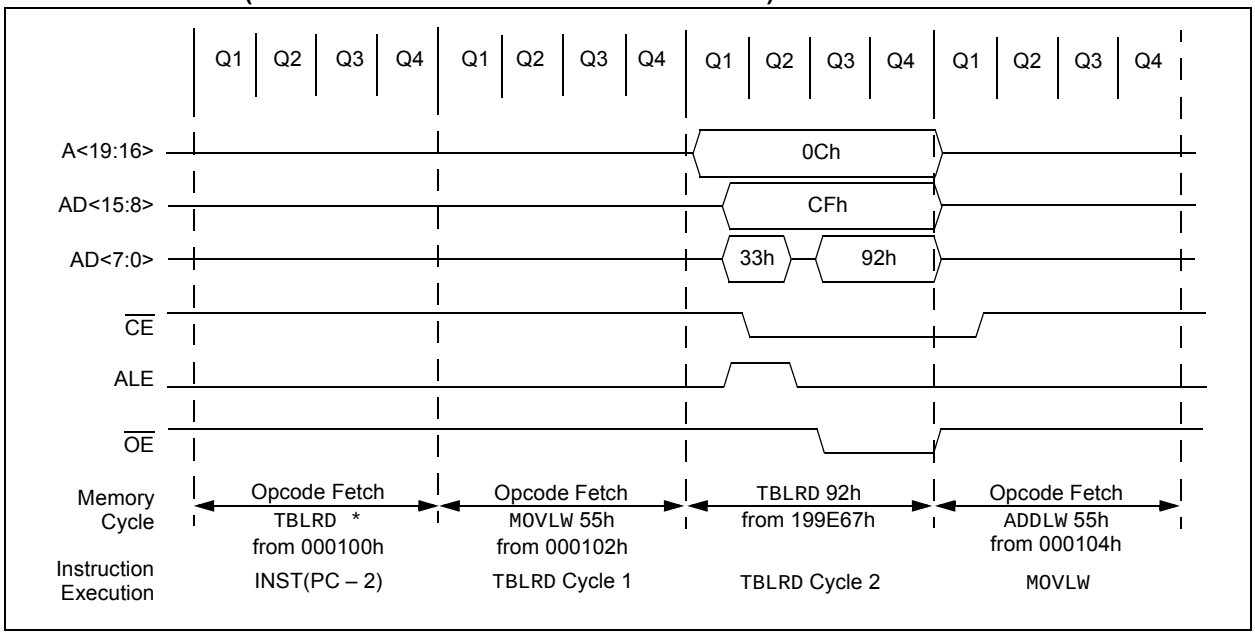
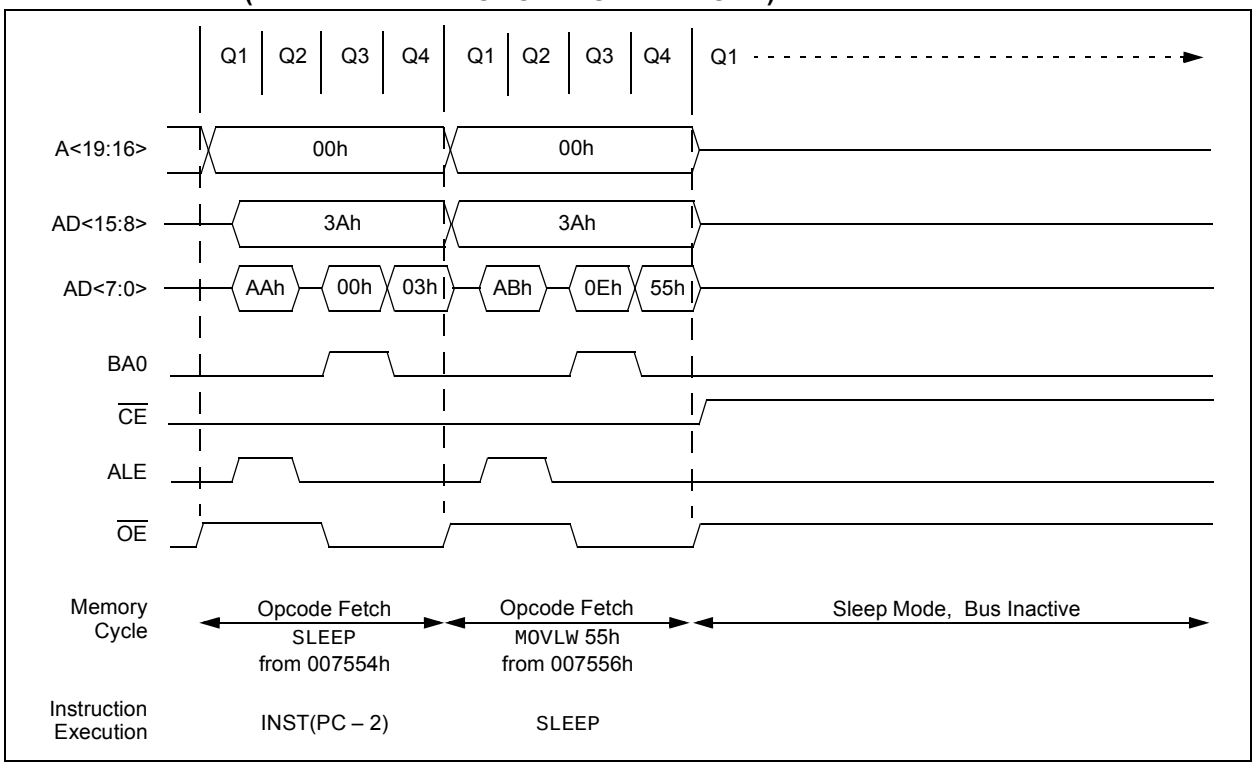


FIGURE 8-8: EXTERNAL MEMORY BUS TIMING FOR SLEEP (EXTENDED MICROCONTROLLER MODE)



8.8 Operation in Power-Managed Modes

In alternate, power-managed Run modes, the external bus continues to operate normally. If a clock source with a lower speed is selected, bus operations will run at that speed. In these cases, excessive access times for the external memory may result if Wait states have been enabled and added to external memory operations. If operations in a lower power Run mode are anticipated, users should provide in their applications for adjusting memory access times at the lower clock speeds.

In Sleep and Idle modes, the microcontroller core does not need to access data; bus operations are suspended. The state of the external bus is frozen, with the address/data pins and most of the control pins holding at the same state they were in when the mode was invoked. The only potential changes are to the $\overline{CE}$, $\overline{LB}$ and $\overline{UB}$ pins, which are held at logic high.

TABLE 8-3: REGISTERS ASSOCIATED WITH THE EXTERNAL MEMORY BUS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
MEMCON(1)	EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
PADCFG	RDPU	REPU	RFPU	RGPU	RHPU	RJPU	RKPU	RLPU
PMD4	CMP1MD	CMP2MD	CMP3MD	USBMD	IOCMD	LVDMD	—	EMBMD

Legend: — = unimplemented, read as '0'. Shaded cells are not used during External Memory Bus access.

Note 1: This register is unimplemented on 64-pin devices read as '0'.

9.0 8 x 8 HARDWARE MULTIPLIER

9.1 Introduction

All PIC18 devices include an 8 x 8 hardware multiplier as part of the ALU. The multiplier performs an unsigned operation and yields a 16-bit result that is stored in the product register pair, PRODH:PRODL. The multiplier's operation does not affect any flags in the STATUS register.

Making multiplication a hardware operation allows it to be completed in a single instruction cycle. This has the advantages of higher computational throughput and reduced code size for multiplication algorithms and allows PIC18 devices to be used in many applications previously reserved for digital-signal processors. A comparison of various hardware and software multiply operations, along with the savings in memory and execution time, is shown in Table 9-1.

9.2 Operation

Example 9-1 shows the instruction sequence for an 8 x 8 unsigned multiplication. Only one instruction is required when one of the arguments is already loaded in the WREG register.

Example 9-2 shows the sequence to do an 8 x 8 signed multiplication. To account for the sign bits of the arguments, each argument's Most Significant bit (MSb) is tested and the appropriate subtractions are done.

EXAMPLE 9-1: 8 x 8 UNSIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W     ;
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
```

EXAMPLE 9-2: 8 x 8 SIGNED MULTIPLY ROUTINE

```
MOVF    ARG1, W
MULWF   ARG2        ; ARG1 * ARG2 ->
                    ; PRODH:PRODL
BTFSC   ARG2, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG1
MOVF    ARG2, W
BTFSC   ARG1, SB    ; Test Sign Bit
SUBWF   PRODH, F    ; PRODH = PRODH
                    ;         - ARG2
```

TABLE 9-1: PERFORMANCE COMPARISON FOR VARIOUS MULTIPLY OPERATIONS

Routine	Multiply Method	Program Memory (Words)	Cycles (Max)	Time			
				@ 64 MHz	@ 48 MHz	@ 10 MHz	@ 4 MHz
8 x 8 Unsigned	Without Hardware Multiply	13	69	4.3 µs	5.7 µs	27.6 µs	69 µs
	Hardware Multiply	1	1	62.5 ns	83.3 ns	400 ns	1 µs
8 x 8 Signed	Without Hardware Multiply	33	91	5.6 µs	7.5 µs	36.4 µs	91 µs
	Hardware Multiply	6	6	375 ns	500 ns	2.4 µs	6 µs
16 x 16 Unsigned	Without Hardware Multiply	21	242	15.1 µs	20.1 µs	96.8 µs	242 µs
	Hardware Multiply	28	28	1.7 µs	2.3 µs	11.2 µs	28 µs
16 x 16 Signed	Without Hardware Multiply	52	254	15.8 µs	21.2 µs	101.6 µs	254 µs
	Hardware Multiply	35	40	2.5 µs	3.3 µs	16.0 µs	40 µs

Example 9-3 shows the sequence to do a 16 x 16 unsigned multiplication. Equation 9-1 shows the algorithm that is used. The 32-bit result is stored in four registers (RES3:RES0).

EQUATION 9-1: 16 x 16 UNSIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \cdot ARG2H:ARG2L \\
&= (ARG1H \cdot ARG2H \cdot 2^{16}) + \\
&\quad (ARG1H \cdot ARG2L \cdot 2^{8}) + \\
&\quad (ARG1L \cdot ARG2H \cdot 2^{8}) + \\
&\quad (ARG1L \cdot ARG2L)
\end{aligned}
$$

EXAMPLE 9-3: 16 x 16 UNSIGNED MULTIPLY ROUTINE

```
    MOVF    ARG1L, W
    MULWF   ARG2L           ; ARG1L * ARG2L->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES1     ;
    MOVFF   PRODL, RES0     ;
;
    MOVF    ARG1H, W
    MULWF   ARG2H           ; ARG1H * ARG2H->
                            ; PRODH:PRODL
    MOVFF   PRODH, RES3     ;
    MOVFF   PRODL, RES2     ;
;
    MOVF    ARG1L, W
    MULWF   ARG2H           ; ARG1L * ARG2H->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
;
    MOVF    ARG1H, W        ;
    MULWF   ARG2L           ; ARG1H * ARG2L->
                            ; PRODH:PRODL
    MOVF    PRODL, W        ;
    ADDWF   RES1, F         ; Add cross
    MOVF    PRODH, W        ; products
    ADDWFC  RES2, F         ;
    CLRF    WREG            ;
    ADDWFC  RES3, F         ;
```

Example 9-4 shows the sequence to do a 16 x 16 signed multiply. Equation 9-2 shows the algorithm used. The 32-bit result is stored in four registers (RES3:RES0). To account for the sign bits of the arguments, the MSb for each argument pair is tested and the appropriate subtractions are done.

EQUATION 9-2: 16 x 16 SIGNED MULTIPLICATION ALGORITHM

$$
\begin{aligned}
RES3:RES0 &= ARG1H:ARG1L \cdot ARG2H:ARG2L \\
&= (ARG1H \cdot ARG2H \cdot 2^{16}) + \\
&\quad (ARG1H \cdot ARG2L \cdot 2^{8}) + \\
&\quad (ARG1L \cdot ARG2H \cdot 2^{8}) + \\
&\quad (ARG1L \cdot ARG2L) + \\
&\quad (-1 \cdot ARG2H\langle 7\rangle \cdot ARG1H:ARG1L \cdot 2^{16}) + \\
&\quad (-1 \cdot ARG1H\langle 7\rangle \cdot ARG2H:ARG2L \cdot 2^{16})
\end{aligned}
$$

EXAMPLE 9-4: 16 x 16 SIGNED MULTIPLY ROUTINE

```
   MOVF    ARG1L, W
   MULWF   ARG2L           ; ARG1L * ARG2L ->
                           ; PRODH:PRODL
   MOVFF   PRODH, RES1     ;
   MOVFF   PRODL, RES0     ;
;
   MOVF    ARG1H, W
   MULWF   ARG2H           ; ARG1H * ARG2H ->
                           ; PRODH:PRODL
   MOVFF   PRODH, RES3     ;
   MOVFF   PRODL, RES2     ;
;
   MOVF    ARG1L, W
   MULWF   ARG2H           ; ARG1L * ARG2H ->
                           ; PRODH:PRODL
   MOVF    PRODL, W        ;
   ADDWF   RES1, F         ; Add cross
   MOVF    PRODH, W        ; products
   ADDWFC  RES2, F         ;
   CLRF    WREG            ;
   ADDWFC  RES3, F         ;
;
   MOVF    ARG1H, W        ;
   MULWF   ARG2L           ; ARG1H * ARG2L ->
                           ; PRODH:PRODL
   MOVF    PRODL, W        ;
   ADDWF   RES1, F         ; Add cross
   MOVF    PRODH, W        ; products
   ADDWFC  RES2, F         ;
   CLRF    WREG            ;
   ADDWFC  RES3, F         ;
;
   BTFSS   ARG2H, 7        ; ARG2H:ARG2L neg?
   BRA  SIGN_ARG1          ; no, check ARG1
   MOVF    ARG1L, W        ;
   SUBWF   RES2            ;
   MOVF    ARG1H, W        ;
   SUBWFB  RES3            ;
SIGN_ARG1
   BTFSS   ARG1H, 7        ; ARG1H:ARG1L neg?
   BRA     CONT_CODE       ; no, done
   MOVF    ARG2L, W        ;
   SUBWF   RES2            ;
   MOVF    ARG2H, W        ;
   SUBWFB  RES3
;
CONT_CODE
   :
```

10.0 INTERRUPTS

Members of the PIC18F97J94 family of devices have multiple interrupt sources and an interrupt priority feature that allows most interrupt sources to be assigned a high-priority level or a low-priority level. The high-priority interrupt vector is at 0008h and the low-priority interrupt vector is at 0018h. High-priority interrupt events will interrupt any low-priority interrupts that may be in progress.

The registers for controlling interrupt operation are:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2, PIR3, PIR4, PIR5 and PIR6
- PIE1, PIE2, PIE3, PIE4, PIE5 and PIE6
- IPR1, IPR2, IPR3, IPR5, IPR5 and IPR6

It is recommended that the Microchip header files, supplied with MPLAB® IDE, be used for the symbolic bit names in these registers. This allows the assembler/compiler to automatically take care of the placement of these bits within the specified register.

In general, interrupt sources have three bits to control their operation. They are:

- **Flag bit** – Indicating that an interrupt event occurred
- **Enable bit** – Enabling program execution to branch to the interrupt vector address when the flag bit is set
- **Priority bit** – Specifying high priority or low priority

10.1 Mid-Range Compatibility

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled and interrupts are compatible with PIC® microcontroller mid-range devices. In Compatibility mode, the interrupt priority bits of the IPRx registers have no effect. The PEIE/GIEL bit of the INTCON register is the global interrupt enable for the peripherals. The PEIE/GIEL bit disables only the peripheral interrupt sources and enables the peripheral interrupt sources when the GIE/GIEH bit is also set. The GIE/GIEH bit of the INTCON register is the global interrupt enable which enables all non-peripheral interrupt sources and disables all interrupt sources, including the peripherals. All interrupts branch to address 0008h in Compatibility mode.

10.2 Interrupt Priority

The interrupt priority feature is enabled by setting the IPEN bit of the RCON register. When interrupt priority is enabled the GIE/GIEH and PEIE/GIEL global interrupt enable bits of Compatibility mode are replaced by the GIEH high priority, and GIEL low priority, global interrupt enables. When set, the GIEH bit of the INTCON register enables all interrupts that have their associated IPRx register or INTCONx register priority bit set (high priority). When clear, the GIEH bit disables all interrupt sources including those selected as low priority. When clear, the GIEL bit of the INTCON register disables only the interrupts that have their associated priority bit cleared (low priority). When set, the GIEL bit enables the low priority sources when the GIEH bit is also set. When the interrupt flag, enable bit and appropriate Global Interrupt Enable (GIE) bit are all set, the interrupt will vector immediately to address 0008h for high priority, or 0018h for low priority, depending on level of the interrupting source's priority bit. Individual interrupts can be disabled through their corresponding interrupt enable bits.

10.3 Interrupt Response

When an interrupt is responded to, the Global Interrupt Enable bit is cleared to disable further interrupts. The GIE/GIEH bit is the global interrupt enable when the IPEN bit is cleared. When the IPEN bit is set, enabling interrupt priority levels, the GIEH bit is the high priority global interrupt enable and the GIEL bit is the low priority global interrupt enable. High priority interrupt sources can interrupt a low priority interrupt. Low priority interrupts are not processed while high priority interrupts are in progress.

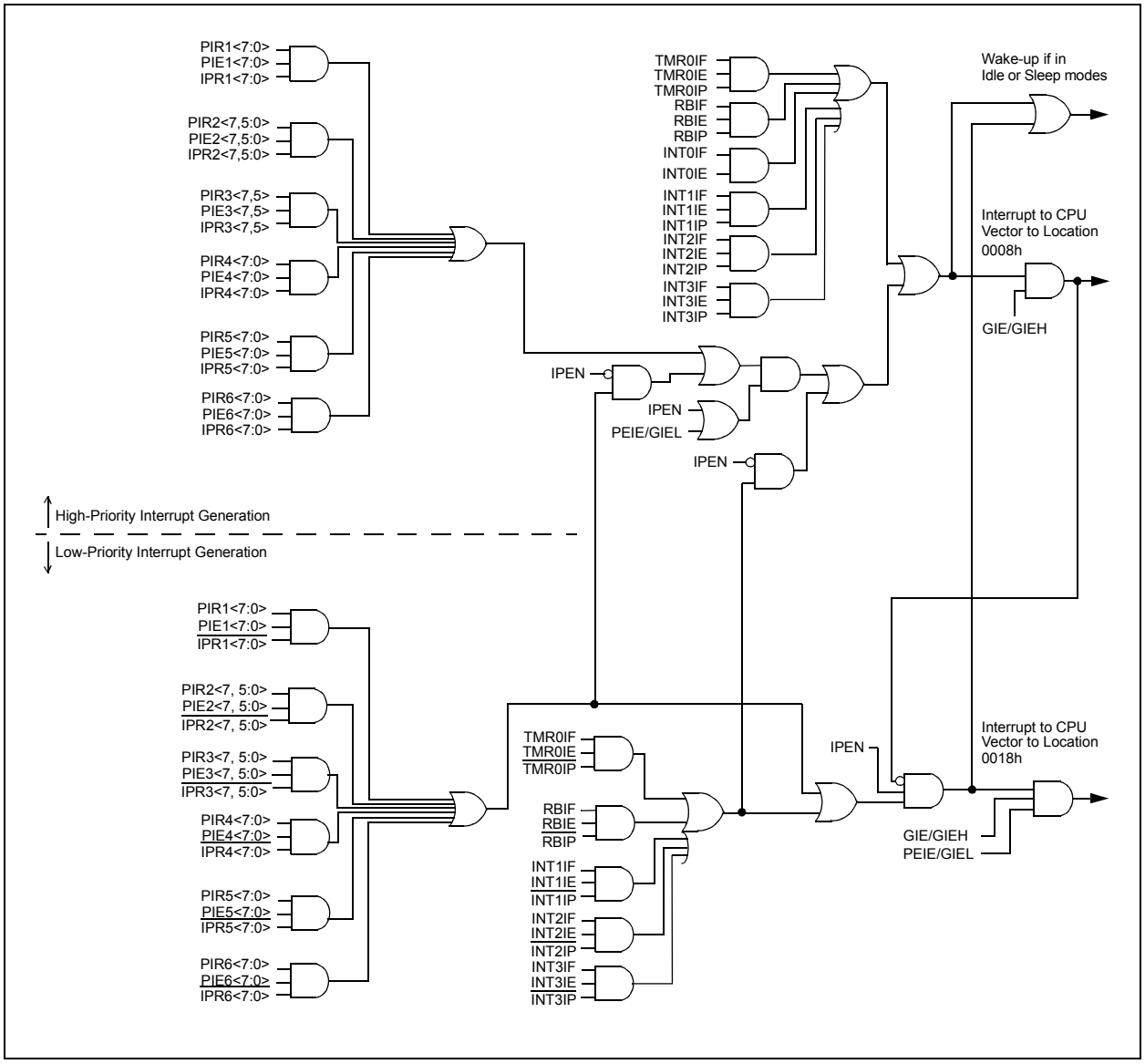
The return address is pushed onto the stack and the PC is loaded with the interrupt vector address (0008h or 0018h). Once in the Interrupt Service Routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits in the INTCONx and PIRx registers. The interrupt flag bits must be cleared by software before re-enabling interrupts to avoid repeating the same interrupt.

The "return from interrupt" instruction, RETFIE, exits the interrupt routine and sets the GIE/GIEH bit (GIEH or GIEL if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INT pins or the PORTB interrupt-on-change, the interrupt latency will be three to four instruction cycles. The exact latency is the same for one-cycle or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bits or the Global Interrupt Enable bit.

Note:	Do not use the MOVFF instruction to modify any of the Interrupt Control registers while **any** interrupt is enabled. Doing so may cause erratic microcontroller behavior.

FIGURE 10-1: PIC18F97J94 FAMILY INTERRUPT LOGIC



10.4 INTCON Registers

The INTCON registers are readable and writable registers that contain various enable, priority and flag bits.

Note: Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 10-1: INTCON: INTERRUPT CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	IOCIE	TMR0IF	INT0IF	IOCIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts including peripherals
When IPEN = 1:
1 = Enables all high-priority interrupts
0 = Disables all interrupts including low priority

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low-priority peripheral interrupts
0 = Disables all low-priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **IOCIE:** I/O Change Interrupt Enable bit
1 = Enables the I/O port change interrupt
0 = Disables the I/O port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register has not overflowed

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **IOCIF:** I/O Port Change Interrupt Flag bit
1 = At least one of the IOC<7:0> pins changed state (must be cleared by clearing all the IOCF bits in the IOC module)
0 = None of the IOC<7:0> pins have changed state

REGISTER 10-2: INTCON2: INTERRUPT CONTROL REGISTER 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG0	INTEDG1	INTEDG2	INTEDG3	TMR0IP	INT3IP	IOCIP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
1 = All PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG0:** External Interrupt 0 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 5 **INTEDG1:** External Interrupt 1 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 4 **INTEDG2:** External Interrupt 2 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 3 **INTEDG3:** External Interrupt 3 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 2 **TMR0IP:** TMR0 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **INT3IP:** INT3 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **IOCIP:** RB Port Change Interrupt Priority bit
1 = High priority
0 = Low priority

Note: Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

REGISTER 10-3: INTCON3: INTERRUPT CONTROL REGISTER 3

R/W-1	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT2IP	INT1IP	INT3IE	INT2IE	INT1IE	INT3IF	INT2IF	INT1IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **INT2IP:** INT2 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **INT1IP:** INT1 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **INT3IE:** INT3 External Interrupt Enable bit
1 = Enables the INT3 external interrupt
0 = Disables the INT3 external interrupt

bit 4 **INT2IE:** INT2 External Interrupt Enable bit
1 = Enables the INT2 external interrupt
0 = Disables the INT2 external interrupt

bit 3 **INT1IE:** INT1 External Interrupt Enable bit
1 = Enables the INT1 external interrupt
0 = Disables the INT1 external interrupt

bit 2 **INT3IF:** INT3 External Interrupt Flag bit
1 = The INT3 external interrupt occurred (must be cleared in software)
0 = The INT3 external interrupt did not occur

bit 1 **INT2IF:** INT2 External Interrupt Flag bit
1 = The INT2 external interrupt occurred (must be cleared in software)
0 = The INT2 external interrupt did not occur

bit 0 **INT1IF:** INT1 External Interrupt Flag bit
1 = The INT1 external interrupt occurred (must be cleared in software)
0 = The INT1 external interrupt did not occur

Note: Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit. User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt. This feature allows for software polling.

10.5 PIR Registers

The PIR registers contain the individual flag bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are six Peripheral Interrupt Request (Flag) registers (PIR1 through PIR5).

Note 1: Interrupt flag bits are set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the Global Interrupt Enable bit, GIE (INTCON<7>).

2: User software should ensure the appropriate interrupt flag bits are cleared prior to enabling an interrupt and after servicing that interrupt.

REGISTER 10-4: PIR1: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 1

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIF	ADIF	RC1IF	TX1IF	SSP1IF	TMR1GIF	TMR2IF	TMR1IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **PSPIF:** Parallel Slave Port Read/Write Interrupt Flag bit
1 = A read or write operation has taken place (must be cleared in software)
0 = No read or write operation has occurred

bit 6 **ADIF:** A/D Converter Interrupt Flag bit
1 = An A/D conversion completed (must be cleared in software)
0 = The A/D conversion is not complete

bit 5 **RC1IF:** EUSART1 Receive Interrupt Flag bit
1 = The EUSART1 receive buffer, RCREG1, is full (cleared when RCREG1 is read)
0 = The EUSART1 receive buffer is empty

bit 4 **TX1IF:** EUSART1 Transmit Interrupt Flag bit
1 = The EUSART1 transmit buffer, TXREG1, is empty (cleared when TXREG1 is written)
0 = The EUSART1 transmit buffer is full

bit 3 **SSP1IF:** Master Synchronous Serial Port 1 Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 2 **TMR1GIF:** Timer1 Gate Interrupt Flag bit
1 = Timer gate interrupt occurred (must be cleared in software)
0 = No timer gate interrupt occurred

bit 1 **TMR2IF:** TMR2 to PR2 Match Interrupt Flag bit
1 = TMR2 to PR2 match occurred (must be cleared in software)
0 = No TMR2 to PR2 match occurred

bit 0 **TMR1IF:** TMR1 Overflow Interrupt Flag bit
1 = TMR1 register overflowed (must be cleared in software)
0 = TMR1 register did not overflow

REGISTER 10-5: PIR2: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIF	SSP2IF	BCL2IF	USBIF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **OSCFIF:** Oscillator Fail Interrupt Flag bit
1 = Device oscillator failed, clock input has changed to INTOSC (bit must be cleared in software)
0 = Device clock operating

bit 6 **SSP2IF:** Master Synchronous Serial Port 2 Interrupt Flag bit
1 = The transmission/reception is complete (must be cleared in software)
0 = Waiting to transmit/receive

bit 5 **BCL2IF:** Bus Collision Interrupt Flag bit
1 = A bus collision has occurred while the MSSP1 module configured in I^2C master was transmitting (must be cleared in software)
0 = No bus collision occurred

bit 4 **USBIF:** Oscillator Fail Interrupt Flag bit
1 = USB requested an interrupt (must be cleared in software)
0 = No USB interrupt request

bit 3 **BCL1IF:** Bus Collision Interrupt Flag bit
1 = A bus collision occurred (bit must be cleared in software)
0 = No bus collision occurred

bit 2 **HLVDIF:** High/Low-Voltage Detect Interrupt Flag bit
1 = A low-voltage condition occurred (bit must be cleared in software)
0 = The device voltage is above the regulator's low-voltage trip point

bit 1 **TMR3IF:** TMR3 Overflow Interrupt Flag bit
1 = TMR3 register overflowed (bit must be cleared in software)
0 = TMR3 register did not overflow

bit 0 **TMR3GIF:** TMR3 Gate Interrupt Flag bit
1 = Timer gate interrupt occurred (bit must be cleared in software)
0 = No timer gate interrupt occurred

REGISTER 10-6: PIR3: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 3

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR5GIF	LCDIF	RC2IF	TX2IF	CTMUIF	CCP2IF	CCP1IF	RTCCIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TMR5GIF:** TMR5 Gate Interrupt Flag bits
1 = TMR gate interrupt occurred (must be cleared in software)
0 = No TMR gate occurred

bit 6 **LCDIF:** LCD Interrupt Flag bit
1 = A write is allowed to the Segment Data Registers
0 = A write is not allowed to the Segment Data Register

bit 5 **RC2IF:** EUSART2 Receive Interrupt Flag bit
1 = The EUSART2 receive buffer, RCREG2, is full (cleared when RCREG2 is read)
0 = The EUSART2 receive buffer is empty

bit 4 **TX2IF:** EUSART2 Transmit Interrupt Flag bit
1 = The EUSART2 transmit buffer, TXREG2, is empty (cleared when TXREG2 is written)
0 = The EUSART2 transmit buffer is full

bit 3 **CTMUIF:** CTMU Interrupt Flag bit
1 = CTMU interrupt occurred (must be cleared in software)
0 = No CTMU interrupt occurred

bit 2 **CCP2IF:** CCP2 Interrupt Flag bit
Capture mode:
1 = A TMR1/TMR3 register capture occurred (must be cleared in software)
0 = No TMR1/TMR3 register capture occurred
Compare mode:
1 = A TMR1/TMR3 register compare match occurred (must be cleared in software)
0 = No TMR1/TMR3 register compare match occurred
PWM mode:
Unused in this mode.

bit 1 **CCP1IF:** ECCP1 Interrupt Flag bit
Capture mode:
1 = A TMR1/TMR3 register capture occurred (must be cleared in software)
0 = No TMR1/TMR3 register capture occurred
Compare mode:
1 = A TMR1/TMR3 register compare match occurred (must be cleared in software)
0 = No TMR1/TMR3 register compare match occurred
PWM mode:
Unused in this mode.

bit 0 **RTCCIF:** RTCC Interrupt Flag bit
1 = RTCC interrupt occurred (must be cleared in software)
0 = No RTCC interrupt occurred

REGISTER 10-7: PIR4: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 4

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CCP10IF	CCP9IF	CCP8IF	CCP7IF	CCP6IF	CCP5IF	CCP4IF	ECCP3IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CCP10IF:** CCP10 Interrupt Flag bits

Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred

Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred

PWM mode:
Not used in PWM mode.

bit 6 **CCP9IF:** CCP9 Interrupt Flag bits

Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred

Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred

PWM mode:
Not used in PWM mode.

bit 5 **CCP8IF:** CCP8 Interrupt Flag bits

Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred

Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred

PWM mode:
Not used in PWM mode.

bit 4 **CCP7IF:** CCP7 Interrupt Flag bit

1 = Interrupt Flag bits

Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred

Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred

PWM mode:
Not used in PWM mode.

REGISTER 10-7: PIR4: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 4 (CONTINUED)

bit 3 **CCP6IF:** CCP6 Interrupt Flag bits
Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred
Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred
PWM mode:
Not used in PWM mode.

bit 2 **CCP5IF:** CCP5 Interrupt Flag bits
Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred
Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred
PWM mode:
Not used in PWM mode.

bit 1 **CCP4IF:** CCP4 Interrupt Flag bits
Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred
Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred
PWM mode:
Not used in PWM mode.

bit 0 **ECCP3IF:** ECCP3 Interrupt Flag bits
Capture mode:
1 = A TMR register capture occurred (bit must be cleared in software)
0 = No TMR register capture occurred
Compare mode:
1 = A TMR register compare match occurred (must be cleared in software)
0 = No TMR register compare match occurred
PWM mode:
Not used in PWM mode.

REGISTER 10-8: PIR5: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 5

U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	ACTORSIF	ACTLOCKIF	TMR8IF	—	TMR6IF	TMR5IF	TMR4IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6 **ACTORSIF:** Active Clock Tuning Out-of-Range Interrupt Flag bit
1 = Active clock tuning out-of-range occurred
0 = Active tuning out-of-range did not occur

bit 5 **ACTLOCKIF:** Active Clock Tuning Lock Interrupt Flag bit
1 = Active clock tuning lock/unlock occurred
0 = Active clock tuning lock/unlock did not occur

bit 4 **TMR8IF:** TMR8 to PR8 Match Interrupt Flag bit
1 = TMR8 to PR8 match occurred (must be cleared in software)
0 = No TMR8 to PR8 match occurred

bit 3 **Unimplemented:** Read as '0'

bit 2 **TMR6IF:** TMR6 to PR6 Match Interrupt Flag bit
1 = TMR6 to PR6 match occurred (must be cleared in software)
0 = No TMR6 to PR6 match occurred

bit 1 **TMR5IF:** TMR5 Overflow Interrupt Flag bit
1 = TMR5 register overflowed (must be cleared in software)
0 = TMR5 register did not overflow

bit 0 **TMR4IF:** TMR4 to PR4 Match Interrupt Flag bit
1 = TMR4 to PR4 match occurred (must be cleared in software)
0 = No TMR4 to PR4 match occurred

REGISTER 10-9: PIR6: PERIPHERAL INTERRUPT REQUEST (FLAG) REGISTER 6

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
RC4IF	TX4IF	RC3IF	TX3IF	—	CMP3IF	CMP2IF	CMP1IF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **RC4IF:** EUSART4 Receive Interrupt Flag bit
1 = The EUSART4 receive buffer is full (cleared by reading RCREG4)
0 = The EUSART4 receive buffer is empty

bit 6 **TX4IF**: EUSART4 Transmit Interrupt Flag bit
1 = The EUSART4 transmit buffer is empty (cleared by writing to TXREG4)
0 = The EUSART4 transmit buffer is full

bit 5 **RC3IF**: EUSART3 Receive Interrupt Flag bit
1 = The EUSART3 receive buffer is full (cleared by reading RCREG3)
0 = The EUSART3 receive buffer is empty

bit 4 **TX3IF:** EUSART3 Transmit Interrupt Flag bit
1 = The EUSART3 transmit buffer is empty (cleared by writing to TXREG3)
0 = The EUSART3 transmit buffer is full

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **CMP3IF:** CMP3 Interrupt Flag bit
1 = CMP3 interrupt occurred (must be cleared in software)
0 = No CMP3 interrupt occurred

bit 1 **CMP2IF:** CMP2 Interrupt Flag bit
1 = CMP2 interrupt occurred (must be cleared in software)
0 = No CMP2 interrupt occurred

bit 0 **CMP1IF:** CM1 Interrupt Flag bit
1 = CMP1 interrupt occurred (must be cleared in software)
0 = No CMP1 interrupt occurred

10.6 PIE Registers

The PIE registers contain the individual enable bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are six Peripheral Interrupt Enable registers (PIE1 through PIE6). When IPEN (RCON<7>) = 0, the PEIE bit must be set to enable any of these peripheral interrupts.

REGISTER 10-10: PIE1: PERIPHERAL INTERRUPT ENABLE REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PSPIE	ADIE	RC1IE	TX1IE	SSP1IE	TMR1GIE	TMR2IE	TMR1IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **PSPIE:** Parallel Slave Port Read/Write Interrupt Enable bit
1 = Enables the PSP read/write interrupt
0 = Disables the PSP read/write interrupt

bit 6 **ADIE:** A/D Converter Interrupt Enable bit
1 = Enables the A/D interrupt
0 = Disables the A/D interrupt

bit 5 **RC1IE:** EUSART1 Receive Interrupt Enable bit
1 = Enables the EUSART1 receive interrupt
0 = Disables the EUSART1 receive interrupt

bit 4 **TX1IE:** EUSART1 Transmit Interrupt Enable bit
1 = Enables the EUSART1 transmit interrupt
0 = Disables the EUSART1 transmit interrupt

bit 3 **SSP1IE:** Master Synchronous Serial Port 1 Interrupt Enable bit
1 = Enables the MSSP1 interrupt
0 = Disables the MSSP1 interrupt

bit 2 **TMR1GIE:** TMR1 Gate Interrupt Enable bit
1 = Enables the gate
0 = Disables the gate

bit 1 **TMR2IE:** TMR2 to PR2 Match Interrupt Enable bit
1 = Enables the TMR2 to PR2 match interrupt
0 = Disables the TMR2 to PR2 match interrupt

bit 0 **TMR1IE:** TMR1 Overflow Interrupt Enable bit
1 = Enables the TMR1 overflow interrupt
0 = Disables the TMR1 overflow interrupt

REGISTER 10-11: PIE2: PERIPHERAL INTERRUPT ENABLE REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
OSCFIE	SSP2IE	BCL2IE	USBIE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **OSCFIE:** Oscillator Fail Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **SSP2IE:** Master Synchronous Serial Port 2 Interrupt Enable bit
1 = Enables the MSSP2 interrupt
0 = Disables the MSSP2 interrupt

bit 5 **BCL2IE:** Bus Collision Interrupt Enable bit (MSSP)
1 = Enabled
0 = Disabled

bit 4 **USBIE:** USB Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **BCLIE:** Bus Collision Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **HLVDIE:** High/Low-Voltage Detect Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **TMR3IE:** TMR3 Overflow Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **TMR3GIE:** Timer3 Gate Interrupt Enable bit
1 = Enabled
0 = Disabled

REGISTER 10-12: PIE3: PERIPHERAL INTERRUPT ENABLE REGISTER 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TMR5GIE	LCDIE	RC2IE	TX2IE	CTMUIE	CCP2IE	CCP1IE	RTCCIE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TMR5GIE:** TMR5 Gate Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **LCDIE:** LCD Ready Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **RC2IE:** EUSART2 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **TX2IE:** EUSART2 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **CTMUIE:** CTMU Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **CCP2IE:** CCP2 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **CCP1IE:** ECCP1 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **RTCCIE:** RTCC Interrupt Enable bit
1 = Enabled
0 = Disabled

REGISTER 10-13: PIE4: PERIPHERAL INTERRUPT ENABLE REGISTER 4

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CCP10IE	CCP9IE	CCP8IE	CCP7IE	CCP6IE	CCP5IE	CCP4IE	ECCP3IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CCP10IE:** CCP10 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **CCP9IE:** CCP9 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **CCP8IE:** CCP8 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **CCP7IE:** CCP7 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **CCP6IE:** CCP6 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 2 **CCP5IE:** CCP5 Interrupt Flag bit
1 = Enabled
0 = Disabled

bit 1 **CCP4IE:** CCP4 Interrupt Flag bit
1 = Enabled
0 = Disabled

bit 0 **ECCP3IE:** ECCP3 Interrupt Flag bit
1 = Enabled
0 = Disabled

REGISTER 10-14: PIE5: PERIPHERAL INTERRUPT ENABLE REGISTER 5

U-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	ACTORSIE	ACTLOCKIE	TMR8IE	—	TMR6IE	TMR5IE	TMR4IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6 **ACTORSIE:** Active Clock Tuning Out-of-Range Interrupt Enable bit
1 = Enables the active clock tuning out-of-range interrupt
0 = Disables the active clock tuning out-of-range interrupt

bit 5 **ACTLOCKIE:** Active Clock Tuning Lock Interrupt Enable bit
1 = Enables the active clock tuning lock/unlock interrupt
0 = Disables the active clock tuning lock/unlock interrupt

bit 4 **TMR8IE:** TMR8 to PR8 Match Interrupt Enable bit
1 = Enables the TMR8 to PR8 match interrupt
0 = Disables the TMR8 to PR8 match interrupt

bit 3 **Unimplemented:** Read as '0'

bit 2 **TMR6IE:** TMR6 to PR6 Match Interrupt Enable bit
1 = Enables the TMR6 to PR6 match interrupt
0 = Disables the TMR6 to PR6 match interrupt

bit 1 **TMR5IE:** TMR5 Overflow Interrupt Enable bit
1 = Enables the TMR5 overflow interrupt
0 = Disables the TMR5 overflow interrupt

bit 0 **TMR4IE:** TMR4 to PR4 Match Interrupt Enable bit
1 = Enables the TMR4 to PR4 match interrupt
0 = Disables the TMR4 to PR4 match interrupt

REGISTER 10-15: PIE6: PERIPHERAL INTERRUPT ENABLE REGISTER 6

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
RC4IE	TX4IE	RC3IE	TX3IE	—	CMP3IE	CMP2IE	CMP1IE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **RC4IE:** EUSART4 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 6 **TX4IE:** EUSART4 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 5 **RC34IE:** EUSART3 Receive Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 4 **TX3IE:** EUSART3 Transmit Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 3 **Unimplemented:** Read as '0'

bit 2 **CMP3IE:** Comparator 3 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 1 **CMP2IE:** Comparator 2 Interrupt Enable bit
1 = Enabled
0 = Disabled

bit 0 **CMP1IE:** Comparator 1 Interrupt Enable bit
1 = Enabled
0 = Disabled

10.7 IPR Registers

The IPR registers contain the individual priority bits for the peripheral interrupts. Due to the number of peripheral interrupt sources, there are six Peripheral Interrupt Priority registers (IPR1 through IPR6). Using the priority bits requires that the Interrupt Priority Enable (IPEN) bit (RCON<7>) be set.

REGISTER 10-16: IPR1: PERIPHERAL INTERRUPT PRIORITY REGISTER 1

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
PSPIP	ADIP	RC1IP	TX1IP	SSP1IP	TMR1GIP	TMR2IP	TMR1IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **PSPIP:** Parallel Slave Port Read/Write Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **ADIP:** A/D Converter Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RC1IP:** EUSART1 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX1IP:** EUSART1 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **SSP1IP:** Master Synchronous Serial Port 1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **TMR1GIP:** Timer1 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR2IP:** TMR2 to PR2 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR1IP:** TMR1 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-17: IPR2: PERIPHERAL INTERRUPT PRIORITY REGISTER 2

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
OSCFIP	SSP2IP	BCL2IP	USBIP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **OSCFIP:** Oscillator Fail Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **SSP2IP:** Master Synchronous Serial Port 2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **BCL2IP:** Bus Collision Interrupt Priority bit (MSSP)
1 = High priority
0 = Low priority

bit 4 **USBIP:** USB Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **BCL1IP:** Bus Collision Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **HLVDIP:** High/Low-Voltage Detect Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR3IP:** TMR3 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR3GIP:** TMR3 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-18: IPR3: PERIPHERAL INTERRUPT PRIORITY REGISTER 3

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR5GIP	LCDIP	RC2IP	TX2IP	CTMUIP	CCP2IP	CCP1IP	RTCCIP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TMR5GIP:** TMR5 Gate Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **LCDIP:** LCD Ready Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RC2IP:** EUSART2 Receive Priority Flag bit
1 = High priority
0 = Low priority

bit 4 **TX2IP:** EUSART2 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **CTMUIP:** CTMU Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP2IP:** CCP2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **CCP1IP:** ECCP1 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **RTCCIP:** RTCC Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-19: IPR4: PERIPHERAL INTERRUPT PRIORITY REGISTER 4

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CCP10IP	CCP9IP	CCP8IP	CCP7IP	CCP6IP	CCP5IP	CCP4IP	ECCP3IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CCP10IP:** CCP10 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **CCP9IP:** CCP9 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **CCP8IP:** CCP8 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **CCP7IP:** CCP7 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **CCP6IP:** CCP6 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 2 **CCP5IP:** CCP5 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **CCP4IP:** CCP4 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **ECCP3IP:** ECCP3 Interrupt Priority bits
1 = High priority
0 = Low priority

REGISTER 10-20: IPR5: PERIPHERAL INTERRUPT PRIORITY REGISTER 5

U-0	R/W-1	R/W-1	R/W-1	U-0	R/W-1	R/W-1	R/W-1
—	ACTORSIP	ACTLOCKIP	TMR8IP	—	TMR6IP	TMR5IP	TMR4IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6 **ACTORSIP:** Active Clock Tuning Out-of-Range Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **ACTLOCKIP:** Active Clock Tuning Lock Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TMR8IP:** TMR8 to PR8 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as '0'

bit 2 **TMR6IP:** TMR6 to PR6 Match Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **TMR5IP:** TMR5 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **TMR4IP:** TMR4 to PR4 Match Interrupt Priority bit
1 = High priority
0 = Low priority

REGISTER 10-21: IPR6: PERIPHERAL INTERRUPT PRIORITY REGISTER 6

R/W-1	R/W-1	R/W-1	R/W-1	U-O	R/W-1	R/W-1	R/W-1
RC4IP	TX4IP	RC3IP	TX3IP	—	CMP3IP	CMP2IP	CMP1IP
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **RCP4IP:** EUSART4 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **TX4IP:** EUSART4 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **RC3IP:** EUSART3 Receive Interrupt Priority bit
1 = High priority
0 = Low priority

bit 4 **TX3IP:** EUSART3 Transmit Interrupt Priority bit
1 = High priority
0 = Low priority

bit 3 **Unimplemented:** Read as '0'

bit 2 **CMP3IP:** CMP3 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **CMP2IP:** CMP2 Interrupt Priority bit
1 = High priority
0 = Low priority

bit 0 **CMP1IP:** CMP1 Interrupt Priority bit
1 = High priority
0 = Low priority

10.8 RCON Register

The RCON register contains bits used to determine the cause of the last Reset or wake-up from Idle or Sleep modes. RCON also contains the bit that enables interrupt priorities (IPEN).

REGISTER 10-22: RCON: RESET CONTROL REGISTER

R/W-0	U-0	R/W-1	R/W-1	R-1	R-1	R/W-0	R/W-0
IPEN	—	$\overline{CM}$	$\overline{RI}$	$\overline{TO}$	$\overline{PD}$	$\overline{POR}$	$\overline{BOR}$
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enables priority levels on interrupts
0 = Disables priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6 **Unimplemented:** Read as '0'

bit 5 **$\overline{CM}$:** Configuration Mismatch Flag bit
1 = A Configuration Mismatch Reset has not occurred
0 = A Configuration Mismatch Reset has occurred (must be subsequently set in software)

bit 4 **$\overline{RI}$:** RESET Instruction Flag bit
For details of bit operation, see Register 5-1.

bit 3 **$\overline{TO}$:** Watchdog Timer Time-out Flag bit
For details of bit operation, see Register 5-1.

bit 2 **$\overline{PD}$:** Power-Down Detection Flag bit
For details of bit operation, see Register 5-1.

bit 1 **$\overline{POR}$:** Power-on Reset Status bit
For details of bit operation, see Register 5-1.

bit 0 **$\overline{BOR}$:** Brown-out Reset Status bit
For details of bit operation, see Register 5-1.

10.9 INTx Pin Interrupts

External interrupts on INT0, INT1, INT2 and INT3 are edge-triggered. INT0 is multiplexed with RB0 pin whereas INT1, INT2 and INT3 can only be used via remappable pins as shown in Table 11-13. If the corresponding INTEDGx bit in the INTCON2 register is set (= 1), the interrupt is triggered by a rising edge. If that bit is clear, the trigger is on the falling edge.

When a valid edge appears on the RBx/INTx pin, the corresponding flag bit, INTxIF, is set. This interrupt can be disabled by clearing the corresponding enable bit, INTxIE. Before re-enabling the interrupt, the flag bit (INTxIF) must be cleared in software in the Interrupt Service Routine.

All external interrupts (INT0, INT1, INT2 and INT3) can wake-up the processor from the power-managed modes if bit, INTxIE, was set prior to going into the power-managed modes. If the Global Interrupt Enable bit (GIE) is set, the processor will branch to the interrupt vector following wake-up.

The interrupt priority for INT1, INT2 and INT3 is determined by the value contained in the Interrupt Priority bits, INT1IP (INTCON3<6>), INT2IP (INTCON3<7>) and INT3IP (INTCON2<1>).

There is no priority bit associated with INT0. It is always a high-priority interrupt source.

10.10 TMR0 Interrupt

In 8-bit mode (the default), an overflow in the TMR0 register (FFh → 00h) will set flag bit, TMR0IF. In 16-bit mode, an overflow in the TMR0H:TMR0L register pair (FFFFh → 0000h) will set TMR0IF.

The interrupt can be enabled/disabled by setting/clearing enable bit, TMR0IE (INTCON<5>). Interrupt priority for Timer0 is determined by the value contained in the interrupt priority bit, TMR0IP (INTCON2<2>). For further details on the Timer0 module, see **Section 14.0 "Timer0 Module"**.

10.11 Edge-Selectable Interrupt-on-Change

Interrupt-on-change pins are selected via the PPS register settings and have the option of generating an interrupt on positive or negative transitions, or both. Positive edge events are enabled by setting the corresponding bits in the IOCP register, while negative edge events are enabled by setting the corresponding bits in the IOCN register. For compatibility with the previous interrupt-on-change feature, both the IOCP and IOCN bits should be set. The interrupt can be enabled by setting/clearing the IOCIE (INTCON<3>) bit. Each individual pin can be disabled by clearing both of the corresponding IOCN/IOCP bits. A change event (either positive or negative edge) will cause the corresponding IOCF flag to be set.

Interrupt priority for the edge selectable interrupt-on-change is determined by the interrupt priority bit, IOCIP (INTCON2<0>).

REGISTER 10-23: IOCP: INTERRUPT-ON-CHANGE POSITIVE EDGE REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IOCP7	IOCP6	IOCP5	IOCP4	IOCP3	IOCP2	IOCP1	IOCP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **IOCP<7:0>:** Interrupt-on-Change Positive Edge Enable bits

1 = Interrupt-on-change is enabled on the pin for a rising edge; associated Status bit and interrupt flag will be set upon detecting an edge

0 = Interrupt-on-change is disabled for the associated pin

REGISTER 10-24: IOCN: INTERRUPT-ON-CHANGE NEGATIVE EDGE REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IOCN7	IOCN6	IOCN5	IOCN4	IOCN3	IOCN2	IOCN1	IOCN0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **IOCN<7:0>:** Interrupt-on-Change Negative Edge Enable bits

1 = Interrupt-on-change is enabled on the pin for a falling edge; associated Status bit and interrupt flag will be set upon detecting an edge

0 = Interrupt-on-change is disabled for the associated pin

REGISTER 10-25: IOCF: INTERRUPT-ON-CHANGE FLAG REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
IOCF7	IOCF6	IOCF5	IOCF4	IOCF3	IOCF2	IOCF1	IOCF0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **IOCF<7:0>:** Interrupt-on-Change Flag bits

1 = An enabled change was detected on the associated pin; this is set when IOCP<x> = 1 and a positive edge was detected on the input pin or when IOCN<x> = 1 and a negative edge was detected on the input pin (clear in software to clear the IOCIF bit)

0 = No change was detected or the user cleared the detected change

10.12 Context Saving During Interrupts

During interrupts, the return PC address is saved on the stack. Additionally, the WREG, STATUS and BSR registers are saved on the Fast Return Stack.

If a fast return from interrupt is not used (see **Section 6.3 "Data Memory Organization"**), the user may need to save the WREG, STATUS and BSR registers on entry to the Interrupt Service Routine (ISR). Depending on the user's application, other registers also may need to be saved.

Example 10-1 saves and restores the WREG, STATUS and BSR registers during an Interrupt Service Routine.

EXAMPLE 10-1: SAVING STATUS, WREG AND BSR REGISTERS IN RAM

```
MOVWF   W_TEMP                  ; W_TEMP is in virtual bank
MOVFF   STATUS, STATUS_TEMP     ; STATUS_TEMP located anywhere
MOVFF   BSR, BSR_TEMP           ; BSR_TMEP located anywhere
;
; USER ISR CODE
;
MOVFF   BSR_TEMP, BSR           ; Restore BSR
MOVF    W_TEMP, W               ; Restore WREG
MOVFF   STATUS_TEMP, STATUS     ; Restore STATUS
```

11.0 I/O PORTS

Depending on the device selected and features enabled, there are up to eleven ports available. Some pins of the I/O ports are multiplexed with an alternate function from the peripheral features on the device. In general, when a peripheral is enabled, that pin may not be used as a general purpose I/O pin.

Each port has three memory mapped registers for its operation:

- TRIS register (Data Direction register)
- PORT register (reads the levels on the pins of the device)
- LAT register (Output Latch register)

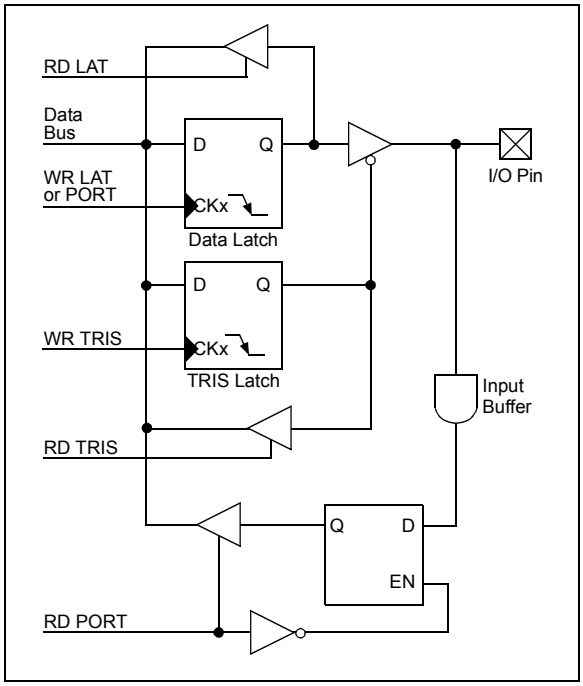
Reading the PORT register reads the current status of the pins, whereas writing to the PORT register, writes to the Output Latch (LAT) register.

Setting a TRIS bit (= 1) makes the corresponding PORT pin an input (putting the corresponding output driver in a High-Impedance mode). Clearing a TRIS bit (= 0) makes the corresponding port pin an output (i.e., driving the contents of the corresponding LAT bit on the selected pin).

The Output Latch (LAT register) is useful for read-modify-write operations on the value that the I/O pins are driving. Read-modify-write operations on the LAT register read and write the latched output value for the PORT register.

A simplified model of a generic I/O port, without the interfaces to other peripherals, is shown in Figure 11-1.

FIGURE 11-1: GENERIC I/O PORT OPERATION



11.1 I/O Port Pin Capabilities

When developing an application, the capabilities of the port pins must be considered.

The Absolute Maximum Ratings of the I/O pins are as follows:

- RA2, RA3 = -300mV to (V_{DD} + 300 mV)
- RA6, RA7, RC0, RC1 = -300 mV to (V_{DD} +300 mV)**(1)**
- RF3/RF4 (the USB D+/D- pins) = supports “USB specific levels” (e.g.: -1.0V to +4.6V, but only when the external source impedance is >/= 28 ohms, and the VUSB3V3 pin voltage is >/= 3.0V, otherwise: -500 mV to (VUSB3V3 +500 mV)
- All other general purpose I/O pins (including $\overline{MCLR}$), when V_{DD} is < 2.0V: -300 mV to +4.0V.
- All other general purpose I/O pins (including $\overline{MCLR}$), when V_{DD} is >= 2.0V: -300 mV to +6.0V**(2)**.

Note 1: When the pins are used to drive a crystal or ceramic resonator, natural oscillation waveforms slightly exceeding the -300 mV to (V_{DD} +300 mV) range may sometimes occur, and if present, such waveforms are allowed. If these pins are instead used as general purpose inputs, the external driving source should adhere to the -300 mV to (V_{DD} +300 mV) specification.

2: In addition to the above absolute maximums, any I/O pin voltage that is actively selected at runtime by the ADC channel select MUX must also meet the V_{AIN} requirements (parameter A25 in Table 30-40).

11.1.1 OUTPUT PIN DRIVE

When used as digital I/O, the output pin drive strengths vary, according to the pins' grouping, to meet the needs for a variety of applications. In general, there are two classes of output pins in terms of drive capability:

- Outputs designed to drive higher current loads, such as LEDs:
  - PORTB
  - PORTC
- Outputs with lower drive levels, but capable of driving normal digital circuit loads with a high input impedance. Able to drive LEDs, but only those with smaller current requirements:
  - PORTA
  - PORTD
  - PORTE
  - PORTF
  - PORTG
  - PORTH(1)
  - PORTJ(1)
  - PORTK(2)
  - PORTL(2)

Note 1: These ports are not available on 64-pin devices.

2: These ports are not available on 64-pin or 80-pin devices.

11.1.2 PULL-UP CONFIGURATION

Nine of the I/O ports (all ports except PORTA and PORTC) implement configurable weak pull-ups on all pins. These are internal pull-ups that allow floating digital input signals to be pulled to a consistent level without the use of external resistors.

Pull-ups for PORTB are enabled by clearing the $\overline{\text{RBPU}}$ bit (INTCON2<7>). PORTB pull-ups are individually selectable through the WPUB register.

Pull-ups for PORTD, PORTE, PORTF, PORTG, PORTH, PORTJ, PORTK and PORTL are enabled through their corresponding enable bits in the PADCFG register, but are not pin-selectable.

REGISTER 11-1: PADCFG1: PAD CONFIGURATION REGISTER 1(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RDPU	REPU	RFPU	RGPU	RHPU	RJPU	RKPU	RLPU
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **RDPU:** PORTD Pull-up Enable bit
1 = PORTD pull-ups are enabled for any input pad
0 = All PORTD pull-ups are disabled

bit 6 **REPU:** PORTE Pull-up Enable bit
1 = PORTE pull-ups are enabled for any input pad
0 = All PORTE pull-ups are disabled

bit 5 **RFPU:** PORTF Pull-up Enable bit
1 = PORTF pull-ups are enabled for any input pad
0 = All PORTF pull-ups are disabled

bit 4 **RGPU:** PORTG Pull-up Enable bit
1 = PORTG pull-ups are enabled for any input pad
0 = All PORTG pull-ups are disabled

bit 3 **RHPU:** PORTH Pull-up Enable bit
1 = PORTH pull-ups are enabled for any input pad
0 = All PORTH pull-ups are disabled

bit 2 **RJPU:** PORTJ Pull-up Enable bit
1 = PORTJ pull-ups are enabled for any input pad
0 = All PORTJ pull-ups are disabled

bit 1 **RKPU:** PORTK Pull-up Enable bit
1 = PORTK pull-ups are enabled for any input pad
0 = All PORTK pull-ups are disabled

bit 0 **RLPU:** PORTL Pull-up Enable bit
1 = PORTL pull-ups are enabled for any input pad
0 = All PORTL pull-ups are disabled

Note 1: If a particular PORT is not available on a package, the corresponding RnPU register bit will be unimplemented and read back as '0'.

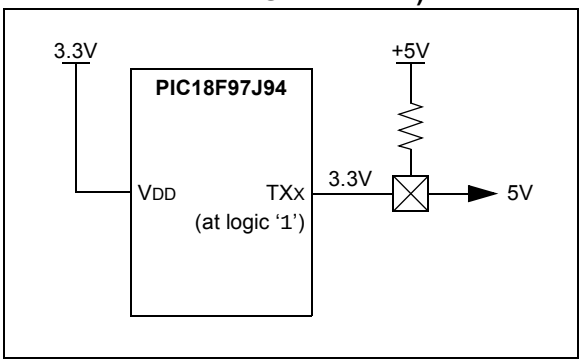
11.1.3 OPEN-DRAIN OUTPUTS

The output pins for several peripherals are also equipped with a configurable, open-drain output option. This allows the peripherals to communicate with external digital logic, operating at a higher voltage level, without the use of level translators.

The open-drain option is implemented on the EUSARTs, the MSSPx modules (in SPI mode) and the CCP modules. These modules are assigned to an I/O pin using the PPS (Peripheral Pin Select) feature. The open-drain option is enabled by setting the open-drain control bits in the ODCON1 and ODCON2 registers.

When the open-drain option is required, the output pin must also be tied through an external pull-up resistor, provided by the user, to a higher voltage level, up to 5V (Figure 11-2). When a digital logic high signal is output, it is pulled up to the higher voltage level.

FIGURE 11-2: USING THE OPEN-DRAIN OUTPUT (USART SHOWN AS EXAMPLE)



REGISTER 11-2: ODCON1: PERIPHERAL OPEN-DRAIN CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCP2OD	ECCP1OD	USART4OD	USART3OD	USART2OD	USART1OD	SSP2OD	SSP1OD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ECCP2OD:** ECCP2 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 6 **ECCP1OD:** ECCP1 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 5 **USART4OD:** EUSART4 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 4 **USART3OD:** EUSART3 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 3 **USART2OD:** EUSART2 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 2 **USART1OD:** EUSART1 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 1 **SSP2OD:** Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 0 **SSP1OD:** SPI1 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

REGISTER 11-3: ODCON2: PERIPHERAL OPEN-DRAIN CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CCP10OD	CCP9OD	CCP8OD	CCP7OD	CCP6OD	CCP5OD	CCP4OD	ECCP3OD
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CCP10OD:** CCP10 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 6 **CCP9OD:** CCP9 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 5 **CCP8OD:** CCP8 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 4 **CCP7OD:** CCP7 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 3 **CCP6OD:** CCP6 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 2 **CCP5OD:** CCP5 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 1 **CCP4OD:** CCP4 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

bit 0 **ECCP3OD:** ECCP3 Open-Drain Output Enable bit
1 = Open-drain capability is enabled
0 = Open-drain capability is disabled

11.1.4 ANALOG AND DIGITAL PORTS

Many of the ports multiplex analog and digital functionality, providing a lot of flexibility for hardware designers. PIC18FXXJ94 devices can make any analog pin analog or digital, depending on an application's needs. The ports' analog/digital functionality is controlled by the registers: ANCON1, ANCON2 and ANCON3.

Setting these registers makes the corresponding pins analog and clearing the registers makes the ports digital. For details on these registers, see **Section 22.0 "12-Bit A/D Converter with Threshold Scan"**

11.2 PORTA, LATA and TRISA Registers

PORTA is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISA and LATA.

All PORTA pins have Schmitt Trigger input levels and full CMOS output drivers.

RA<5:0> are multiplexed with analog inputs for the A/D Converter.

The operation of the analog inputs as A/D Converter inputs is selected by clearing or setting the ANSELx control bits in the ANCON1 register. The corresponding TRISA bits control the direction of these pins, even when they are being used as analog inputs. The user must ensure the bits in the TRISA register are maintained set when using them as analog inputs.

Note: RA<5:0> are configured as analog inputs on any Reset and are read as '0'.

OSC2/CLKO/RA6 and OSC1/CLKI/RA7 normally serve as the external circuit connections for the External (Primary) Oscillator circuit (HS Oscillator modes), or the external clock input and output (EC Oscillator modes). In these cases, RA6 and RA7 are not available as digital I/O, and their corresponding TRIS and LAT bits are read as '0'. When the device is configured to use either the FRC or LPRC Internal Oscillators as the default oscillator mode, RA6 and RA7 are automatically configured as digital I/O; the oscillator and clock in/clock out functions are disabled.

EXAMPLE 11-1: INITIALIZING PORTA

```
CLRF     PORTA    ;  Initialize PORTA by
                  ; clearing output latches
CLRF     LATA     ; Alternate method to
                  ; clear output data latches
BANKSEL  ANCON1   ; Select bank with ANCON1 register
MOVLW    00h      ; Configure A/D
MOVWF    ANCON1   ; for digital inputs
BANKSEL  TRISA    ; Select bank with TRISA register
MOVLW    0BFh     ; Value used to initialize
                  ; data direction
MOVWF    TRISA    ; Set RA<7, 5:0> as inputs,
                  ; RA<6> as output
```

TABLE 11-1: PORTA FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RA0/AN0/AN1-/RP0/SEG19	RA0	0	O	DIG	LATA<0> data output; not affected by analog input.
		1	I	ST	PORTA<0> data input; disabled when analog input is enabled.
	AN0	1	I	ANA	A/D Input Channel 0. Default input configuration on POR; does not affect digital output.
	AN1-	1	I	ANA	Quasi-differential A/D negative input channel.
	RP0	x	x	DIG	Reconfigurable Pin 0 for PPS-Lite; TRIS must be set to match input/output of the module.
	SEG19	0	O	ANA	LCD Segment 19 output; disables all other pin functions.
RA1/AN1/RP1/SEG18	RA1	0	O	DIG	LATA<1> data output; not affected by analog input.
		1	I	ST	PORTA<1> data input; disabled when analog input is enabled.
	AN1	1	I	ANA	A/D Input Channel 1. Default input configuration on POR; does not affect digital output.
	RP1	x	x	DIG	Reconfigurable Pin 1 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG18	0	O	ANA	LCD Segment 18 output; disables all other pin functions.
RA2/AN2/VREF-/RP2/SEG21	RA2	0	O	DIG	LATA<2> data output; not affected by analog input.
		1	I	ST	PORTA<2> data input; disabled when analog input enabled.
	AN2	1	I	ANA	A/D Input Channel 2. Default input configuration on POR; does not affect digital output.
	VREF-	1	I	ANA	A/D and Comparator Low Reference Voltage input.
	RP2	x	x	DIG	Reconfigurable Pin 2 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG21	0	O	ANA	LCD Segment 21 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-1: PORTA FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RA3/AN3/VREF+/RP3	RA3	0	O	DIG	LATA<3> data output; not affected by analog input.
		1	I	ST	PORTA<3> data input; disabled when analog input is enabled.
	AN3	1	I	ANA	A/D Input Channel 3. Default input configuration on POR; does not affect digital output.
	VREF+	1	I	ANA	A/D and Comparator High Reference Voltage input.
	RP3	x	x	DIG	Reconfigurable Pin 3 for PPS-Lite; TRIS must be set to match input/output of module.
RA4/AN6/RP4/SEG14	RA4	0	O	DIG	LATA<4> data output; not affected by analog input.
		1	I	ST	PORTA<4> data input; disabled when analog input is enabled.
	AN6	1	I	ANA	A/D Input Channel 6. Default input configuration on POR; does not affect digital output.
	RP4	x	x	DIG	Reconfigurable Pin 4 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG14	0	O	ANA	LCD Segment 14 output; disables all other pin functions.
RA5/AN4/RP5/LVDIN/ C1INA/C2INA/C3INA/ SEG15	RA5	0	O	DIG	LATA<5> data output; not affected by analog input.
		1	I	ST	PORTA<5> data input; disabled when analog input is enabled.
	AN4	1	I	ANA	A/D Input Channel 4. Default input configuration on POR; does not affect digital output.
	RP5	x	x	DIG	Reconfigurable Pin 5 for PPS-Lite; TRIS must be set to match input/output of module.
	LVDIN	1	I	ANA	High/Low-Voltage Detect (HLVD) external trip point input.
	C1INA	1	I	ANA	Comparator 1 Input A.
	C2INA	1	I	ANA	Comparator 2 Input A.
	C3INA	1	I	ANA	Comparator 3 Input A.
	SEG15	0	O	ANA	LCD Segment 15 output; disables all other pin functions.
RA6/RP6/CLKO/OSC2	RA6	0	O	DIG	LATA<6> data output; disabled when OSC2 Configuration bit is set.
		1	I	ST	PORTA<6> data input; disabled when OSC2 Configuration bit is set.
	RP6	x	x	DIG	Reconfigurable Pin 6 for PPS-Lite; TRIS must be set to match input/output of module.
	CLKO	x	O	DIG	System cycle clock output (FOSC/4, EC and Internal Oscillator modes).
	OSC2	x	O	ANA	Main oscillator feedback output connection (HS, MS and LP modes).
RA7/RP10/CLKI/OSC1	RA7	0	O	DIG	LATA<7> data output; disabled when OSC2 Configuration bit is set.
		1	I	ST	PORTA<7> data input; disabled when OSC2 Configuration bit is set.
	RP10	x	x	DIG	Reconfigurable Pin 10 for PPS-Lite; TRIS must be set to match input/output of module.
	CLKI	x	O	DIG	Main external clock source input (EC modes).
	OSC1	x	O	ANA	Main oscillator input connection (HS, MS and LP modes).

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input,
x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.3 PORTB, LATB and TRISB Registers

PORTB is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISB and LATB. All pins on PORTB are digital only.

EXAMPLE 11-2: INITIALIZING PORTB

```
CLRF    PORTB   ; Initialize PORTB by
                ; clearing output
                ; data latches
CLRF    LATB    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISB   ; Set RB<3:0> as inputs
                ; RB<5:4> as outputs
                ; RB<7:6> as inputs
```

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit, RBPU (INTCON2<7>), and setting the associated WPUB bit. The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

The RB<3:2> pins are multiplexed as CTMU edge inputs.

TABLE 11-2: PORTB FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RB0/INT0/CTED13/ RP8/VLCAP1	RB0	0	O	DIG	LATB<0> data output.
		1	I	ST	PORTB<0> data input.
	INT0	1	I	ST	External Interrupt 0 input.
	CTED13	1	I	ST	CTMU Edge 13 input.
	RP8	x	x	DIG	Reconfigurable Pin 8 for PPS-Lite; TRIS must be set to match input/output of module.
	VLCAP1	x	x	ANA	External capacitor connection for LCD module.
RB1/RP9/VLCAP2	RB1	0	O	DIG	LATB<1> data output.
		1	I	ST	PORTB<1> data input.
	RP9	x	x	DIG	Reconfigurable Pin 9 for PPS-Lite; TRIS must be set to match input/output of module.
	VLCAP2	x	x	ANA	External capacitor connection for LCD module.
RB2/CTED1/RP14/ SEG9	RB2	0	O	DIG	LATB<2> data output.
		1	I	ST	PORTB<2> data input.
	CTED1	1	I	ST	CTMU Edge 1 input.
	RP14	x	x	DIG	Reconfigurable Pin 14 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG9	0	O	ANA	LCD Segment 9 output; disables all other pin functions.
RB3/CTED2/RP7/ SEG10	RB3	0	O	DIG	LATB<3> data output.
		1	I	ST	PORTB<3> data input.
	CTED2	1	I	ST	CTMU Edge 2 input.
	RP7	x	x	DIG	Reconfigurable Pin 7 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG10	0	O	ANA	LCD Segment 10 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-2: PORTB FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RB4/CTED3/RP12/ SEG11	RB4	0	O	DIG	LATB<4> data output.
		1	I	ST	PORTB<4> data input.
	CTED3	1	I	ST	CTMU Edge 3 input.
	RP12	x	x	DIG	Reconfigurable Pin 12 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG11	0	O	ANA	LCD Segment 11 output; disables all other pin functions.
RB5/CTED4/RP13/ SEG8	RB5	0	O	DIG	LATB<5> data output.
		1	I	ST	PORTB<5> data input.
	CTED4	1	I	ST	CTMU Edge 4 input.
	RP13	x	x	DIG	Reconfigurable Pin 13 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG8	0	O	ANA	LCD Segment 8 output; disables all other pin functions.
RB6/CTED5/PGC	RB6	0	O	DIG	LATB<6> data output.
		1	I	ST	PORTB<6> data input.
	CTED5	1	I	ST	CTMU Edge 5 input.
	PGC	x	I	ST	Serial execution (ICSP™) clock input for ICSP and ICD operations.
RB7/CTED6/PGD	RB7	0	O	DIG	LATB<7> data output.
		1	I	ST	PORTB<7> data input.
	CTED6	1	I	ST	CTMU Edge 6 input.
	PGD	x	I/O	ST/DIG	Serial execution (ICSP™) data input/output for ICSP and ICD operations.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.4 PORTC, LATC and TRISC Registers

PORTC is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISC and LATC. Only PORTC pins, RC2 through RC7, are digital only pins. The pins have Schmitt Trigger input buffers.

When enabling peripheral functions, use care in defining TRIS bits for each PORTC pin. Some peripherals can override the TRIS bit to make a pin an output or input. Consult the corresponding peripheral section for the correct TRIS bit settings.

Note: These pins are configured as digital inputs on any device Reset.

The contents of the TRISC register are affected by peripheral overrides. Reading TRISC always returns the current contents, even though a peripheral device may be overriding one or more of the pins.

EXAMPLE 11-3: INITIALIZING PORTC

```
CLRF    PORTC   ; Initialize PORTC by
                ; clearing output
                ; data latches
CLRF    LATC    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISC   ; Set RC<3:0> as inputs
                ; RC<5:4> as outputs
                ; RC<7:6> as inputs
```

TABLE 11-3: PORTC FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RC0/ PWRLCLK/ SCLKI/SOSCO	RC0	1	I	ST	PORTC<0> data input.
	PWRLCLK	1	I	ST	Optional RTCC input from power line clock (50 or 60 Hz).
	SCLKI	x	I	ST	Digital SOSC input.
	SOSCO	x	O	ANA	Secondary Oscillator (SOSC) feedback output connection.
RC1/SOSCI	RC1	1	I	ST	PORTC<1> data input.
	SOSCI	x	I	ANA	Secondary Oscillator (SOSC) input connection.
RC2/CTED7/ RP11/AN9/ SEG13	RC2	0	O	DIG	LATC<2> data output; not affected by analog input.
		1	I	ST	PORTC<2> data input; disabled when analog input is enabled.
	CTED7	1	I	ST	CTMU Edge 7 input.
	RP11	x	x	DIG	Reconfigurable Pin 11 for PPS-Lite; TRIS must be set to match input/output of module.
	AN9	1	I	ANA	A/D Input Channel 9. Default input configuration on POR; does not affect digital output.
	SEG13	0	O	ANA	LCD Segment 13 output; disables all other pin functions.
RC3/CTED8/ RP15/SCL1/ SEG17	RC3	0	O	DIG	LATC<3> data output.
		1	I	ST	PORTC<3> data input.
	CTED8	1	I	ST	CTMU Edge 8 input.
	RP15	x	x	DIG	Reconfigurable Pin 15 for PPS-Lite; TRIS must be set to match input/output of module.
	SCL1	x	I/O	I^2C	Synchronous serial clock input/output for I^2C mode.
	SEG17	0	O	ANA	LCD Segment 17 output; disables all other pin functions
RC4/CTED9/ RP17/SDA1/ SEG16	RC4	0	O	DIG	LATC<4> data output.
		1	I	ST	PORTC<4> data input.
	CTED9	1	I	ST	CTMU Edge 9 input.
	RP17	x	x	DIG	Reconfigurable Pin 17 for PPS-Lite; TRIS must be set to match input/output of module.
	SDA1	x	I/O	I^2C	I^2C mode data I/O
	SEG16	0	O	ANA	LCD Segment 16 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, I^2C = I^2C/SMBus, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-3: PORTC FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RC5/CTED10/ RP16/SEG12	RC5	0	O	DIG	LATC<5> data output.
		1	I	ST	PORTC<5> data input.
	CTED10	1	I	ST	CTMU Edge 10 input.
	RP16	x	x	DIG	Reconfigurable Pin 16 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG12	0	O	ANA	LCD Segment 12 output; disables all other pin functions.
RC6/CTED11/ $\overline{\text{UOE}}$/RP18/ SEG27	RC6	0	O	DIG	LATC<6> data output.
		1	I	ST	PORTC<6> data input.
	CTED11	1	I	ST	CTMU Edge 11 input.
	$\overline{\text{UOE}}$	0	O	DIG	USB Output Enable control (for external transceiver).
	RP18	x	x	DIG	Reconfigurable Pin 18 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG27	0	O	ANA	LCD Segment 27 output; disables all other pin functions.
RC7/CTED12/ RP19/SEG22	RC7	0	O	DIG	LATC<7> data output.
		1	I	ST	PORTC<7> data input.
	CTED12	1	I	ST	CTMU Edge 12 input.
	RP19	x	x	DIG	Reconfigurable Pin 19 for PPS-Lite; TRIS must be set to match input/output of module.
	SEG22	0	O	ANA	LCD Segment 22 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, I^2C = I^2C/SMBus, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.5 PORTD, LATD and TRISD Registers

PORTD is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISD and LATD.

All pins on PORTD are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note: These pins are configured as digital inputs on any device Reset.

Each of the PORTD pins has a weak internal pull-up. A single control bit can turn off all the pull-ups. This is performed by setting bit, RDPU (PADCFG<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on all device Resets.

On 80-pin and 100-pin devices, PORTD is multiplexed with the system bus as part of the external memory interface. I/O port and other functions are only available when the interface is disabled by setting the EBDIS bit (MEMCON<7>). When the interface is enabled, PORTD is the low-order byte of the multiplexed Address/Data bus (AD<7:0>). The TRISD bits are also overridden.

PORTD can also be configured as an 8-bit wide microprocessor port (Parallel Slave Port) by setting control bit, PSPMODE (PSPCON<4>). In this mode, the input buffers are TTL. For additional information, see **Section 11.13 "Parallel Slave Port"**.

PORTD also has I^2C functionality on RD5 and RD6.

EXAMPLE 11-4: INITIALIZING PORTD

```
CLRF    PORTD   ; Initialize PORTD by
                ; clearing output
                ; data latches
CLRF    LATD    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISD   ; Set RD<3:0> as inputs
                ; RD<5:4> as outputs
                ; RD<7:6> as inputs
```

TABLE 11-4: PORTD FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD0/PSP0/ RP20/SEG0/AD0	RD0	0	O	DIG	LATD<0> data output.
		1	I	ST	PORTD<0> data input.
	PSP0	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 0.
	RP20	x	x	DIG	Reconfigurable Pin 20 for PPS-Lite; TRIS must be set to match input/ output of module.
	SEG0	0	O	ANA	LCD Segment 0 output; disables all other pin functions.
	AD0	x	I/O	ST/DIG	External Memory Bus Address Line 0.
RD1/PSP1/ RP21/SEG1/AD1	RD1	0	O	DIG	LATD<1> data output.
		1	I	ST	PORTD<1> data input.
	PSP1	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 1.
	RP21	x	x	DIG	Reconfigurable Pin 21 for PPS-Lite; TRIS must be set to match input/ output of module.
	SEG1	0	O	ANA	LCD Segment 1 output; disables all other pin functions.
	AD1	x	I/O	ST/DIG	External Memory Bus Address Line 1.
RD2/PSP2/ RP22/SEG2/AD2	RD2	0	O	DIG	LATD<2> data output.
		1	I	ST	PORTD<2> data input.
	PSP2	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 2.
	RP22	x	x	DIG	Reconfigurable Pin 22 for PPS-Lite; TRIS must be set to match input/ output of module.
	SEG2	0	O	ANA	LCD Segment 2 output; disables all other pin functions.
	AD2	x	I/O	ST/DIG	External Memory Bus Address Line 2.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, I^2C = I^2C/SMBus, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-4: PORTD FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RD3/PSP3/ RP23/SEG3/AD3	RD3	0	O	DIG	LATD<3> data output.
		1	I	ST	PORTD<3> data input.
	PSP3	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 3.
	RP23	x	x	DIG	Reconfigurable Pin 23 for PPS-Lite; TRIS must be set to match input/ output of module.
	SEG3	0	O	ANA	LCD Segment 3 output; disables all other pin functions.
	AD3	x	I/O	ST/DIG	External Memory Bus Address Line 3.
RD4/PSP4/ RP24/SEG4/AD4	RD4	0	O	DIG	LATD<4> data output.
		1	I	ST	PORTD<4> data input.
	PSP4	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 4.
	RP24	x	x	DIG	Reconfigurable Pin 24 for PPS-Lite; TRIS must be set to match input/ output of module.
	SEG4	0	O	ANA	LCD Segment 4 output; disables all other pin functions.
	AD4	x	I/O	ST/DIG	External Memory Bus Address Line 4.
RD5/PSP5/ RP25/SDA2/ SEG5/AD5	RD5	0	O	DIG	LATD<5> data output.
		1	I	ST	PORTD<5> data input.
	PSP5	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 5.
	RP25	x	x	DIG	Reconfigurable Pin 25 for PPS-Lite; TRIS must be set to match input/ output of module.
	SDA2	x	I/O	ST/DIG	I^2C mode data I/O.
	SEG5	0	O	ANA	LCD Segment 5 output; disables all other pin functions.
	AD5	x	I/O	ST/DIG	External Memory Bus Address Line 5.
RD6/PSP6/ RP26/SCL2/ SEG6/AD6	RD6	0	O	DIG	LATD<6> data output.
		1	I	ST	PORTD<6> data input.
	PSP6	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 6.
	RP26	x	x	DIG	Reconfigurable Pin 26 for PPS-Lite; TRIS must be set to match input/ output of module.
	SCL2	x	I/O	I^2C	Synchronous serial clock input/output for I^2C mode.
	SEG6	0	O	ANA	LCD Segment 6 output; disables all other pin functions.
	AD6	x	I/O	ST/DIG	External Memory Bus Address Line 6.
RD7/PSP7/ RP27/REFO2/ SEG7/AD7	RD7	0	O	DIG	LATD<7> data output.
		1	I	ST	PORTD<7> data input.
	PSP7	x	I/O	ST/DIG	Parallel Slave Port Data Bus Bit 7.
	RP27	x	x	DIG	Reconfigurable Pin 27 for PPS-Lite; TRIS must be set to match input/ output of module.
	REFO2	0	O	DIG	Reference Clock 2 output.
	SEG7	0	O	ANA	LCD Segment 7 output; disables all other pin functions.
	AD7	x	I/O	ST/DIG	External Memory Bus Address Line 7.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, I^2C = I^2C/SMBus, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.6 PORTE, LATE and TRISE Registers

PORTE is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISE and LATE.

All pins on PORTE are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note: These pins are configured as digital inputs on any device Reset.

Each of the PORTE pins has a weak internal pull-up. A single control bit can turn off all the pull-ups. This is performed by setting bit, REPU (PADCFG<6>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on any device Reset.

For devices operating in Microcontroller mode, the RE7 pin can be configured as the alternate peripheral pin for the ECCP2 module and Enhanced PWM Output 2A.

PORTE is also multiplexed with the Parallel Slave Port address lines. RE2, RE1 and RE0 are multiplexed with the control signals, $\overline{CS}$, $\overline{WR}$ and $\overline{RD}$.

RE3 can also be configured as the Reference Clock Output (REFO) from the system clock. For further details, see **Section 3.4 "Reference Clock Output Control Module"**.

EXAMPLE 11-5: INITIALIZING PORTE

```
CLRF    PORTE   ; Initialize PORTE by
                ; clearing output
                ; data latches
CLRF    LATE    ; Alternate method
                ; to clear output
                ; data latches
MOVLW   03h     ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISE   ; Set RE<1:0> as inputs
                ; RE<7:2> as outputs
```

TABLE 11-5: PORTE FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RE0//$\overline{RD}$/RP28/ LCDBIAS1/AD8	RE0	0	O	DIG	LATE<0> data output.
		1	I	ST	PORTE<0> data input.
	$\overline{RD}$	1	I	ST	Parallel Slave Port (PSP) Read ($\overline{RD}$) signal.
	RP28	x	x	DIG	Reconfigurable Pin 28 for PPS-Lite; TRIS must be set to match input/ output of module.
	LCDBIAS1	x	I	ANA	LCD Module Bias Voltage Input 1.
	AD8	x	I/O	ST/DIG	External Memory Bus Address Line 8.
RE1//$\overline{WR}$/RP29/ LCDBIAS2/AD9	RE1	0	O	DIG	LATE<1> data output.
		1	I	ST	PORTE<1> data input.
	$\overline{WR}$	1	I	ST	Parallel Slave Port (PSP) Write ($\overline{WR}$) signal.
	RP29	x	x	DIG	Reconfigurable Pin 29 for PPS-Lite; TRIS must be set to match input/ output of module.
	LCDBIAS2	x	I	ANA	LCD Module Bias Voltage Input 2.
	AD9	x	I/O	ST/DIG	External Memory Bus Address Line 9.
RE2/$\overline{CS}$/RP30/ LCDBIAS3/AD10	RE2	0	O	DIG	LATE<2> data output.
		1	I	ST	PORTE<2> data input.
	$\overline{CS}$	1	I	ST	Parallel Slave Port (PSP) Chip Select ($\overline{CS}$) signal.
	RP30	x	x	DIG	Reconfigurable Pin 30 for PPS-Lite; TRIS must be set to match input/ output of module.
	LCDBIAS3	x	I	ANA	LCD Module Bias Voltage Input 3.
	AD10	x	I/O	ST/DIG	External Memory Bus Address Line 10.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-5: PORTE FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RE3/REFO1/ RP33/COM0/ AD11	RE3	0	O	DIG	LATE<3> data output.
		1	I	ST	PORTE<3> data input.
	REFO1	0	O	DIG	Reference Clock Output 1.
	RP33	x	x	DIG	Reconfigurable Pin 33 for PPS-Lite; TRIS must be set to match input/ output of module.
	COM0	x	O	ANA	LCD Common 0 output; disables all other outputs.
	AD11	x	I/O	ST/DIG	External Memory Bus Address Line 11.
RE4/RP32/ COM1/AD12	RE4	0	O	DIG	LATE<4> data output.
		1	I	ST	PORTE<4> data input.
	RP32	x	x	DIG	Reconfigurable Pin 32 for PPS-Lite; TRIS must be set to match input/ output of module.
	COM1	x	O	ANA	LCD Common 1 output; disables all other outputs.
	AD12	x	I/O	ST/DIG	External Memory Bus Address Line 12.
RE5/RP37/ COM2/AD13	RE5	0	O	DIG	LATE<5> data output.
		1	I	ST	PORTE<5> data input.
	RP37	x	x	DIG	Reconfigurable Pin 37 for PPS-Lite; TRIS must be set to match input/ output of module.
	COM2	x	O	ANA	LCD Common 2 output; disables all other outputs.
	AD13	x	I/O	ST/DIG	External Memory Bus Address Line 13.
RE6/RP34/ COM3/AD14	RE6	0	O	DIG	LATE<6> data output.
		1	I	ST	PORTE<6> data input.
	RP34	x	x	DIG	Reconfigurable Pin 34 for PPS-Lite; TRIS must be set to match input/ output of module.
	COM3	x	O	ANA	LCD Common 3 output; disables all other outputs.
	AD14	x	I/O	ST/DIG	External Memory Bus Address Line 14.
RE7/RP31/ LCDBIAS0/ AD15	RE7	0	O	DIG	LATE<7> data output.
		1	I	ST	PORTE<7> data input.
	RP31	x	x	DIG	Reconfigurable Pin 31 for PPS-Lite; TRIS must be set to match input/ output of module.
	LCDBIAS0	x	I	ANA	LCD Module Bias Voltage Input 0.
	AD15	x	I/O	ST/DIG	External Memory Bus Address Line 15.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.7 PORTF, LATF and TRISF Registers

PORTF is a 6-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISF and LATF. All pins on PORTF are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Pins, RF2 through RF6, may be used as comparator inputs or outputs by setting the appropriate bits in the CMCON register. To use RF<7:2> as digital inputs, it is also necessary to turn off the comparators.

Note 1: On device Resets, pins, RF<7:2>, are configured as analog inputs and are read as '0'.

2: To configure PORTF as a digital I/O, turn off the comparators and clear ANCON1 and ANCON2 to digital.

EXAMPLE 11-6: INITIALIZING PORTF

```
CLRF     PORTF    ; Initialize PORTF by
                  ; clearing output
                  ; data latches
CLRF     LATF     ; Alternate method
                  ; to clear output
                  ; data latches
BANKSEL  ANCON1   ; Select bank with ANCON1 register
MOVLW    BFh      ; Make RF2 digital
MOVWF    ANCON1   ;
BANKSELANCON2;
MOVLW    F1h      ; Make RF5, RF6, RF7 digital
MOVWF    ANCON2   ;
BANKSEL  TRISF    ; Select bank with TRISF register
MOVLW    0F3h     ; Value used to
                  ; initialize data
                  ; direction
MOVWF    TRISF    ; Set RF3:RF2 as outputs
                  ; RF7:RF4 as inputs
```

TABLE 11-6: PORTF FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RF0	—	—	—	—	PORTF<0> is not implemented.
RF1	—	—	—	—	PORTF<1> is not implemented.
RF2/RP36/C2INB/ CTMUI/SEG20/ AN7	RF2	0	O	DIG	LATF<2> data output.
		1	I	ST	PORTF<2> data input.
	RP36	x	x	DIG	Reconfigurable Pin 36 for PPS-Lite; TRIS must be set to match input/ output of module.
	C2INB	1	I	ANA	Comparator 2 Input B.
	CTMUI	1	I	ANA	CTMU comparator input.
	SEG20	0	O	ANA	LCD Segment 20 output; disables all other pin functions.
	AN7	1	I	ANA	A/D Input Channel 7. Default input configuration on POR; does not affect digital output.
RF3/D-	RF3	1	I	ST	PORTF<3> data input.
	D-	x	I	XCVR	USB bus minus line output.
		x	O	XCVR	USB bus minus line input.
RF4/D+	RF4	1	I	ST	PORTF<4> data input.
	D+	x	I	XCVR	USB bus plus line input.
		x	O	XCVR	USB bus plus line output.
RF5/RP35/C1INB/ AN10/CVREF/ SEG23	RF5	0	O	DIG	LATF<5> data output.
		1	I	ST	PORTF<5> data input.
	RP35	x	x	DIG	Reconfigurable Pin 35 for PPS-Lite; TRIS must be set to match input/ output of module.
	C1INB	1	I	ANA	Comparator 1 Input B.
	AN10	1	I	ANA	A/D Input Channel 10. Default input configuration on POR; does not affect digital output.
	CVREF	0	O	ANA	Comparator reference voltage output.
	SEG23	0	O	ANA	LCD Segment 23 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, XCVR = USB Transceiver, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-6: PORTF FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RF6/RP40/C1INA/ AN11/SEG24	RF6	0	O	DIG	LATF<6> data output.
		1	I	ST	PORTF<6> data input.
	RP40	x	x	DIG	Reconfigurable Pin 40 for PPS-Lite; TRIS must be set to match input/ output of module.
	C1INA	1	I	ANA	Comparator 1 Input A.
	AN11	1	I	ANA	A/D Input Channel 11. Default input configuration on POR; does not affect digital output.
	SEG24	0	O	ANA	LCD Segment 24 output; disables all other pin functions.
RF7/RP38/AN5/ SEG25	RF7	0	O	DIG	LATF<7> data output.
		1	I	ST	PORTF<7> data input.
	RP38	x	x	DIG	Reconfigurable Pin 38 for PPS-Lite; TRIS must be set to match input/ output of module.
	AN5	1	I	ANA	A/D Input Channel 5. Default input configuration on POR; does not affect digital output.
	SEG25	0	O	ANA	LCD Segment 25 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, XCVR = USB Transceiver, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.8 PORTG, LATG and TRISG Registers

PORTG width varies depending on pin count. For 64- and 80-pin devices, PORTG is a 6-bit wide, bidirectional port. For 100-pin devices, PORTG is an 8-bit wide bidirectional port. The corresponding Data Direction and Output Latch registers are TRISG and LATG.

PORTG is multiplexed with the EUSART, and CCP, ECCP, Analog, Comparator, RTCC and Timer input functions (Table 11-7). When operating as I/O, all PORTG pins have Schmitt Trigger input buffers. The open-drain functionality for the CCPx and EUSARTx can be configured using ODCONx.

When enabling peripheral functions, care should be taken in defining TRIS bits for each PORTG pin. Some peripherals override the TRIS bit to make a pin an output, while other peripherals override the TRIS bit to make a pin an input. The user should refer to the corresponding peripheral section for the correct TRIS bit settings. The pin override value is not loaded into the TRIS register. This allows read-modify-write of the TRIS register without concern due to peripheral overrides.

EXAMPLE 11-7: INITIALIZING PORTG

```
CLRF    PORTG       ;  Initialize PORTG by
                    ; clearing output
                    ; data latches
BCF     CM1CON, CON ; disable
                    ; comparator 1
CLRF    LATG        ; Alternate method
                    ; to clear output
                    ; data latches
BANKSEL ANCON2      ; Select bank with ACON2 register
MOVLW   0F0h        ; make AN16 to AN19
                    ; digital
MOVWF   ANCON2
BANKSEL TRISG       ; Select bank with TRISG register
MOVLW   04h         ; Value used to
                    ; initialize data
                    ; direction
MOVWF   TRISG       ; Set RG1:RG0 as
                    ; outputs
                    ; RG2 as input
                    ; RG4:RG3 as inputs
```

TABLE 11-7: PORTG FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RG0/RP46/AN8/ SEG28/COM4	RG0	0	O	DIG	LATG<0> data output; not affected by analog input.
		1	I	ST	PORTG<0> data input; disabled when analog input is enabled.
	RP46	x	x	DIG	Reconfigurable Pin 46 for PPS-Lite; TRIS must be set to match input/ output of module.
	AN8	1	I	ANA	A/D Input Channel 8. Default input configuration on POR; does not affect digital output.
	SEG28	0	O	ANA	LCD Segment 28 output; disables all other pin functions.
	COM4	x	O	ANA	LCD Common 4 output; disables all other outputs.
RG1/RP39/ AN19/SEG29/ COM5	RG1	0	O	DIG	LATG<1> data output; not affected by analog input.
		1	I	ST	PORTG<1> data input; disabled when analog input is enabled.
	RP39	x	x	DIG	Reconfigurable Pin 39 for PPS-Lite; TRIS must be set to match input/ output of module.
	AN19	1	I	ANA	A/D Input Channel 19. Default input configuration on POR; does not affect digital output.
	SEG29	0	O	ANA	LCD Segment 29 output; disables all other pin functions.
	COM5	x	O	ANA	LCD Common 5 output; disables all other outputs.
RG2/RP42/ C3INA/AN18/ SEG30/COM6	RG2	0	O	DIG	LATG<2> data output; not affected by analog input.
		1	I	ST	PORTG<2> data input; disabled when analog input is enabled.
	RP42	x	x	DIG	Reconfigurable Pin 42 for PPS-Lite; TRIS must be set to match input/ output of module.
	C3INA	1	I	ANA	Comparator 3 Input A.
	AN18	1	I	ANA	A/D Input Channel 18. Default input configuration on POR; does not affect digital output.
	SEG30	0	O	ANA	LCD Segment 30 output; disables all other pin functions.
	COM6	x	O	ANA	LCD Common 6 output; disables all other outputs.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-7: PORTG FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RG3/RP43/ C3INB/AN17/ SEG31/COM7	RG3	0	O	DIG	LATG<3> data output; not affected by analog input.
		1	I	ST	PORTG<3> data input; disabled when analog input is enabled.
	RP43	x	x	DIG	Reconfigurable Pin 43 for PPS-Lite; TRIS must be set to match input/ output of module.
	C3INB	1	I	ANA	Comparator 3 Input B.
	AN17	1	I	ANA	A/D Input Channel 17. Default input configuration on POR; does not affect digital output.
	SEG31	0	O	ANA	LCD Segment 31 output; disables all other pin functions.
	COM7	x	O	ANA	LCD Common 7 output; disables all other outputs.
RG4/RTCC/ RP44/C3INC/ AN16/SEG26	RG4	0	O	DIG	LATG<4> data output; not affected by analog input.
		1	I	ST	PORTG<4> data input; disabled when analog input is enabled.
	RTCC	x	O	DIG	RTCC output.
	RP44	x	x	DIG	Reconfigurable Pin 44 for PPS-Lite; TRIS must be set to match input/ output of module.
	C3INC	1	I	ANA	Comparator 3 Input C.
	AN16	1	I	ANA	A/D Input Channel 16. Default input configuration on POR; does not affect digital output.
	SEG26	0	O	ANA	LCD Segment 26 output; disables all other pin functions.
RG6	RG6	0	O	DIG	LATG<6> data output.
		1	I	ST	PORTG<6> data input.
RG7	RG7	0	O	DIG	LATG<7> data output.
		1	I	ST	PORTG<7> data input.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.9 PORTH, LATH and TRISH Registers

Note: PORTH is available only on 80-pin and 100-pin devices.

PORTH is an 8-bit wide, bidirectional I/O port. The corresponding Data Direction and Output Latch registers are TRISH and LATH.

All pins on PORTH are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

EXAMPLE 11-8: INITIALIZING PORTH

```
CLRF     PORTH    ; Initialize PORTH by
                  ; clearing output
                  ; data latches
CLRF     LATH     ; Alternate method
                  ; to clear output
                  ; data latches
BANKSEL  ANCON2   ; Select bank with ANCON2 register
MOVLW    0Fh      ; Configure PORTH as
MOVWF    ANCON2   ; digital I/O
MOVLW    0Fh      ; Configure PORTH as
MOVWF    ANCON1   ; digital I/O
BANKSEL  TRISH    ; Select bank with TRISH register
MOVLW    0CFh     ; Value used to
                  ; initialize data
                  ; direction
MOVWF    TRISH    ; Set RH3:RH0 as inputs
                  ; RH5:RH4 as outputs
                  ; RH7:RH6 as inputs
```

TABLE 11-8: PORTH FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RH0/AN23/ SEG47/A16	RH0	0	O	DIG	LATH<0> data output; not affected by analog input.
		1	I	ST	PORTH<0> data input.
	AN23	1	I	ANA	A/D Input Channel 23. Default input configuration on POR; does not affect digital output.
	SEG47	0	O	ANA	LCD Segment 47 output; disables all other pin functions.
	A16	x	O	DIG	External Memory Bus Address<16> output.
RH1/AN22/ SEG46/A17	RH1	0	O	DIG	LATH<1> data output; not affected by analog input.
		1	I	ST	PORTH<1> data input.
	AN22	1	I	ANA	A/D Input Channel 22. Default input configuration on POR; does not affect digital output.
	SEG46	0	O	ANA	LCD Segment 46 output; disables all other pin functions.
	A17	x	O	DIG	External Memory Bus Address<17> output.
RH2/AN21/ SEG45/A18	RH2	0	O	DIG	LATH<2> data output; not affected by analog input.
		1	I	ST	PORTH<2> data input.
	AN21	1	I	ANA	A/D Input Channel 21. Default input configuration on POR; does not affect digital output.
	SEG45	0	O	ANA	LCD Segment 45 output; disables all other pin functions.
	A18	x	O	DIG	External Memory Bus Address<18> output.
RH3/AN20/ SEG44/A19	RH3	0	O	DIG	LATH<3> data output; not affected by analog input.
		1	I	ST	PORTH<3> data input.
	AN20	1	I	ANA	A/D Input Channel 20. Default input configuration on POR; does not affect digital output.
	SEG44	0	O	ANA	LCD Segment 44 output; disables all other pin functions.
	A19	x	O	DIG	External Memory Bus Address<19> output.
RH4/C2INC/ AN12/SEG40	RH4	0	O	DIG	LATH<4> data output; not affected by analog input.
		1	I	ST	PORTH<4> data input; disabled when analog input is enabled.
	C2INC	1	I	ANA	Comparator 2 Input C.
	AN12	1	I	ANA	A/D Input Channel 12. Default input configuration on POR; does not affect digital output.
	SEG40	0	O	ANA	LCD Segment 40 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-8: PORTH FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RH5/C2IND/ AN13/SEG41	RH5	0	O	DIG	LATH<5> data output; not affected by analog input.
		1	I	ST	PORTH<5> data input; disabled when analog input is enabled.
	C2IND	1	I	ANA	Comparator 2 Input D.
	AN13	1	I	ANA	A/D Input Channel 13. Default input configuration on POR; does not affect digital output.
	SEG41	0	O	ANA	LCD Segment 41 output; disables all other pin functions.
RH6/C1INC/ AN14/SEG42	RH6	0	O	DIG	LATH<6> data output; not affected by analog input.
		1	I	ST	PORTH<6> data input; disabled when analog input is enabled.
	C1INC	1	I	ANA	Comparator 1 Input C.
	AN14	1	I	ANA	A/D Input Channel 14. Default input configuration on POR; does not affect digital output.
	SEG42	0	O	ANA	LCD Segment 42 output; disables all other pin functions.
RH7/AN15/ SEG43	RH7	0	O	DIG	LATH<7> data output; not affected by analog input.
		1	I	ST	PORTH<7> data input; disabled when analog input is enabled.
	AN15	1	I	ANA	A/D Input Channel 15. Default input configuration on POR; does not affect digital output.
	SEG43	0	O	ANA	LCD Segment 43 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.10 PORTJ, LATJ and TRISJ Registers

Note: PORTJ is available only on 80-pin and 100-pin devices.

PORTJ is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISJ and LATJ.

All pins on PORTJ are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Note: These pins are configured as digital inputs on any device Reset.

When the external memory interface is enabled, all of the PORTJ pins function as control outputs for the interface. This occurs automatically when the interface is enabled by clearing the EBDIS control bit (MEMCON<7>). The TRISJ bits are also overridden.

Each of the PORTJ pins has a weak internal pull-up. The pull-ups are provided to keep the inputs at a known state for the external memory interface while powering up. A single control bit can turn off all the pull-ups. This is performed by clearing bit, RJPU (PADCFG<2>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on any device Reset.

EXAMPLE 11-9: INITIALIZING PORTJ

```
CLRF    PORTJ   ; Initialize PORTJ by
                ; clearing output latches
CLRF    LATJ    ; Alternate method
                ; to clear output latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISJ   ; Set RJ3:RJ0 as inputs
                ; RJ5:RJ4 as output
                ; RJ7:RJ6 as inputs
```

TABLE 11-9: PORTJ FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RJ0/SEG32/ ALE	RJ0	0	O	DIG	LATJ<0> data output.
		1	I	ST	PORTJ<0> data input.
	SEG32	0	O	ANA	LCD Segment 32 output; disables all other pin functions.
	ALE	x	O	DIG	External Memory Bus Address Latch Enable (ALE) signal.
RJ1/SEG33/$\overline{OE}$	RJ1	0	O	DIG	LATJ<1> data output.
		1	I	ST	PORTJ<1> data input.
	SEG33	0	O	ANA	LCD Segment 33 output; disables all other pin functions.
	$\overline{OE}$	x	O	DIG	External Memory Bus Address Latch Enable ($\overline{OE}$) signal.
RJ2/SEG34/ $\overline{WRL}$	RJ2	0	O	DIG	LATJ<2> data output.
		1	I	ST	PORTJ<2> data input.
	SEG34	0	O	ANA	LCD Segment 34 output; disables all other pin functions.
	$\overline{WRL}$	x	O	DIG	External Memory Bus Write Low ($\overline{WRL}$) signal.
RJ3/SEG35/ $\overline{WRH}$	RJ3	0	O	DIG	LATJ<3> data output.
		1	I	ST	PORTJ<3> data input.
	SEG35	0	O	ANA	LCD Segment 35 output; disables all other pin functions.
	$\overline{WRH}$	x	O	DIG	External Memory Bus Write High ($\overline{WRH}$) signal.
RJ4/SEG39/ BA0	RJ4	0	O	DIG	LATJ<4> data output.
		1	I	ST	PORTJ<4> data input.
	SEG39	0	O	ANA	LCD Segment 39 output; disables all other pin functions.
	BA0	x	O	DIG	External Memory Bus Byte Access 0 (BA0) signal.
RJ5/SEG38/$\overline{CE}$	RJ5	0	O	DIG	LATJ<5> data output.
		1	I	ST	PORTJ<5> data input.
	SEG38	0	O	ANA	LCD Segment 38 output; disables all other pin functions.
	$\overline{CE}$	x	O	DIG	External Memory Bus Chip Enable ($\overline{CE}$) signal.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

TABLE 11-9: PORTJ FUNCTIONS (CONTINUED)

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RJ6/SEG37/$\overline{LB}$	RJ6	0	O	DIG	LATJ<6> data output.
		1	I	ST	PORTJ<6> data input.
	SEG37	0	O	ANA	LCD Segment 37 output; disables all other pin functions.
	$\overline{LB}$	x	O	DIG	External Memory Bus Lower Byte ($\overline{LB}$) signal.
RJ7/SEG36/$\overline{UB}$	RJ7	0	O	DIG	LATJ<7> data output.
		1	I	ST	PORTJ<7> data input.
	SEG36	0	O	ANA	LCD Segment 36 output; disables all other pin functions.
	$\overline{UB}$	x	O	DIG	External Memory Bus Upper Byte ($\overline{UB}$) signal.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input,
x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.11 PORTK, LATK and TRISK Registers

Note: PORTK is available only on 100-pin devices.

PORTK is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISK and LATK.

All pins on PORTK are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Each of the PORTK pins has a weak internal pull-up. The pull-ups are provided to keep the inputs at a known state for the external memory interface while powering up. A single control bit can turn off all the pull-ups. This is performed by clearing bit, RKPU (PADCFG<1>).

The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on any device Reset.

EXAMPLE 11-10: INITIALIZING PORTK

```
BANKSEL  LATK    ; select bank with LATK register
CLRF     LATK    ; Initialize LATK
                 ; by clearing output
                 ; data latches
BANKSEL  TRISK   ; Select bank with TRISK register
MOVLW    0CFh    ; Value used to
                 ; initialize data
                 ; direction
MOVWF    TRISK   ; Set RH3:RH0 as inputs
                 ; RH5:RH4 as outputs
                 ; RH7:RH6 as inputs
```

TABLE 11-10: PORTK FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RK0/SEG56	RK0	0	O	DIG	LATK<0> data output.
		1	I	ST	PORTK<0> data input.
	SEG56	0	O	ANA	LCD Segment 56 output; disables all other pin functions.
RK1/SEG57	RK1	0	O	DIG	LATK<1> data output.
		1	I	ST	PORTK<1> data input.
	SEG57	0	O	ANA	LCD Segment 57 output; disables all other pin functions.
RK2/SEG58	RK2	0	O	DIG	LATK<2> data output.
		1	I	ST	PORTK<2> data input.
	SEG58	0	O	ANA	LCD Segment 58 output; disables all other pin functions.
RK3/SEG59	RK3	0	O	DIG	LATK<3> data output.
		1	I	ST	PORTK<3> data input.
	SEG59	0	O	ANA	LCD Segment 59 output; disables all other pin functions.
RK4/SEG60	RK4	0	O	DIG	LATK<4> data output.
		1	I	ST	PORTK<4> data input.
	SEG60	0	O	ANA	LCD Segment 60 output; disables all other pin functions.
RK5/SEG61	RK5	0	O	DIG	LATK<5> data output.
		1	I	ST	PORTK<5> data input.
	SEG61	0	O	ANA	LCD Segment 61 output; disables all other pin functions.
RK6/SEG62	RK6	0	O	DIG	LATK<6> data output.
		1	I	ST	PORTK<6> data input.
	SEG62	0	O	ANA	LCD Segment 62 output; disables all other pin functions.
RK7/SEG63	RK7	0	O	DIG	LATK<7> data output.
		1	I	ST	PORTK<7> data input.
	SEG63	0	O	ANA	LCD Segment 63 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.12 PORTL, LATL and TRISL Registers

Note: PORTL is available only on 100-pin devices.

PORTL is an 8-bit wide, bidirectional port. The corresponding Data Direction and Output Latch registers are TRISL and LATL.

All pins on PORTL are implemented with Schmitt Trigger input buffers. Each pin is individually configurable as an input or output.

Each of the PORTL pins has a weak internal pull-up.

The pull-ups are provided to keep the inputs at a known state for the external memory interface while powering up. A single control bit can turn off all the pull-ups. This is performed by clearing bit, RLPU (PADCFG<0>).

The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on any device Reset.

EXAMPLE 11-11: INITIALIZING PORTL

```
BANKSEL PORTL   ; select correct bank
CLRF    PORTL   ; Initialize PORTL by
                ; clearing output latches
CLRF    LATL    ; Alternate method
                ; to clear output latches
MOVLW   0CFh    ; Value used to
                ; initialize data
                ; direction
MOVWF   TRISL   ; Set RL3:RL0 as inputs
                ; RL5:RL4 as output
                ; RL7:RL6 as inputs
```

TABLE 11-11: PORTL FUNCTIONS

Pin Name	Function	TRIS Setting	I/O	I/O Type	Description
RL0/SEG48	RL0	0	O	DIG	LATL<0> data output.
		1	I	ST	PORTL<0> data input.
	SEG48	0	O	ANA	LCD Segment 48 output; disables all other pin functions.
RL1/SEG49	RL1	0	O	DIG	LATL<1> data output.
		1	I	ST	PORTL<1> data input.
	SEG49	0	O	ANA	LCD Segment 49 output; disables all other pin functions.
RL2/SEG50	RL2	0	O	DIG	LATL<2> data output.
		1	I	ST	PORTL<2> data input.
	SEG50	0	O	ANA	LCD Segment 50 output; disables all other pin functions.
RL3/SEG51	RL3	0	O	DIG	LATL<3> data output.
		1	I	ST	PORTL<3> data input.
	SEG51	0	O	ANA	LCD Segment 51 output; disables all other pin functions.
RL4/SEG52	RL4	0	O	DIG	LATL<4> data output.
		1	I	ST	PORTL<4> data input.
	SEG52	0	O	ANA	LCD Segment 52 output; disables all other pin functions.
RL5/SEG53	RL5	0	O	DIG	LATL<5> data output.
		1	I	ST	PORTL<5> data input.
	SEG53	0	O	ANA	LCD Segment 53 output; disables all other pin functions.
RL6/SEG54	RL6	0	O	DIG	LATL<6> data output.
		1	I	ST	PORTL<6> data input.
	SEG54	0	O	ANA	LCD Segment 54 output; disables all other pin functions.
RL7/SEG55	RL7	0	O	DIG	LATL<7> data output.
		1	I	ST	PORTL<7> data input.
	SEG55	0	O	ANA	LCD Segment 55 output; disables all other pin functions.

Legend: O = Output, I = Input, ANA = Analog Signal, DIG = Digital Output, ST = Schmitt Trigger Buffer Input, x = Don't care (TRIS bit does not affect port direction or is overridden for this option).

11.13 Parallel Slave Port

PORTD can function as an 8-bit-wide Parallel Slave Port (PSP), or microprocessor port, when control bit, PSPMODE (PSPCON<4>), is set. The port is asynchronously readable and writable by the external world through the $\overline{RD}$ control input pin (RE0/AD8/LCDBIAS1/RP28/$\overline{RD}$) and $\overline{WR}$ control input pin (RE1/AD9/LCDBIAS2/RP29/$\overline{WR}$).

Note:	The Parallel Slave Port is available only in Microcontroller mode.

The PSP can directly interface to an 8-bit microprocessor data bus. The external microprocessor can read or write the PORTD latch as an 8-bit latch.

Setting bit, PSPMODE, enables port pin, RE0/AD8/LCDBIAS1/RP28/$\overline{RD}$, to be the $\overline{RD}$ input, RE1/AD9/LCDBIAS2/RP29/$\overline{WR}$ to be the $\overline{WR}$ input and RE2/AD10/LCDBIAS3/RP30/$\overline{CS}$ to be the $\overline{CS}$ (Chip Select) input. For this functionality, the corresponding data direction bits of the TRISE register (TRISE<2:0>) must be configured as inputs (‘111’).

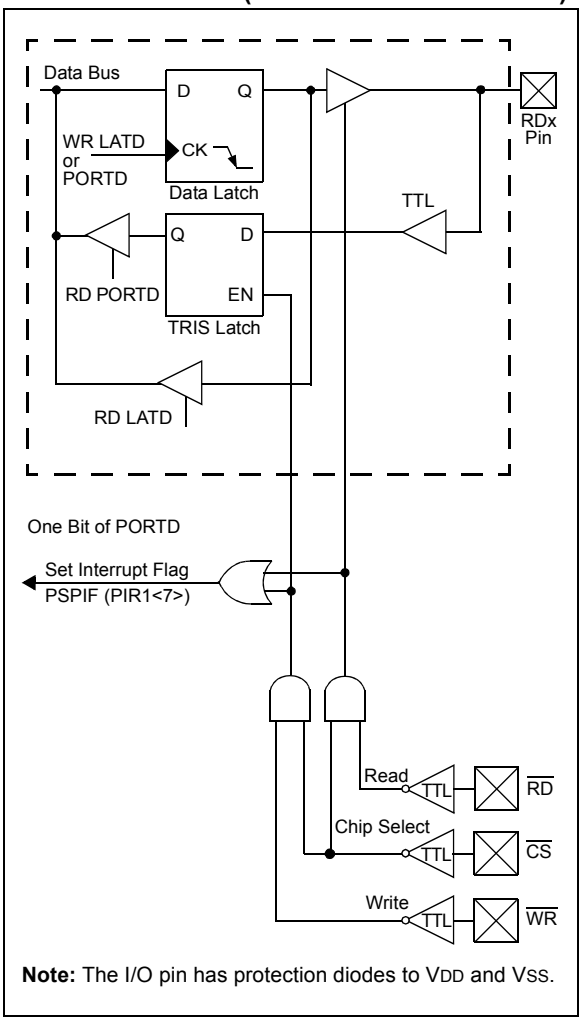
A write to the PSP occurs when both the $\overline{CS}$ and $\overline{WR}$ lines are first detected low and ends when either are detected high. The PSPIF and IBF flag bits (PIR1<7> and PSPCON<7>, respectively) are set when the write ends.

A read from the PSP occurs when both the $\overline{CS}$ and $\overline{RD}$ lines are first detected low. The data in PORTD is read out and the OBF bit (PSPCON<6>) is set. If the user writes new data to PORTD to set OBF, the data is immediately read out, but the OBF bit is not set.

When either the $\overline{CS}$ or $\overline{RD}$ line is detected high, the PORTD pins return to the input state and the PSPIF bit is set. User applications should wait for PSPIF to be set before servicing the PSP. When this happens, the IBF and OBF bits can be polled and the appropriate action taken.

The timing for the control signals in Write and Read modes is shown in Figure 11-4 and Figure 11-5, respectively.

FIGURE 11-3: PORTD AND PORTE BLOCK DIAGRAM (PARALLEL SLAVE PORT)



Note: The I/O pin has protection diodes to VDD and VSS.

REGISTER 11-4: PSPCON: PARALLEL SLAVE PORT CONTROL REGISTER

R-0	R-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0
IBF	OBF	IBOV	PSPMODE	—	—	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **IBF:** Input Buffer Full Status bit
1 = A word has been received and is waiting to be read by the CPU
0 = No word has been received

bit 6 **OBF:** Output Buffer Full Status bit
1 = The output buffer still holds a previously written word
0 = The output buffer has been read

bit 5 **IBOV:** Input Buffer Overflow Detect bit
1 = A write occurred when a previously input word had not been read (must be cleared in software)
0 = No overflow occurred

bit 4 **PSPMODE:** Parallel Slave Port Mode Select bit
1 = Parallel Slave Port mode
0 = General Purpose I/O mode

bit 3-0 **Unimplemented:** Read as '0'

FIGURE 11-4: PARALLEL SLAVE PORT WRITE WAVEFORMS

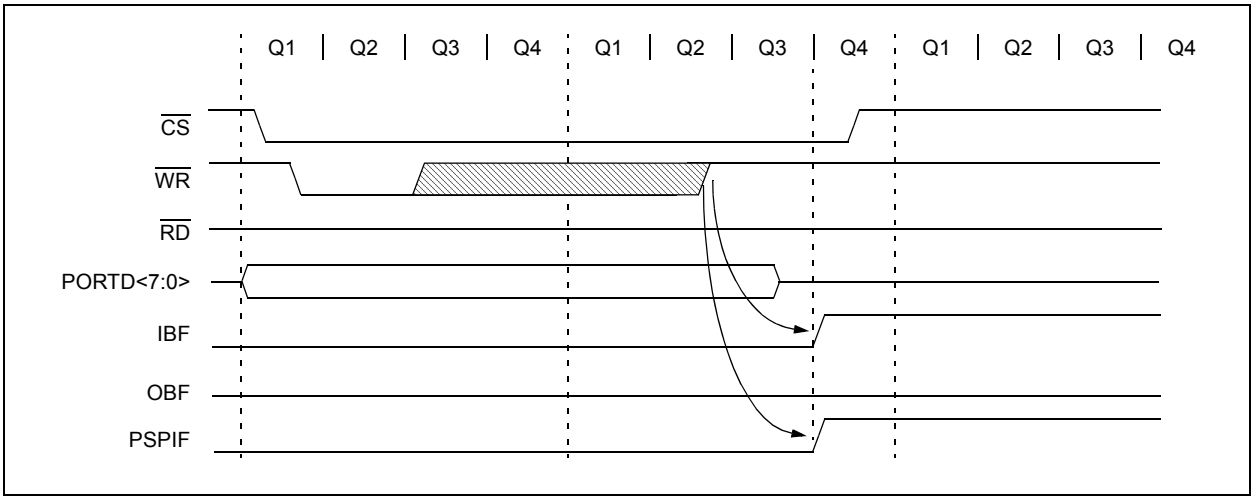
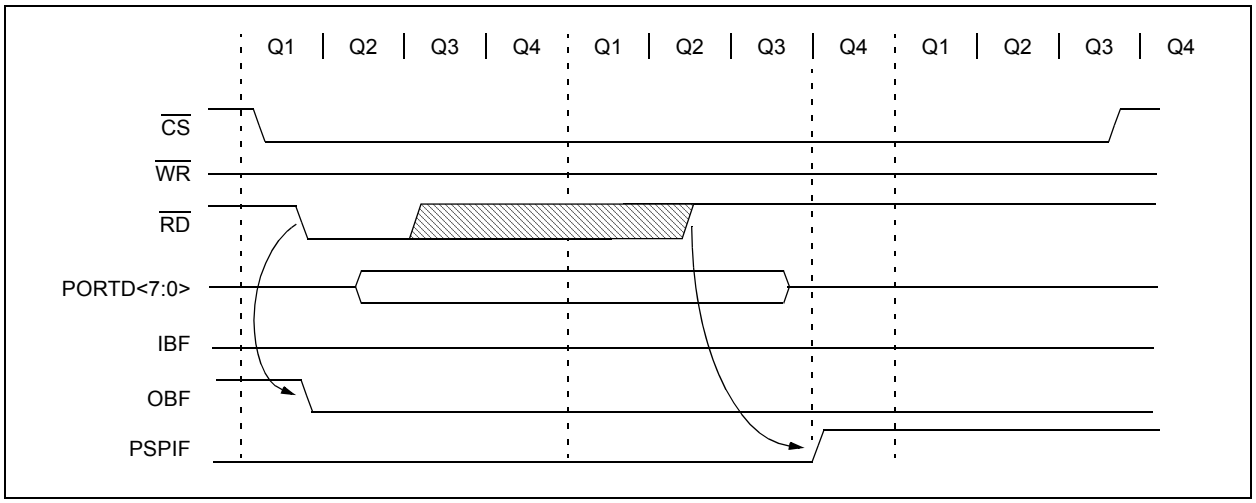


FIGURE 11-5: PARALLEL SLAVE PORT READ WAVEFORMS



11.14 Virtual PORT

This device includes a single virtual port, which is used to construct a logically addressed 8-bit PORT from 8 physically unrelated pins on the device. The virtual PORT is controlled through the PORTVP, LATVP and TRISVP registers. These function identically to the PORT, LAT and TRIS registers of the actual I/O ports. Refer to **Section 11.1 "I/O Port Pin Capabilities"** for more information.

11.15 PPS-Lite

Previous PIC18 devices had I/O pins that were "hard-wired" to a set of peripherals. For example, a port pin might have had the option of serving as an I/O pin, an analog input or as an interrupt source. In an effort to increase the flexibility of the parts, PIC18FXXJ94 devices contain PPS-Lite (Peripheral Pin Select-Lite), which allows the developer to connect an internal peripheral to a subset of pins. PPS-Lite is similar to PPS (available on PIC18F products), but limits the user to interconnections within four sets of pin/peripheral groups.

The PPS-Lite feature allows some flexibility in choosing which peripheral connects to any particular pin. This allows designs to be maximized for layout efficiency, and also may allow component changes without changing the printed circuit board design. The Peripheral Pin Select feature operates over a fixed subset of digital I/O pins (those designated as RPn pins). Users may independently map the input and/or output of most digital peripherals to a limited set of these I/O pins. The PPS-Lite configuration is performed in software and does not require the device to be reprogrammed. Hardware safeguards are included that prevent accidental or spurious changes to the peripheral mapping once it has been established.

11.15.1 AVAILABLE PINS

The PPS-Lite feature is used with a range of pins. All devices in the PIC18FXXJ94 family contain a total of 47 remappable peripheral pins, labeled RP0 through RP46. Pins that support PPS-Lite feature include the designation, "RPn" in their full pin designation, where "RP" designates a remappable peripheral and "n" is the remappable pin number. For PIC18FXXJ94 devices, RP41 through RP45 are digital inputs only.

11.15.2 AVAILABLE PERIPHERALS

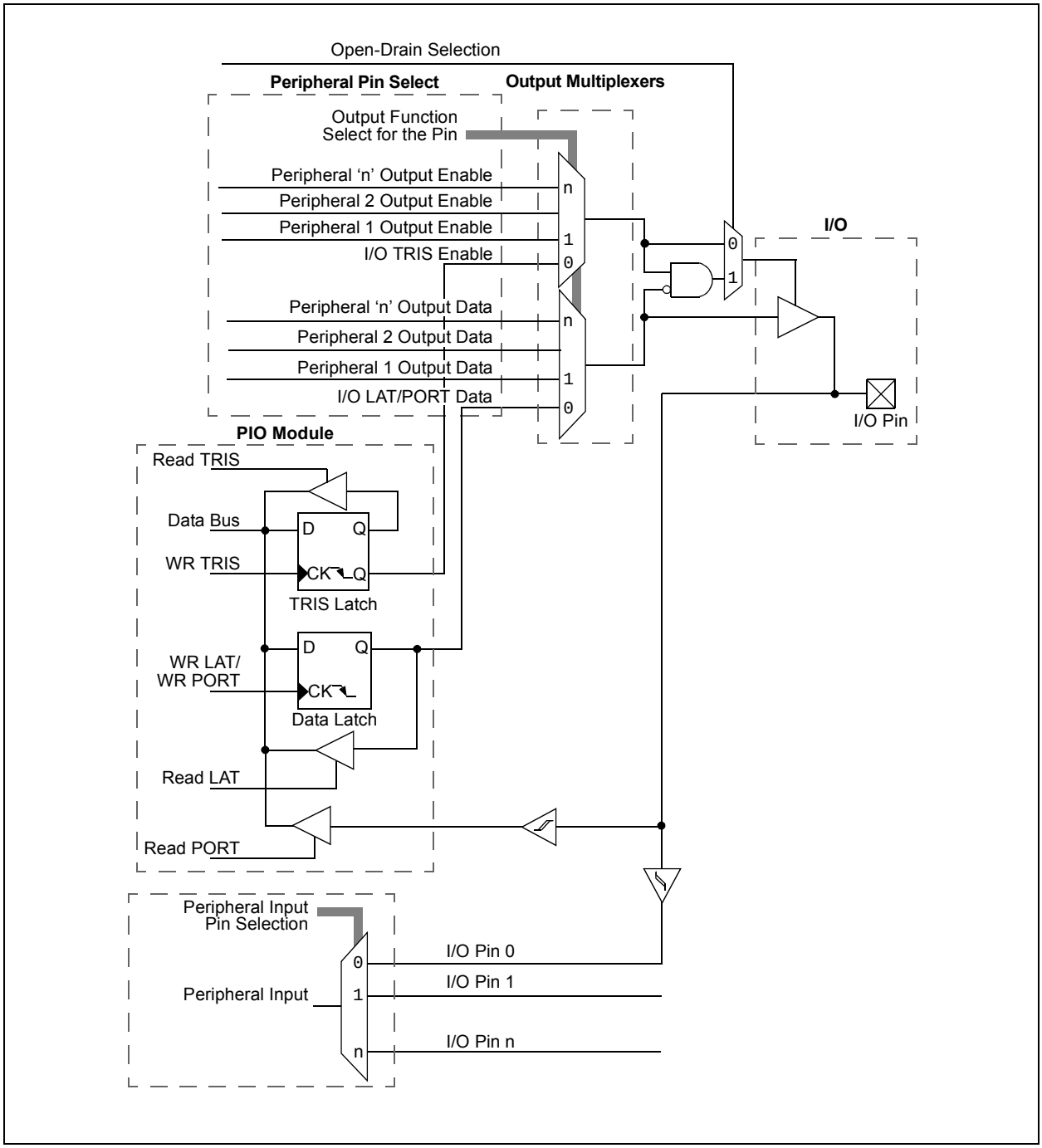
The peripherals managed by the Peripheral Pin Select are all "digital only" peripherals. These include general serial communications (USART and SPI), general purpose timer clock inputs, timer related peripherals (input capture and output compare) and external interrupt inputs.

In comparison, some digital only peripheral modules are not currently included in the Peripheral Pin Select feature. This is because the peripheral's function requires special I/O circuitry on a specific port and cannot be easily connected to multiple pins. These modules include I^2C, USB, change notification inputs, RTCC alarm output and all modules with analog inputs, such as the A/D Converter.

A key difference between remappable and non-remappable peripherals is that remappable peripherals are not associated with a default I/O pin. The peripheral must always be assigned to a specific I/O pin before it can be used. In contrast, non-remappable peripherals are always available on a default pin, assuming that the peripheral is active and not conflicting with another peripheral.

When a remappable peripheral is active on a given I/O pin, it takes priority over all other digital I/O and digital communication peripherals associated with the pin. Priority is given, regardless of the type of peripheral that is mapped. Remappable peripherals never take priority over any analog functions associated with the pin.

FIGURE 11-6: STRUCTURE OF PORT SHARED WITH PPS PERIPHERALS



11.15.3 CONTROLLING PERIPHERAL PIN SELECT

Peripheral Pin Select features are controlled through two sets of Special Function Registers (SFRs): one to map peripheral inputs and one to map peripheral outputs. Because they are separately controlled, a particular peripheral's input and output (if the peripheral has both) can be placed on any selectable function with the only constraint being that RPn peripherals and pins can only be mapped within their own group. It is not possible to map a peripheral to a pin outside of its group or vice versa.

The association of a peripheral to a peripheral-selectable pin is handled in two different ways, depending if an input or output is being mapped.

11.15.3.1 Input Mapping

The inputs of the Peripheral Pin Select options are mapped on the basis of the peripheral; that is, a bit field associated with a peripheral dictates the pin it will be mapped to. The RPINRx registers (refer to registers in Table 11-12 and Table 11-13) contain sets of 4-bit fields, with each set associated with one of the remappable peripherals. Programming a given peripheral's bit field with an RPn value maps the RPn pin to that peripheral. For any given device, the valid range of values for any of the bit fields corresponds to the maximum number of peripheral Pin Selections supported by the device.

The PPS-Lite peripheral inputs and associated RPn pins have been organized into four groups. It is not possible to map a peripheral to an RPn pin which is outside of its group. To map a peripheral input signal to an RPn pin, use the 4-step process as indicated in Table 11-13. Choose the signal and the RPn pin, and the column on the right shows which value to write to the associated RPIN register.

The peripheral inputs that support Peripheral Pin Selection have no default pins. Since the implemented bit fields of RPINRx registers reset to all '1's, the inputs are all tied to VSS in the device's default (Reset) state.

For example, to assign U1RX to RP3, write the value, h'0, to RPINR0_1<3:0>. Figure 11-7 illustrates remappable pin selection for the U1RX input.

FIGURE 11-7: REMAPPABLE INPUT FOR U1RX

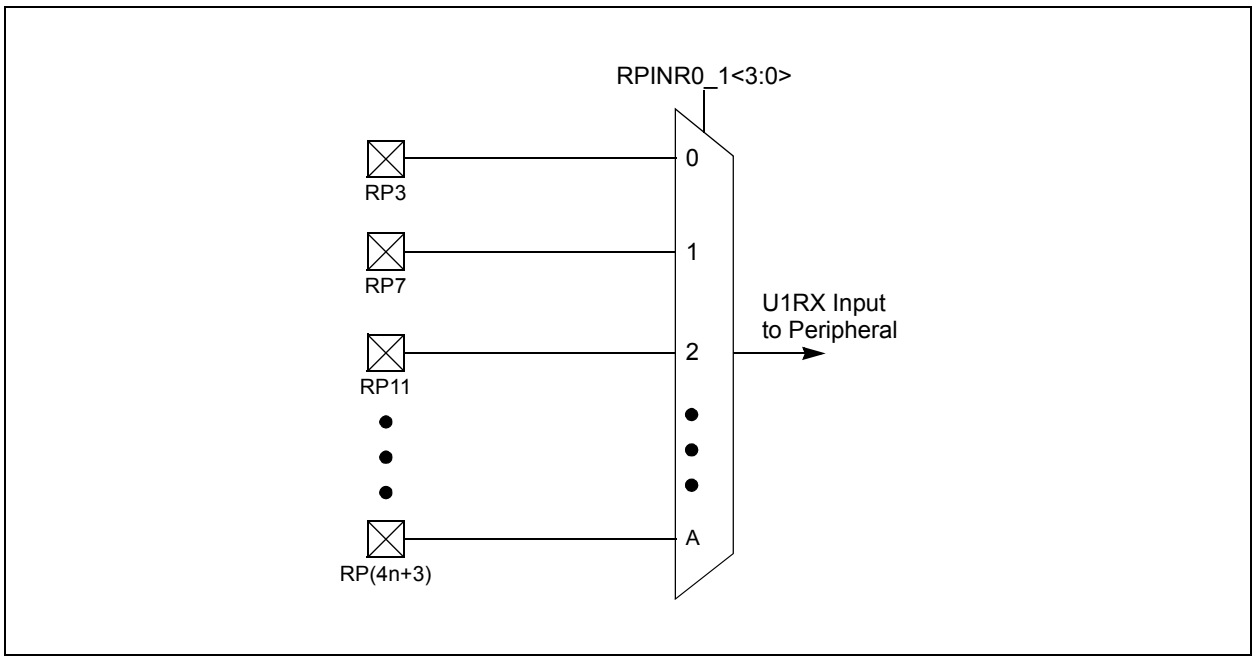


TABLE 11-12: RPINR REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RPINR52_53	RVP7R3	RVP7R2	RVP7R1	RVP7R0	RVP6R3	RVP6R2	RVP6R1	RVP6R0
RPINR50_51	RVP5R3	RVP5R2	RVP5R1	RVP5R0	RVP4R3	RVP4R2	RVP4R1	RVP4R0
RPINR48_49	RVP3R3	RVP3R2	RVP3R1	RVP3R0	RVP2R3	RVP2R2	RVP2R1	RVP2R0
RPINR46_47	RVP1R3	RVP1R2	RVP1R1	RVP1R0	RVP0R3	RVP0R2	RVP0R1	RVP0R0
RPINR44_45	T5CKIR3	T5CKIR2	T5CKIR1	T5CKIR0	T5GR3	T5GR2	T5GR1	T5GR0
RPINR42_43	T3CKIR3	T3CKIR2	T3CKIR1	T3CKIR0	T3GR3	T3GR2	T3GR1	T3GR0
RPINR40_41	T1CKIR3	T1CKIR2	T1CKIR1	T1CKIR0	T1GR3	T1GR2	T1GR1	T1GR0
RPINR38_39	T0CKIR3	T0CKIR2	T0CKIR1	T0CKIR0	CCP10R3	CCP10R2	CCP10R1	CCP10R0
RPINR36_37	CCP9R3	CCP9R2	CCP9R1	CCP9R0	CCP8R3	CCP8R2	CCP8R1	CCP8R0
RPINR34_35	CCP7R3	CCP7R2	CCP7R1	CCP7R0	CCP6R3	CCP6R2	CCP6R1	CCP6R0
RPINR32_33	CCP5R3	CCP5R2	CCP5R1	CCP5R0	CCP4R3	CCP4R2	CCP4R1	CCP4R0
RPINR30_31	MDCIN2R3	MDCIN2R2	MDCIN2R1	MDCIN2R0	MDCIN1R3	MDCIN1R2	MDCIN1R1	MDCIN1R0
RPINR28_29	MDMINR3	MDMINR2	MDMINR1	MDMINR0	INT3R3	INT3R2	INT3R1	INT3R0
RPINR26_27	INT2R3	INT2R2	INT2R1	INT2R0	INT1R3	INT1R2	INT1R1	INT1R0
RPINR24_25	IOC7R3	IOC7R2	IOC7R1	IOC7R0	IOC6R3	IOC6R2	IOC6R1	IOC6R0
RPINR22_23	IOC5R3	IOC5R2	IOC5R1	IOC5R0	IOC4R3	IOC4R2	IOC4R1	IOC4R0
RPINR20_21	IOC3R3	IOC3R2	IOC3R1	IOC3R0	IOC2R3	IOC2R2	IOC2R1	IOC2R0
RPINR18_19	IOC1R3	IOC1R2	IOC1R1	IOC1R0	IOC0R3	IOC0R2	IOC0R1	IOC0R0
RPINR16_17	ECCP3R3	ECCP3R2	ECCP3R1	ECCP3R0	ECCP2R3	ECCP2R2	ECCP2R1	ECCP2R0
RPINR14_15	ECCP1R3	ECCP1R2	ECCP1R1	ECCP1R0	FLT0R3	FLT0R2	FLT0R1	FLT0R0
RPINR12_13	SS2R3	SS2R2	SS2R1	SS2R0	SDI2R3	SDI2R2	SDI2R1	SDI2R0
RPINR10_11	SCK2R3	SCK2R2	SCK2R1	SCK2R0	SS1R3	SS1R2	SS1R1	SS1R0
RPINR8_9	SDI1R3	SDI1R2	SDI1R1	SDI1R0	SCK1R3	SCK1R2	SCK1R1	SCK1R0
RPINR6_7	U4TXR3	U4TXR2	U4TXR1	U4TXR0	U4RXR3	U4RXR2	U4RXR1	U4RXR0
RPINR4_5	U3TXR3	U3TXR2	U3TXR1	U3TXR0	U3RXR3	U3RXR2	U3RXR1	U3RXR0
RPINR2_3	U2TXR3	U2TXR2	U2TXR1	U2TXR0	U2RXR3	U2RXR2	U2RXR1	U2RXR0
RPINR0_1	U1TXR3	U1TXR2	U1TXR1	U1TXR0	U1RXR3	U1RXR2	U1RXR1	U1RXR0

TABLE 11-13: RPIN REGISTERS AND AVAILABLE FUNCTIONS

PPS-Lite Input Peripheral Group 4n	
(1) To Map this signal	(4) to the Associated RPIN Register
SDI1	RPINR8_9<7:4>
FLT0	RPINR14_15<3:0>
IOC0	RPINR18_19<3:0>
IOC4	RPINR22_23<3:0>
MDCIN1	RPINR30_31<3:0>
T0CKI	RPINR38_39<7:4>
T5G	RPINR44_45<3:0>
U3RX	RPINR4_5<3:0>
U4RX	RPINR6_7<3:0>
CCP5	RPINR32_33<7:4>
CCP8	RPINR36_37<3:0>
RVP0	RPINR46_47<3:0>
RVP4	RPINR50_51<3:0>
(2) with this RPn Pin	(3) Write this Corresponding Value
RP0	h'0
RP4	h'1
RP8	h'2
RP12	h'3
RP16	h'4
RP20	h'5
RP24	h'6
RP28	h'7
RP32	h'8
RP36	h'9
RP40	h'A
RP44	h'B
—	h'C
—	h'D
—	h'E
Vss	h'F

PPS-Lite Input Peripheral Group 4n + 1	
(1) To Map this Signal	(4) to the Associated RPIN Register
SDI2	RPINR12_13<3:0>
INT1	RPINR26_27<3:0>
IOC1	RPINR18_19<7:4>
IOC5	RPINR22_23<7:4>
MDCIN2	RPINR30_31<7:4>
T1CKI	RPINR40_41<7:4>
T1G	RPINR40_41<3:0>
T3CKI	RPINR42_43<7:4>
T3G	RPINR42_43<3:0>
T5CKI	RPINR44_45<7:4>
U3TX	RPINR4_5<7:4>
U4TX	RPINR6_7<7:4>
CCP7	RPINR34_35<7:4>
CCP9	RPINR36_37<7:4>
RVP1	RPINR46_47<7:4>
RVP5	RPINR50_51<7:4>
(2) with this RPn Pin	(3) Write this Corresponding Value
RP1	h'0
RP5	h'1
RP9	h'2
RP13	h'3
RP17	h'4
RP21	h'5
RP25	h'6
RP29	h'7
RP33	h'8
RP37	h'9
RP41	h'A
RP45	h'B
—	h'C
—	h'D
—	h'E
Vss	h'F

TABLE 11-13: RPIN REGISTERS AND AVAILABLE FUNCTIONS (CONTINUED)

PPS-Lite Input Peripheral Group 4n + 2	
(1) To Map this Signal	(4) to the Associated RPIN Register
$\overline{SS1}$	RPINR10_11<3:0>
INT2	RPINR26_27<7:4>
IOC2	RPINR20_21<3:0>
IOC6	RPINR24_25<3:0>
MDMIN	RPINR28_29<7:4>
U1TX	RPINR0_1<7:4>
U2RX	RPINR2_3<3:0>
SCK2	RPINR10_11<7:4>
ECCP3	RPINR16_17<7:4>
CCP6	RPINR34_35<3:0>
CCP10	RPINR38_39<3:0>
RVP2	RPINR48_49<3:0>
RVP6	RPINR52_53<3:0>
(2) with this RPn Pin	(3) Write this Corresponding Value
RP2	h'0
RP6	h'1
RP10	h'2
RP14	h'3
RP18	h'4
RP22	h'5
RP26	h'6
RP30	h'7
RP34	h'8
RP38	h'9
RP42	h'A
RP46	h'B
—	h'C
—	h'D
—	h'E
Vss	h'F

PPS-Lite Input Peripheral Group 4n + 3	
(1) To Map this Signal	(4) to the Associated RPIN Register
$\overline{SS2}$	RPINR12_13<7:4>
INT3	RPINR28_29<3:0>
IOC3	RPINR20_21<7:4>
IOC7	RPINR24_25<7:4>
U1RX	RPINR0_1<3:0>
U2TX	RPINR2_3<7:4>
SCK1	RPINR8_9<3:0>
ECCP1	RPINR14_15<7:4>
ECCP2	RPINR16_17<3:0>
CCP4	RPINR32_33<3:0>
RVP3	RPINR48_49<7:4>
RVP7	RPINR52_53<7:4>
(2) with this RPn Pin	(3) Write this Corresponding Value
RP3	h'0
RP7	h'1
RP11	h'2
RP15	h'3
RP19	h'4
RP23	h'5
RP27	h'6
RP31	h'7
RP35	h'8
RP39	h'9
RP43	h'A
—	h'B
—	h'C
—	h'D
—	h'E
Vss	h'F

11.15.3.2 Output Mapping

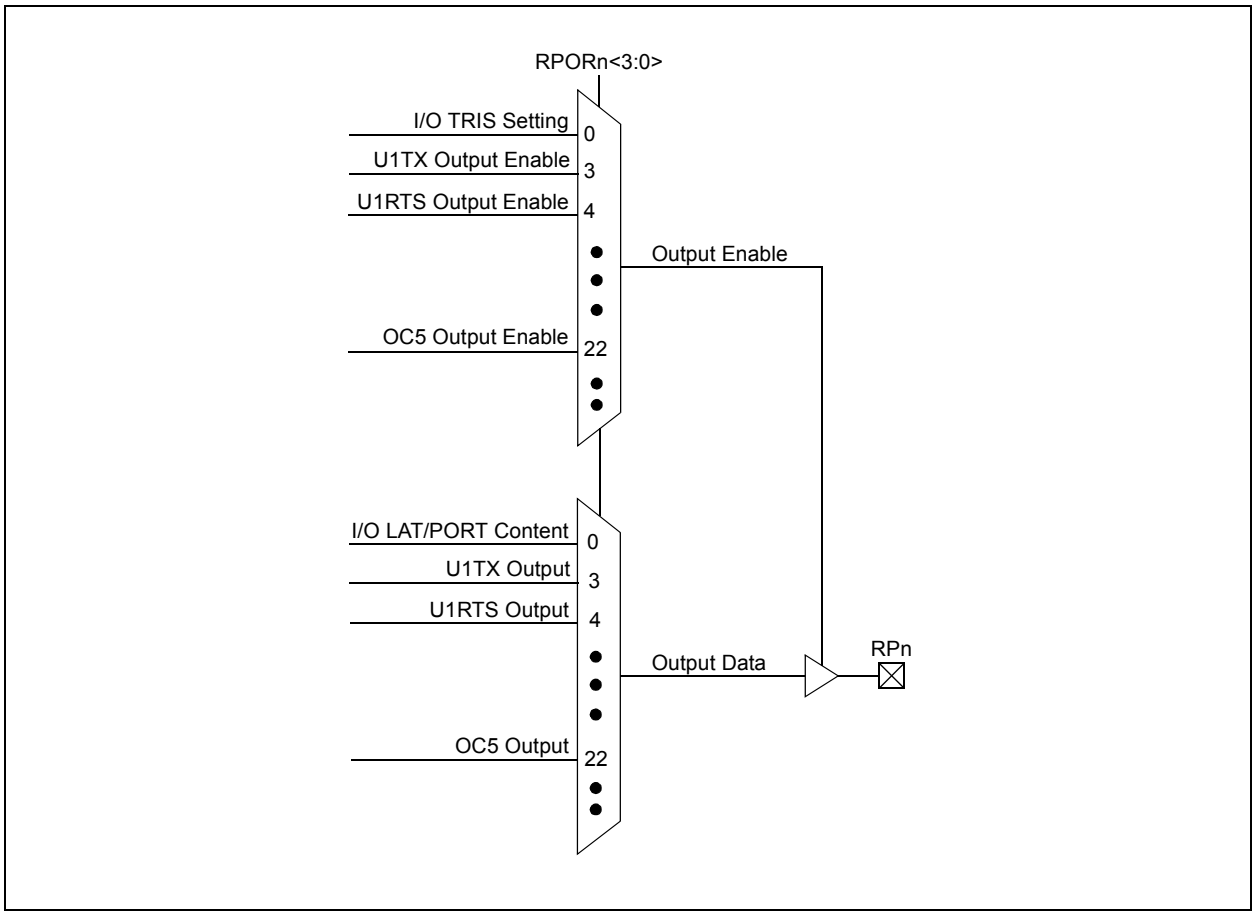
In contrast to the inputs, the outputs of the Peripheral Pin Select options are mapped on the basis of the pin. In this case, a bit field associated with a particular pin dictates the peripheral output to be mapped. The RPORx registers contain sets of 4-bit fields, with each associated with one RPn pin (see Register 11-5). The value of the bit field corresponds to one of the peripherals and that peripheral's output is mapped to the pin. Each pin has a limited set of peripherals to choose from.

The PPS-Lite peripheral outputs and associated RPn pins have been organized into four groups. It is not possible to map a peripheral to an RPn pin which is outside of its group. To map a peripheral output signal to an RPn pin, use the 4-step process, as indicated in Table 11-14. Choose the RPn pin and the signal; the column on the right shows which value to write to the associated RPORx register.

The peripheral outputs that support Peripheral Pin Selection have no default pins. Since the RPORx registers reset to all '0's, the outputs are all disconnected in the device's default (Reset) state.

The list of peripherals for output mapping also includes a null value of b'0000' because of the mapping technique. This allows unused peripherals to not be connected to a pin. Not all peripherals are available on all pins. For example, the "SDO2" signal is only available on RP0, RP4, RP8, etc. The "SDO2" signal is not available on RP1.

FIGURE 11-8: MULTIPLEXING OF REMAPPABLE OUTPUT FOR RPn



REGISTER 11-5: RPORn_n: REMAPPED PERIPHERAL OUTPUT REGISTER n (FUNCTION MAPS TO PIN)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RPORn_3	RPORn_2	RPORn_1	RPORn_0	RPmR_3	RPmR_2	RPmR_1	RPmR_0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-4 **RPORn_<3:0>:** RPn peripheral output function mapping

bit 3-0 **RPmR<3:0>:** RPm peripheral output function mapping

Note 1: Register values can only be changed if IOLOCK = 0.

TABLE 11-14: PPS-LITE OUTPUT

PPS-Lite Output Peripheral Group 4n	
(1) To Map this RPn Pin	(4) to the Associated RPOR Register
RP0	RPOR0_1<3:0>
RP4	RPOR4_5<3:0>
RP8	RPOR8_9<3:0>
RP12	RPOR12_13<3:0>
RP16	RPOR16_17<3:0>
RP20	RPOR20_21<3:0>
RP24	RPOR24_25<3:0>
RP28	RPOR28_29<3:0>
RP32	RPOR32_33<3:0>
RP36	RPOR36_37<3:0>
RP40	RPOR40_41<3:0>
RP44	RPOR44_45<3:0>
(2) with this Output Signal	(3) Write this Corresponding Value
Disabled	h'0
U2BCLK	h'1
U3RX_DT	h'2
U4RX_DT	h'3
SDO2	h'4
P1D	h'5
P2D	h'6
P3B	h'7
CTPLS	h'8
CCP5	h'9
CCP8	h'A
C1OUT	h'B
Unused	h'C
RVP0	h'D
RVP4	h'E
Reserved	h'F

PPS-Lite Output Peripheral Group 4n + 1	
(1) To Map this RPn Pin	(4) to the Associated RPOR Register
RP1	RPOR0_1<7:4>
RP5	RPOR4_5<7:4>
RP9	RPOR8_9<7:4>
RP13	RPOR12_13<7:4>
RP17	RPOR16_17<7:4>
RP21	RPOR20_21<7:4>
RP25	RPOR24_25<7:4>
RP29	RPOR28_29<7:4>
RP33	RPOR32_33<7:4>
RP37	RPOR36_37<7:4>
RP41	RPOR40_41<7:4>
RP45	RPOR44_45<7:4>
(2) with this Output Signal	(3) Write this Corresponding Value
Disabled	h'0
U1BCLK	h'1
U3TX_CK	h'2
U4TX_CK	h'3
SDO1	h'4
P1C	h'5
P2C	h'6
P3C	h'7
CCP7	h'8
CCP9	h'9
C2OUT	h'A
Unused	h'B
Unused	h'C
RVP1	h'D
RVP5	h'E
Reserved	h'F

TABLE 11-14: PPS-LITE OUTPUT (CONTINUED)

PPS-Lite Output Peripheral Group 4n + 2	
(1) To Map this RPn Pin	(4) to the Associated RPOR Register
RP2	RPOR2_3<3:0>
RP6	RPOR6_7<3:0>
RP10	RPOR10_11<3:0>
RP14	RPOR14_15<3:0>
RP18	RPOR18_19<3:0>
RP22	RPOR22_23<3:0>
RP26	RPOR26_27<3:0>
RP30	RPOR30_31<3:0>
RP34	RPOR34_35<3:0>
RP38	RPOR38_39<3:0>
RP42	RPOR42_43<3:0>
RP46	RPOR46<3:0>

(2) with this Output Signal	(3) Write this Corresponding Value
Disabled	h'0
U1TX_CK	h'1
U2RX_DT	h'2
U3BCLK	h'3
U4BCLK	h'4
SCK2	h'5
P1B	h'6
P2B	h'7
ECCP3/P3A	h'8
CCP6	h'9
CCP10	h'A
Unused	h'B
Unused	h'C
RVP2	h'D
RVP6	h'E
Reserved	h'F

PPS-Lite Output Peripheral Group 4n +3	
(1) To Map this RPn Pin	(4) to the Associated RPOR Register
RP3	RPOR2_3<7:4>
RP7	RPOR6_7<7:4>
RP11	RPOR10_11<7:4>
RP15	RPOR14_15<7:4>
RP19	RPOR18_19<7:4>
RP23	RPOR22_23<7:4>
RP27	RPOR26_27<7:4>
RP31	RPOR30_31<7:4>
RP35	RPOR34_35<7:4>
RP39	RPOR38_39<7:4>
RP43	RPOR42_43<7:4>

(2) with this Output Signal	(3) Write this Corresponding Value
Disabled	h'0
U1RX_DT	h'1
U2TX_CK	h'2
SCK1	h'3
ECCP1/P1A	h'4
ECCP2/P2A	h'5
P3D	h'6
MDOUT	h'7
CCP4	h'8
C3OUT	h'9
Unused	h'A
Unused	h'B
Unused	h'C
RVP3	h'D
RVP7	h'E
Reserved	h'F

11.15.3.3 I/O Mapping

While most peripheral signals are defined as either input or output, some peripheral signals switch between input and output: UnRX_DT, UnTX_CK, PBIO and CCP. Most commonly, these signals are mapped so that both the input and output map to the same RPn pin. If desired, the input and output can be mapped to separate pins. For standard peripheral operation, ensure that both the input and output mapping configurations select the same RPn pin.

11.15.3.4 Mapping Limitations

The control schema of Peripheral Select Pins is not limited to a small range of fixed peripheral configurations. There are no mutual or hardware enforced lockouts between any of the peripheral mapping SFRs. While such mappings may be technically possible from a configuration point of view, the user must ensure the selected configurations are supportable from an electrical point of view.

11.15.4 CONTROLLING CONFIGURATION CHANGES

Because peripheral remapping can be changed during run time, some restrictions on peripheral remapping are needed to prevent accidental configuration changes. PIC18FXXJ94 devices include two features to prevent alterations to the peripheral map:

- Continuous state monitoring
- Configuration bit remapping lock

11.15.4.1 Control Register Lock

The contents of RPINRx and RPORx registers are constantly monitored in hardware by shadow registers. If an unexpected change in any of the registers occurs (such as cell disturbances caused by ESD or other external events), a Configuration Mismatch Reset will trigger.

11.15.4.2 Configuration Bit Pin Select Lock

As an additional level of safety, the device can be configured to prevent more than one write session to the RPINRx and RPORx registers. The IOL1WAY Configuration bit (CONFIG5H<0>) blocks the IOLOCK bit from being cleared after it has been set once.

In the default (unprogrammed) state, IOL1WAY is set, restricting users to one write session. Programming IOL1WAY allows users unlimited access to the Peripheral Pin Select registers. It is good programming practice to always set the IOLOCK bit (OSCCON2<6>) after all changes have been made to PPS-Lite registers.

11.15.5 CONSIDERATIONS FOR PERIPHERAL PIN SELECTION

The ability to control Peripheral Pin Selection introduces several considerations into application design that should be considered. This is particularly true for several common peripherals which are only available as remappable peripherals.

Before any other application code is executed, the user must initialize the device with the proper peripheral configuration. Since the IOLOCK is not active in the Reset state, the peripherals can be configured, and the IOLOCK bit can be set when configuration is complete.

Choosing the configuration requires the review of all Peripheral Pin Selects and their pin assignments, especially those that will not be used in the application. In all cases, unused pin-selected peripherals should be disabled. Unused peripherals should have their inputs assigned to Vss. I/O pins with unused RPn functions should be configured with the NULL (‘0’) peripheral output.

The assignment of an RPn pin to the peripheral input or output depends on the peripheral and its use in the application. It is good programming practice to map peripherals to pins immediately after Reset. This should be done before any configuration changes to the peripheral itself.

The assignment of a peripheral output to a particular pin does not automatically perform any other configuration of the pin’s I/O circuitry. This means adding a pin-selectable output to a pin may mean inadvertently driving an existing peripheral input when the output is driven. Users must be familiar with the behavior of other fixed peripherals that share a remappable pin. To be safe, fixed digital peripherals that share the same pin should be disabled when not in use.

Configuring a remappable pin for a specific peripheral input does not automatically turn that feature on. The peripheral must be specifically configured for operation and enabled, as if it were tied to a fixed pin.

A final consideration is that Peripheral Pin Select functions neither override analog inputs, nor reconfigure pins with analog functions for digital I/O. If a pin is configured as an analog input on device Reset, it must be explicitly reconfigured as digital I/O when used with a Peripheral Pin Select.

11.15.5.1 Basic Steps to Use Peripheral Pin Selection Lite (PPS-Lite)

1. Disable any fixed digital peripherals on the pins to be used.
2. Switch pins to be used for digital functionality (if they have analog functionality) using the ANCONx registers.
3. Clear the IOLOCK bit (OSCCON<6>) if needed (not needed after a device Reset).
4. Set RPINRx and RPORx registers appropriately.
5. Set the IOLOCK bit (OSCCON<6>).
6. Enable and configure newly mapped PPS-Lite peripherals.

12.0 DATA SIGNAL MODULATOR

The Data Signal Modulator (DSM) is a peripheral which allows the user to mix a data stream, also known as a modulator signal, with a carrier signal to produce a modulated output.

Both the carrier and the modulator signals are supplied to the DSM module, either internally from the output of a peripheral, or externally through an input pin.

The carrier signal is comprised of two distinct and separate signals: a Carrier High (CARH) signal and a Carrier Low (CARL) signal. During the time in which the Modulator (MOD) signal is in a logic high state, the DSM mixes the Carrier High signal with the Modulator signal. When the Modulator signal is in a logic low state, the DSM mixes the Carrier Low signal with the Modulator signal.

Using this method, the DSM can generate the following types of key modulation schemes:

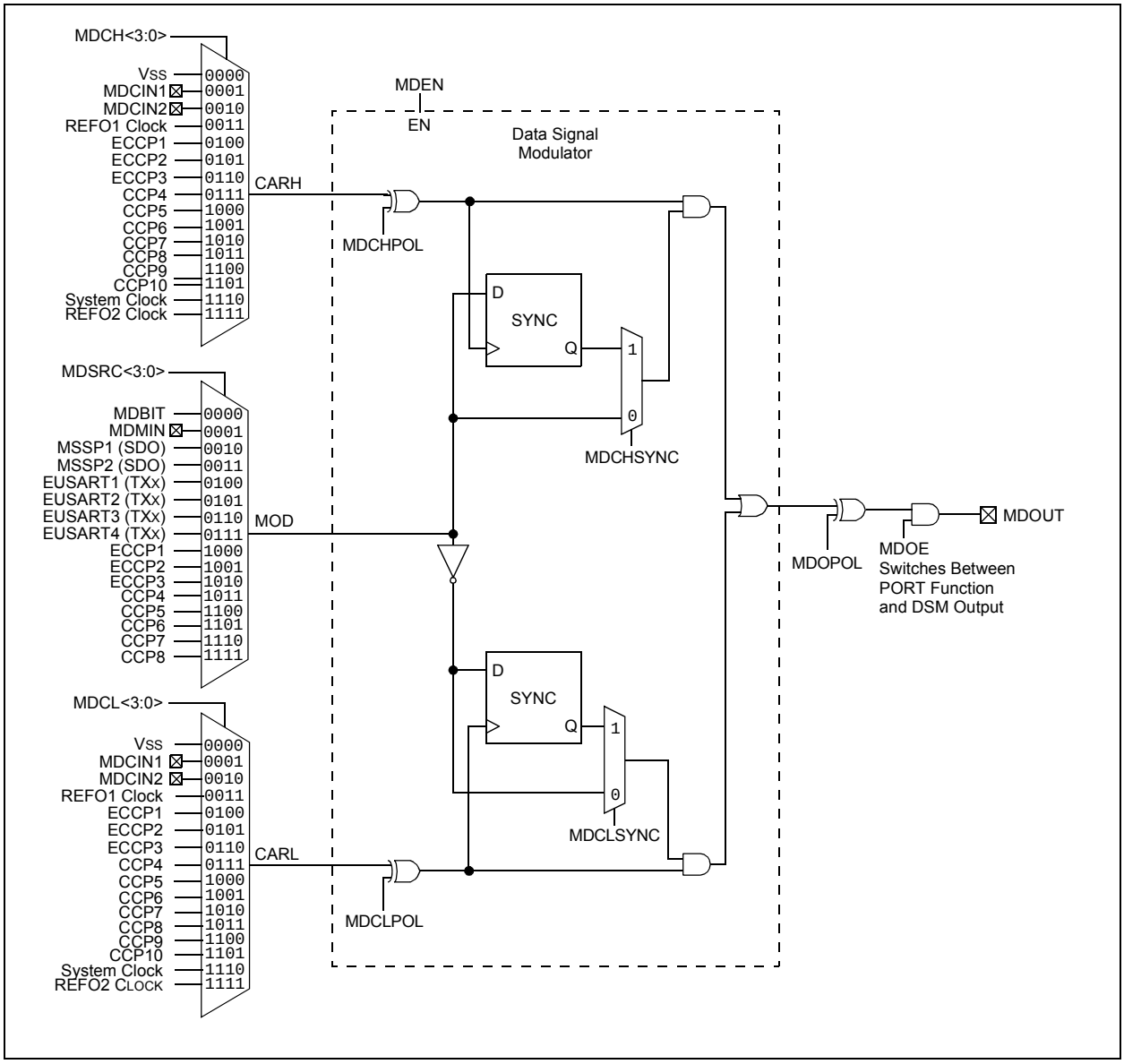
- Frequency-Shift Keying (FSK)
- Phase-Shift Keying (PSK)
- On-Off Keying (OOK)

Additionally, the following features are provided within the DSM module:

- Carrier Synchronization
- Carrier Source Polarity Select
- Carrier Source Pin Disable
- Programmable Modulator Data
- Modulator Source Pin Disable
- Modulator Output Polarity Select
- Slew Rate Control

Figure 12-1 shows a simplified block diagram of the Data Signal Modulator peripheral.

FIGURE 12-1: SIMPLIFIED BLOCK DIAGRAM OF THE DATA SIGNAL MODULATOR



12.1 DSM Operation

The DSM module can be enabled by setting the MDEN bit in the MDCON register. Clearing the MDEN bit in the MDCON register disables the DSM module by automatically switching the Carrier High and Carrier Low signals to the VSS signal source. The Modulator signal source is also switched to the MDBIT in the MDCON register. This not only assures that the DSM module is inactive, but that it is also consuming the least amount of current.

The Modulation Carrier High and Modulation Carrier Low Control registers are not affected when the MDEN bit is cleared, and the DSM module is disabled. The values inside these registers remain unchanged while the DSM is inactive. The sources for the Carrier High, Carrier Low and Modulator signals will once again be selected when the MDEN bit is set, and the DSM module is again enabled and active.

The modulated output signal can be disabled without shutting down the DSM module. The DSM module will remain active and continue to mix signals, but the output value will not be sent to the MDOUT pin. During the time that the output is disabled, the MDOUT pin will remain low. The modulated output can be disabled by clearing the MDOE bit in the MDCON register.

12.2 Modulator Signal Sources

The Modulator signal can be supplied from the following sources:

- ECCP1 Signal
- ECCP2 Signal
- ECCP3 Signal
- CCP2 Signal
- CCP3 Signal
- CCP4 Signal
- CCP5 Signal
- CCP6 Signal
- CCP7 Signal
- CCP8 Signal
- MSSP1 SDO Signal (SPI mode only)
- MSSP2 SDO Signal (SPI mode only)
- EUSART1 TX1 Signal
- EUSART2 TX2 Signal
- EUSART3 TX3 Signal
- EUSART4 TX4 Signal
- External Signal on MDMIN Pin (RF0/MDMIN)
- MDBIT bit in the MDCON Register

The Modulator signal is selected by configuring the MDSRC<3:0> bits in the MDSRC register.

12.3 Carrier Signal Sources

The Carrier High signal and Carrier Low signal can be supplied from the following sources:

- ECCP1 Signal
- ECCP2 Signal
- ECCP3 Signal
- CCP5 Signal
- CCP6 Signal
- CCP7 Signal
- CCP8 Signal
- CCP9 Signal
- CCP10 Signal
- Reference Clock Output Module Signal (REFO1)
- Reference Clock Output Module Signal (REFO2)
- System Clock
- External Signals on the MDCIN1 and MDCIN2 pins are available though PPS. Refer to **Section 11.15 "PPS-Lite"** for setup.
- VSS

The Carrier High signal is selected by configuring the MDCH<3:0> bits in the MDCARH register. The Carrier Low signal is selected by configuring the MDCL<3:0> bits in the MDCARL register.

12.4 Carrier Synchronization

During the time when the DSM switches between Carrier High and Carrier Low signal sources, the carrier data in the modulated output signal can become truncated. To prevent this, the carrier signal can be synchronized to the Modulator signal. When synchronization is enabled, the carrier pulse that is being mixed at the time of the transition is allowed to transition low before the DSM switches over to the next carrier source.

Synchronization is enabled separately for the Carrier High and Carrier Low signal sources. Synchronization for the Carrier High signal can be enabled by setting the MDCHSYNC bit in the MDCARH register. Synchronization for the Carrier Low signal can be enabled by setting the MDCLSYNC bit in the MDCARL register.

Figure 12-1 through Figure 12-6 show timing diagrams using various synchronization methods.

FIGURE 12-2: ON-OFF KEYING (OOK) SYNCHRONIZATION

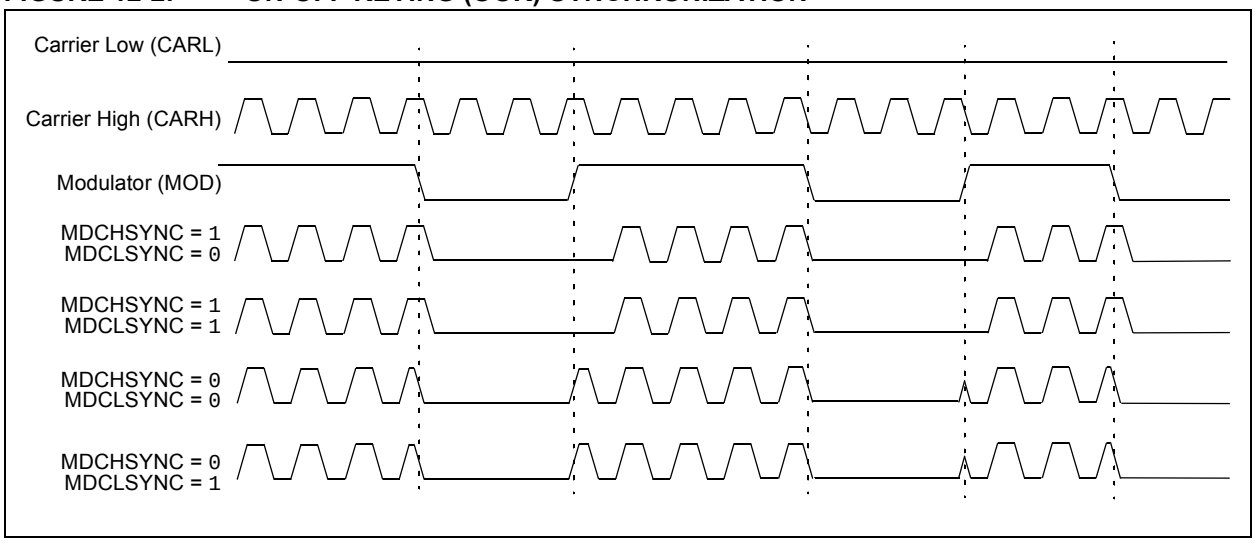


FIGURE 12-3: NO SYNCHRONIZATION (MDCHSYNC = 0, MDCLSYNC = 0)

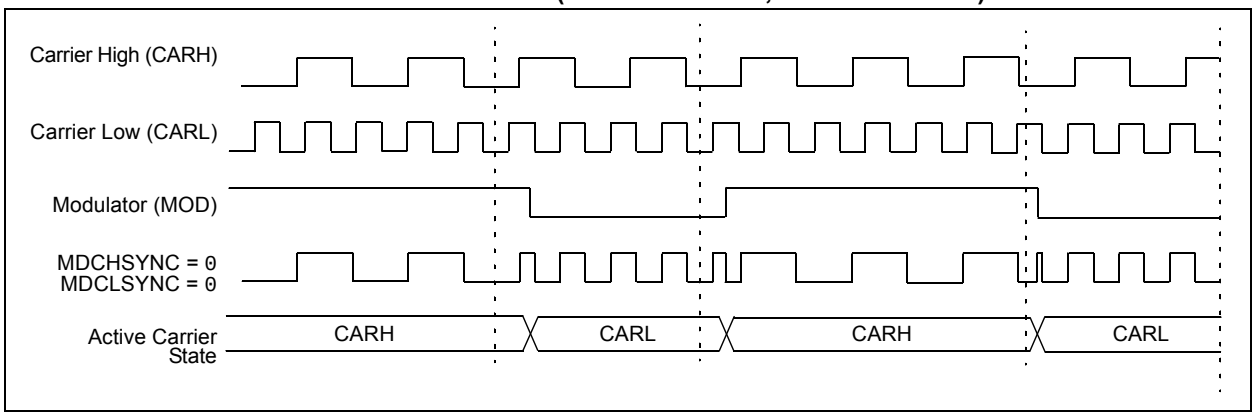


FIGURE 12-4: CARRIER HIGH SYNCHRONIZATION (MDCHSYNC = 1, MDCLSYNC = 0)

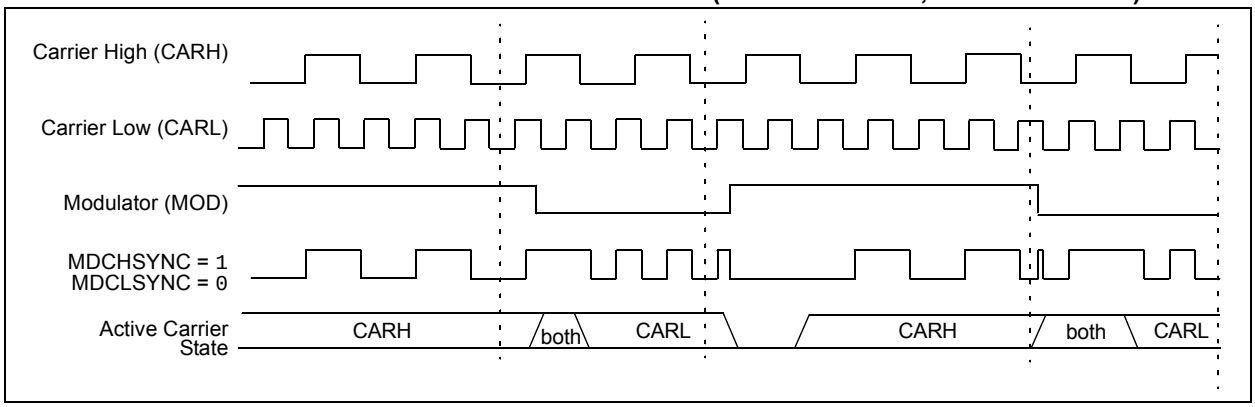


FIGURE 12-5: CARRIER LOW SYNCHRONIZATION (MDCHSYNC = 0, MDCLSYNC = 1)

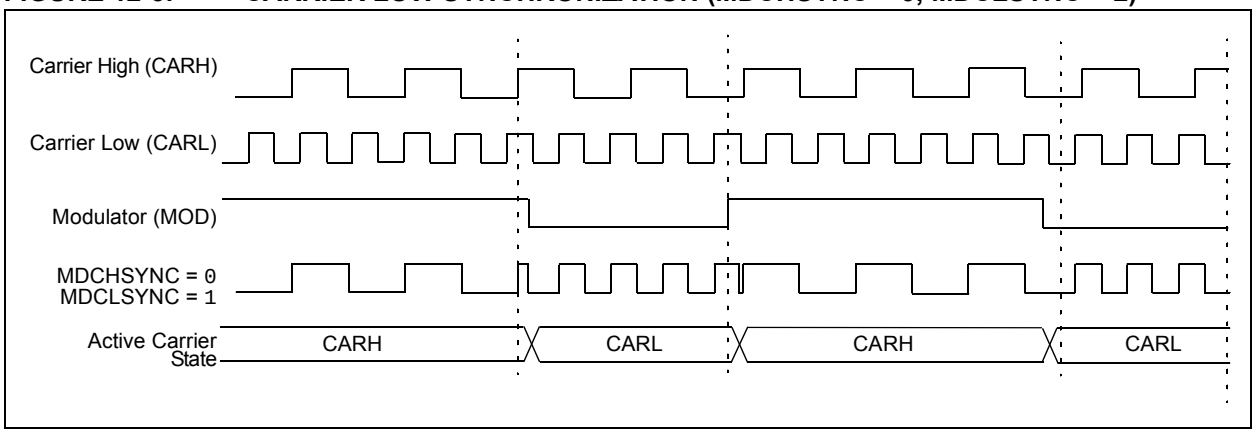
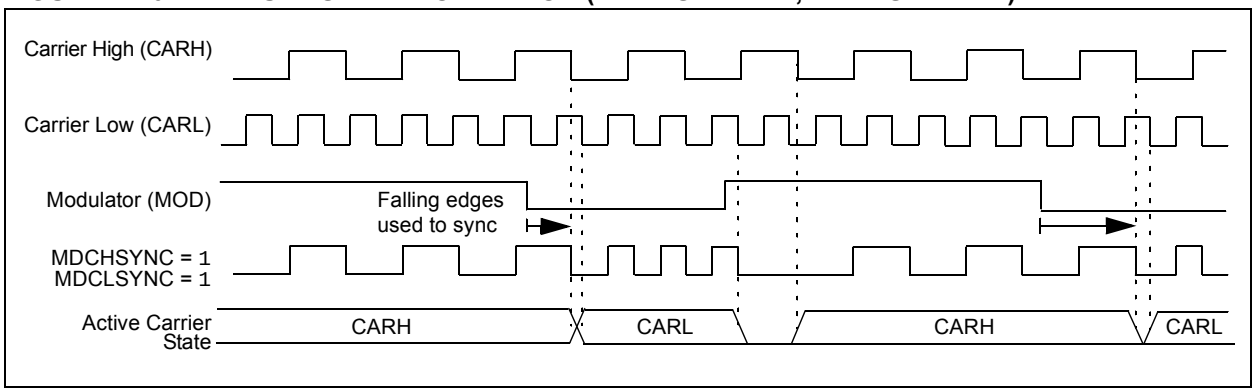


FIGURE 12-6: FULL SYNCHRONIZATION (MDCHSYNC = 1, MDCLSYNC = 1)



12.5 Carrier Source Polarity Select

The signal provided from any selected input source for the Carrier High and Carrier Low signals can be inverted. Inverting the signal for the Carrier High source is enabled by setting the MDCHPOL bit of the MDCARH register. Inverting the signal for the Carrier Low source is enabled by setting the MDCLPOL bit of the MDCARL register.

12.6 Carrier Source Pin Disable

Some peripherals assert control over their corresponding output pin when they are enabled. For example, when the CCP1 module is enabled, the output of CCP1 is connected to the CCP1 pin.

This default connection to a pin can be disabled by setting the MDCHODIS bit in the MDCARH register for the Carrier High source and the MDCLODIS bit in the MDCARL register for the Carrier Low source.

12.7 Programmable Modulator Data

The MDBIT of the MDCON register can be selected as the source for the Modulator signal. This gives the user the ability to program the value used for modulation.

12.8 Modulator Source Pin Disable

The Modulator source default connection to a pin can be disabled by setting the MDSODIS bit in the MDSRC register.

12.9 Modulated Output Polarity

The modulated output signal provided on the MDOUT pin can also be inverted. Inverting the modulated output signal is enabled by setting the MDOPOL bit of the MDCON register.

12.10 Slew Rate Control

When modulated data streams of 20 MHz or greater are required, the slew rate limitation on the output port pin can be disabled. The slew rate limitation can be removed by clearing the MDSLR bit in the MDCON register.

12.11 Operation In Sleep Mode

The DSM module is not affected by Sleep mode. The DSM can still operate during Sleep if the carrier and Modulator input sources are also still operable during Sleep.

12.12 Effects of a Reset

Upon any device Reset, the Modulator data signal module is disabled. The user's firmware is responsible for initializing the module before enabling the output. The registers are reset to their default values.

REGISTER 12-1: MDCON: MODULATION CONTROL REGISTER

R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	U-0	U-0	R/W-0
MDEN	MDOE	MDSLR	MDOPOL	MDOUT(2)	—	—	MDBIT(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **MDEN:** Modulator Module Enable bit
1 = Modulator module is enabled and mixing input signals
0 = Modulator module is disabled and has no output

bit 6 **MDOE:** Modulator Module Pin Output Enable bit
1 = Modulator pin output is enabled
0 = Modulator pin output is disabled

bit 5 **MDSLR:** MDOUT Pin Slew Rate Limiting bit
1 = MDOUT pin slew rate limiting is enabled
0 = MDOUT pin slew rate limiting is disabled

bit 4 **MDOPOL:** Modulator Output Polarity Select bit
1 = Modulator output signal is inverted
0 = Modulator output signal is not inverted

bit 3 **MDOUT:** Modulator Output bit(2)
Displays the current output value of the Modulator module.

bit 2-1 **Unimplemented:** Read as '0'

bit 0 **MDBIT:** Modulator Source Input bit(1)
Allows software to manually set modulation source input to the module.

Note 1: The MDBIT must be selected as the modulation source in the MDCON register for this operation.

2: The modulated output frequency can be greater and asynchronous from the clock that updates this register bit. The bit value may not be valid for higher speed Modulator or carrier signals.

REGISTER 12-2: MDSRC: MODULATION SOURCE CONTROL REGISTER

R/W-x	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
MDSODIS	—	—	—	MDSRC3	MDSRC2	MDSRC1	MDSRC0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **MDSODIS:** Modulation Source Output Disable bit

1 = Output signal driving the peripheral output pin (selected by MDMS<3:0>) is disabled
0 = Output signal driving the peripheral output pin (selected by MDMS<3:0>) is enabled

bit 6-4 **Unimplemented:** Read as '0'

bit 3-0 **MDSRC<3:0>** Modulation Source Selection bits

1111 = CCP8 output (PWM Output mode only)
1110 = CCP7 output (PWM Output mode only)
1101 = CCP6 output (PWM Output mode only)
1100 = CCP5 output (PWM Output mode only)
1011 = CCP4 output (PWM Output mode only)
1010 = ECCP3 output (PWM Output mode only)
1001 = ECCP2 output (PWM Output mode only)
1000 = ECCP1 output (PWM Output mode only)
0111 = EUSART4 TXx output
0110 = EUSART3 TXx output
0101 = EUSART2 TXx output
0100 = EUSART1 TXx output
0011 = MSSP2 SDO signal (SPI mode only)
0010 = MSSP1 SDO signal (SPI mode only)
0001 = MDMIN pin
0000 = MDBIT bit of MDCON register is the modulation source

REGISTER 12-3: MDCARH: MODULATION CARRIER HIGH CONTROL REGISTER

R/W-x	R/W-x	R/W-x	U-0	R/W-x	R/W-x	R/W-x	R/W-x
MDCHODIS	MDCHPOL	MDCHSYNC	—	MDCH3(1)	MDCH2(1)	MDCH1(1)	MDCH0(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **MDCHODIS:** Modulator Carrier High Output Disable bit
- 1 = Output signal driving the peripheral output pin (selected by MDCH<3:0>) is disabled
- 0 = Output signal driving the peripheral output pin (selected by MDCH<3:0>) is enabled

bit 6 **MDCHPOL:** Modulator Carrier High Polarity Select bit
- 1 = Selected Carrier High signal is inverted
- 0 = Selected Carrier High signal is not inverted

bit 5 **MDCHSYNC:** Modulator Carrier High Synchronization Enable bit
- 1 = Modulator waits for a falling edge on the Carrier High time signal before allowing a switch to the Carrier Low time
- 0 = Modulator output is not synchronized to the Carrier High time signal(1)

bit 4 **Unimplemented:** Read as '0'

bit 3-0 **MDCH<3:0>:** Modulator Data Carrier High Selection bits(1)
- 1111 = Reference Clock Output Module 2 (REFO2) signal
- 1110 = System clock
- 1101 = CCP10 output (PWM Output mode only)
- 1100 = CCP9 output (PWM Output mode only)
- 1011 = CCP8 output (PWM Output mode only)
- 1010 = CCP7 output (PWM Output mode only)
- 1001 = CCP6 output (PWM Output mode only)
- 1000 = CCP5 output (PWM Output mode only)
- 0111 = CCP4 output (PWM Output mode only)
- 0110 = ECCP3 output (PWM Output mode only)
- 0101 = ECCP2 output (PWM Output mode only)
- 0100 = ECCP1 output (PWM Output mode only)
- 0011 = Reference Clock Output Module 1 (REFO1) signal
- 0010 = MDCIN2 pin
- 0001 = MDCIN1 pin
- 0000 = No carrier input (tied to ground)

Note 1: Narrowed carrier pulse widths or spurs may occur in the signal stream during transitions.

REGISTER 12-4: MDCARL: MODULATION CARRIER LOW CONTROL REGISTER

R/W-x	R/W-x	R/W-x	U-0	R/W-x	R/W-x	R/W-x	R/W-x
MDCLODIS	MDCLPOL	MDCLSYNC	—	MDCL3(1)	MDCL2(1)	MDCL1(1)	MDCL0(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **MDCLODIS:** Modulator Carrier Low Output Disable bit
1 = Output signal driving the peripheral output pin (selected by MDCL<3:0>) is disabled
0 = Output signal driving the peripheral output pin (selected by MDCL<3:0>) is enabled

bit 6 **MDCLPOL:** Modulator Carrier Low Polarity Select bit
1 = Selected Carrier Low signal is inverted
0 = Selected Carrier Low signal is not inverted

bit 5 **MDCLSYNC:** Modulator Carrier Low Synchronization Enable bit
1 = Modulator waits for a falling edge on the Carrier Low time signal before allowing a switch to the Carrier High time
0 = Modulator output is not synchronized to the Carrier Low time signal(1)

bit 4 **Unimplemented:** Read as '0'

bit 3-0 **MDCL<3:0>:** Modulator Data Carrier Low Selection bits(1)
1111 = Reference Clock Output Module 2 (REFO2) signal
1110 = System clock
1101 = CCP10 output (PWM Output mode only)
1100 = CCP9 output (PWM Output mode only)
1011 = CCP8 output (PWM Output mode only)
1010 = CCP7 output (PWM Output mode only)
1001 = CCP6 output (PWM Output mode only)
1000 = CCP5 output (PWM Output mode only)
0111 = CCP4 output (PWM Output mode only)
0110 = ECCP3 output (PWM Output mode only)
0101 = ECCP2 output (PWM Output mode only)
0100 = ECCP1 output (PWM Output mode only)
0011 = Reference Clock Output Module 1 (REFO1) signal
0010 = MDCIN2 pin
0001 = MDCIN1 pin
0000 = No carrier input (tied to ground)

Note 1: Narrowed carrier pulse widths or spurs may occur in the signal stream during transitions.

13.0 LIQUID CRYSTAL DISPLAY (LCD) CONTROLLER

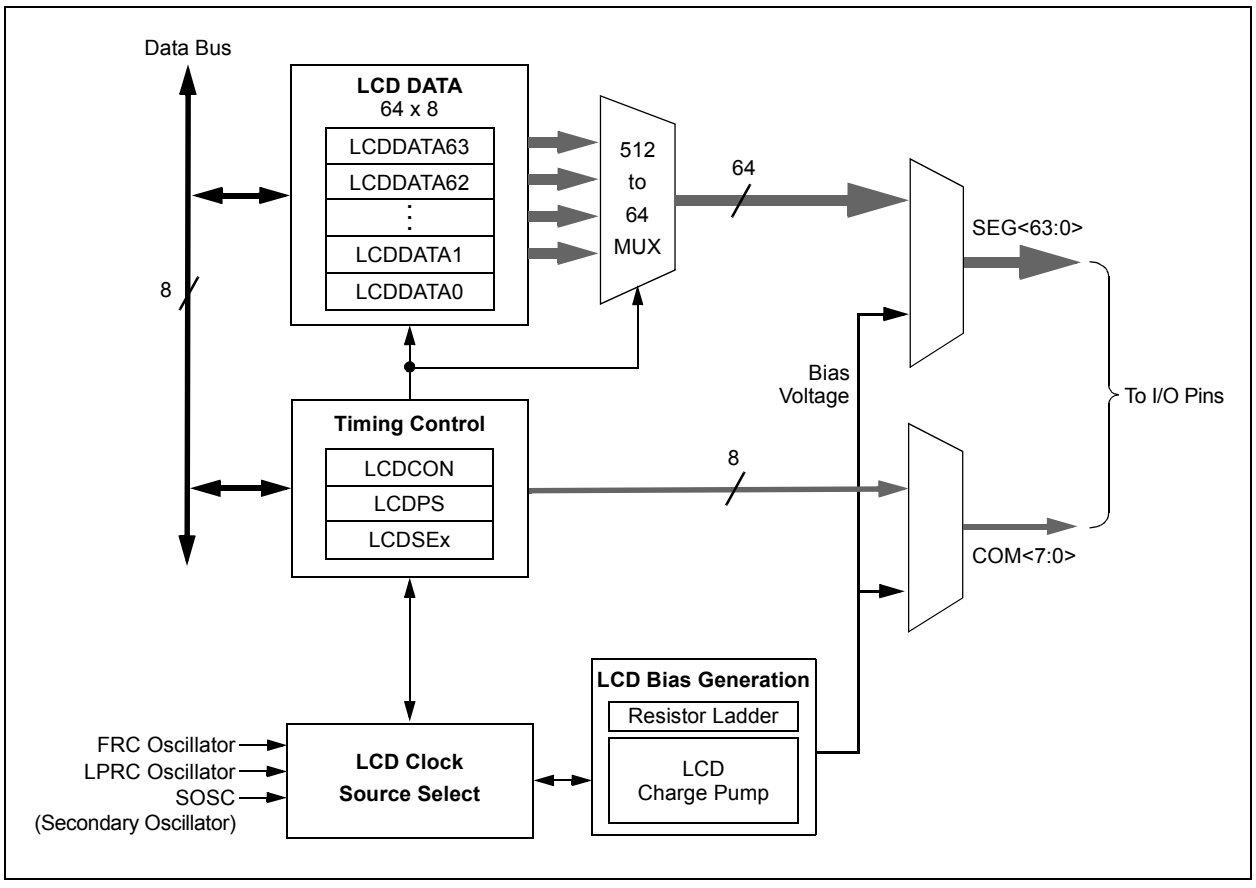
The Liquid Crystal Display (LCD) driver module generates the timing control to drive a static or multiplexed LCD panel. In 100-pin devices (PIC18F97J94), the module drives panels of up to eight commons and up to 60 segments when 5 to 8 commons are used, and up to 64 segments when 1 to 4 commons are used. It also provides control of the LCD pixel data.

The LCD driver module supports:

- Direct driving of LCD panel
- Three LCD clock sources with selectable prescaler
- Up to eight commons:
  - Static (One common)
  - 1/2 multiplex (two commons)
  - 1/3 multiplex (three commons)
  - 1/8 multiplex (eight commons)
- Up to 60 segments (in 100-pin devices when 1/5-1/8 multiplex is selected), 64 (in 100-pin devices when up to 1/4 multiplex is selected), 46 (in 80-pin devices when 1/5-1/8 multiplex is selected), 50 (in 80-pin devices when up to 1/4 multiplex is selected), 30 (in 64-pin devices when 1/5-1/8 multiplex is selected) and 34 (in 64-pin devices when up to 1/4 multiplex is selected)
- Static, 1/2 or 1/3 LCD bias
- On-chip bias generator with dedicated charge pump to support a range of fixed and variable bias options
- Internal resistors for bias voltage generation
- Software contrast control for LCD using the internal biasing

A simplified block diagram of the module is shown in Figure 13-1.

FIGURE 13-1: LCD CONTROLLER MODULE BLOCK DIAGRAM



13.1 LCD Registers

The LCD controller has up to 77 registers:

- LCD Control Register (LCDCON)
- LCD Phase Register (LCDPS)
- LCD Voltage Regulator Control Register (LCDREG)
- LCD Reference Ladder Control Register (LCDREF and LCDRL)
- Eight LCD Segment Enable Registers (LCDSE7:LCDSE0)
- 64 LCD Data Registers (LCDDATA63:LCDDATA0)

The LCDCON register, shown in Register 13-1, controls the overall operation of the module. Once the module is configured, the LCDEN (LCDCON<7>) bit is used to enable or disable the LCD module. The LCD panel can also operate during Sleep by clearing the SLPEN (LCDCON<6>) bit.

The LCDPS register, shown in Register 13-3, configures the LCD clock source prescaler and the type of waveform: Type-A or Type-B. For details on these features, see **Section 13.3 "LCD Clock Source Selection"** and **Section 13.12 "LCD Waveform Generation"**.

REGISTER 13-1: LCDCON: LCD CONTROL REGISTER

R/W-0	R/W-0	R/C-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDEN	SLPEN	WERR	CS1	CS0	LMUX2	LMUX1	LMUX0
bit 7							bit 0

Legend:	C = Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **LCDEN**: LCD Driver Enable bit
1 = LCD driver module is enabled
0 = LCD driver module is disabled

bit 6 **SLPEN**: LCD Driver Enable in Sleep mode bit
1 = LCD driver module is disabled in Sleep mode
0 = LCD driver module is enabled in Sleep mode

bit 5 **WERR**: LCD Write Failed Error bit
1 = LCDDATAx register is written while WA (LCDPS<4>) = 0 (must be cleared in software)
0 = No LCD write error

bit 4-3 **CS<1:0>**: Clock Source Select bits
00 = FRC (8 MHz)/8192
01 = SOSC Oscillator (32.768 kHz)/32
1x = INTRC (31.25 kHz)/32

bit 2-0 **LMUX<2:0>**: Commons Select bits

LMUX<2:0>	Multiplex	Bias
111	1/8 MUX (COM<7:0>)	1/3
110	1/7 MUX (COM<6:0>)	1/3
101	1/6 MUX (COM<5:0>)	1/3
100	1/5 MUX (COM<4:0>)	1/3
011	1/4 MUX (COM<3:0>)	1/3
010	1/3 MUX (COM<2:0>)	1/2 or 1/3
001	1/2 MUX (COM<1:0>)	1/2 or 1/3
000	Static (COM0)	Static

REGISTER 13-2: LCDREG: LCD CHARGE PUMP CONTROL REGISTER

R/W-0	U-0	RW-1	RW-1	RW-1	RW-1	RW-0	RW-0
CPEN	—	BIAS2	BIAS1	BIAS0	MODE13	CLKSEL1	CLKSEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CPEN:** 3.6V Charge Pump Enable bit
1 = The regulator generates the highest (3.6V) voltage
0 = Highest voltage in the system is supplied externally (V_{DD})

bit 6 **Unimplemented:** Read as '0'

bit 5-3 **BIAS<2:0>:** Regulator Voltage Output Control bits
111 =3.60V peak (offset on LCDBIAS0 of 0V)
110 =3.47V peak (offset on LCDBIAS0 of 0.13V)
101 =3.34V peak (offset on LCDBIAS0 of 0.26V)
100 =3.21V peak (offset on LCDBIAS0 of 0.39V)
011 =3.08V peak (offset on LCDBIAS0 of 0.52V)
010 =2.95V peak (offset on LCDBIAS0 of 0.65V)
001 =2.82V peak (offset on LCDBIAS0 of 0.78V)
000 =2.69V peak (offset on LCDBIAS0 of 0.91V)

bit 2 **MODE13:** 1/3 LCD BIAS Enable bit
1 = Regulator output supports 1/3 LCD BIAS mode
0 = Regulator output supports Static LCD BIAS mode

bit 1-0 **CLKSEL<1:0>**: Regulator Clock Select Control bits
11 =LPRC
10 =FRC
01 =SOSC
00 =Disable regulator and float regulator voltage output.

REGISTER 13-3: LCDPS: LCD PHASE REGISTER

R/W-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0
WFT	BIASMD	LCDA	WA	LP3	LP2	LP1	LP0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **WFT**: Waveform Type Select bit
1 = Type-B waveform (phase changes on each frame boundary)
0 = Type-A waveform (phase changes within each common type)

bit 6 **BIASMD**: Bias Mode Select bit
When LMUX<2:0> = 000 or 011 through 111:
0 = Static Bias mode (LMUX<2:0> = 000) / 1/3 Bias mode (LMUX<2:0> = 011 through 111) (do not set this bit to '1')
When LMUX<2:0> = 001 or 010:
1 = 1/2 Bias mode
0 = 1/3 Bias mode

bit 5 **LCDA**: LCD Active Status bit
1 = LCD driver module is active
0 = LCD driver module is inactive

bit 4 **WA**: LCD Write Allow Status bit
1 = Writes into the LCDDATAx registers is allowed
0 = Writes into the LCDDATAx registers is not allowed

bit 3-0 **LP<3:0>**: LCD Prescaler Select bits
1111 = 1:16
1110 = 1:15
1101 = 1:14
1100 = 1:13
1011 = 1:12
1010 = 1:11
1001 = 1:10
1000 = 1:9
0111 = 1:8
0110 = 1:7
0101 = 1:6
0100 = 1:5
0011 = 1:4
0010 = 1:3
0001 = 1:2
0000 = 1:1

13.2 LCD Segment Pins Configuration

The LCDSEx registers configure the functions of the port pins. Setting the segment enable bit for a particular segment configures that pin as an LCD driver. There are four LCD Segment Enable registers, as shown in Table 13-1. The prototype LCDSEx register is shown in Register 13-4.

TABLE 13-1: LCDSEx REGISTERS AND ASSOCIATED SEGMENTS

Register	Segments
LCDSE0	Seg 7:Seg 0
LCDSE1	Seg 15:Seg 8
LCDSE2	Seg 23:Seg 16
LCDSE3	Seg 31:Seg 24
LCDSE4	Seg 39:Seg 32
LCDSE5	Seg 47:Seg 40
LCDSE6	Seg 55:Seg 48
LCDSE7	Seg 63:Seg 56

Once the module is initialized for the LCD panel, the individual bits of the LCDDATAx registers are cleared or set to represent a clear or dark pixel, respectively.

Specific sets of LCDDATA registers are used with specific segments and common signals. Each bit represents a unique combination of a specific segment connected to a specific common.

Individual LCDDATA bits are named by the convention, “SxxCy”, with “xx” as the segment number and “y” as the common number. The relationship is summarized in Register 13-3. The prototype LCDDATAx register is shown in Register 13-5.

REGISTER 13-4: LCDSEx: LCD SEGMENT x ENABLE REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SE(n)	SE(n)	SE(n)	SE(n)	SE(n)	SE(n)	SE(n)	SE(n)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **SE(n)**: Segment Enable bits

For LCDSE0: n = 0-7
For LCDSE1: n = 8-15
For LCDSE2: n = 16-23
For LCDSE3: n = 24-31
For LCDSE0: n = 32-39
For LCDSE0: n = 40-47
For LCDSE0: n = 48-55
For LCDSE0: n = 56-63

1 = Segment function of the pin is enabled, digital I/O is disabled
0 = Segment function of the pin is disabled, digital I/O is enabled

REGISTER 13-5: LCDDATAx: LCD DATA x REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy	S(n)Cy
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **S(n)Cy:** Pixel On bits

For registers LCDDATA0 through LCDDATA7: n = (0-63), y = 0
For registers LCDDATA8 through LCDDATA15: n = (0-63), y = 1
For registers LCDDATA16 through LCDDATA23: n = (0-63), y = 2
For registers LCDDATA24 through LCDDATA31: n = (0-63), y = 3
For registers LCDDATA32 through LCDDATA39: n = (0-63), y = 4
For registers LCDDATA40 through LCDDATA47: n = (0-63), y = 5
For registers LCDDATA48 through LCDDATA55: n = (0-63), y = 6
For registers LCDDATA56 through LCDDATA63: n = (0-63), y = 7

1 = Pixel on
0 = Pixel off

TABLE 13-2: LCDDATA REGISTERS AND BITS FOR SEGMENT AND COM COMBINATIONS

COM Lines	Segments							
	0 to 7	8 to 15	16 to 23	24 to 31	32 to 39	40 to 47	48 to 55	56 to 63
0	LCDDATA0 S00C0:S07C0	LCDDATA1 S08C0:S15C0	LCDDATA2 S16C0:S23C0	LCDDATA3 S24C0:S31C0	LCDDATA4 S32C0:S39C0	LCDDATA5 S40C0:S47C0	LCDDATA6 S48C0:S55C0	LCDDATA7 S56C0:S63C0
1	LCDDATA8 S00C1:S07C1	LCDDATA9 S08C1:S15C1	LCDDATA10 S16C1:S23C1	LCDDATA11 S24C1:S31C1	LCDDATA12 S32C1:S39C1	LCDDATA13 S40C1:S47C1	LCDDATA14 S48C1:S55C1	LCDDATA15 S56C1:S63C1
2	LCDDATA16 S00C2:S07C2	LCDDATA17 S08C2:S15C2	LCDDATA18 S16C2:S23C2	LCDDATA19 S24C2:S31C2	LCDDATA20 S32C2:S39C2	LCDDATA21 40C2:S47C2	LCDDATA22 S48C2:S55C2	LCDDATA23 S56C2:S63C2
3	LCDDATA24 S00C3:S07C3	LCDDATA25 S08C3:S15C3	LCDDATA26 S16C3:S23C3	LCDDATA27 S24C3:S31C3	LCDDATA28 S32C3:S39C3	LCDDATA29 S40C3:S47C3	LCDDATA30 S48C3:S55C3	LCDDATA31 S56C3:S63C3
4	LCDDATA32 S00C4:S07C4	LCDDATA33 S08C4:S15C4	LCDDATA34 S16C4:S23C4	LCDDATA35 S24C4:S31C4	LCDDATA36 S32C4:S39C4	LCDDATA37 S40C4:S47C4	LCDDATA38 S48C4:S55C4	LCDDATA39 S56C4:S63C4
5	LCDDATA40 S00C5:S07C5	LCDDATA41 S08C5:S15C5	LCDDATA42 S16C5:S23C5	LCDDATA43 S24C5:S31C5	LCDDATA44 S32C5:S39C5	LCDDATA45 S40C5:S47C5	LCDDATA46 S48C5:S55C5	LCDDATA47 S56C5:S63C5
6	LCDDATA48 S00C6:S07C6	LCDDATA49 S08C6:S15C6	LCDDATA50 S16C6:S23C6	LCDDATA51 S24C6:S31C6	LCDDATA52 S32C6:S39C6	LCDDATA53 S40C6:S47C6	LCDDATA54 S48C6:S55C6	LCDDATA55 S56C6:S63C6
7	LCDDATA56 S00C7:S07C7	LCDDATA57 S08C7:S15C7	LCDDATA58 S16C7:S23C7	LCDDATA59 S24C7:S31C7	LCDDATA60 S32C7:S39C7	LCDDATA61 S40C7:S47C7	LCDDATA62 S48C7:S55C7	LCDDATA63 S56C7:S63C7

13.3 LCD Clock Source Selection

The LCD driver module has three possible clock sources:

- FRC/8192
- SOSC Clock/32
- LPRC/32

The first clock source is the 8 MHz Fast Internal RC (FRC) Oscillator divided by 8,192. This divider ratio is chosen to provide about 1 kHz output. The divider is not programmable. Instead, the LCD prescaler bits, LCDPS<3:0>, are used to set the LCD frame clock rate.

The second clock source is the SOSC Oscillator/32. This also outputs about 1 kHz when a 32.768 kHz crystal is used with the SOSC Oscillator. To use the SOSC Oscillator as a clock source, set the SOSCEN (T1CON<3>) bit.

The third clock source is a 31.25 kHz internal LPRC Oscillator/32 that provides approximately 1 kHz output.

The second and third clock sources may be used to continue running the LCD while the processor is in Sleep.

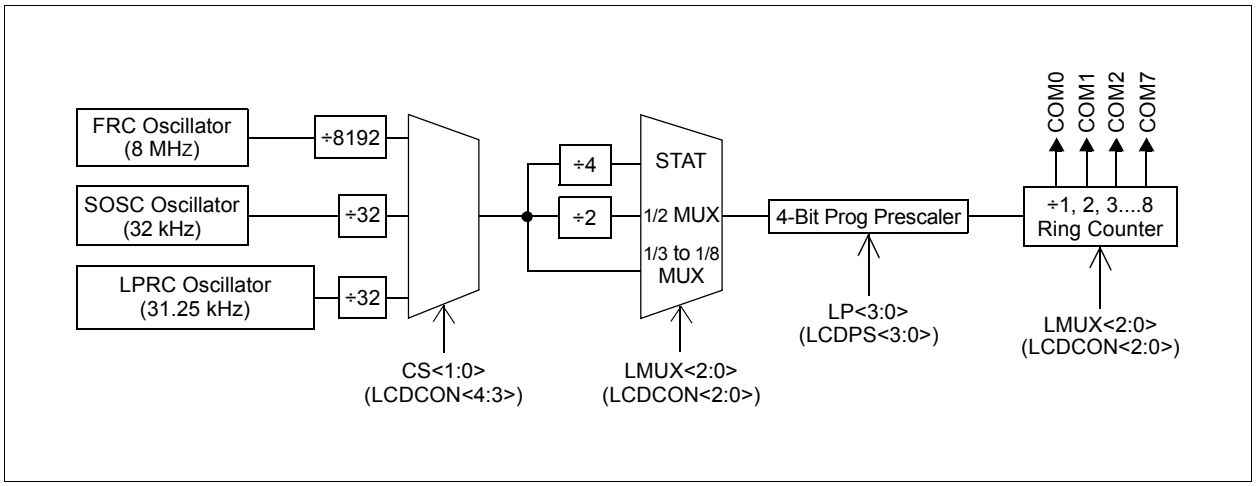
These clock sources are selected through the bits, CS<1:0> (LCDCON<4:3>).

13.3.1 LCD PRESCALER

A 16-bit counter is available as a prescaler for the LCD clock. The prescaler is not directly readable or writable. Its value is set by the LP<3:0> bits (LCDPS<3:0>) that determine the prescaler assignment and prescale ratio.

Selectable prescale values are from 1:1 through 1:16, in increments of one.

FIGURE 13-2: LCD CLOCK GENERATION



13.4 LCD Bias Types

The LCD module can be configured in one of three bias types:

- Static bias (two voltage levels: VSS and VDD)
- 1/2 bias (three voltage levels: VSS, 1/2 VDD and VDD)
- 1/3 bias (four voltage levels: VSS, 1/3 VDD, 2/3 VDD and VDD)

LCD bias voltages can be generated with internal resistor ladders, internal bias generator or external resistor ladder.

13.5 Internal Resistor Biasing

This mode does not use external resistors, but rather internal resistor ladders that are configured to generate the bias voltage.

The internal reference ladder actually consists of three separate ladders. Disabling the internal reference ladder disconnects all of the ladders, allowing external voltages to be supplied.

Depending on the total resistance of the resistor ladders, the biasing can be classified as low, medium or high power.

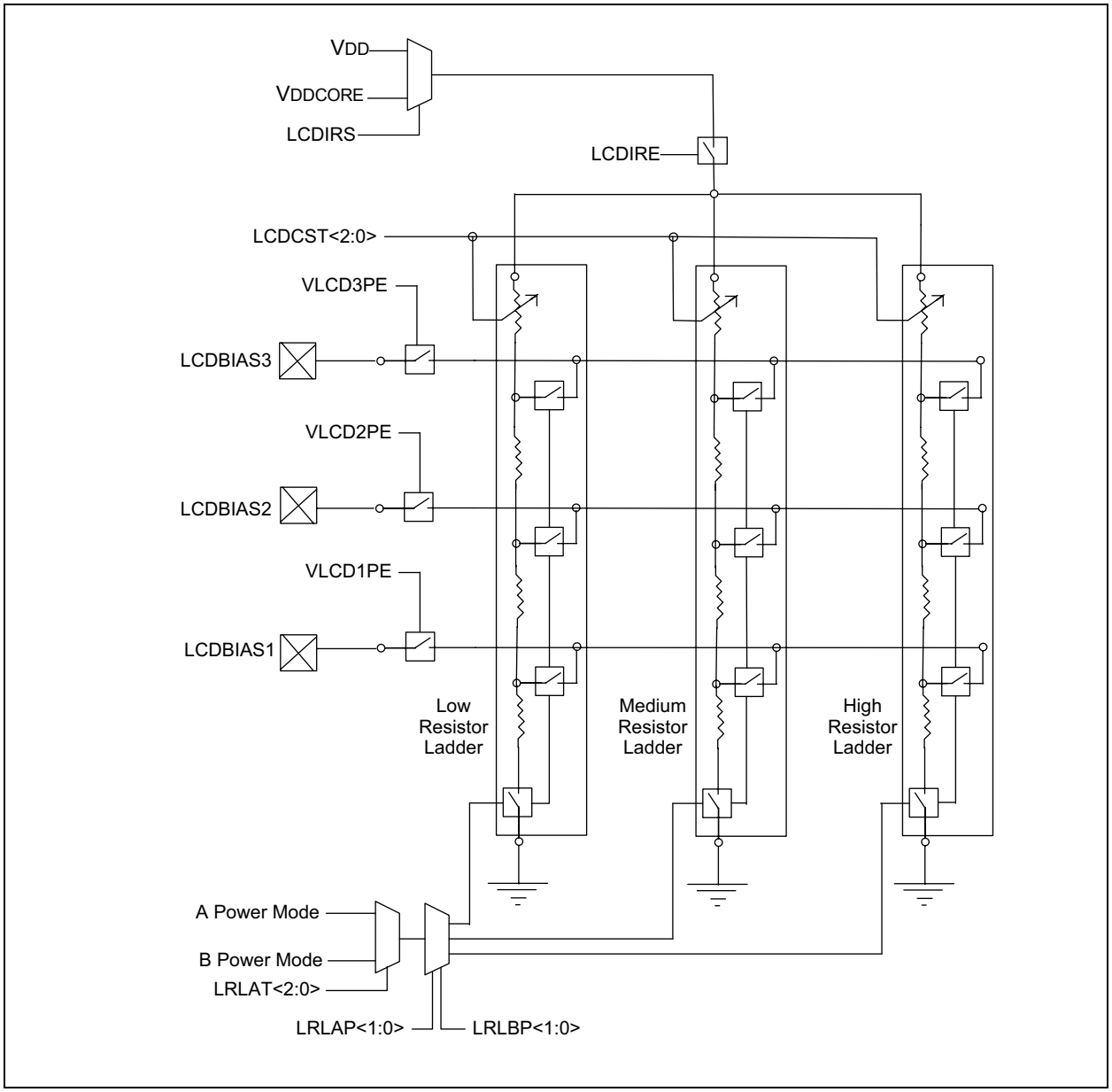
Table 13-3 shows the total resistance of each of the ladders. Table 13-3 shows the internal resister ladder connections. When the internal resistor ladder is selected, the bias voltage can either be from VDD or from VDDCORE, depending on the LCDIRS setting. It can also provide software contrast control (using LCDCST<2:0>)

.

TABLE 13-3: INTERNAL RESISTANCE LADDER POWER MODES

Power Mode	Nominal Resistance of Entire Ladder	IDD
Low	3 MΩ	1 µA
Medium	300 kΩ	10 µA
High	30 kΩ	100 µA

FIGURE 13-3: LCD BIAS INTERNAL RESISTOR LADDER CONNECTION DIAGRAM



There are two power modes, designated as “Mode A” and “Mode B”. Mode A is set by the LRLAP<1:0> bits and Mode B by the LRLB<1:0> bits. The resistor ladder to use for Modes A and B are selected by the bits, LRLAP<1:0> and LRLBP<1:0>, respectively.

Each ladder has a matching contrast control ladder, tuned to the nominal resistance of the reference ladder. This contrast control resistor can be controlled by the LCDCST<2:0> bits (LCDREF<5:3>). Disabling the internal reference ladder results in all of the ladders being disconnected, allowing external voltages to be supplied.

To get additional current in High-Power mode, when LRLAP<1:0> (LCDRL<7:6>) = 11, both the medium and high-power resistor ladders are activated.

Whenever the LCD module is inactive, LCDA (LCDPS<5>) = 0), the reference ladder will be turned off.

13.5.1 AUTOMATIC POWER MODE SWITCHING

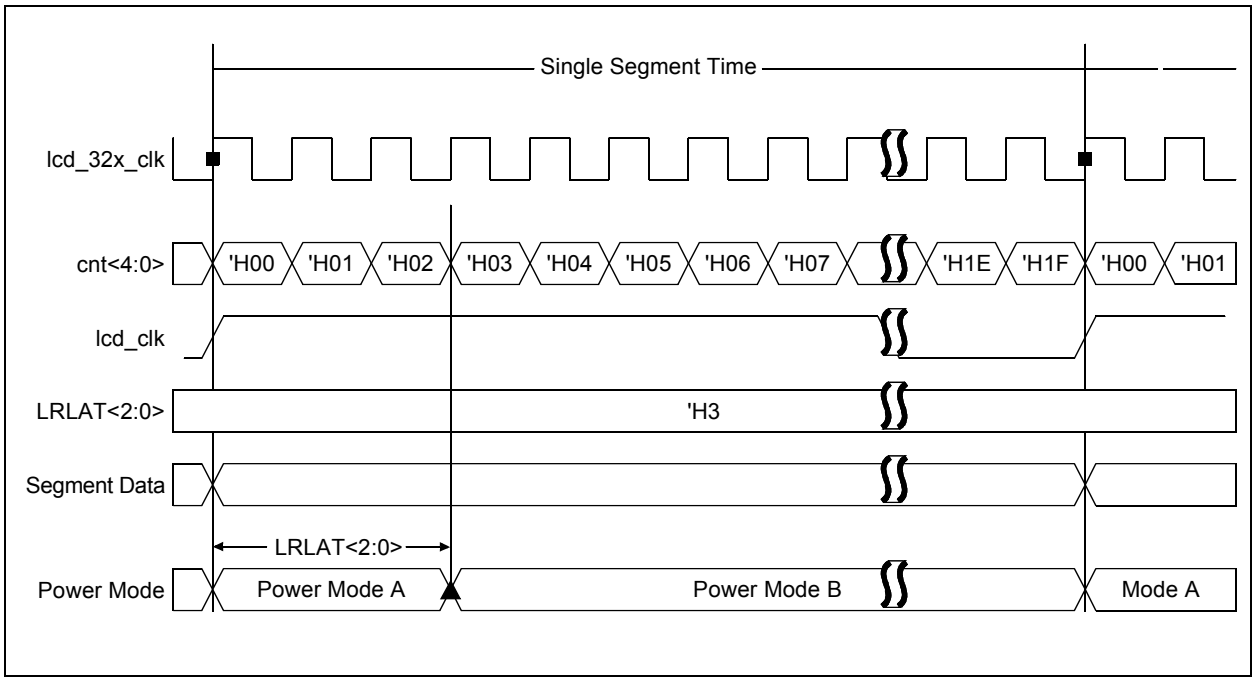
As an LCD segment is electrically only a capacitor, current is drawn only during the interval when the voltage is switching. To minimize total device current, the LCD reference ladder can be operated in a different power mode for the transition portion of the duration. This is controlled by the LCDREF and LCDRL registers.

Mode A Power mode is active for a programmable time, beginning at the time when the LCD segment waveform is transitioning. The LRLAT<2:0> bits (LCDRL<2:0>) select how long or if the Mode A is active. Mode B Power mode is active for the remaining time before the segments or commons change again.

As shown in Figure 13-4, there are 32 counts in a single segment time. Type-A can be chosen during the time when the wave form is in transition. Type-B can be used when the clock is stable or not in transition.

By using this feature of automatic power switching using Type-A/Type-B, the power consumption can be optimized for a given contrast.

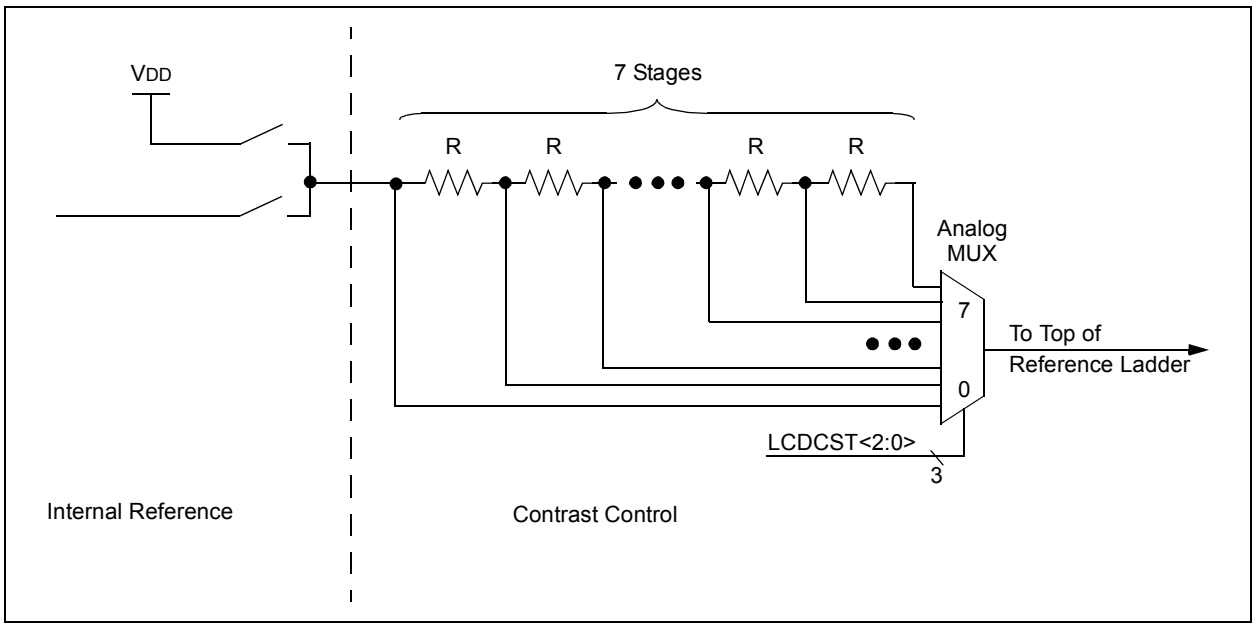
FIGURE 13-4: LCD REFERENCE LADDER POWER MODE SWITCHING DIAGRAM



13.5.2 CONTRAST CONTROL

The LCD contrast control circuit consists of a 7-tap resistor ladder, controlled by the LCDCSTx bits (see Figure 13-5)

FIGURE 13-5: INTERNAL REFERENCE AND CONTRAST CONTROL BLOCK DIAGRAM



13.5.3 INTERNAL REFERENCE

Under firmware control, an internal reference for the LCD bias voltages can be enabled. When enabled, the source of this voltage can be VDD.

When no internal reference is selected, the LCD contrast control circuit is disabled and LCD bias must be provided externally. Whenever the LCD module is inactive (LCDA = 0), the internal reference will be turned off.

13.5.4 VLCDxPE PINS

The VLCD3PE, VLCD2PE and VLCD1PE pins provide the ability for an external LCD bias network to be used instead of the internal ladder. Use of the VLCDxPE pins does not prevent use of the internal ladder.

Each VLCDxPE pin has an independent control in the LCDREF register, allowing access to any or all of the LCD bias signals.

This architecture allows for maximum flexibility in different applications. The VLCDxPE pins could be used to add capacitors to the internal reference ladder for increasing the drive capacity. For applications where the internal contrast control is insufficient, the firmware can choose to enable only the VLCD3PE pin, allowing an external contrast control circuit to use the internal reference divider.

REGISTER 13-6: LCDREF: LCD REFERENCE LADDER CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
LCDIRE	—	LCDCST2	LCDCST1	LCDCST0	VLCD3PE	VLCD2PE	VLCD1PE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **LCDIRE**: LCD Internal Reference Enable bit
1 = Internal LCD reference is enabled and connected to the internal contrast control circuit
0 = Internal LCD reference is disabled

bit 6 **Unimplemented**: Read as '0'

bit 5-3 **LCDCST<2:0>**: LCD Contrast Control bits
Selects the Resistance of the LCD Contrast Control Resistor Ladder:
111 =Resistor ladder is at maximum resistance (minimum contrast)
110 =Resistor ladder is at 6/7th of maximum resistance
101 =Resistor ladder is at 5/7th of maximum resistance
100 =Resistor ladder is at 4/7th of maximum resistance
011 =Resistor ladder is at 3/7th of maximum resistance
010 =Resistor ladder is at 2/7th of maximum resistance
001 =Resistor ladder is at 1/7th of maximum resistance
000 =Minimum resistance (maximum contrast); resistor ladder is shorted

bit 2 **VLCD3PE**: Bias3 Pin Enable bit
1 = BIAS3 level is connected to the external pin, LCDBIAS3
0 = BIAS3 level is internal (internal resistor ladder)

bit 1 **VLCD2PE**: Bias2 Pin Enable bit
1 = BIAS2 level is connected to the external pin, LCDBIAS2
0 = BIAS2 level is internal (internal resistor ladder)

bit 0 **VLCD1PE**: Bias1 Pin Enable bit
1 = BIAS1 level is connected to the external pin, LCDBIAS1
0 = BIAS1 level is internal (internal resistor ladder)

REGISTER 13-7: LCDRL: LCD REFERENCE LADDER CONTROL REGISTER LOW

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
LRLAP1	LRLAP0	LRLBP1	LRLBP0	—	LRLAT2	LRLAT1	LRLAT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **LRLAP<1:0>**: LCD Reference Ladder A Time Power Control bits

During Time Interval A:

11 = Internal LCD reference ladder is powered in High-Power mode
10 = Internal LCD reference ladder is powered in Medium Power mode
01 = Internal LCD reference ladder is powered in Low-Power mode
00 = Internal LCD reference ladder is powered down and unconnected

bit 5-4 **LRLBP<1:0>**: LCD Reference Ladder B Time Power Control bits

During Time Interval B:

11 = Internal LCD reference ladder is powered in High-Power mode
10 = Internal LCD reference ladder is powered in Medium Power mode
01 = Internal LCD reference ladder is powered in Low-Power mode
00 = Internal LCD reference ladder is powered down and unconnected

bit 3 **Unimplemented**: Read as '0'

bit 2-0 **LRLAT<2:0>**: LCD Reference Ladder A Time Interval Control bits

Sets the number of 32 clock counts when the A Time Interval Power mode is active.

For Type-A Waveforms (WFT = 0):

111 = Internal LCD reference ladder is in A Power mode for 7 clocks and B Power mode for 9 clocks
110 = Internal LCD reference ladder is in A Power mode for 6 clocks and B Power mode for 10 clocks
101 = Internal LCD reference ladder is in A Power mode for 5 clocks and B Power mode for 11 clocks
100 = Internal LCD reference ladder is in A Power mode for 4 clocks and B Power mode for 12 clocks
011 = Internal LCD reference ladder is in A Power mode for 3 clocks and B Power mode for 13 clocks
010 = Internal LCD reference ladder is in A Power mode for 2 clocks and B Power mode for 14 clocks
001 = Internal LCD reference ladder is in A Power mode for 1 clock and B Power mode for 15 clocks
000 = Internal LCD reference ladder is always in B Power mode

For Type-B Waveforms (WFT = 1):

111 = Internal LCD reference ladder is in A Power mode for 7 clocks and B Power mode for 25 clocks
110 = Internal LCD reference ladder is in A Power mode for 6 clocks and B Power mode for 26 clocks
101 = Internal LCD reference ladder is in A Power mode for 5 clocks and B Power mode for 27 clocks
100 = Internal LCD reference ladder is in A Power mode for 4 clocks and B Power mode for 28 clocks
011 = Internal LCD reference ladder is in A Power mode for 3 clocks and B Power mode for 29 clocks
010 = Internal LCD reference ladder is in A Power mode for 2 clocks and B Power mode for 30 clocks
001 = Internal LCD reference ladder is in A Power mode for 1 clock and B Power mode for 31 clocks
000 = Internal LCD reference ladder is always in B Power mode

13.5.5 LCD BIAS GENERATION

The LCD driver module is capable of generating the required bias voltages for LCD operation with a minimum of external components. This includes the ability to generate the different voltage levels required by the different bias types that are required by the LCD. The driver module can also provide bias voltages, both above and below microcontroller VDD, through the use of an on-chip LCD voltage regulator.

13.5.6 LCD BIAS TYPES

PIC18F97J94 family devices support three bias types, based on the waveforms generated to control segments and commons:

- Static (two discrete levels)
- 1/2 Bias (three discrete levels)
- 1/3 Bias (four discrete levels)

The use of different waveforms in driving the LCD is discussed in more detail in **Section 13.12 "LCD Waveform Generation"**.

13.5.7 LCD VOLTAGE REGULATOR

The purpose of the LCD regulator is to provide proper bias voltage and good contrast for the LCD, regardless of VDD levels. This module contains a charge pump and internal voltage reference. The regulator can be configured by using external components to boost bias voltage above VDD. It can also operate a display at a constant voltage below VDD. The regulator can also be selectively disabled to allow bias voltages to be generated by an external resistor network.

The LCD regulator is controlled through the LCDREG register. It is enabled or disabled using the CLKSEL<1:0> bits, while the charge pump can be selectively enabled using the CPEN bit. When the regulator is enabled, the MODE13 bit is used to select the bias type. The peak LCD bias voltage, measured as a difference between the potentials of LCDBIAS3 and LCDBIAS0, is configured with the BIAS bits.

REGISTER 13-8: LCDREG: LCD VOLTAGE REGULATOR CONTROL REGISTER

R/W-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0
CPEN	—	BIAS2	BIAS1	BIAS0	MODE13	CLKSEL1	CLKSEL 0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CPEN**: LCD Charge Pump Enable bit
1 = Charge pump is enabled; highest LCD bias voltage is 3.6V
0 = Charge pump is disabled; highest LCD bias voltage is VDD

bit 6 **Unimplemented**: Read as '0'

bit 5-3 **BIAS<2:0>**: Regulator Voltage Output Control bits
111 = 3.60V peak (offset on LCDBIAS0 of 0V)
110 = 3.47V peak (offset on LCDBIAS0 of 0.13V)
101 = 3.34V peak (offset on LCDBIAS0 of 0.26V)
100 = 3.21V peak (offset on LCDBIAS0 of 0.39V)
011 = 3.08V peak (offset on LCDBIAS0 of 0.52V)
010 = 2.95V peak (offset on LCDBIAS0 of 0.65V)
001 = 2.82V peak (offset on LCDBIAS0 of 0.78V)
000 = 2.69V peak (offset on LCDBIAS0 of 0.91V)

bit 2 **MODE13**: 1/3 LCD Bias Enable bit
1 = Regulator output supports 1/3 LCD Bias mode
0 = Regulator output supports Static LCD Bias mode

bit 1-0 **CLKSEL<1:0>**: Regulator Clock Source Select bits
11 = 31 kHz LPRC
10 = 8 MHz FRC
01 = SOSC
00 = LCD regulator disabled

13.6 BIAS CONFIGURATIONS

PIC18F97J94 family devices have four distinct circuit configurations for LCD bias generation:

- M0: Regulator with Boost
- M1: Regulator without Boost
- M2: Resistor Ladder with Software Contrast
- M3: Resistor Ladder with Hardware Contrast

13.6.1 M0 (REGULATOR WITH BOOST)

In M0 operation, the LCD charge pump feature is enabled. This allows the regulator to generate voltages up to +3.6V to the LCD (as measured at LCDBIAS3).

M0 uses a flyback capacitor connected between VLCAP1 and VLCAP2, as well as filter capacitors on LCDBIAS0 through LCDBIAS3, to obtain the required voltage boost (Figure 13-6). The output voltage (VBIAS) is the difference of the potential between LCDBIAS3 and LCDBIAS0. It is set by the BIAS<2:0> bits which adjust the offset between LCDBIAS0 and VSS. The flyback capacitor (CFLY) acts as a charge storage element for large LCD loads. This mode is useful in those cases where the voltage requirements of the LCD are higher than the microcontroller's VDD. It also permits software control of the display's contrast, by adjustment of bias voltage, by changing the value of the BIAS bits.

M0 supports static and 1/3 bias types. Generation of the voltage levels for 1/3 bias is handled automatically, but must be configured in software.

M0 is enabled by selecting a valid regulator clock source (CLKSEL<1:0> set to any value except '00') and setting the CPEN bit. If static bias type is required, the MODE13 bit must be cleared.

13.6.2 M1 (REGULATOR WITHOUT BOOST)

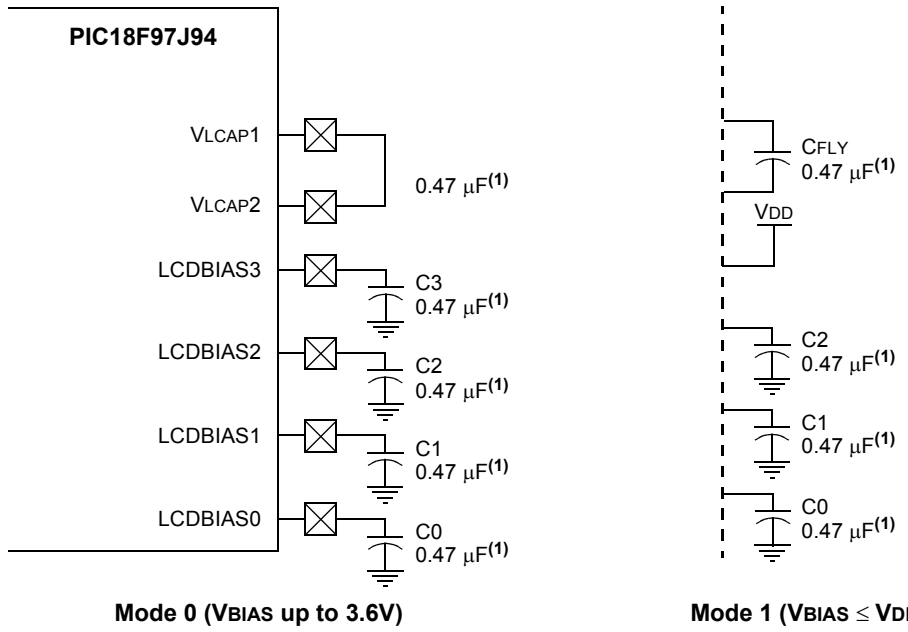
M1 operation is similar to M0, but does not use the LCD charge pump. It can provide VBIAS up to the voltage level supplied directly to LCDBIAS3. It can be used in cases where VDD for the application is expected to never drop below a level that can provide adequate contrast for the LCD. The connection of external components is very similar to M0, except that LCDBIAS3 must be tied directly to VDD (Figure 13-6).

Note:	When the device is put to Sleep while operating in mode M0 or M1, make sure that the bias capacitors are fully discharged to get the lowest Sleep current.

The BIAS<2:0> bits can still be used to adjust contrast in software by changing the VBIAS. As with M0, changing these bits changes the offset between LCDBIAS0 and VSS. In M1, this is reflected in the change between the LCDBIAS0 and the voltage tied to LCDBIAS3. Thus, if VDD should change, VBIAS will also change; where in M0, the level of VBIAS is constant.

Like M0, M1 supports static and 1/3 bias types. Generation of the voltage levels for 1/3 bias is handled automatically but must be configured in software. M1 is enabled by selecting a valid regulator clock source (CLKSEL<1:0> set to any value except '00') and clearing the CPEN bit. If 1/3 bias type is required, the MODE13 bit should also be set.

FIGURE 13-6: LCD REGULATOR CONNECTIONS FOR M0 AND M1 CONFIGURATIONS



Mode 0 (V_{BIAS} up to 3.6V) **Mode 1 ($V_{BIAS} \leq V_{DD}$)**

Note 1: These values are provided for design guidance only; they should be optimized for the application by the designer based on the actual LCD specifications.

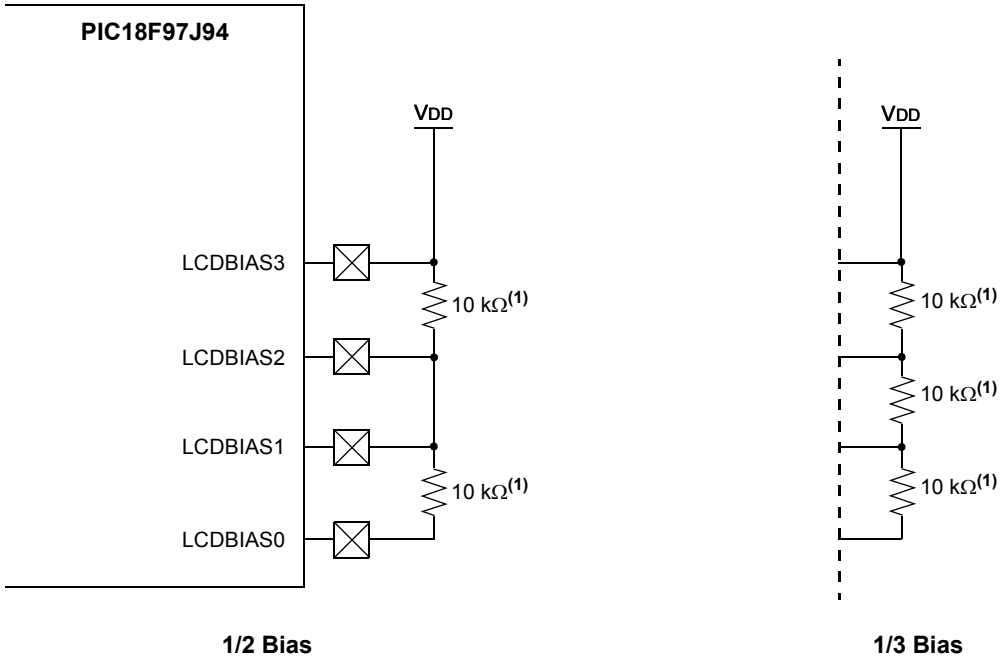
13.6.3 M2 (EXTERNAL RESISTOR LADDER WITH SOFTWARE CONTRAST)

M2 operation also uses the LCD regulator but disables the charge pump. The regulator's internal voltage reference remains active as a way to regulate contrast. It is used in cases where the current requirements of the LCD exceed the capacity of the regulator's charge pump.

In this configuration, the LCD bias voltage levels are created by an external resistor voltage divider, connected across LCDBIAS0 through LCDBIAS3, with the top of the divider tied to VDD (Figure 13-7). The potential at the bottom of the ladder is determined by the LCD regulator's voltage reference, tied internally to LCDBIAS0. The bias type is determined by the voltages on the LCDBIAS pins, which are controlled by the configuration of the resistor ladder. Most applications, using M2, will use a 1/3 or 1/2 bias type. While static bias can also be used, it offers extremely limited contrast range and additional current consumption over other bias generation modes.

Like M1, the LCDBIAS bits can be used to control contrast, limited by the level of VDD supplied to the device. Also, since there is no capacitor required across VLCAP1 and VLCAP2, these pins are available as digital I/O ports, RG2 and RG3. M2 is selected by clearing the CLKSEL<1:0> bits and setting the CPEN bit.

FIGURE 13-7: RESISTOR LADDER CONNECTIONS FOR M2 CONFIGURATION



Bias Level at Pin	Bias Type	
	1/2 Bias	1/3 Bias
LCDBIAS0	(Internal Low Reference Voltage)	(Internal Low Reference Voltage)
LCDBIAS1	1/2 VBIAS	1/3 VBIAS
LCDBIAS2	1/2 VBIAS	2/3 VBIAS
LCDBIAS3	VBIAS (up to VDD)	VBIAS (up to VDD)

Note 1: These values are provided for design guidance only; they should be optimized for the application by the designer based on the actual LCD specifications.

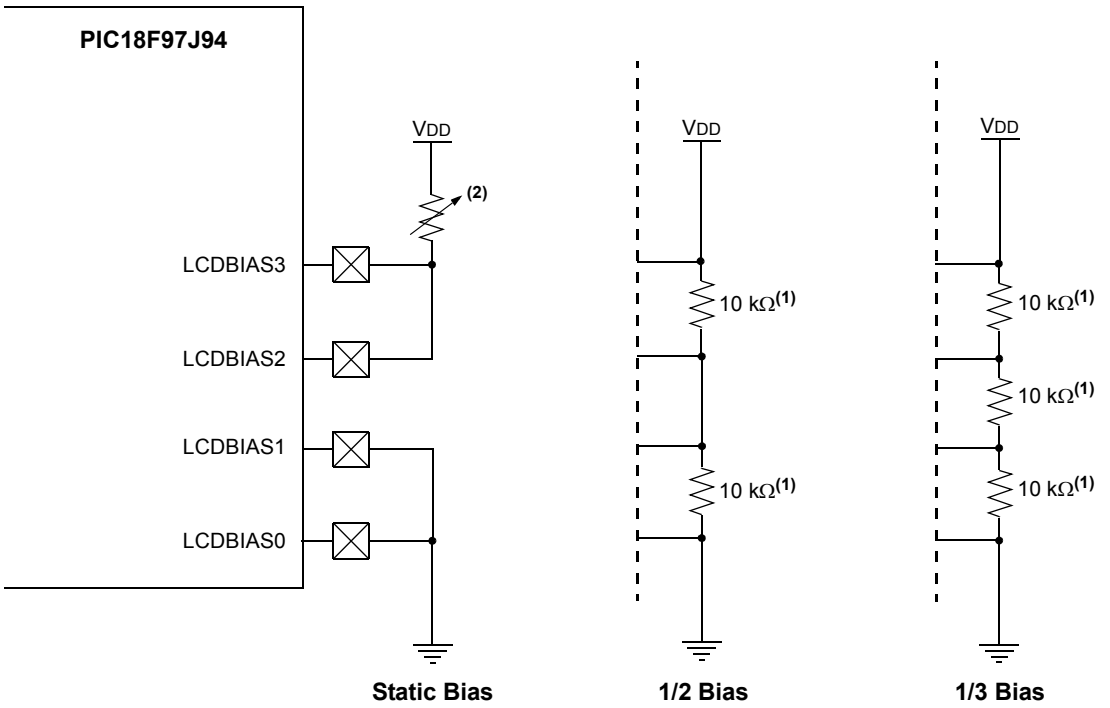
13.6.4 M3 (HARDWARE CONTRAST)

In M3, the LCD regulator is completely disabled. Like M2, LCD bias levels are tied to VDD and are generated using an external divider. The difference is that the internal voltage reference is also disabled and the bottom of the ladder is tied to ground (VSS); see Figure 13-8. The value of the resistors, and the difference between VSS and VDD, determine the contrast range; no software adjustment is possible. This configuration is also used where the LCD's current requirements exceed the capacity of the charge pump and software contrast control is not needed.

Depending on the bias type required, resistors are connected between some or all of the pins. A potentiometer can also be connected between LCDBIAS3 and VDD to allow for hardware controlled contrast adjustment.

M3 is selected by clearing the CLKSEL<1:0> and CPEN bits.

FIGURE 13-8: RESISTOR LADDER CONNECTIONS FOR M3 CONFIGURATION



Bias Level at Pin	Bias Type		
	Static	1/2 Bias	1/3 Bias
LCDBIAS0	AVSS	AVSS	AVSS
LCDBIAS1	AVSS	1/2 VDD	1/3 VDD
LCDBIAS2	VDD	1/2 VDD	2/3 VDD
LCDBIAS3	VDD	VDD	VDD

Note 1: These values are provided for design guidance only; they should be optimized for the application by the designer based on the actual LCD specifications.

2: A potentiometer for manual contrast adjustment is optional; it may be omitted entirely.

13.7 Design Considerations for the LCD Charge Pump

When designing applications that use the LCD regulator with the charge pump enabled, users must always consider both the dynamic current and RMS (static) current requirements of the display, and what the charge pump can deliver. Both dynamic and static current can be determined by Equation 13-1:

EQUATION 13-1: LCD STATIC, DYNAMIC CURRENT

$$I = C \times \frac{dV}{dt}$$

For dynamic current, C, is the value of the capacitors attached to LCDBIAS3 and LCDBIAS2. The variable, dV, is the voltage drop allowed on C2 and C3 during a voltage switch on the LCD display, and dt is the duration of the transient current after a clock pulse occurs.

For practical design purposes, it will be assumed to be 0.047 µF for C, 0.1V for dV and 1 µs for dt. This yields a dynamic current of 4.7 mA for 1 µs.

RMS current is determined by the value of C_{FLY} for C, the voltage across V_{LCAP1} and V_{LCAP2} for dV and the regulator clock period (TPER) for dt. Assuming a C_{FLY} value of 0.047 µF, a value of 1.02V across C_{FLY} and TPER of 30 µs, the maximum theoretical static current will be 1.8 mA. Since the charge pump must charge five capacitors, the maximum current becomes 360 µA.

For a real-world assumption of 50% efficiency, this yields a practical current of 180 µA. Users should compare the calculated current capacity against the requirements of the LCD. While dV and dt are relatively fixed by device design, the values of C_{FLY} and the capacitors on the LCDBIAS pins can be changed to increase or decrease current. As always, any changes should be evaluated in the actual circuit for their impact on the application.

13.8 LCD Multiplex Types

The LCD driver module can be configured into four multiplex types:

- Static (only COM0 used)
- 1/2 multiplex (COM0 and COM1 are used)
- 1/3 multiplex (COM0, COM1 and COM2 are used)
- 1/4 multiplex (COM0, COM1, COM2 and COM3 are used)
- 1/5 multiplex (COM0, COM1, COM2, COM3 and COM4 are used)
- 1/6 multiplex (COM0, COM1, COM2, COM3, COM4 and COM5 are used)
- 1/7 multiplex (COM0, COM1, COM2, COM3, COM4, COM5 and COM6 are used)
- 1/8 multiplex (COM0, COM1, COM2, COM3, COM4, COM5, COM6 and COM7 are used)

The LMUX<2:0> setting (LCDCON<2:0>) decides the function of the COM pins. (For details, see Table 13-4).

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. If the pin is a COM drive, the TRIS setting of that pin is overridden.

Note: On a Power-on Reset, the LMUX<2:0> bits are '000'.

TABLE 13-4: COM<7:0> PIN FUNCTIONS

LMUX<2:0>	COM7 Pin	COM6 Pin	COM5 Pin	COM4 Pin	COM3 Pin	COM2 Pin	COM1 Pin	COM0 Pin
111	COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0
110	I/O Pin	COM6	COM5	COM4	COM3	COM2	COM1	COM0
101	I/O Pin	I/O Pin	COM5	COM4	COM3	COM2	COM1	COM0
100	I/O Pin	I/O Pin	I/O Pin	COM4	COM3	COM2	COM1	COM0
011	I/O Pin	I/O Pin	I/O Pin	I/O Pin	COM3	COM2	COM1	COM0
010	I/O Pin	I/O Pin	I/O Pin	I/O Pin	I/O Pin	COM2	COM1	COM0
001	I/O Pin	I/O Pin	I/O Pin	I/O Pin	I/O Pin	I/O Pin	COM1	COM0
000	I/O Pin	I/O Pin	I/O Pin	I/O Pin	I/O Pin	I/O Pin	I/O Pin	COM0

Note: Pins, COM<7:4>, can also be used as SEG pins when ¼ multiplex to static multiplex are used. These pins can be used as I/O pins only if respective bits in the LCDSEx registers are set to '0'.

13.9 Segment Enables

The LCDSEx registers are used to select the pin function for each segment pin. The selection allows each pin to operate as either an LCD segment driver or a digital only pin. To configure the pin as a segment pin, the corresponding bits in the LCDSEx registers must be set to '1'.

If the pin is a digital I/O, the corresponding TRIS bit controls the data direction. Any bit set in the LCDSEx registers overrides any bit settings in the corresponding TRIS register.

Note: On a Power-on Reset, these pins are configured as digital I/O.

13.10 Pixel Control

The LCDDATAx registers contain bits that define the state of each pixel. Each bit defines one unique pixel. Table 13-2 shows the correlation of each bit in the LCDDATAx registers to the respective common and segment signals.

Any LCD pixel location not being used for display can be used as general purpose RAM.

13.11 LCD Frame Frequency

The rate at which the COM and SEG outputs change is called the LCD frame frequency.

TABLE 13-5: FRAME FREQUENCY FORMULAS

Multiplex	Frame Frequency =
Static (‘000’)	Clock Source/(4 x 1 x (LP<3:0> + 1))
1/2 (‘001’)	Clock Source/(2 x 2 x (LP<3:0> + 1))
1/3 (‘010’)	Clock Source/(1 x 3 x (LP<3:0> + 1))
1/4 (‘011’)	Clock Source/(1 x 4 x (LP<3:0> + 1))
1/5 (‘100’)	Clock Source/(1 x 5 x (LP<3:0> + 1))
1/6 (‘101’)	Clock Source/(1 x 6 x (LP<3:0> + 1))
1/7 (‘110’)	Clock Source/(1 x 7 x (LP<3:0> + 1))
1/8 (‘111’)	Clock Source/(1 x 8 x (LP<3:0> + 1))

Note: The clock source is FRC/8192, SOSC/32 or LPRC/32.

13.12 LCD Waveform Generation

LCD waveform generation is based on the philosophy that the net AC voltage across the dark pixel should be maximized and the net AC voltage across the clear pixel should be minimized. The net DC voltage across any pixel should be zero.

The COM signal represents the time slice for each common, while the SEG contains the pixel data.

The pixel signal (COM-SEG) will have no DC component and can take only one of the two rms values. The higher rms value will create a dark pixel and a lower rms value will create a clear pixel.

As the number of commons increases, the delta between the two rms values decreases. The delta represents the maximum contrast that the display can have.

The LCDs can be driven by two types of waveforms: Type-A and Type-B. In a Type-A waveform, the phase changes within each common type, whereas a Type-B waveform’s phase changes on each frame boundary. Thus, Type-A waveforms maintain 0 VDC over a single frame, whereas Type-B waveforms take two frames.

Note:	If Sleep has to be executed with LCD Sleep enabled (SLPEN (LCDCON<6>) = 1), care must be taken to execute Sleep only when VDC on all the pixels is ‘0’.

Figure 13-9 through Figure 13-21 provide waveforms for static, half-multiplex, one-third multiplex and quarter multiplex drives for Type-A and Type-B waveforms.

FIGURE 13-9: TYPE-A/TYPE-B WAVEFORMS IN STATIC DRIVE

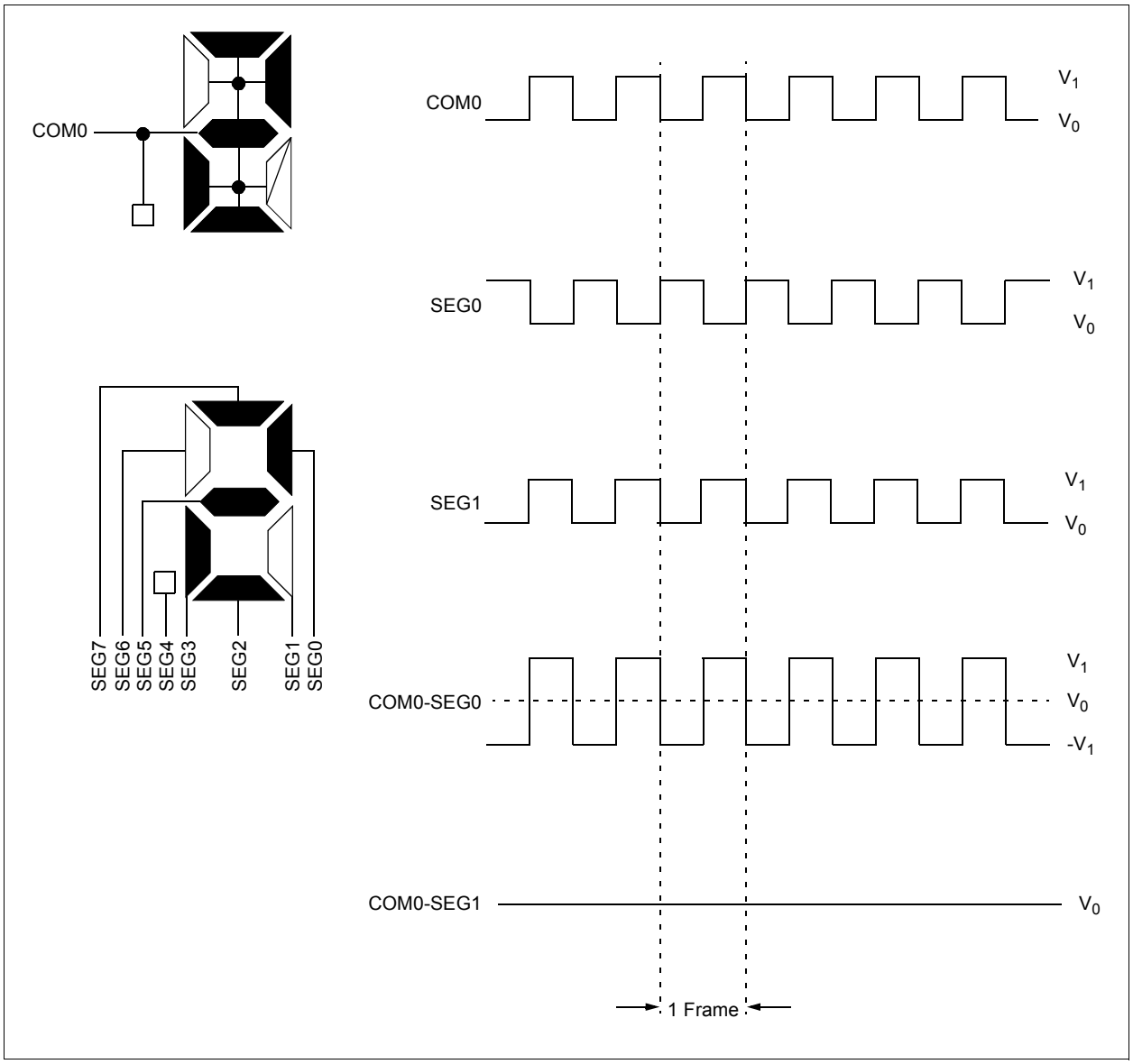


FIGURE 13-10: TYPE-A WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE

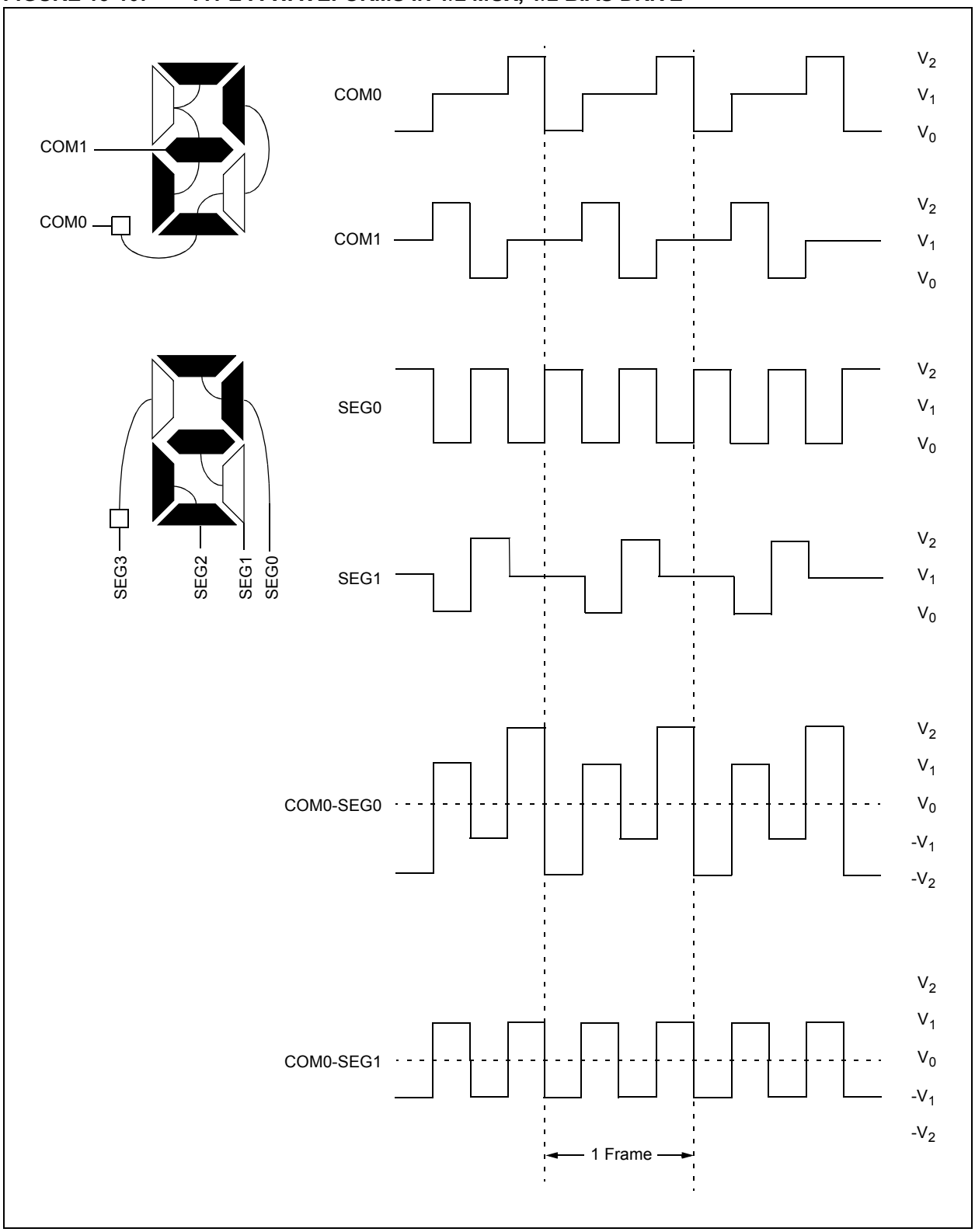


FIGURE 13-11: TYPE-B WAVEFORMS IN 1/2 MUX, 1/2 BIAS DRIVE

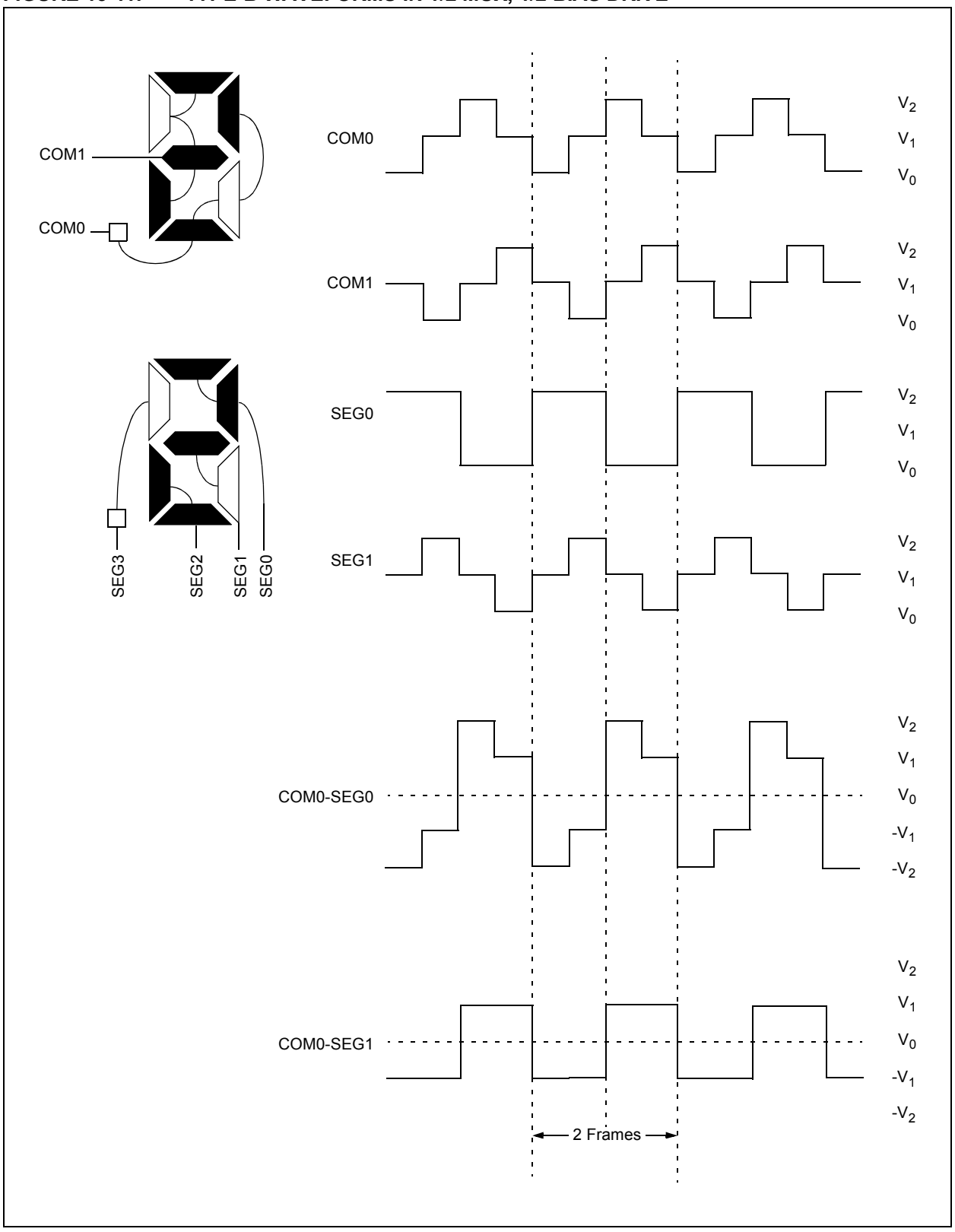


FIGURE 13-12: TYPE-A WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE

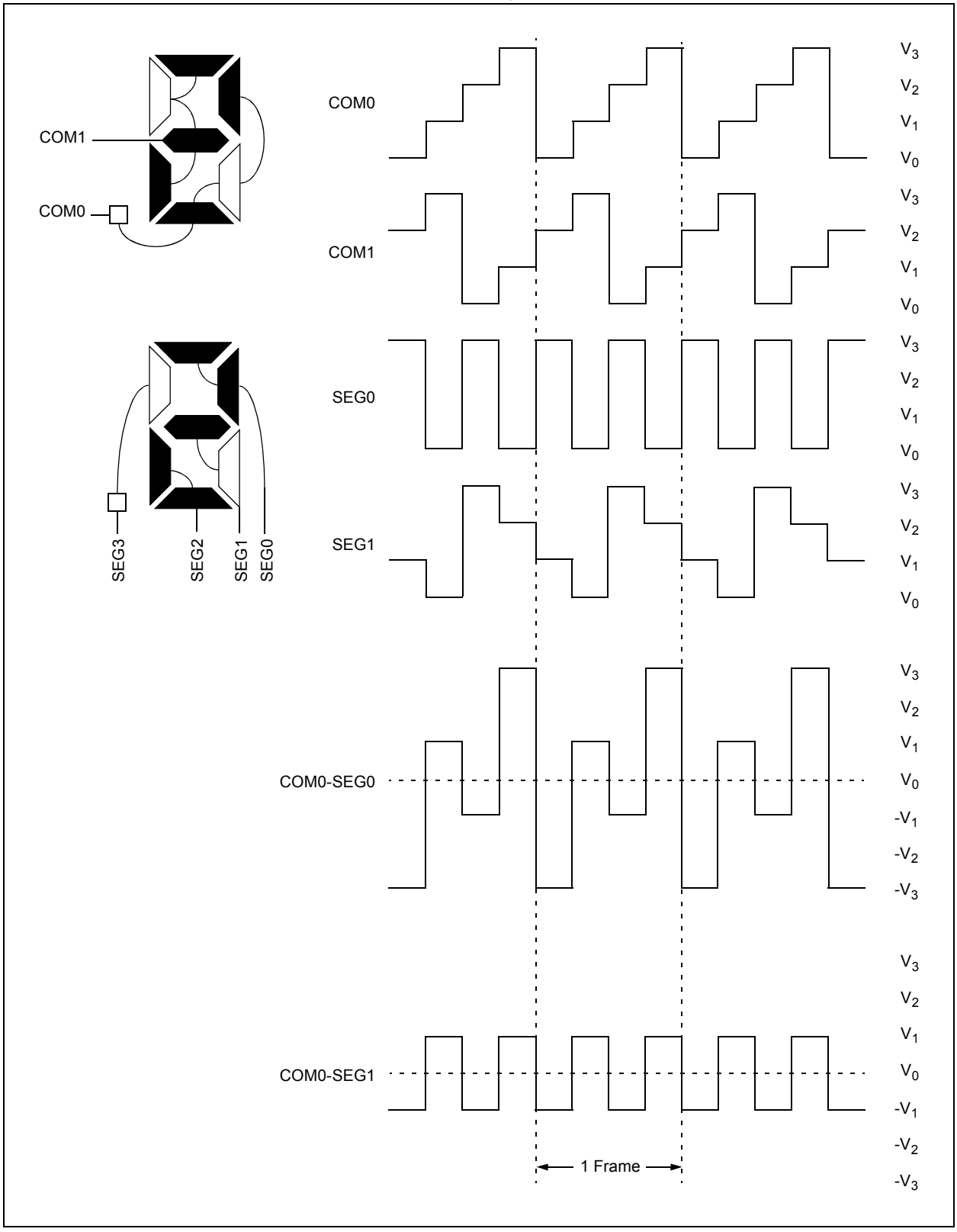


FIGURE 13-13: TYPE-B WAVEFORMS IN 1/2 MUX, 1/3 BIAS DRIVE

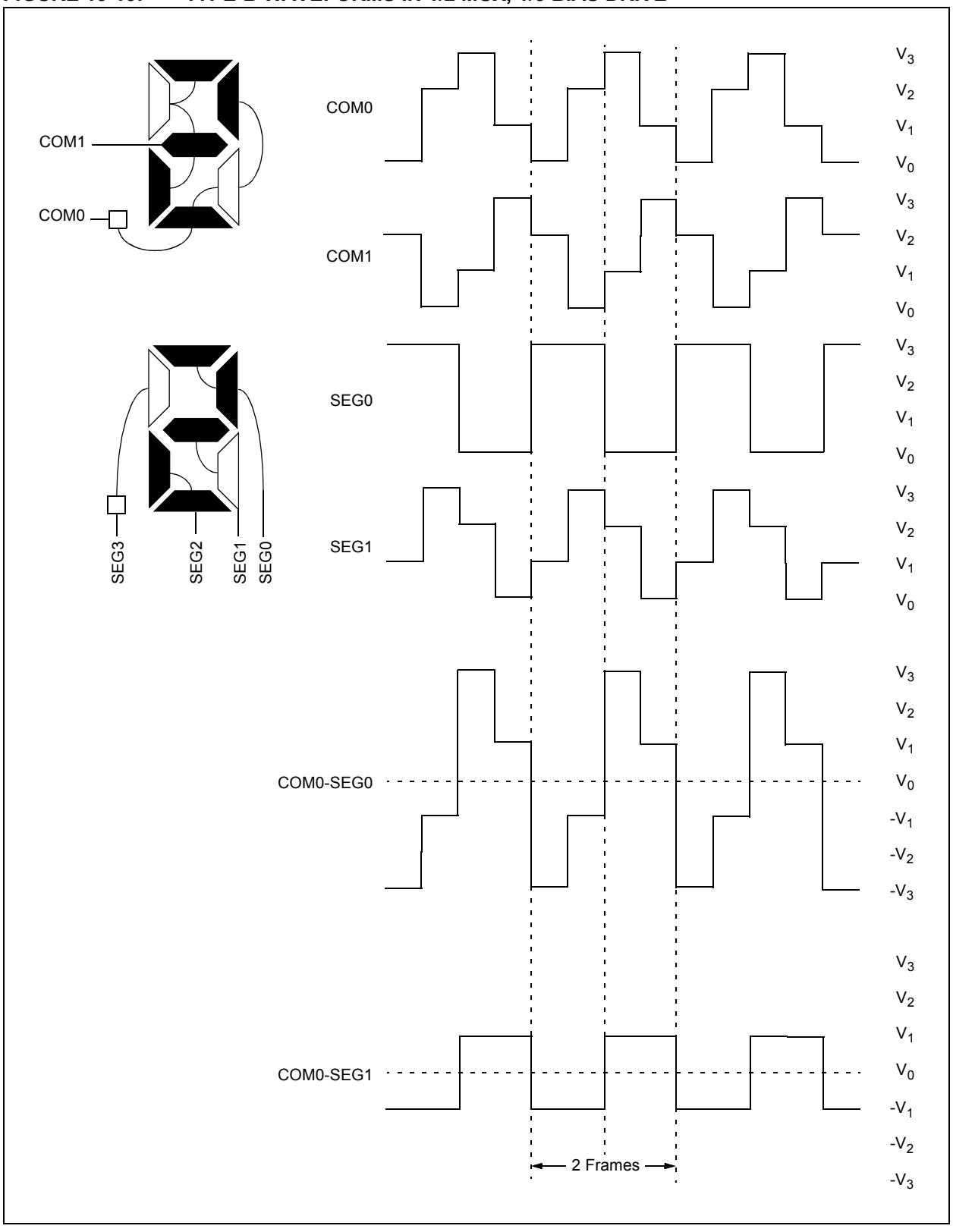


FIGURE 13-14: TYPE-A WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE

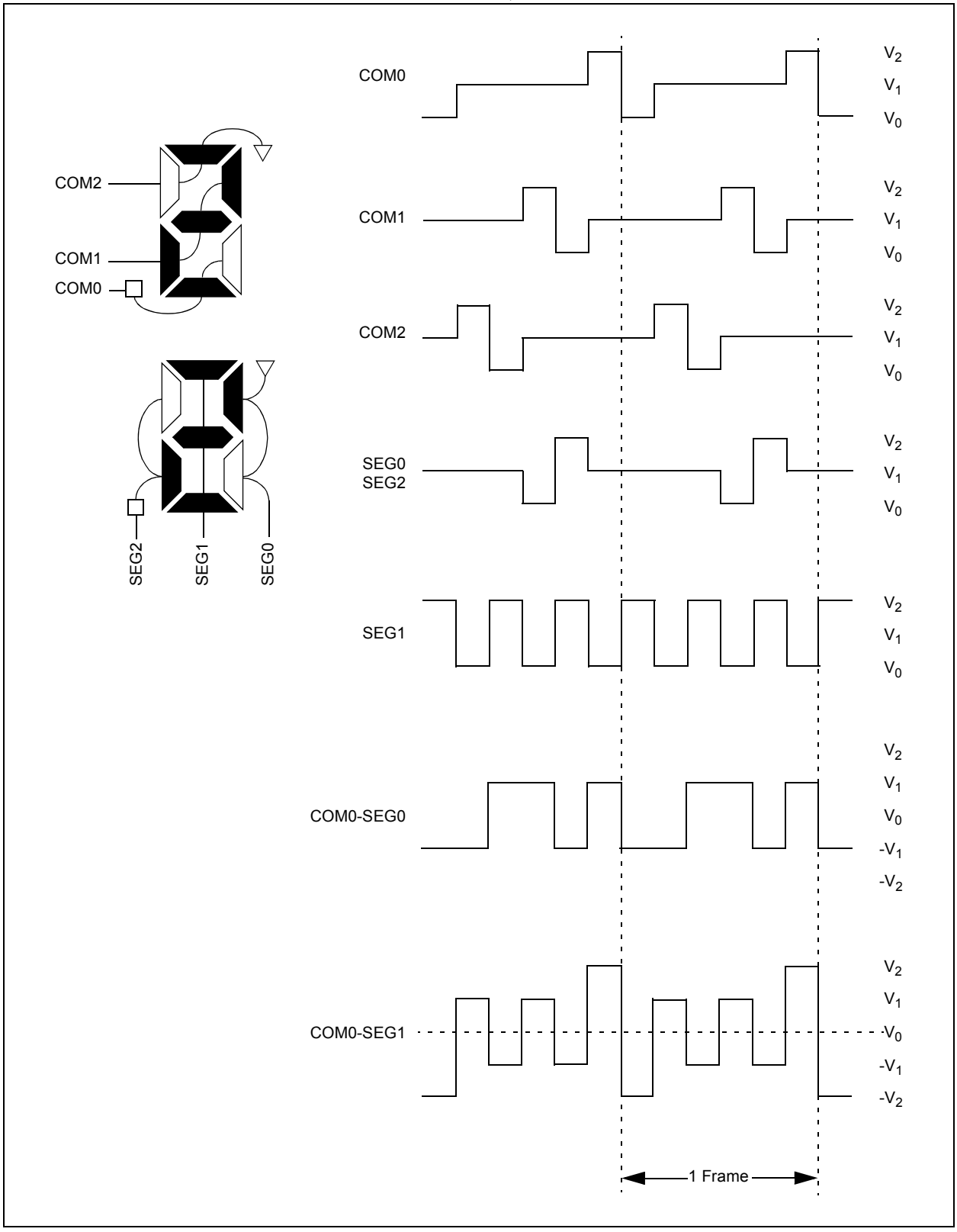


FIGURE 13-15: TYPE-B WAVEFORMS IN 1/3 MUX, 1/2 BIAS DRIVE

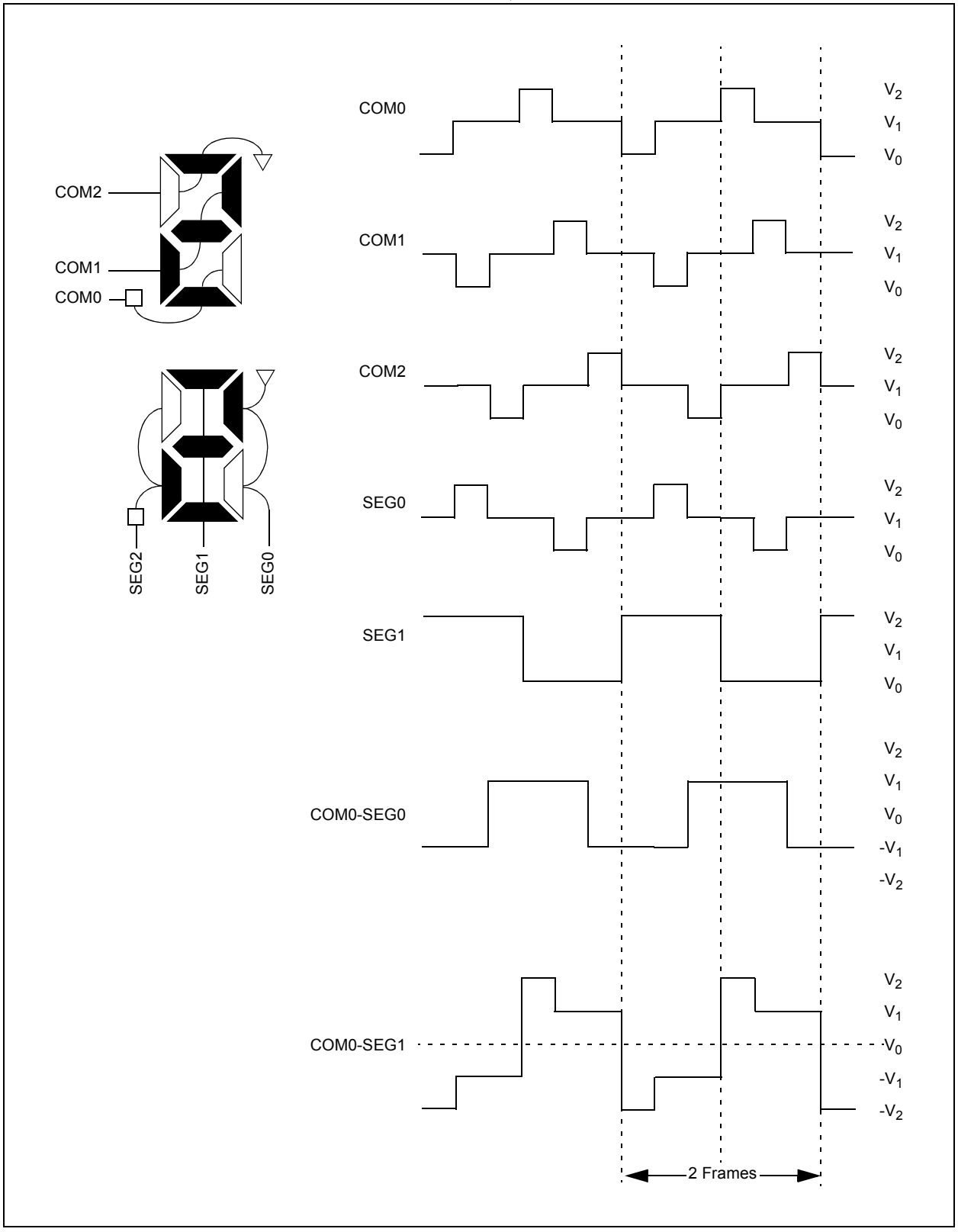


FIGURE 13-16: TYPE-A WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE

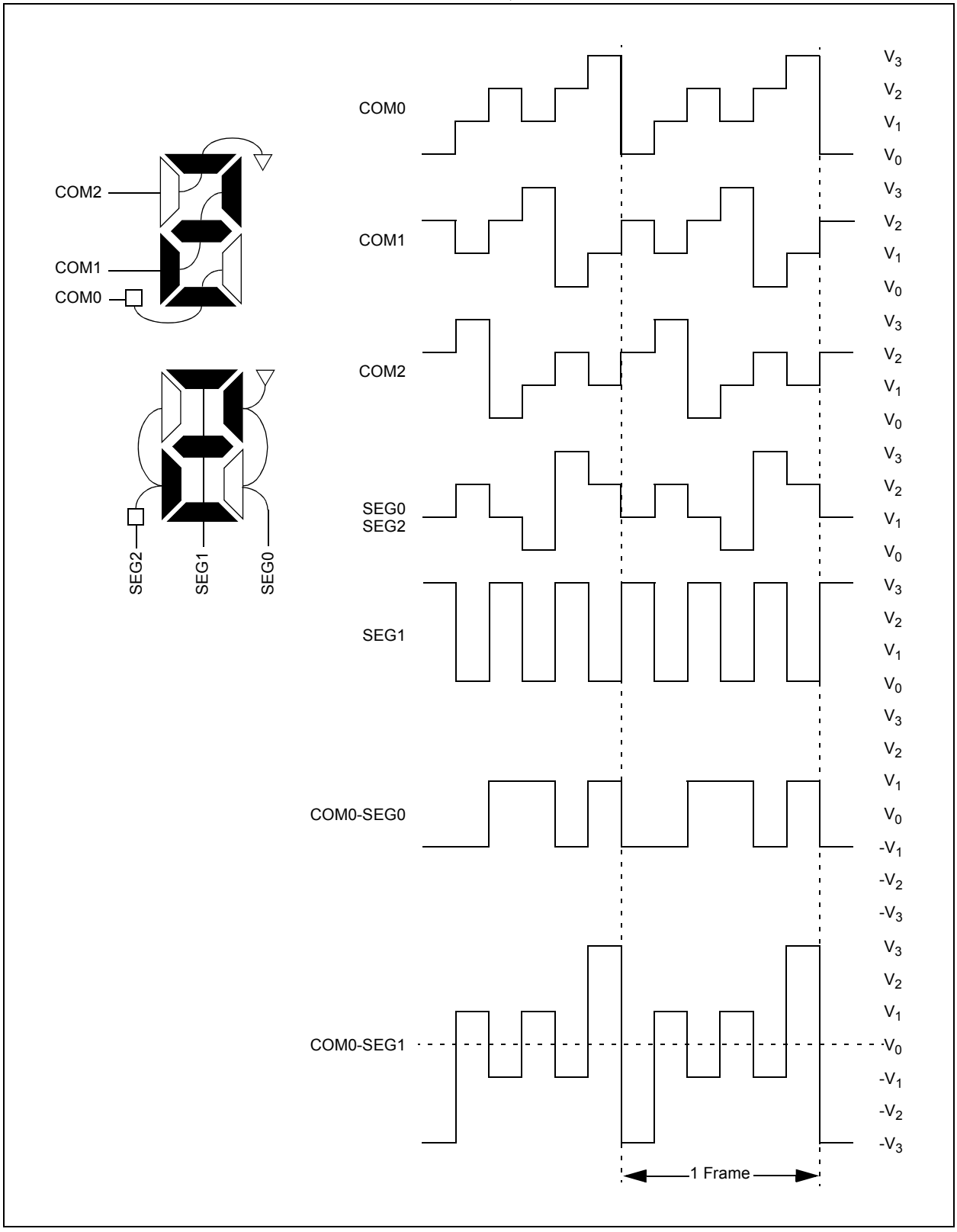


FIGURE 13-17: TYPE-B WAVEFORMS IN 1/3 MUX, 1/3 BIAS DRIVE

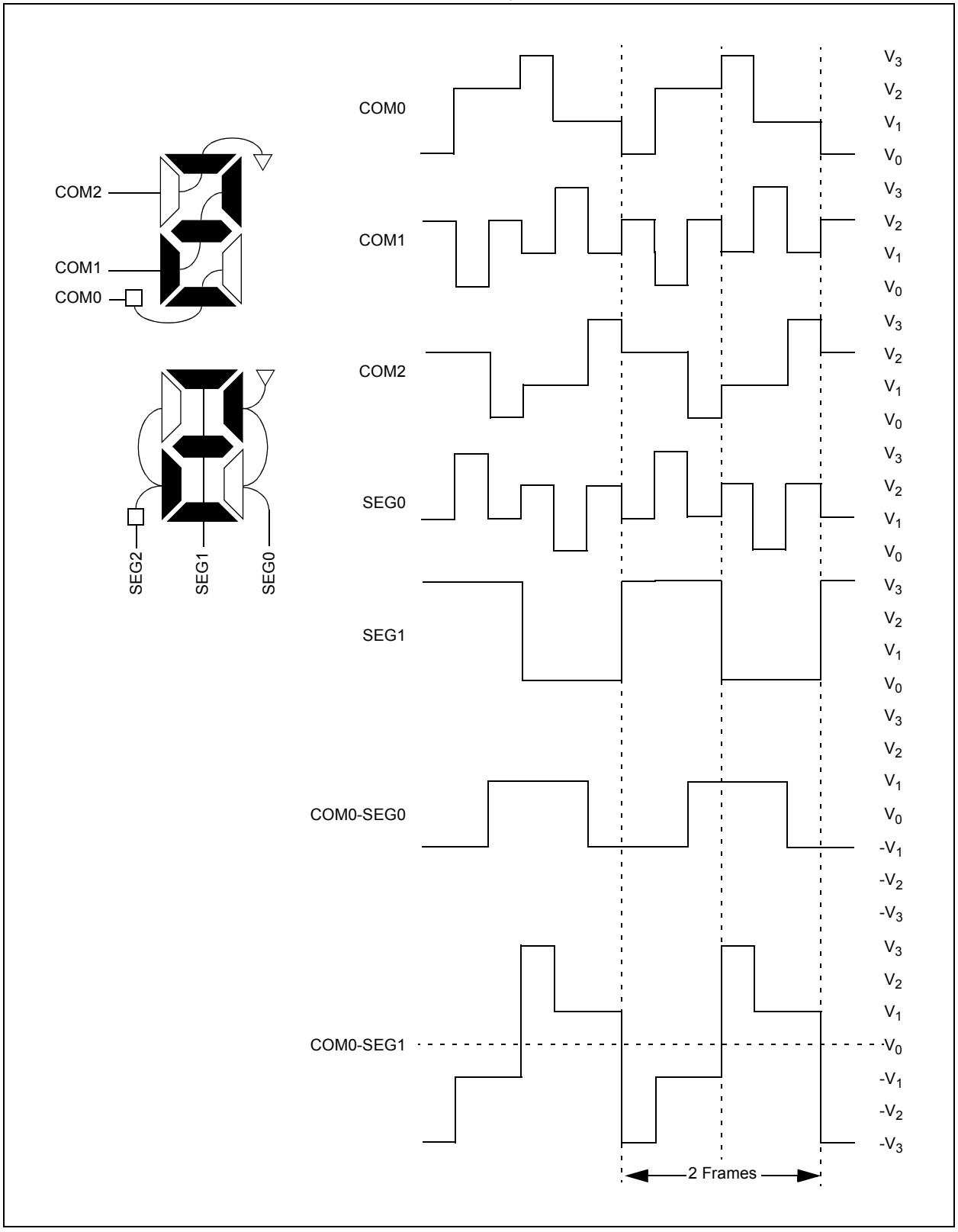


FIGURE 13-18: TYPE-A WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE

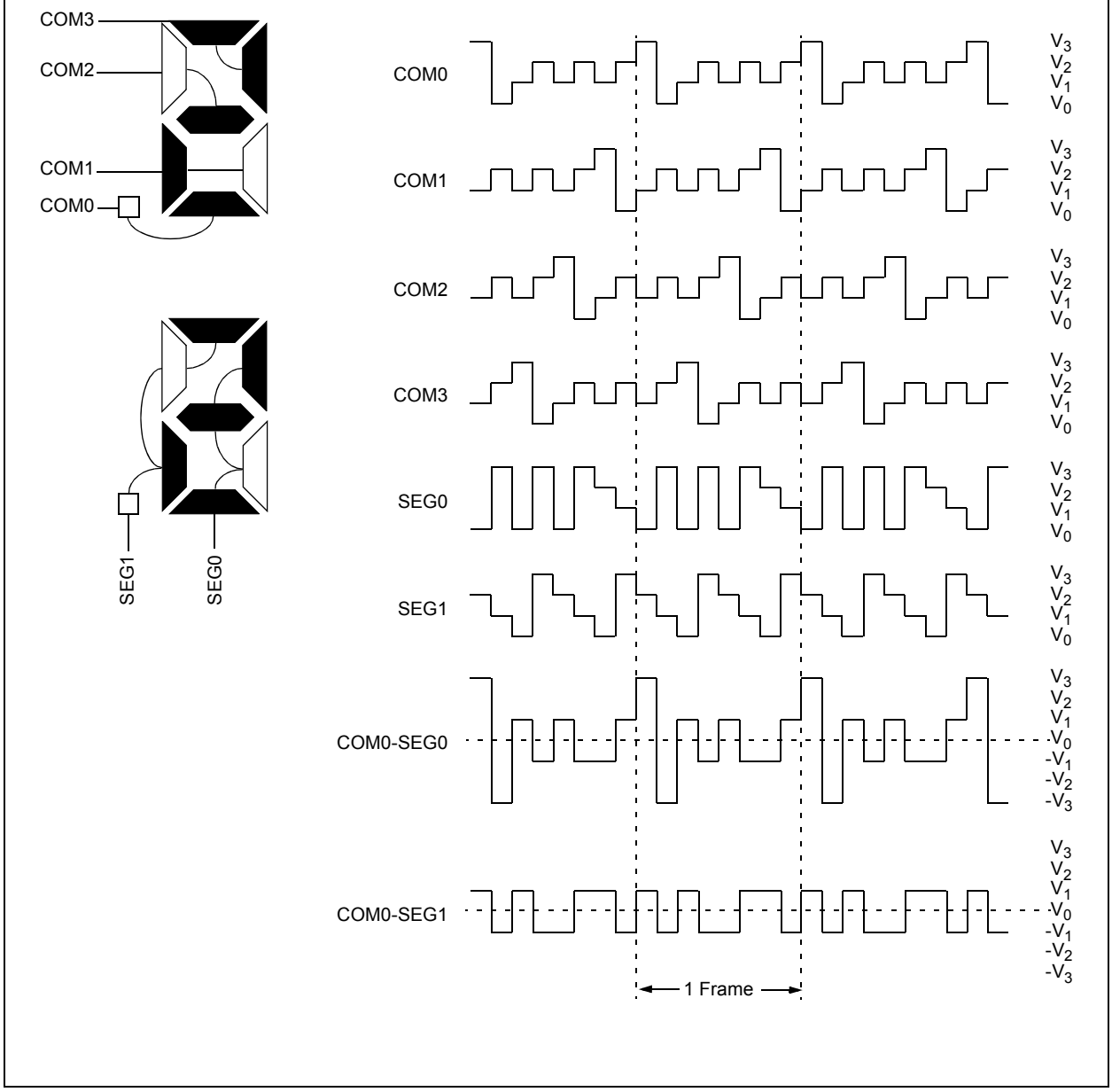


FIGURE 13-19: TYPE-B WAVEFORMS IN 1/4 MUX, 1/3 BIAS DRIVE

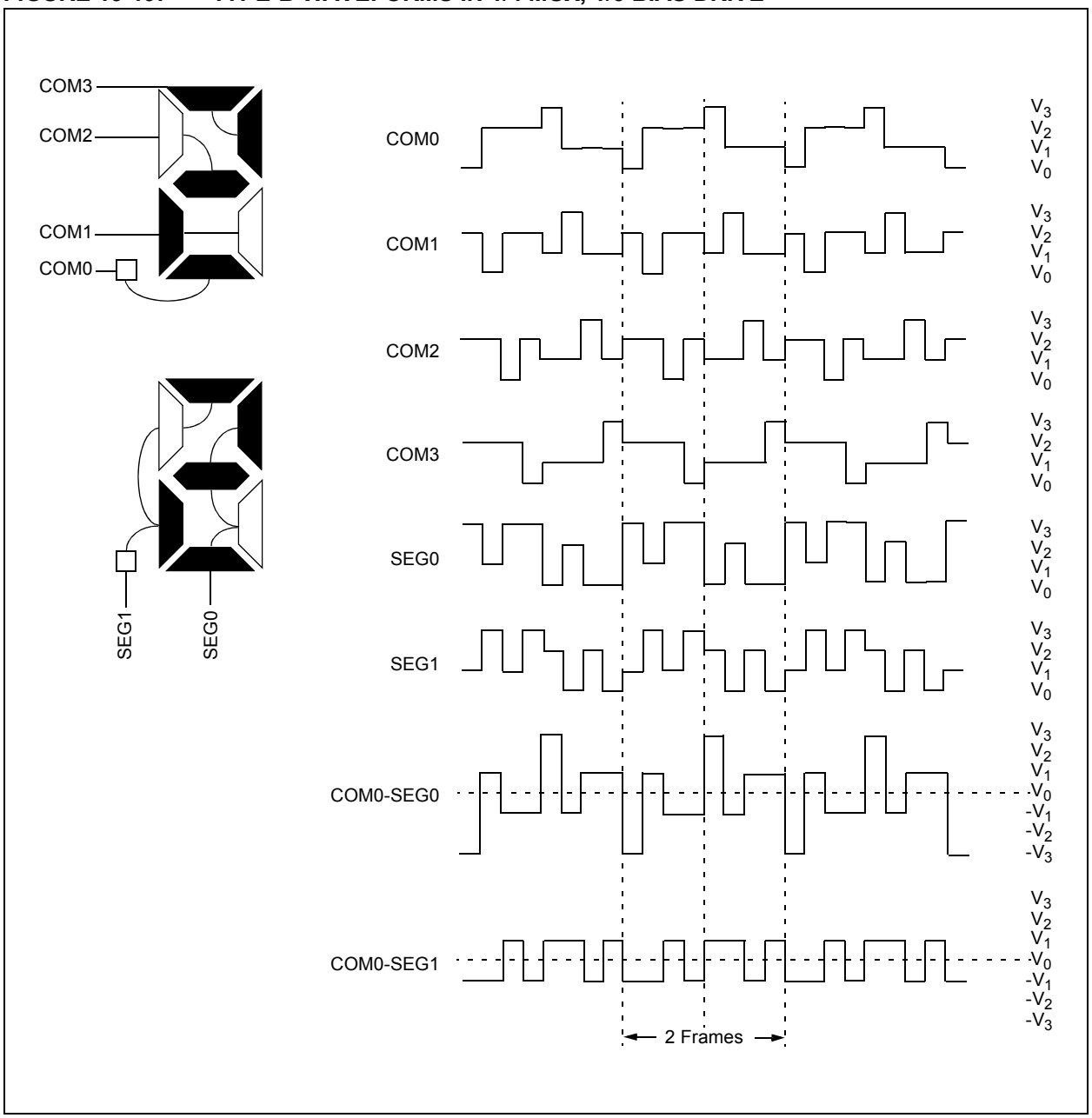


FIGURE 13-20: TYPE-A WAVEFORMS IN 1/8 MUX, 1/3 BIAS DRIVE

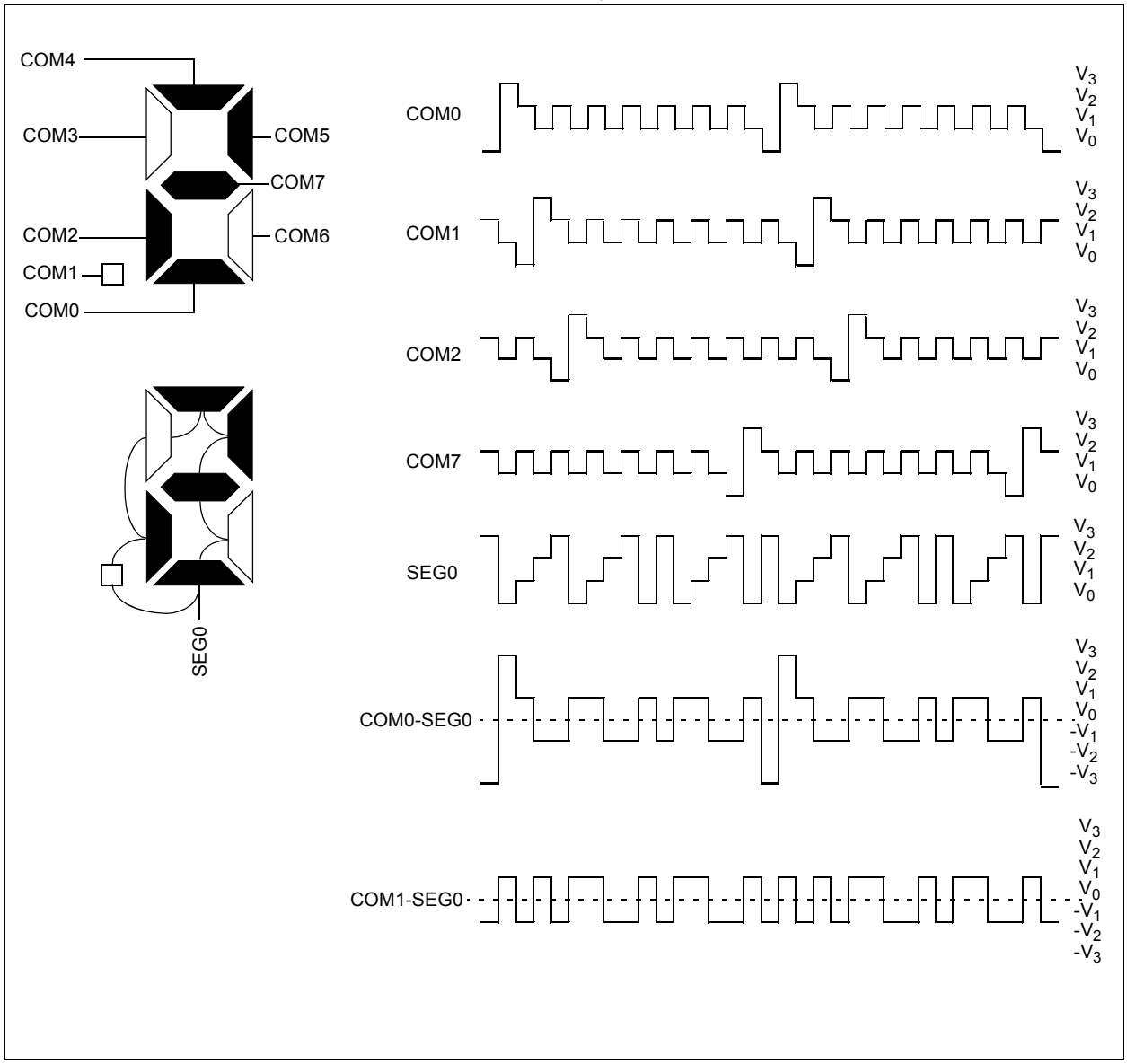
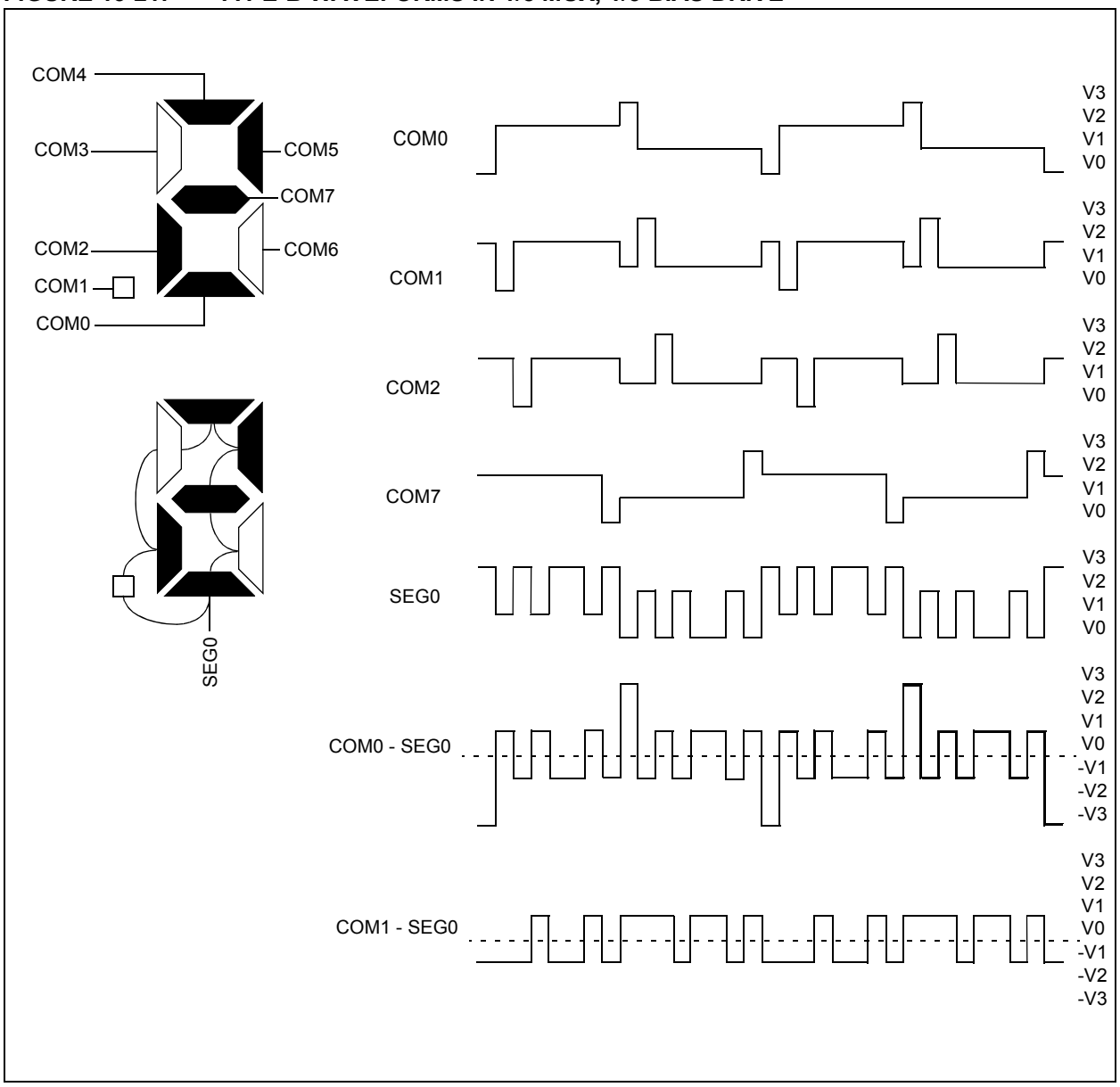


FIGURE 13-21: TYPE-B WAVEFORMS IN 1/8 MUX, 1/3 BIAS DRIVE



13.13 LCD Interrupts

The LCD timing generation provides an interrupt that defines the LCD frame timing. This interrupt can be used to coordinate the writing of the pixel data with the start of a new frame, which produces a visually crisp transition of the image.

This interrupt can also be used to synchronize external events to the LCD. For example, the interface to an external segment driver can be synchronized for segment data updates to the LCD frame.

A new frame is defined as beginning at the leading edge of the COM0 common signal. The interrupt will be set immediately after the LCD controller completes accessing all pixel data required for a frame. This will occur at a fixed interval before the frame boundary (TFINT), as shown in Figure 13-22.

The LCD controller will begin to access data for the next frame within the interval from the interrupt to when the controller begins accessing data after the interrupt (TFWR). New data must be written within TFWR, as this is when the LCD controller will begin to access the data for the next frame.

When the LCD driver is running with Type-B waveforms, and the LMUX<2:0> bits are not equal to '000', there are some additional issues.

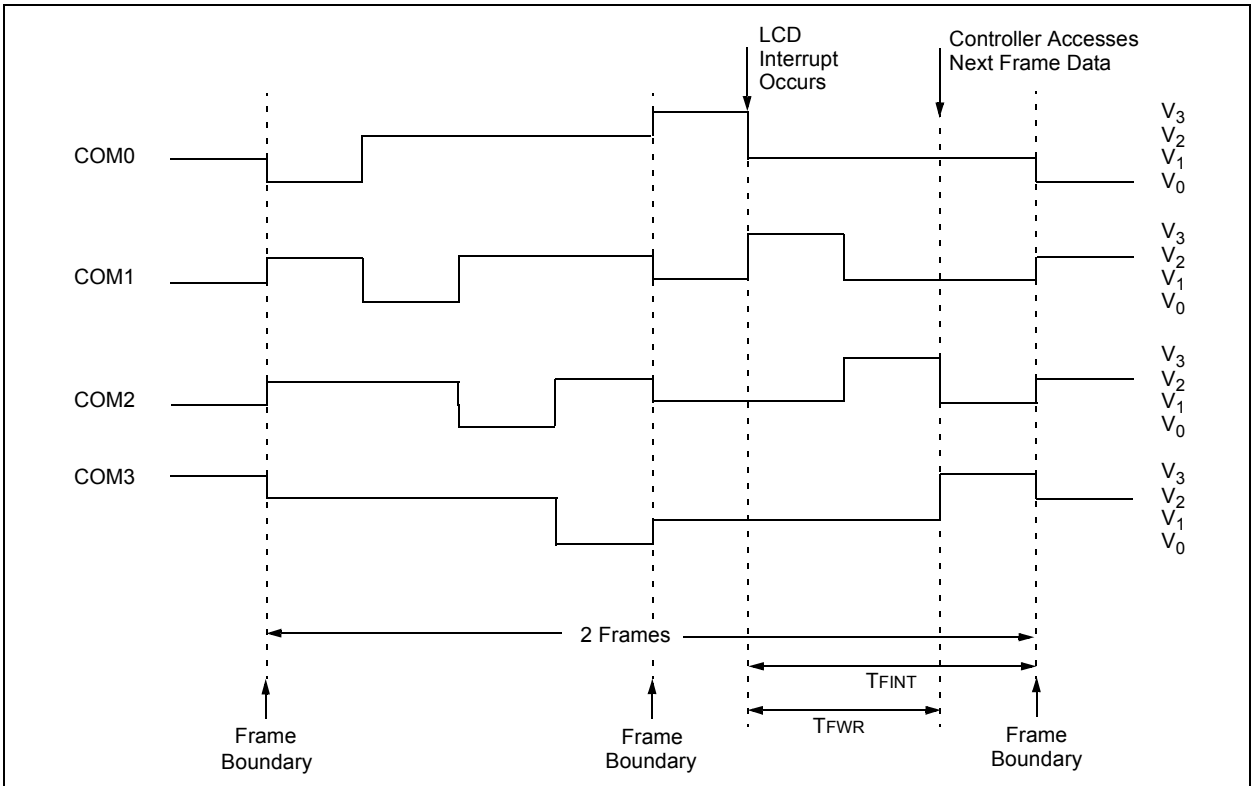
Since the DC voltage on the pixel takes two frames to maintain 0V, the pixel data must not change between subsequent frames. If the pixel data were allowed to change, the waveform for the odd frames would not necessarily be the complement of the waveform generated in the even frames and a DC component would be introduced into the panel.

Because of this, using Type-B waveforms requires synchronizing the LCD pixel updates to occur within a subframe after the frame interrupt.

To correctly sequence writing in Type-B, the interrupt only occurs on complete phase intervals. If the user attempts to write when the write is disabled, the WERR bit (LCDCON<5>) is set.

Note:	The interrupt is not generated when the Type-A waveform is selected and when the Type-B with no multiplex (static) is selected.

FIGURE 13-22: EXAMPLE WAVEFORMS AND INTERRUPT TIMING IN QUARTER DUTY CYCLE DRIVE



TFWR = TFRAME/2 * (LMUX<2:0> + 1) + TCY/2

TFINT = (TFWR/2 – (2 TCY + 40 ns)) → Minimum = 1.5(TFRAME/4) – (2 TCY + 40 ns)
(TFWR/2 – (1 TCY + 40 ns)) → Maximum = 1.5(TFRAME/4) – (1 TCY + 40 ns)

13.14 Configuring the LCD Module

To configure the LCD module.

1. Select the frame clock prescale using bits, LP<3:0> (LCDPS<3:0>).
2. Configure the appropriate pins to function as segment drivers using the LCDSEx registers.
3. If using the internal reference resistors for biasing, enable the internal reference ladder and:
   - Define the Mode A and Mode B interval by using the LRLAT<2:0> bits (LCDRL<2:0>)
   - Define the low, medium or high ladder for Mode A and Mode B by using the LRLAP<1:0> bits (LCDRL<7:6>) and the LRLBP<1:0> bits (LCDRL<5:4>), respectively
   - Set the VLCDxPE bits and enable the LCDIRE bit (LCDREF<7>)
4. Configure the following LCD module functions using the LCDCON register:
   - Multiplex and Bias mode – LMUX<2:0> bits
   - Timing Source – CS<1:0> bits
   - Sleep mode – SLPEN bit
5. Write initial values to the Pixel Data registers, LCDDATA0 through LCDDATA63.
6. Clear the LCD Interrupt Flag, LCDIF, and if desired, enable the interrupt by setting bit, LCDIE.
7. Enable the LCD module by setting the LCDEN bit (LCDCON<7>)

13.15 Operation During Sleep

The LCD module can operate during Sleep. The selection is controlled by the SLPEN bit (LCDCON<6>). Setting the SLPEN bit allows the LCD module to go to Sleep. Clearing the SLPEN bit allows the module to continue to operate during Sleep.

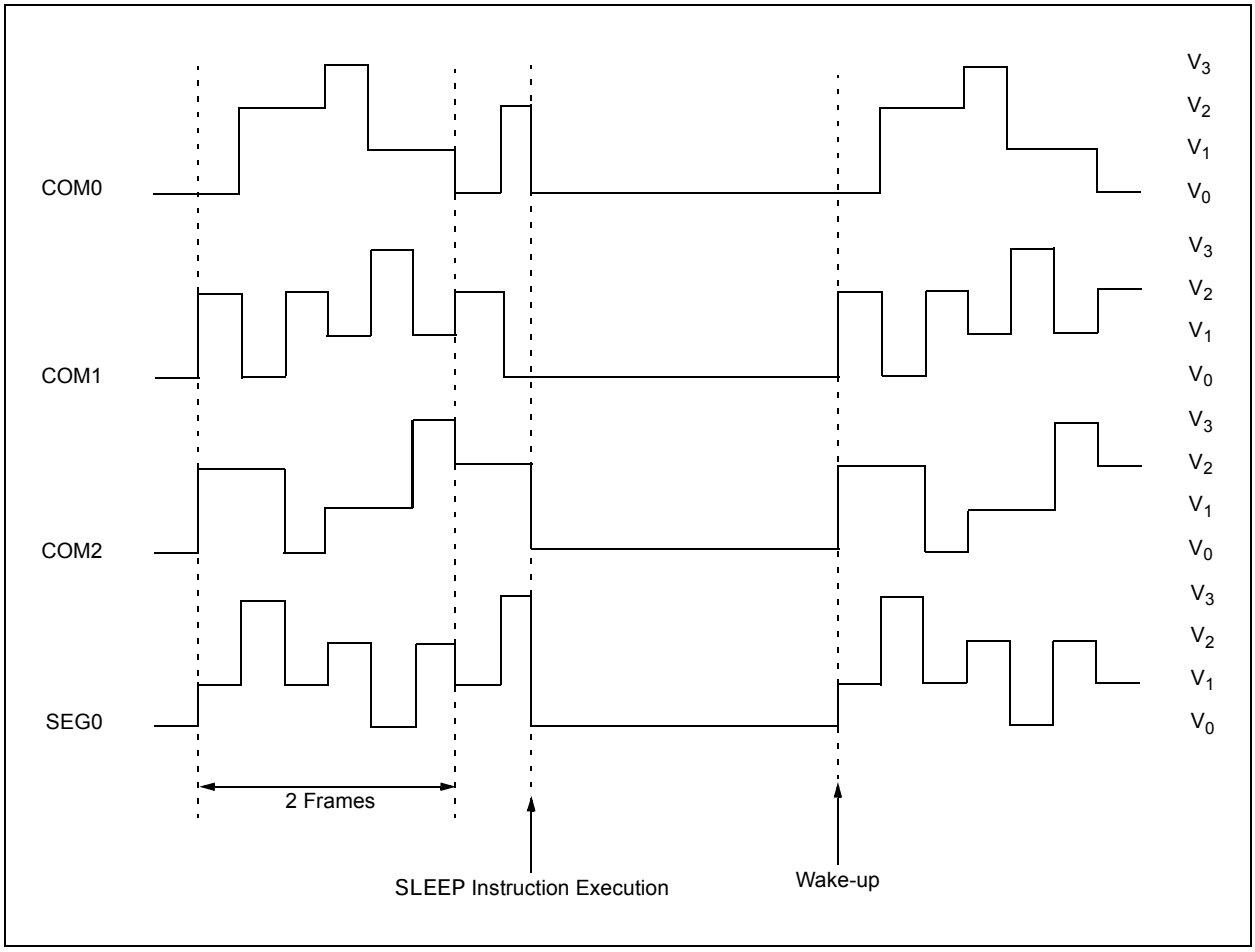
If a SLEEP instruction is executed and SLPEN = 1, the LCD module will cease all functions and go into a very Low-Current Consumption mode. The module will stop operation immediately and drive the minimum LCD voltage on both segment and common lines. Figure 13-23 shows this operation.

The LCD module current consumption will not decrease in this mode, but the overall consumption of the device will be lower due to shut down of the core and other peripheral functions.

To ensure that no DC component is introduced on the panel, the SLEEP instruction should be executed immediately after an LCD frame boundary. The LCD interrupt can be used to determine the frame boundary. See **Section 13.13 "LCD Interrupts"** for the formulas to calculate the delay.

If a SLEEP instruction is executed and SLPEN = 0, the module will continue to display the current contents of the LCDDATA registers. The LCD data cannot be changed.

FIGURE 13-23: SLEEP ENTRY/EXIT WHEN SLPEN = 1 OR CS<1:0> = 00.



14.0 TIMER0 MODULE

The Timer0 module incorporates the following features:

- Software-selectable operation as a timer or counter in both 8-bit or 16-bit modes
- Readable and writable registers
- Dedicated 8-bit, software programmable prescaler
- Selectable clock source (internal or external)
- Edge select for external clock
- Interrupt-on-overflow

The T0CON register (Register 14-1) controls all aspects of the module's operation, including the prescale selection. It is both readable and writable.

Figure 14-1 provides a simplified block diagram of the Timer0 module in 8-bit mode. Figure 14-2 provides a simplified block diagram of the Timer0 module in 16-bit mode.

REGISTER 14-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS1	T0CS0	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TMR0ON:** Timer0 On/Off Control bit
1 = Enables Timer0
0 = Stops Timer0

bit 6 **T08BIT**: Timer0 8-Bit/16-Bit Control bit
1 = Timer0 is configured as an 8-bit timer/counter
0 = Timer0 is configured as a 16-bit timer/counter

bit 5-4 **T0CS<1:0>**: Timer0 Clock Source Select bit
11 = Increment on high-to-low transition on T0CKI pin
10 = Increment on low-to-high transition on T0CKI pin
01 = Internal clock (FOSC/4)
00 = INTOSC

bit 3 **PSA**: Timer0 Prescaler Assignment bit
1 = Timer0 prescaler is not assigned; Timer0 clock input bypasses prescaler
0 = Timer0 prescaler is assigned; Timer0 clock input comes from prescaler output

bit 2-0 **T0PS<2:0>**: Timer0 Prescaler Select bits
111 = 1:256 Prescale value
110 = 1:128 Prescale value
101 = 1:64 Prescale value
100 = 1:32 Prescale value
011 = 1:16 Prescale value
010 = 1:8 Prescale value
001 = 1:4 Prescale value
000 = 1:2 Prescale value

14.1 Timer0 Operation

Timer0 can operate in one of these two modes:

- As an 8-bit (T08BIT = 1) or 16-bit (T08BIT = 0) timer
- As an asynchronous 8-bit (T08BIT = 1) or 16-bit (T08BIT = 0) counter

14.1.1 TIMER MODE

In Timer mode, Timer0 either increments every CPU clock cycle, or every instruction cycle, depending on the clock select bit, TMR0CS<1:0> (T0CON<7:6>).

14.1.2 COUNTER MODE

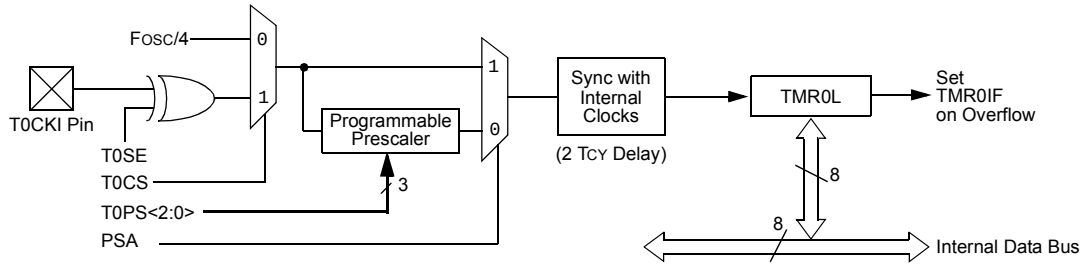
In this mode, Timer0 is incremented via a rising or falling edge of an external source on the T0CKI pin. The clock select bits, TMR0CS<1:0>, must be set to '1x'.

14.2 Timer0 Reads and Writes in 16-Bit Mode

TMR0H is not the actual high byte of Timer0 in 16-bit mode. It is actually a buffered version of the real high byte of Timer0, which is not directly readable nor writable (see Figure 14-2). TMR0H is updated with the contents of the high byte of Timer0 during a read of TMR0L. This provides the ability to read all 16 bits of Timer0 without having to verify that the read of the high and low byte were valid, due to a rollover between successive reads of the high and low byte.

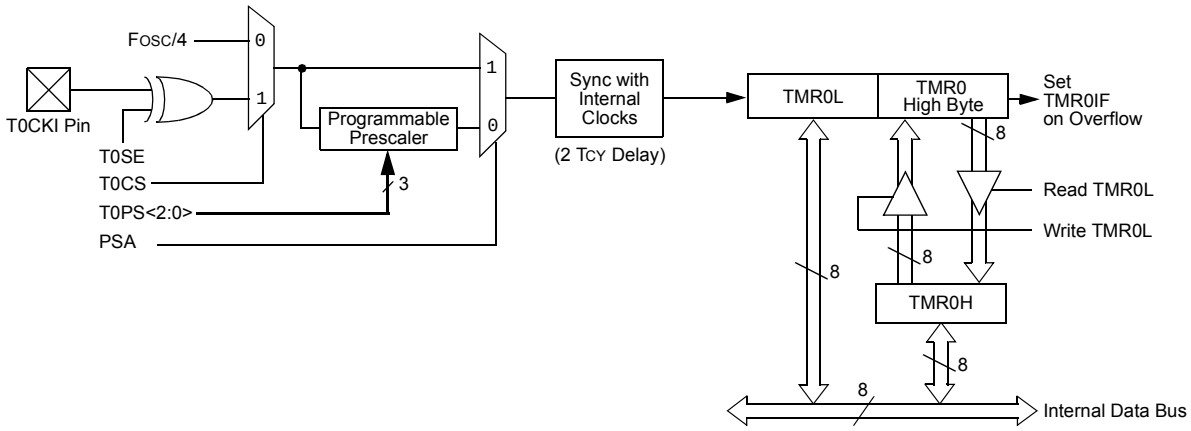
Similarly, a write to the high byte of Timer0 must also take place through the TMR0H Buffer register. The high byte is updated with the contents of TMR0H when a write occurs to TMR0L. This allows all 16 bits of Timer0 to be updated at once.

FIGURE 14-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

FIGURE 14-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)



Note: Upon Reset, Timer0 is enabled in 8-bit mode with clock input from T0CKI max. prescale.

14.3 Prescaler

An 8-bit counter is available as a prescaler for the Timer0 module. The prescaler is not directly readable or writable. Its value is set by the PSA and T0PS<2:0> bits (T0CON<3:0>), which determine the prescaler assignment and prescale ratio.

Clearing the PSA bit assigns the prescaler to the Timer0 module. When it is assigned, prescale values from 1:2 through 1:256 in power-of-two increments are selectable.

When assigned to the Timer0 module, all instructions writing to the TMR0 register (for example, `CLRF TMR0`, `MOVWF TMR0`, `BSF TMR0`) clear the prescaler count.

Note:	Writing to TMR0 when the prescaler is assigned to Timer0 will clear the prescaler count but will not change the prescaler assignment.

14.3.1 SWITCHING PRESCALER ASSIGNMENT

The prescaler assignment is fully under software control and can be changed “on-the-fly” during program execution.

14.4 Timer0 Interrupt

The TMR0 interrupt is generated when the TMR0 register overflows from FFh to 00h in 8-bit mode, or from FFFFh to 0000h in 16-bit mode. This overflow sets the TMR0IF flag bit. The interrupt can be masked by clearing the TMR0IE bit (INTCON<5>). Before re-enabling the interrupt, the TMR0IF bit must be cleared in software by the Interrupt Service Routine (ISR).

Since Timer0 is shutdown in Sleep mode, the TMR0 interrupt cannot awaken the processor from Sleep.

15.0 TIMER1/3/5 MODULES

The Timer1/3/5 timer/counter modules incorporate these features:

- Software-selectable operation as a 16-bit timer or counter
- Readable and writable 8-bit registers (TMRxH and TMRxL)
- Selectable clock source (internal or external) with device clock or SOSC Oscillator internal options
- Interrupt-on-overflow
- Module Reset on ECCP Special Event Trigger

Note: Throughout this section, generic references are used for register and bit names that are the same – except for an 'x' variable that indicates the item's association with the Timer1, Timer3 or Timer5 module. For example, the control register is named TxCON and refers to T1CON, T3CON and T5CON.

A simplified block diagram of the Timer1/3/5 module is shown in Figure 15-1.

The Timer1/3/5 module is controlled through the TxCON register (Register 15-1). It also selects the clock source options for the ECCP modules. (For more information, see **Section 18.1.1 "ECCP Module and Timer Resources"**).

The FOSC clock source should not be used with the ECCP capture/compare features. If the timer will be used with the capture or compare features, always select one of the other timer clocking options.

REGISTER 15-1: TxCON: TIMERx CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
TMRxCS1	TMRxCS0	TxCKPS1	TxCKPS0	SOSCEN	$\overline{\text{TxSYNC}}$	RD16	TMRxON
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **TMRxCS<1:0>:** Timerx Clock Source Select bits
11 = Timerx Clock source is INTOSC
10 = Timerx clock source depends on the SOSCEN bit:
SOSCEN = 0:
External clock from the TxCKI pin (on the rising edge).
SOSCEN = 1:
Depending on the SOSCSEL fuses, either a crystal oscillator on the SOSCI/SOSCO pins or an external clock from the SCLKI pin.
01 = Timerx clock source is the system clock (FOSC)(1)
00 = Timerx clock source is the instruction clock (FOSC/4)

bit 5-4 **TxCKPS<1:0>**: Timerx Input Clock Prescale Select bits
11 = 1:8 Prescale value
10 = 1:4 Prescale value
01 = 1:2 Prescale value
00 = 1:1 Prescale value

bit 3 **SOSCEN**: SOSC Oscillator Enable bit
1 = SOSC/SCLKI are enabled for Timerx (based on the SOSCSEL fuses)
0 = SOSC/SCLKI are disabled for Timerx and TxCKI is enabled

bit 2 **$\overline{\text{TxSYNC}}$:** Timerx External Clock Input Synchronization Control bit
(Not usable if the device clock comes from Timer1/3/5.)
When TMRxCS<1:0> = 10:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMRxCS<1:0> = 0x:
This bit is ignored; Timer1/3/5 uses the internal clock.

bit 1 **RD16:** 16-Bit Read/Write Mode Enable bit
1 = Enables register read/write of Timerx in one 16-bit operation
0 = Enables register read/write of Timerx in two 8-bit operations

bit 0 **TMRxON:** Timerx On bit
1 = Enables Timerx
0 = Stops Timerx

Note 1: The FOSC clock source should not be selected if the timer will be used with the ECCP capture/compare features.

15.1 Timer1/3/5 Gate Control Register

The Timer1/3/5 Gate Control register (TxGCON), provided in Register 15-2, is used to control the Timerx gate.

REGISTER 15-2: TxGCON: TIMERx GATE CONTROL REGISTER(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-x	R/W-0	R/W-0
TMRxGE	TxGPOL	TxGTM	TxGSPM	TxGGO/$\overline{\text{TxDONE}}$	TxGVAL	TxGSS1	TxGSS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **TMRxGE:** Timerx Gate Enable bit

If TMRxON = 0:
This bit is ignored.

If TMRxON = 1:
1 = Timerx counting is controlled by the Timerx gate function
0 = Timerx counts regardless of Timerx gate function

bit 6 **TxGPOL:** Timerx Gate Polarity bit

1 = Timerx gate is active-high (Timerx counts when gate is high)
0 = Timerx gate is active-low (Timerx counts when gate is low)

bit 5 **TxGTM:** Timerx Gate Toggle Mode bit

1 = Timerx Gate Toggle mode is enabled.
0 = Timerx Gate Toggle mode is disabled and toggle flip-flop is cleared
Timerx gate flip-flop toggles on every rising edge.

bit 4 **TxGSPM:** Timerx Gate Single Pulse Mode bit

1 = Timerx Gate Single Pulse mode is enabled and is controlling Timerx gate
0 = Timerx Gate Single Pulse mode is disabled

bit 3 **TxGGO/$\overline{\text{TxDONE}}$:** Timerx Gate Single Pulse Acquisition Status bit

1 = Timerx gate single pulse acquisition is ready, waiting for an edge
0 = Timerx gate single pulse acquisition has completed or has not been started
This bit is automatically cleared when TxGSPM is cleared.

bit 2 **TxGVAL:** Timerx Gate Current State bit

Indicates the current state of the Timerx gate that could be provided to TMRxH:TMRxL; unaffected by the Timerx Gate Enable (TMRxGE) bit.

bit 1-0 **TxGSS<1:0>:** Timerx Gate Source Select bits

11 = Comparator 2 output
10 = Comparator 1 output
01 = TMR(x+1) to match PR(x+1) output(2)
00 = Timer1 gate pin
The Watchdog Timer Oscillator is turned on if TMRxGE = 1, regardless of the state of TMRxON.

Note 1: Programming the TxGCON prior to TxCON is recommended.
2: Timer(x+1) will be Timer1/3/5 for Timerx (Timer1/3/5), respectively.

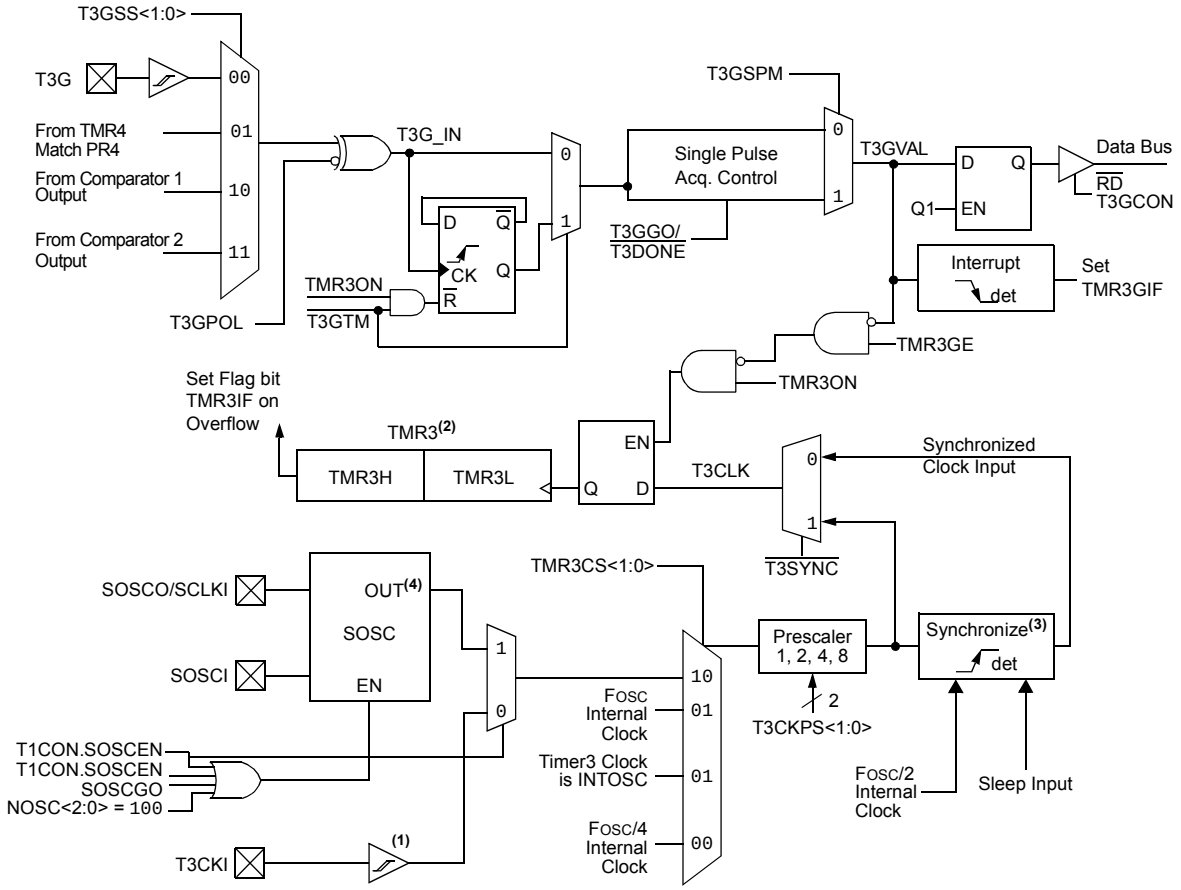
15.2 Timer1/3/5 Operation

Timer1, Timer3 and Timer5 can operate in these modes:

- Timer
- Synchronous Counter
- Asynchronous Counter
- Timer with Gated Control

The operating mode is determined by the clock select bits, TMRxCSx (TxCON<7:6>). When the TMRxCSx bits are cleared (= 00), Timer1/3/5 increments on every internal instruction cycle (FOSC/4). When TMRxCSx = 01, the Timer1/3/5 clock source is the system clock (FOSC). When it is '10', Timer1/3/5 works as a counter from the external clock from the TxCKI pin (on the rising edge after the first falling edge) or the SOSC Oscillator. When it is '11', the Timer1/3/5 clock source is INTOSC.

FIGURE 15-1: TIMER1/3/5 BLOCK DIAGRAM



Note 1: ST buffer is a high-speed type when using T3CKI.
2: Timer3 registers increment on the rising edge.
3: Synchronization does not operate while in Sleep.
4: The output of SOSC is determined by the SOSCSEL Configuration bits.

15.3 Timer1/3/5 16-Bit Read/Write Mode

Timer1/3/5 can be configured for 16-bit reads and writes (see Figure 15-3). When the RD16 control bit (TxCON<1>) is set, the address for TMRxH is mapped to a buffer register for the high byte of Timer1/3/5. A read from TMRxL will load the contents of the high byte of Timer1/3/5 into the Timerx High Byte Buffer register. This provides users with the ability to accurately read all 16 bits of Timer1/3/5 without having to determine whether a read of the high byte, followed by a read of the low byte, has become invalid due to a rollover between reads.

A write to the high byte of Timer1/3/5 must also take place through the TMRxH Buffer register. The Timer1/3/5 high byte is updated with the contents of TMRxH when a write occurs to TMRxL. This allows users to write all 16 bits to both the high and low bytes of Timer1/3/5 at once.

The high byte of Timer1/3/5 is not directly readable or writable in this mode. All reads and writes must take place through the Timerx High Byte Buffer register.

Writes to TMRxH do not clear the Timer1/3/5 prescaler. The prescaler is only cleared on writes to TMRxL.

15.4 Using the SOSC Oscillator as the Timer1/3/5 Clock Source

The SOSC Internal Oscillator may be used as the clock source for Timer1/3/5. It can be enabled in one of these ways:

- Setting the SOSCEN bit in either of the TxCON registers (TxCON<3>)
- Setting the SOSCGO bit in the OSCCON2 register (OSCCON2<1>)
- Setting the NOSC bits to secondary clock source in the OSCCON register (OSCCON<2:0> = `100`)

The SOSCGO bit is used to warm up the SOSC so that it is ready before any peripheral requests it.

To use it as the Timer3 clock source, the TMR3CSx bits must also be set. As previously noted, this also configures Timer3 to increment on every rising edge of the oscillator source.

The SOSC Oscillator is described in **Section 15.4 "Using the SOSC Oscillator as the Timer1/3/5 Clock Source"**.

15.5 Timer1/3/5 Gates

Timer1/3/5 can be configured to count freely or the count can be enabled and disabled using the Timer1/3/5 gate circuitry. This is also referred to as the Timer1/3/5 gate count enable.

The Timer1/3/5 gate can also be driven by multiple selectable sources.

15.5.1 TIMER1/3/5 GATE COUNT ENABLE

The Timerx Gate Enable mode is enabled by setting the TMRxGE bit (TxGCON<7>). The polarity of the Timerx Gate Enable mode is configured using the TxGPOL bit (TxGCON<6>).

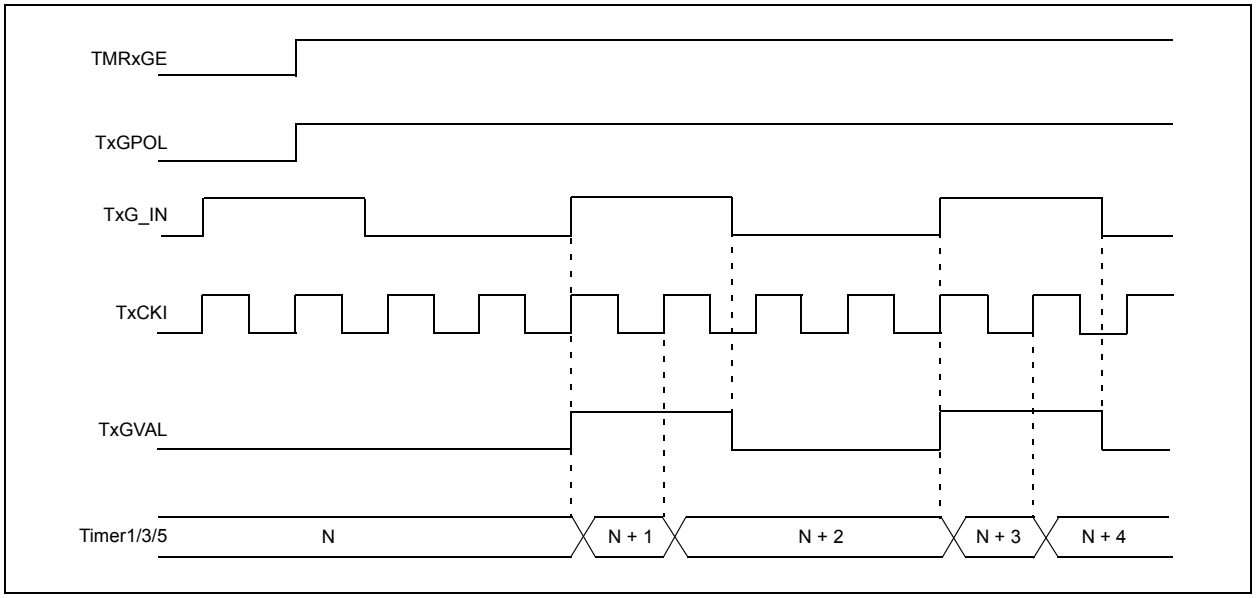
When Timerx Gate Enable mode is enabled, Timer1/3/5 will increment on the rising edge of the Timer1/3/5 clock source. When Timerx Gate Enable mode is disabled, no incrementing will occur and Timer1/3/5 will hold the current count. See Figure 15-2 for timing details.

TABLE 15-1: TIMER1/3/5 GATE ENABLE SELECTIONS

TxCLK(†)	TxGPOL (TxGCON<6>)	TxG Pin	Timerx Operation
↑	0	0	Counts
↑	0	1	Holds Count
↑	1	0	Holds Count
↑	1	1	Counts

† The clock on which TMR1/3/5 is running. For more information, see TxCLK in Figure 15-1.

FIGURE 15-2: TIMER1/3/5 GATE COUNT ENABLE MODE



15.5.2 TIMER1/3/5 GATE SOURCE SELECTION

The Timer1/3/5 gate source can be selected from one of four different sources. Source selection is controlled by the TxGSS<1:0> bits (TxGCON<1:0>). The polarity for each available source is also selectable and is controlled by the TxGPOL bit (TxGCON <6>).

TABLE 15-2: TIMER1/3/5 GATE SOURCES

TxGSS<1:0>	Timerx Gate Source
00	Timerx Gate Pin
01	TMR(x+1) to Match PR(x+1) (TMR(x+1) increments to match PR(x+1))
10	Comparator 1 Output (comparator logic high output)
11	Comparator 2 Output (comparator logic high output)

15.5.2.1 TxG Pin Gate Operation

The TxG pin is one source for Timer1/3/5 gate control. It can be used to supply an external source to the Timerx gate circuitry.

15.5.2.2 Timer2/4/6/8 Match Gate Operation

The TMR(x+1) register will increment until it matches the value in the PR(x+1) register. On the very next increment cycle, TMR2 will be reset to 00h. When this Reset occurs, a low-to-high pulse will automatically be generated and internally supplied to the Timerx gate circuitry. The pulse will remain high for one instruction cycle and will return back to a low state until the next match.

Depending on TxGPOL, Timerx increments differently when TMR(x+1) matches PR(x+1). When TxGPOL = 1, Timerx increments for a single instruction cycle following a TMR(x+1) match with PR(x+1). When TxGPOL = 0, Timerx increments continuously, except for the cycle following the match, when the gate signal goes from low-to-high.

15.5.2.3 Comparator 1 Output Gate Operation

The output of Comparator1 can be internally supplied to the Timerx gate circuitry. After setting up Comparator 1 with the CM1CON register, Timerx will increment depending on the transitions of the C1OUT (CMSTAT<0>) bit.

15.5.2.4 Comparator 2 Output Gate Operation

The output of Comparator 2 can be internally supplied to the Timerx gate circuitry. After setting up Comparator 2 with the CM2CON register, Timerx will increment depending on the transitions of the C2OUT (CMSTAT<1>) bit.

15.5.3 TIMER1/3/5 GATE TOGGLE MODE

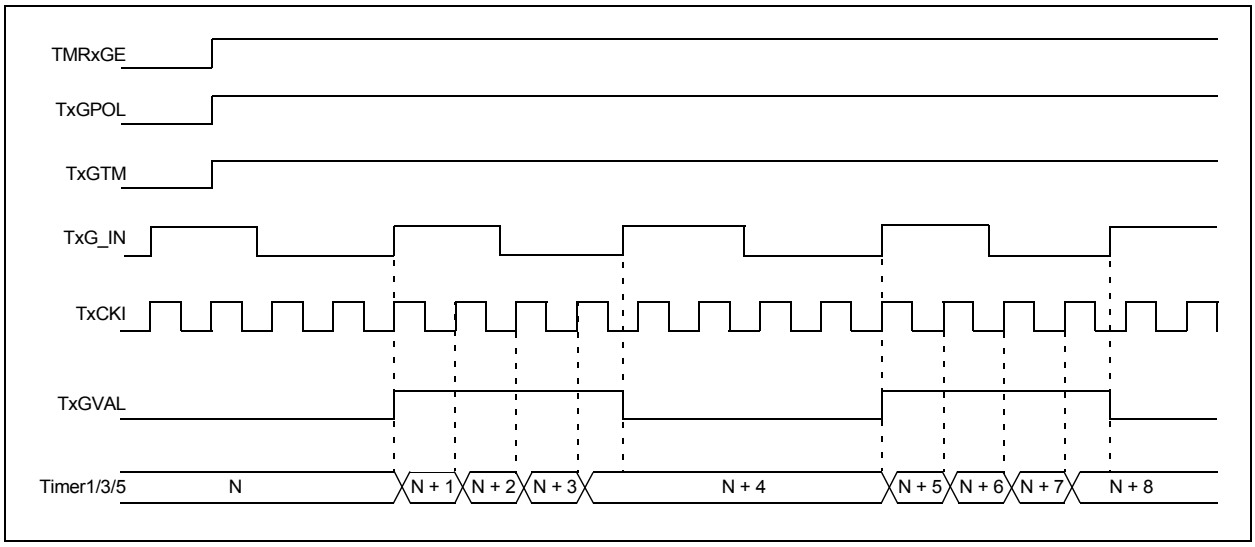
When Timer1/3/5 Gate Toggle mode is enabled, it is possible to measure the full cycle length of a Timer1/3/5 gate signal, as opposed to the duration of a single level pulse.

The Timerx gate source is routed through a flip-flop that changes state on every incrementing edge of the signal. (For timing details, see Figure 15-3.)

The TxGVAL bit will indicate when the Toggled mode is active and the timer is counting.

Timer1/3/5 Gate Toggle mode is enabled by setting the TxGTM bit (TxGCON<5>). When the TxGTM bit is cleared, the flip-flop is cleared and held clear. This is necessary in order to control which edge is measured.

FIGURE 15-3: TIMER1/3/5 GATE TOGGLE MODE



15.5.4 TIMER1/3/5 GATE SINGLE PULSE MODE

When Timer1/3/5 Gate Single Pulse mode is enabled, it is possible to capture a single pulse gate event. Timer1/3/5 Gate Single Pulse mode is first enabled by setting the TxGSPM bit (TxGCON<4>). Next, the TxGGO/$\overline{\text{TxDONE}}$ bit (TxGCON<3>) must be set.

The Timer1/3/5 will be fully enabled on the next incrementing edge. On the next trailing edge of the pulse, the TxGGO/$\overline{\text{TxDONE}}$ bit will automatically be cleared.

No other gate events will be allowed to increment Timer1/3/5 until the TxGGO/$\overline{\text{TxDONE}}$ bit is once again set in software.

Clearing the TxGSPM bit also will clear the TxGGO/$\overline{\text{TxDONE}}$ bit. (For timing details, see Figure 15-4.)

Simultaneously enabling the Toggle mode and the Single Pulse mode will permit both sections to work together. This allows the cycle times on the Timer1/3/5 gate source to be measured. (For timing details, see Figure 15-5.)

FIGURE 15-4: TIMER1/3/5 GATE SINGLE PULSE MODE

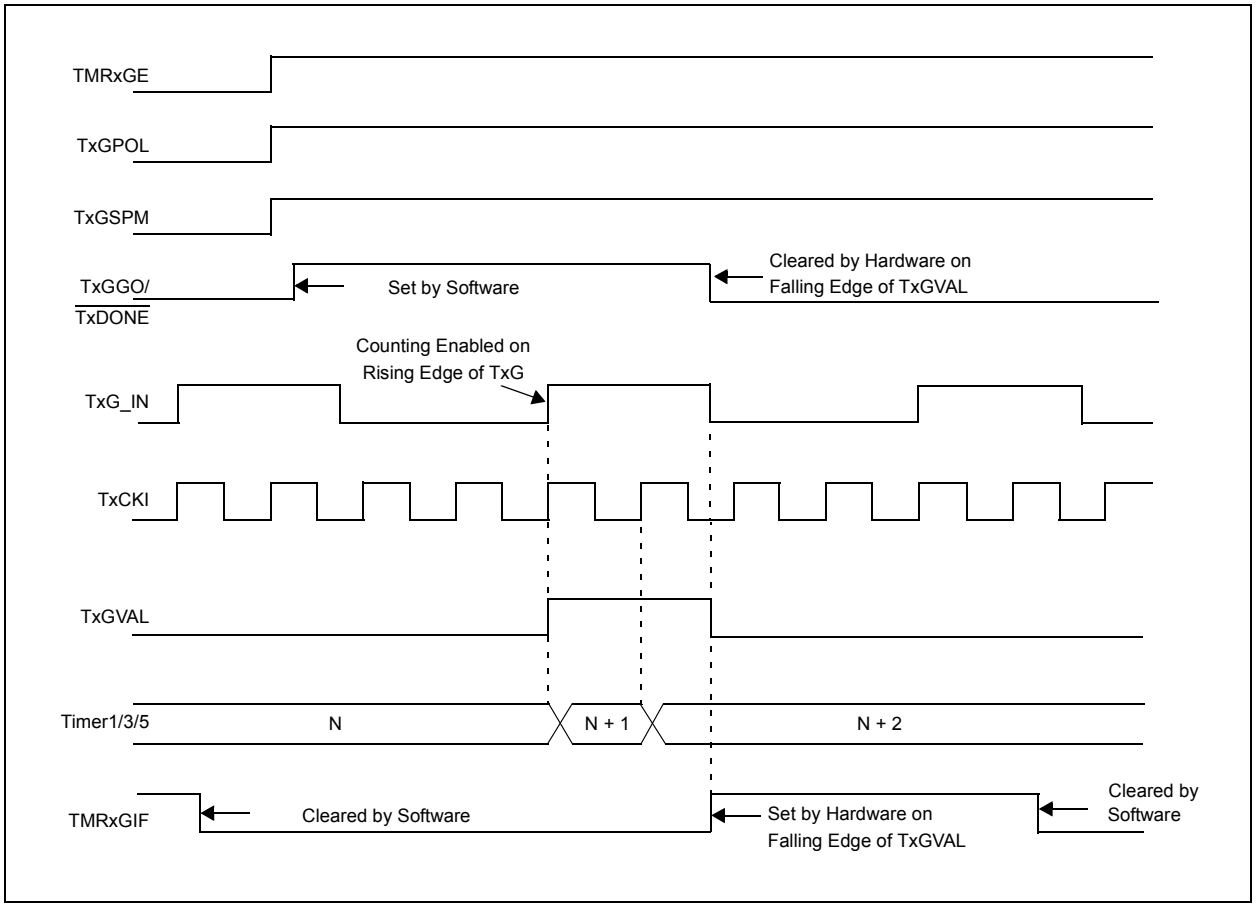
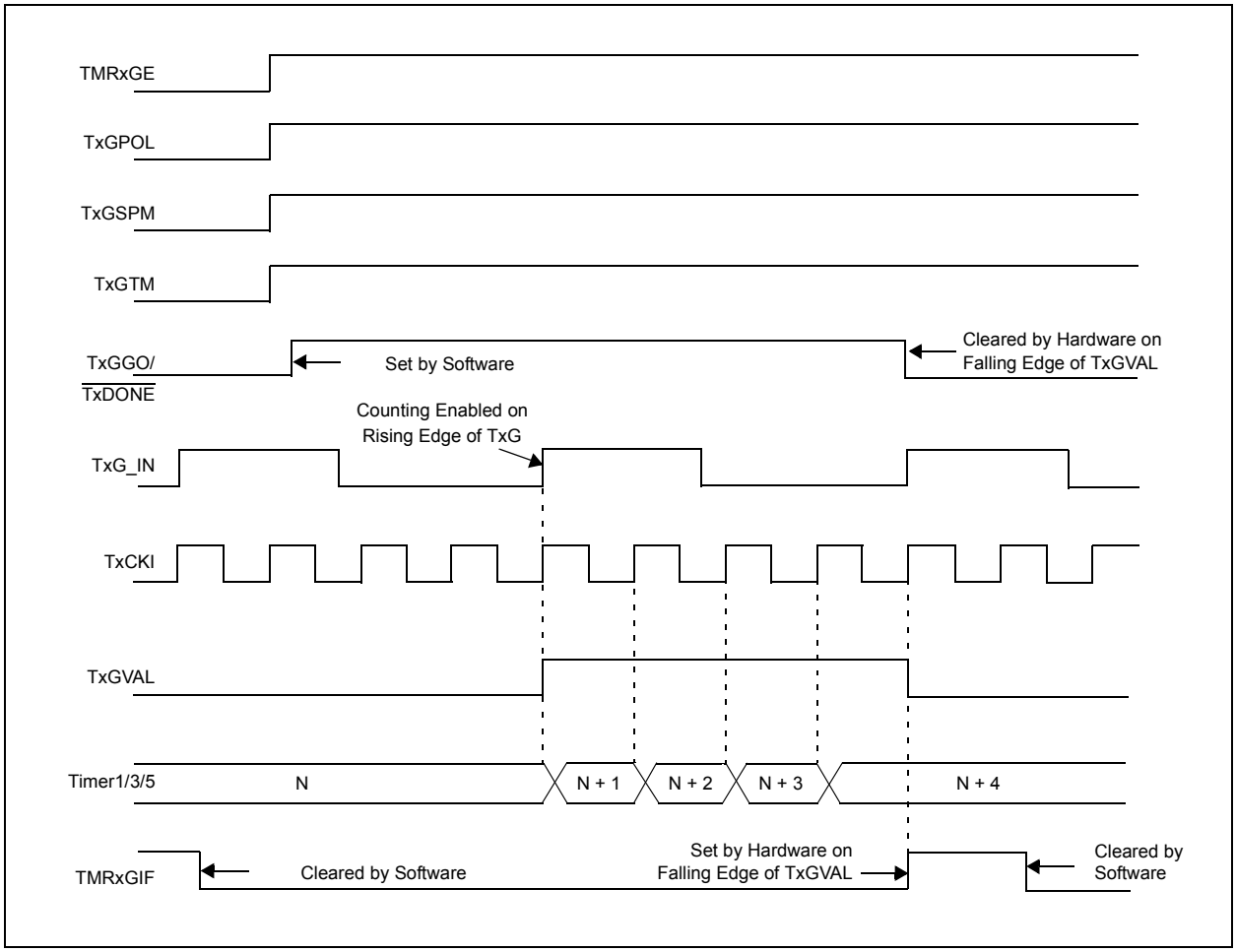


FIGURE 15-5: TIMER1/3/5 GATE SINGLE PULSE AND TOGGLE COMBINED MODE



15.5.5 TIMER1/3/5 GATE VALUE STATUS

When Timer1/3/5 gate value status is utilized, it is possible to read the most current level of the gate control value. The value is stored in the TxGVAL bit (TxGCON<2>). The TxGVAL bit is valid even when the Timer1/3/5 gate is not enabled (TMRxGE bit is cleared).

15.5.6 TIMER1/3/5 GATE EVENT INTERRUPT

When the Timer1/3/5 gate event interrupt is enabled, it is possible to generate an interrupt upon the completion of a gate event. When the falling edge of TxGVAL occurs, the TMRxGIF flag bit in the PIRx register will be set. If the TMRxGIE bit in the PIEx register is set, then an interrupt will be recognized.

The TMRxGIF flag bit operates even when the Timer1/3/5 gate is not enabled (TMRxGE bit is cleared).

15.6 Timer1/3/5 Interrupt

The TMRx register pair (TMRxH:TMRxL) increments from 0000h to FFFFh and overflows to 0000h. The Timerx interrupt, if enabled, is generated on overflow and is latched in the interrupt flag bit, TMRxIF. Table 15-3 gives each module's flag bit.

TABLE 15-3: TIMER1/3/5 INTERRUPT FLAG BITS

Timer Module	Flag Bit
1	PIR1<0>
3	PIR2<1>
5	PIR5<1>

This interrupt can be enabled or disabled by setting or clearing the TMRxIE bit, respectively. Table 15-4 gives each module's enable bit.

TABLE 15-4: TIMER1/3/5 INTERRUPT ENABLE BITS

Timer Module	Flag Bit
1	PIE1<0>
3	PIE2<1>
5	PIE5<1>

15.7 Resetting Timer1/3/5 Using the ECCP Special Event Trigger

If the ECCP modules are configured to use Timerx and to generate a Special Event Trigger in Compare mode (CCPxM<3:0> = 1011), this signal will reset Timerx. The trigger from ECCP2 will also start an A/D conversion if the A/D module is enabled (For more information, see **Section 18.3.4 "Special Event Trigger"**.)

The module must be configured as either a timer or synchronous counter to take advantage of this feature. When used this way, the CCPRxH:CCPRxL register pair effectively becomes a Period register for Timerx.

If Timerx is running in Asynchronous Counter mode, the Reset operation may not work.

In the event that a write to Timerx coincides with a Special Event Trigger from an ECCP module, the write will take precedence.

Note: The Special Event Triggers from the ECCPx module will only clear the TMR3 register's content, but not set the TMR3IF interrupt flag bit (PIR1<0>).

Note: The CCP and ECCP modules use Timers, 1 through 8, for some modes. The assignment of a particular timer to a CCP/ECCP module is determined by the Timer to CCP enable bits in the CCPTMRSx registers. For more details, see Register 18-2, Register 18-3 and Register 19-2

16.0 TIMER2/4/6/8 MODULES

The Timer2/4/6/8 timer modules have the following features:

- 8-Bit Timer register (TMRx)
- 8-Bit Period register (PRx)
- Readable and Writable (all registers)
- Software Programmable Prescaler (1:1, 1:4, 1:16)
- Software Programmable Postscaler (1:1 to 1:16)
- Interrupt on TMRx Match of PRx

Note: Throughout this section, generic references are used for register and bit names that are the same, except for an 'x' variable that indicates the item's association with the Timer2, Timer4, Timer6 or Timer8 module. For example, the control register is named TxCON and refers to T2CON, T4CON, T6CON and T8CON.

The Timer2/4/6/8 modules have a control register, shown in Register 16-1. Timer2/4/6/8 can be shut off by clearing control bit, TMRxON (TxCON<2>), to minimize power consumption. The prescaler and postscaler selection of Timer2/4/6/8 also are controlled by this register. Figure 16-1 is a simplified block diagram of the Timer2/4/6/8 modules.

16.1 Timer2/4/6/8 Operation

Timer2/4/6/8 can be used as the PWM time base for the PWM mode of the ECCP modules. The TMRx registers are readable and writable, and are cleared on any device Reset. The input clock (FOSC/4) has a prescale option of 1:1, 1:4 or 1:16, selected by control bits, TxCKPS<1:0> (TxCON<1:0>). The match output of TMRx goes through a four-bit postscaler (that gives a 1:1 to 1:16 inclusive scaling) to generate a TMRx interrupt, latched in the flag bit, TMRxIF. Table 16-1 gives each module's flag bit.

TABLE 16-1: TIMER2/4/6/8 FLAG BITS

Timer Module	Flag Bit
2	PIR1<1>
4	PIR5<0>
6	PIR5<2>
8	PIR5<4>

The interrupt can be enabled or disabled by setting or clearing the Timerx Interrupt Enable bit (TMRxIE), shown in Table 16-2.

TABLE 16-2: TIMER2/4/6/8 INTERRUPT ENABLE BITS

Timer Module	Flag Bit
2	PIE1<1>
4	PIE5<0>
6	PIE5<2>
8	PIE5<4>

The prescaler and postscaler counters are cleared when any of the following occurs:

- A write to the TMRx register
- A write to the TxCON register
- Any device Reset – Power-on Reset (POR), MCLR Reset, Watchdog Timer Reset (WDTR) or Brown-out Reset (BOR)

A TMRx is not cleared when a TxCON is written.

Note: The CCP and ECCP modules use Timers, 1 through 8, for some modes. The assignment of a particular timer to a CCP/ECCP module is determined by the Timer to CCP enable bits in the CCPTMRSx registers. For more details, see Register 18-2, Register 18-3 and Register 19-2.

REGISTER 16-1: TxCON: TIMERx CONTROL REGISTER (TIMER2/4/6/8)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	TxOUTPS3	TxOUTPS2	TxOUTPS1	TxOUTPS0	TMRxON	TxCKPS1	TxCKPS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-3 **TxOUTPS<3:0>:** Timerx Output Postscale Select bits
0000 = 1:1 Postscale
0001 = 1:2 Postscale
•
•
•
1111 = 1:16 Postscale

bit 2 **TMRxON**: Timerx On bit
1 = Timerx is on
0 = Timerx is off

bit 1-0 **TxCKPS<1:0>:** Timerx Clock Prescale Select bits
00 = Prescaler is 1
01 = Prescaler is 4
1x = Prescaler is 16

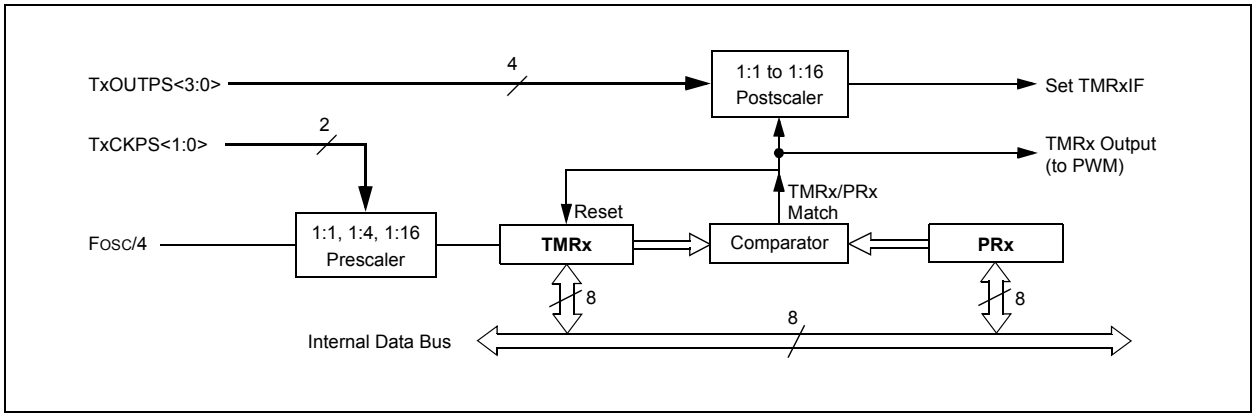
16.2 Timer2/4/6/8 Interrupt

The Timer2/4/6/8 modules have 8-bit Period registers, PRx, that are both readable and writable. Timer2/4/6/8 increment from 00h until they match PR2/4/6/8 and then reset to 00h on the next increment cycle. The PRx registers are initialized to FFh upon Reset.

16.3 Output of TMRx

The outputs of TMRx (before the postscaler) are used only as a PWM time base for the ECCP modules. They are not used as baud rate clocks for the MSSPx modules as is the Timer2 output.

FIGURE 16-1: TIMER2/4/6/8 BLOCK DIAGRAM



17.0 REAL-TIME CLOCK AND CALENDAR (RTCC)

The key features of the Real-Time Clock and Calendar (RTCC) module are:

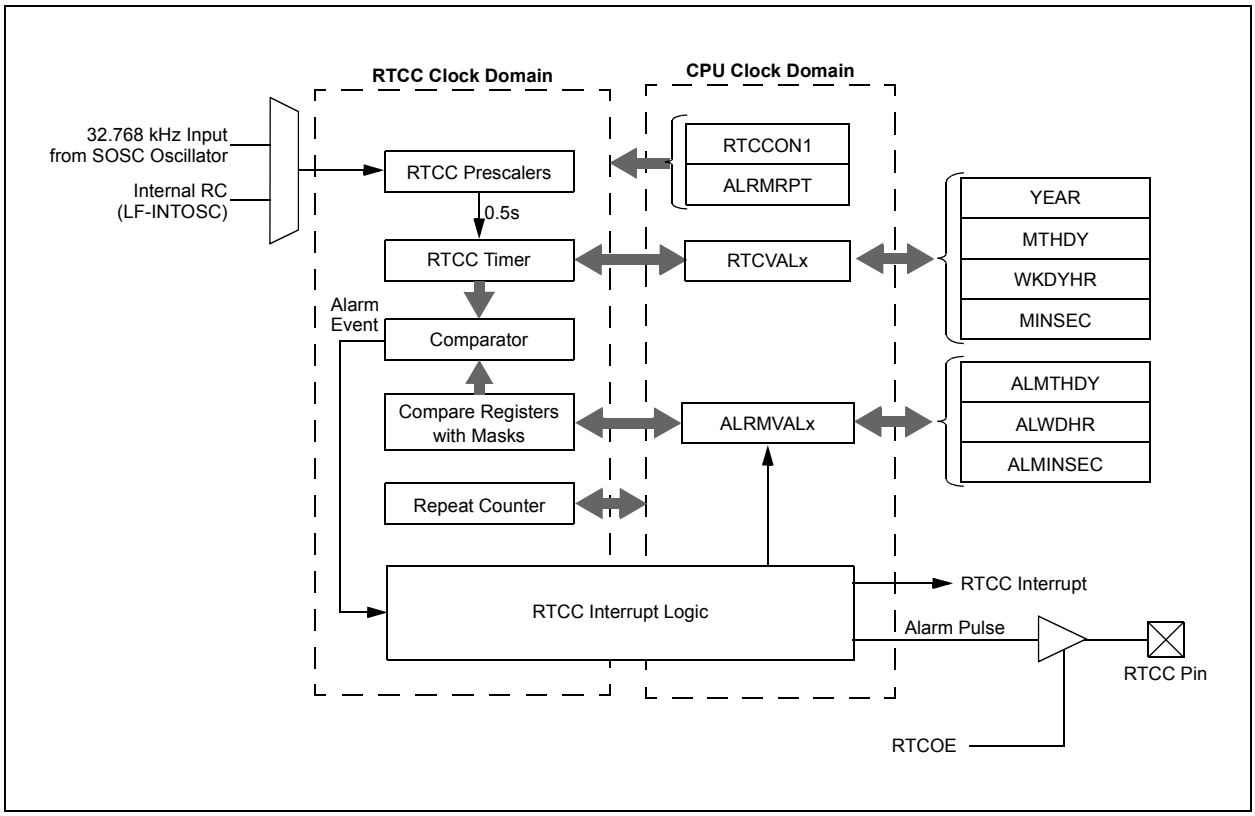
- Hardware Real-Time Clock and Calendar (RTCC)
- Provides hours, minutes and seconds using 24- hour format
- Visibility of one-half second period
- Provides calendar – weekday, date, month and year
- Alarm configurable for half a second, one second, 10 seconds, one minute, 10 minutes, one hour, one day, one week or one month
- Alarm repeat with decrementing counter
- Alarm with indefinite repeat – chime
- Year 2000 to 2099 leap year correction
- BCD format for smaller software overhead
- Optimized for long term battery operation
- Fractional second synchronization
- Multiple clock sources
  - SOSC
  - LPRC
  - 50 Hz
  - 60 Hz
- User calibration of the 32.768 kHz clock crystal frequency with periodic auto-adjust
- Calibration to within ±2.64 seconds error per month
- Calibrates up to 260 ppm of crystal error

The RTCC module is intended for applications where accurate time must be maintained for an extended period with minimum to no intervention from the CPU. The module is optimized for low-power usage in order to provide extended battery life, while keeping track of time.

The module is a 100-year clock and calendar with automatic leap year detection. The range of the clock is from 00:00:00 (midnight) on January 1, 2000 to 23:59:59 on December 31, 2099.

Hours are measured in 24-hour (military time) format. The clock provides a granularity of one second with half-second visibility to the user.

FIGURE 17-1: RTCC BLOCK DIAGRAM



17.1 RTCC MODULE REGISTERS

The RTCC module registers are divided into the following categories:

RTCC Control Registers

- RTCCON1
- RTCCON2
- RTCCAL
- PADCFG
- ALRMCFG
- ALRMRPT

RTCC Value Registers

- RTCVALH
- RTCVALL
  Both registers access the following registers:
  - YEAR
  - MONTH
  - DAY
  - WEEKDAY
  - HOUR
  - MINUTE
  - SECOND

Alarm Value Registers

- ALRMVALH
- ALRMVALL
  Both registers access the following registers:
  - ALRMMNTH
  - ALRMDAY
  - ALRMWD
  - ALRMHR
  - ALRMMIN
  - ALRMSEC

Note: The RTCVALH and RTCVALL registers can be accessed through RTCRPT<1:0> (RTCCON1<1:0>). ALRMVALH and ALRMVALL can be accessed through ALRMPTR<1:0> (ALRMCFG<1:0>).

17.1.1 RTCC CONTROL REGISTERS

REGISTER 17-1: RTCCON1: RTCC CONFIGURATION REGISTER 1(1)

R/W-0	U-0	R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0
RTCEN(2)	—	RTCWREN(4)	RTCSYNC	HALFSEC(3)	RTCOE	RTCPTR1	RTCPTR0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **RTCEN:** RTCC Enable bit(2)
1 = RTCC module is enabled
0 = RTCC module is disabled

bit 6 **Unimplemented:** Read as '0'

bit 5 **RTCWREN:** RTCC Value Registers Write Enable bit(4)
1 = RTCVALH, RTCVALL and RTCCON2 registers can be written to by the user
0 = RTCVALH, RTCVALL and RTCCON2 registers are locked out from being written to by the user

bit 4 **RTCSYNC:** RTCC Value Registers Read Synchronization bit
1 = RTCVALH, RTCVALL and ALRMRPT registers can change while reading if a rollover ripple results in an invalid data read. If the register is read twice and results in the same data, the data can be assumed to be valid.
0 = RTCVALH, RTCVALL or ALRMRPT registers can be read without concern over a rollover ripple

bit 3 **HALFSEC:** Half-Second Status bit(3)
1 = Second half period of a second
0 = First half period of a second

bit 2 **RTCOE:** RTCC Output Enable bit
1 = RTCC clock output is enabled
0 = RTCC clock output is disabled

bit 1-0 **RTCPTR<1:0>:** RTCC Value Register Window Pointer bits
Points to the corresponding RTCC Value registers when reading the RTCVALH and RTCVALL registers. The RTCPTR<1:0> value decrements on every read or write of RTCVALH<15:8> until it reaches '00'.
RTCVALH:
00 = Minutes
01 = Weekday
10 = Month
11 = Reserved
RTCVALL:
00 = Seconds
01 = Hours
10 = Day
11 = Year

Note 1: The RTCCON1 register is only affected by a POR.
2: A write to the RTCEN bit is only allowed when RTCWREN = 1.
3: This bit is read-only; it is cleared to '0' on a write to the lower half of the MINSEC register.
4: RTCWREN can only be written with the unlock sequence (see Example 17-1).

REGISTER 17-2: RTCCAL: RTCC CALIBRATION REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CAL<7:0>:** RTC Drift Calibration bits

`01111111` = Maximum positive adjustment; adds 508 RTC clock pulses every minute
.
.
.
`00000001` = Minimum positive adjustment; adds four RTC clock pulses every minute
`00000000` = No adjustment
`11111111` = Minimum negative adjustment; subtracts four RTC clock pulses every minute
.
.
.
`10000000` = Maximum negative adjustment; subtracts 512 RTC clock pulses every minute

Register 17-3: RTCCON2: RTC CONFIGURATION REGISTER 2(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PWCEN(1)	PWCPOL(1)	PWCCPRE(1)	PWCSPRE(1)	RTCCLKSEL1	RTCCLKSEL0	RTCSECSEL1	RTCSECSEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **PWCEN:** Power Control Enable bit(1)
- 1 = Power control is enabled
- 0 = Power control is disabled

bit 6 **PWCPOL:** Power Control Polarity bit(1)
- 1 = Power control output is active-high
- 0 = Power control output is active-low

bit 5 **PWCCPRE:** Power Control/Stability Prescaler bits(1)
- 1 = PWC stability window clock is divide-by-2 of source RTCC clock
- 0 = PWC stability window clock is divide-by-1 of source RTCC clock

bit 4 **PWCSPRE:** Power Control Sample Prescaler bits(1)
- 01 =PWC sample window clock is divide-by-2 of source RTCC clock
- 00 =PWC sample window clock is divide-by-1 of source RTCC clock

bit 3-2 **RTCCLKSEL<1:0>:** RTCC Clock Select bits
Determines the source of the internal RTCC clock, which is used for all RTCC timer operations.
- 11 =60 Hz Powerline
- 10 =50 Hz Powerline
- 01 =INTOSC
- 00 =SOSC

bit 1-0 **RTSECSEL<1:0>:** RTCC Seconds Clock Output Select bit
- 11 =Power control
- 10 =RTCC source clock is selected for the RTCC pin (pin can be LF-INTOSC or SOSC, depending on the RTCOSC (CONFIG3L<1>) bit setting
- 01 =RTCC seconds clock is selected for the RTCC pin
- 00 =RTCC alarm pulse is selected for the RTCC pin

Note 1: The RTCCON2 register is only affected by a POR.

REGISTER 17-4: ALRMCFG: ALARM CONFIGURATION REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ALRMEN:** Alarm Enable bit

1 = Alarm is enabled (cleared automatically after an alarm event whenever ARPT<7:0> = 00h and CHIME = 0)
0 = Alarm is disabled

bit 6 **CHIME:** Chime Enable bit

1 = Chime is enabled; ARPT<7:0> bits are allowed to roll over from 00h to FFh
0 = Chime is disabled; ARPT<7:0> bits stop once they reach 00h

bit 5-2 **AMASK<3:0>:** Alarm Mask Configuration bits

0000 = Every half second
0001 = Every second
0010 = Every 10 seconds
0011 = Every minute
0100 = Every 10 minutes
0101 = Every hour
0110 = Once a day
0111 = Once a week
1000 = Once a month
1001 = Once a year (except when configured for February 29th, once every four years)
101x = Reserved – Do not use
11xx = Reserved – Do not use

bit 1-0 **ALRMPTR<1:0>:** Alarm Value Register Window Pointer bits

Points to the corresponding Alarm Value registers when reading the ALRMVALH and ALRMVALL registers. The ALRMPTR<1:0> value decrements on every read or write of ALRMVALH until it reaches '00'.

ALRMVALH:
00 = ALRMMIN
01 = ALRMWD
10 = ALRMMNTH
11 = Unimplemented

ALRMVALL:
00 = ALRMSEC
01 = ALRMHR
10 = ALRMDAY
11 = Unimplemented

REGISTER 17-5: ALRMRPT: ALARM REPEAT REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **ARPT<7:0>:** Alarm Repeat Counter Value bits

11111111 = Alarm will repeat 255 more times

.

.

.

00000000 = Alarm will not repeat

The counter decrements on any alarm event. The counter is prevented from rolling over from 00h to FFh unless CHIME = 1.

17.1.2 RTCVALH AND RTCVALL REGISTER MAPPINGS

The registers described in this section are the targets or sources for writes or reads to the RTCVALH and RTCVALL in the order they will appear when accessed through the RTCCON1<RTCPTR> pointer. For more information on RTCVAL register mapping, see **Section 17.2.8 "Register Mapping"**.

REGISTER 17-6: RESERVED REGISTER (RTCVALH when RTCPTR<1:0> = 11)

U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
—	—	—	—	—	—	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **Unimplemented:** Read as '0'

Note:	A read or write to the RTCVALH register when RTCPTR<1:0> = 11 is necessary to automatically decrement RTCPTR.

REGISTER 17-7: YEAR: YEAR VALUE REGISTER(1) (RTCVALL when RTCPTR<1:0> = 11)

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
YRTEN3	YRTEN2	YRTEN1	YRTEN0	YRONE3	YRONE2	YRONE1	YRONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **YRTEN<3:0>:** Binary Coded Decimal Value of Year's Tens Digit bits
Contains a value from 0 to 9.

bit 3-0 **YRONE<3:0>:** Binary Coded Decimal Value of Year's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to the YEAR register is only allowed when RTCWREN = 1.

REGISTER 17-8: MONTH: MONTH VALUE REGISTER(1) (RTCVALH when RTCPTR<1:0> = 10)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **MTHTEN0:** Binary Coded Decimal Value of Month's Tens Digit bit
Contains a value of 0 or 1.

bit 3-0 **MTHONE<3:0>:** Binary Coded Decimal Value of Month's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-9: DAY: DAY VALUE REGISTER(1) (RTCVALL when RTCPTR<1:0> = 10)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **DAYTEN<1:0>:** Binary Coded Decimal value of Day's Tens Digit bits
Contains a value from 0 to 3.

bit 3-0 **DAYONE<3:0>:** Binary Coded Decimal Value of Day's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-10: WEEKDAY: WEEKDAY VALUE REGISTER(1) (RTCVALH when RTCPTR<1:0> = 01)

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **WDAY<2:0>:** Binary Coded Decimal Value of Weekday Digit bits
Contains a value from 0 to 6.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-11: HOUR: HOUR VALUE REGISTER(1) (RTCVALL when RTCPTR<1:0> = 01)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **HRTEN<1:0>:** Binary Coded Decimal Value of Hour's Tens Digit bits
Contains a value from 0 to 2.

bit 3-0 **HRONE<3:0>:** Binary Coded Decimal Value of Hour's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-12: MINUTE: MINUTE VALUE REGISTER (RTCVALH when RTCPTR<1:0> = 00)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-4 **MINTEN<2:0>:** Binary Coded Decimal Value of Minute's Tens Digit bits
Contains a value from 0 to 5.

bit 3-0 **MINONE<3:0>:** Binary Coded Decimal Value of Minute's Ones Digit bits
Contains a value from 0 to 9.

REGISTER 17-13: SECOND: SECOND VALUE REGISTER (RTCVALL when RTCPTR<1:0> = 00)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-4 **SECTEN<2:0>:** Binary Coded Decimal Value of Second's Tens Digit bits
Contains a value from 0 to 5.

bit 3-0 **SECONE<3:0>:** Binary Coded Decimal Value of Second's Ones Digit bits
Contains a value from 0 to 9.

17.1.3 ALRMVALH AND ALRMVALL REGISTER MAPPINGS

The registers described in this section are the targets or sources for writes or reads to the ALRMVALH and ALRMVALL in the order they will appear when accessed through the ALRMCFG<ALRMPTR> pointer. For more information on ALRMVAL register mapping, see **Section 17.2.8 "Register Mapping"**.

REGISTER 17-14: ALRMMNTH: ALARM MONTH VALUE REGISTER(1) (ALRMVALH when ALRMPTR<1:0> = 10)

U-0	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	—	MTHTEN0	MTHONE3	MTHONE2	MTHONE1	MTHONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **MTHTEN0:** Binary Coded Decimal Value of Month's Tens Digit bits
Contains a value of 0 or 1.

bit 3-0 **MTHONE<3:0>:** Binary Coded Decimal Value of Month's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-15: ALRMDAY: ALARM DAY VALUE REGISTER(1) (ALRMVALL when ALRMPTR<1:0> = 10)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	DAYTEN1	DAYTEN0	DAYONE3	DAYONE2	DAYONE1	DAYONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **DAYTEN<1:0>:** Binary Coded Decimal Value of Day's Tens Digit bits
Contains a value from 0 to 3.

bit 3-0 **DAYONE<3:0>:** Binary Coded Decimal Value of Day's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-16: ALRMWD: ALARM WEEKDAY VALUE REGISTER(1) (ALRMVALH WHEN ALRMPTR<1:0> = 01)

U-0	U-0	U-0	U-0	U-0	R/W-x	R/W-x	R/W-x
—	—	—	—	—	WDAY2	WDAY1	WDAY0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **WDAY<2:0>:** Binary Coded Decimal Value of Weekday Digit bits
Contains a value from 0 to 6.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-17: ALRMHR: ALARM HOURS VALUE REGISTER(1) (ALRMVALL when ALRMPTR<1:0> = 01)

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	HRTEN1	HRTEN0	HRONE3	HRONE2	HRONE1	HRONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **HRTEN<1:0>:** Binary Coded Decimal Value of Hour's Tens Digit bits
Contains a value from 0 to 2.

bit 3-0 **HRONE<3:0>:** Binary Coded Decimal Value of Hour's Ones Digit bits
Contains a value from 0 to 9.

Note 1: A write to this register is only allowed when RTCWREN = 1.

REGISTER 17-18: ALRMMIN: ALARM MINUTES VALUE REGISTER (ALRMVALH when ALRMPTR<1:0> = 00)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	MINTEN2	MINTEN1	MINTEN0	MINONE3	MINONE2	MINONE1	MINONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-4 **MINTEN<2:0>:** Binary Coded Decimal Value of Minute's Tens Digit bits
Contains a value from 0 to 5.

bit 3-0 **MINONE<3:0>:** Binary Coded Decimal Value of Minute's Ones Digit bits
Contains a value from 0 to 9.

REGISTER 17-19: ALRMSEC: ALARM SECONDS VALUE REGISTER (ALRMVALL when ALRMPTR<1:0> = 00)

U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	SECTEN2	SECTEN1	SECTEN0	SECONE3	SECONE2	SECONE1	SECONE0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-4 **SECTEN<2:0>:** Binary Coded Decimal Value of Second's Tens Digit bits
Contains a value from 0 to 5.

bit 3-0 **SECONE<3:0>:** Binary Coded Decimal Value of Second's Ones Digit bits
Contains a value from 0 to 9.

17.1.4 RTCEN BIT WRITE

RTCWREN (RTCCON1<5>) must be set before a write to RTCEN can take place. Any write to the RTCEN bit, while RTCWREN = 0, will be ignored.

Like the RTCEN bit, the RTCVALH and RTCVALL registers can only be written to when RTCWREN = 1. A write to these registers, while RTCWREN = 0, will be ignored.

17.2 Operation

17.2.1 REGISTER INTERFACE

The register interface for the RTCC and alarm values is implemented using the Binary Coded Decimal (BCD) format. This simplifies the firmware when using the module, as each of the digits is contained within its own 4-bit value (see Figure 17-2 and Figure 17-3).

FIGURE 17-2: TIMER DIGIT FORMAT

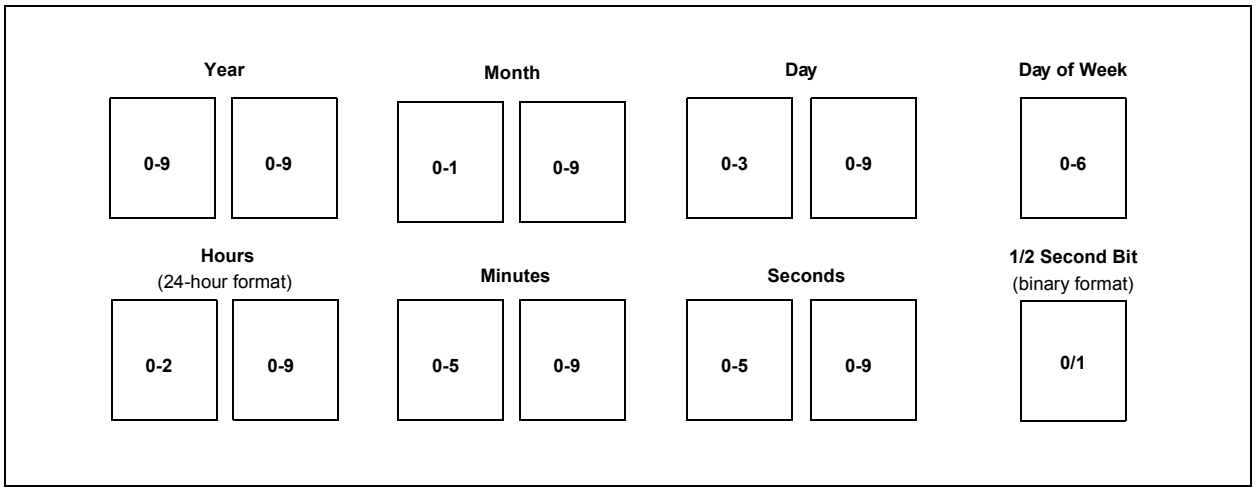
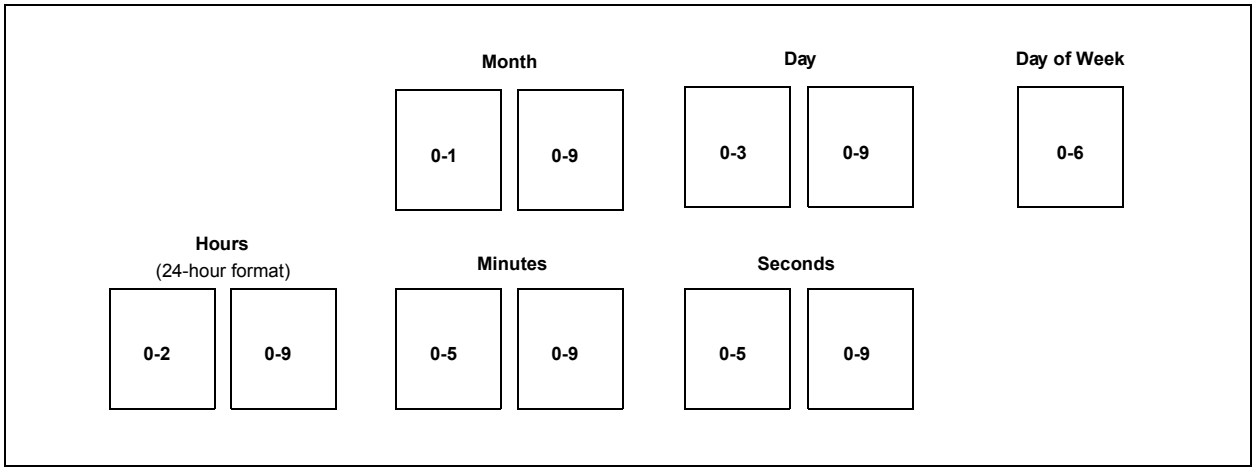


FIGURE 17-3: ALARM DIGIT FORMAT

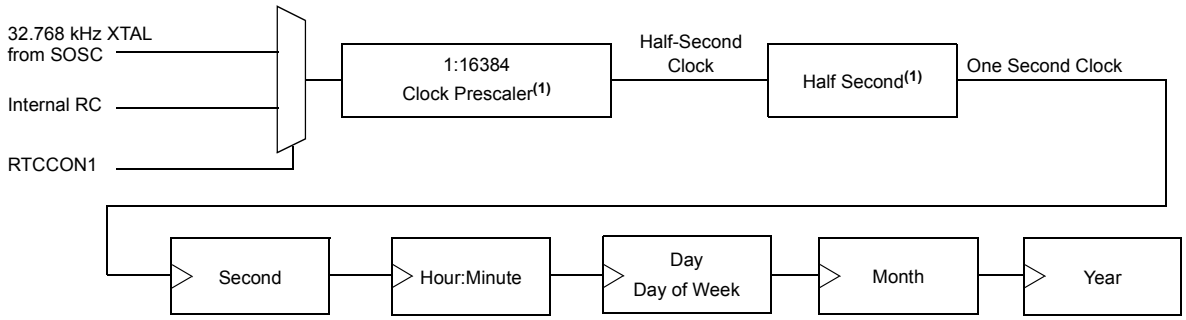


17.2.2 CLOCK SOURCE

As mentioned earlier, the RTCC module is intended to be clocked by an external Real-Time Clock (RTC) crystal, oscillating at 32.768 kHz, but an internal oscillator can be used. The RTCC clock selection is decided by the RTCOSC bit (CONFIG3L<0>).

Calibration of the crystal can be done through this module to yield an error of 3 seconds or less per month. (For further details, see **Section 17.2.9 "Calibration"**.)

FIGURE 17-4: CLOCK SOURCE MULTIPLEXING



Note 1: Writing to the lower half of the MINSEC register resets all counters, allowing fraction of a second synchronization; clock prescaler is held in Reset when RTCEN = 0.

17.2.2.1 Real-Time Clock Enable

The RTCC module can be clocked by an external, 32.768 kHz crystal (SOSC Oscillator) or the LF-INTOSC Oscillator, which can be selected in CONFIG3L<0>.

If the external clock is used, the SOSC Oscillator should be enabled. If LF-INTOSC is providing the clock, the INTOSC clock can be brought out to the RTCC pin by the RTSECSEL<1:0> bits (RTCCON2<1:0>).

17.2.3 DIGIT CARRY RULES

This section explains which timer values are affected when there is a rollover:

- Time of Day: From 23:59:59 to 00:00:00 with a carry to the Day field
- Month: From 12/31 to 01/01 with a carry to the Year field
- Day of Week: From 6 to 0 with no carry (see Table 17-1)
- Year Carry: From 99 to 00; this also surpasses the use of the RTCC

For the day-to-month rollover schedule, see Table 17-2.

Because the following values are in BCD format, the carry to the upper BCD digit occurs at the count of 10, not 16 (SECONDS, MINUTES, HOURS, WEEKDAY, DAYS and MONTHS).

TABLE 17-1: DAY OF WEEK SCHEDULE

Day of Week	
Sunday	0
Monday	1
Tuesday	2
Wednesday	3
Thursday	4
Friday	5
Saturday	6

TABLE 17-2: DAY TO MONTH ROLLOVER SCHEDULE

Month	Maximum Day Field
01 (January)	31
02 (February)	28 or 29(1)
03 (March)	31
04 (April)	30
05 (May)	31
06 (June)	30
07 (July)	31
08 (August)	31
09 (September)	30
10 (October)	31
11 (November)	30
12 (December)	31

Note 1: See **Section 17.2.4 "Leap Year"**.

17.2.4 LEAP YEAR

Since the year range on the RTCC module is 2000 to 2099, the leap year calculation is determined by any year divisible by four in the above range. Only February is affected in a leap year.

February will have 29 days in a leap year and 28 days in any other year.

17.2.5 GENERAL FUNCTIONALITY

All Timer registers containing a time value of seconds or greater are writable. The user configures the time by writing the required year, month, day, hour, minutes and seconds to the Timer registers, via register pointers. (See **Section 17.2.8 "Register Mapping"**.)

The timer uses the newly written values and proceeds with the count from the required starting point.

The RTCC is enabled by setting the RTCEN bit (RTCCON1<7>). If enabled, while adjusting these registers, the timer still continues to increment. However, any time the MINSEC register is written to, both of the timer prescalers are reset to '0'. This allows fraction of a second synchronization.

The Timer registers are updated in the same cycle as the WRITE instruction's execution by the CPU. The user must ensure that when RTCEN = 1, the updated registers will not be incremented at the same time. This can be accomplished in several ways:

- By checking the RTCSYNC bit (RTCCON1<4>)
- By checking the preceding digits from which a carry can occur
- By updating the registers immediately following the seconds pulse (or an alarm interrupt)

The user has visibility to the half-second field of the counter. This value is read-only and can be reset only by writing to the lower half of the SECONDS register.

17.2.6 SAFETY WINDOW FOR REGISTER READS AND WRITES

The RTCSYNC bit indicates a time window during which the RTCC clock domain registers can be safely read and written without concern about a rollover. When RTCSYNC = 0, the registers can be safely accessed by the CPU.

Whether RTCSYNC = 1 or 0, the user should employ a firmware solution to ensure that the data read did not fall on a rollover boundary, resulting in an invalid or partial read. This firmware solution would consist of reading each register twice and then comparing the two values. If the two values matched, then a rollover did not occur.

17.2.7 WRITE LOCK

In order to perform a write to any of the RTCC Timer registers, the RTCWREN bit (RTCCON1<5>) must be set.

To avoid accidental writes to the RTCC Timer register, it is recommended that the RTCWREN bit (RTCCON1<5>) be kept clear when not writing to the register. For the RTCWREN bit to be set, there is only one instruction cycle time window allowed between the 55h/AA sequence and the setting of RTCWREN. For that reason, it is recommended that users follow the code example in Example 17-1.

EXAMPLE 17-1: SETTING THE RTCWREN BIT

```
movlw   0x55
movwf   EECON2
movlw   0xAA
movwf   EECON2
bsf     RTCCON1,RTCWREN
```

17.2.8 REGISTER MAPPING

To limit the register interface, the RTCC Timer and Alarm Timer registers are accessed through corresponding register pointers. The RTCC Value register window (RTCVALH and RTCVALL) uses the RTCPTRx bits (RTCCON1<1:0>) to select the required Timer register pair.

By reading or writing to the RTCVALH register, the RTCC Pointer value (RTCPTR<1:0>) decrements by '1' until it reaches '00'. When '00' is reached, the MINUTES and SECONDS value is accessible through RTCVALH and RTCVALL until the pointer value is manually changed.

TABLE 17-3: RTCVALH AND RTCVALL REGISTER MAPPING

RTCPTR<1:0>	RTCC Value Register Window	
	RTCVALH	RTCVALL
00	MINUTES	SECONDS
01	WEEKDAY	HOURS
10	MONTH	DAY
11	—	YEAR

The Alarm Value register windows (ALRMVALH and ALRMVALL) use the ALRMPTR bits (ALRMCFG<1:0>) to select the desired Alarm register pair.

By reading or writing to the ALRMVALH register, the Alarm Pointer value, ALRMPTR<1:0>, decrements by '1' until it reaches '00'. When it reaches '00', the ALRMMIN and ALRMSEC values are accessible through ALRMVALH and ALRMVALL until the pointer value is manually changed.

TABLE 17-4: ALRMVAL REGISTER MAPPING

ALRMPTR<1:0>	Alarm Value Register Window	
	ALRMVALH	ALRMVALL
00	ALRMMIN	ALRMSEC
01	ALRMWD	ALRMHR
10	ALRMMNTH	ALRMDAY
11	—	—

17.2.9 CALIBRATION

The real-time crystal input can be calibrated using the periodic auto-adjust feature. When properly calibrated, the RTCC can provide an error of less than three seconds per month.

To perform this calibration, find the number of error clock pulses and store the value into the lower half of the RTCCAL register. The 8-bit signed value, loaded into RTCCAL, is multiplied by four and will either be added or subtracted from the RTCC timer, once every minute.

To calibrate the RTCC module:

1. Use another timer resource on the device to find the error of the 32.768 kHz crystal.
2. Convert the number of error clock pulses per minute (see Equation 17-1).

EQUATION 17-1: CONVERTING ERROR CLOCK PULSES

(Ideal Frequency (32,758) – Measured Frequency) * 60 = Error Clocks per Minute

   - If the oscillator is *faster* than ideal (negative result from Step 2), the RCFGCALL register value needs to be negative. This causes the specified number of clock pulses to be subtracted from the timer counter once every minute.
   - If the oscillator is *slower* than ideal (positive result from Step 2), the RCFGCALL register value needs to be positive. This causes the specified number of clock pulses to be added to the timer counter once every minute.
3. Load the RTCCAL register with the correct value.

Writes to the RTCCAL register should occur only when the timer is turned off or immediately after the rising edge of the seconds pulse.

Note:	In determining the crystal's error value, it is the user's responsibility to include the crystal's initial error from drift due to temperature or crystal aging.

17.3 Alarm

The Alarm features and characteristics are:

- Configurable from half a second to one year
- Enabled using the ALRMEN bit (ALRMCFG<7>, Register 17-4)
- Offers one-time and repeat alarm options

17.3.1 CONFIGURING THE ALARM

The alarm feature is enabled using the ALRMEN bit.

This bit is cleared when an alarm is issued. The bit will not be cleared if the CHIME bit = 1 or if ALRMRPT ≠ 0.

The interval selection of the alarm is configured through the ALRMCFG bits (AMASK<3:0>); see Figure 17-5. These bits determine which and how many digits of the alarm must match the clock value for the alarm to occur.

The alarm can also be configured to repeat based on a preconfigured interval. The number of times this occurs, after the alarm is enabled, is stored in the ALRMRPT register.

Note:	While the alarm is enabled (ALRMEN = 1), changing any of the registers, other than the RTCCAL, ALRMCFG and ALRMRPT registers and the CHIME bit, can result in a false alarm event leading to a false alarm interrupt. To avoid this, only change the timer and alarm values while the alarm is disabled (ALRMEN = 0). It is recommended that the ALRMCFG and ALRMRPT registers and CHIME bit be changed when RTCSYNC = 0.

FIGURE 17-5: ALARM MASK SETTINGS

Alarm Mask Setting AMASK<3:0>	Day of the Week	Month	Day	Hours	Minutes	Seconds
0000 – Every half second 0001 – Every second						
0010 – Every 10 seconds						s
0011 – Every minute						s s
0100 – Every 10 minutes					m	s s
0101 – Every hour					m m	s s
0110 – Every day				h h	m m	s s
0111 – Every week	d			h h	m m	s s
1000 – Every month			d d	h h	m m	s s
1001 – Every year(1)		m m	d d	h h	m m	s s

Note 1: Annually, except when configured for February 29.

When ALRMCFG = 00 and the CHIME bit = 0 (ALRMCFG<6>), the repeat function is disabled and only a single alarm will occur. The alarm can be repeated up to 255 times by loading the ALRMRPT register with FFh.

After each alarm is issued, the ALRMRPT register is decremented by one. Once the register has reached '00', the alarm will be issued one last time.

After the alarm is issued a last time, the ALRMEN bit is cleared automatically and the alarm turned off. Indefinite repetition of the alarm can occur if the CHIME bit = 1.

When CHIME = 1, the alarm is not disabled when the ALRMRPT register reaches '00', but it rolls over to FF and continues counting indefinitely.

17.3.2 ALARM INTERRUPT

At every alarm event, an interrupt is generated. Additionally, an alarm pulse output is provided that operates at half the frequency of the alarm.

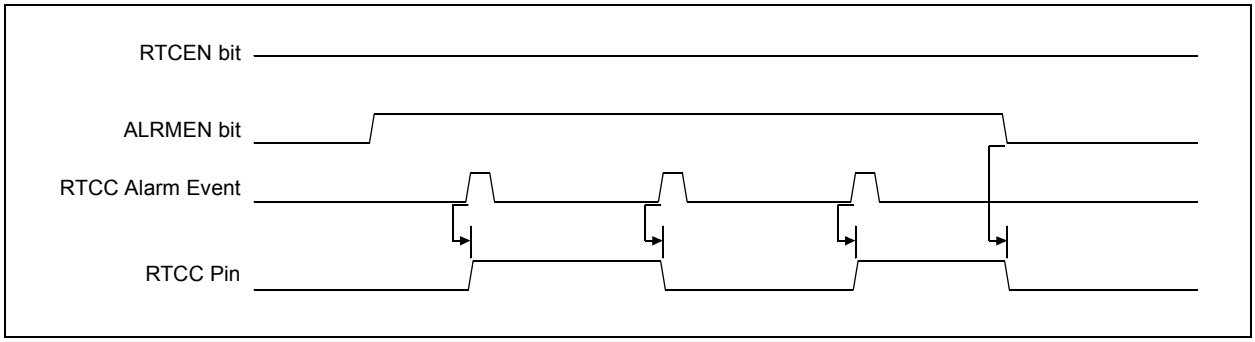
The alarm pulse output is completely synchronous with the RTCC clock and can be used as a trigger clock to other peripherals. This output is available on the RTCC pin. The output pulse is a clock with a 50% duty cycle and a frequency half that of the alarm event (see Figure 17-6).

The RTCC pin can also output the seconds clock. The user can select between the alarm pulse, generated by the RTCC module, or the seconds clock output.

The RTSECSEL<1:0> bits (RTCCON2<1:0>) select between these two outputs:

- Alarm pulse – RTSECSEL<1:0> = 00
- Seconds clock – RTSECSEL<1:0> = 01

FIGURE 17-6: TIMER PULSE GENERATION



17.4 Sleep Mode

The timer and alarm continue to operate while in Sleep mode. The operation of the alarm is not affected by Sleep, as an alarm event can always wake-up the CPU.

The Idle mode does not affect the operation of the timer or alarm.

17.5 Reset

17.5.1 DEVICE RESET

When a device Reset occurs, the ALRMRPT register is forced to its Reset state, causing the alarm to be disabled (if enabled prior to the Reset). If the RTCC was enabled, it will continue to operate when a basic device Reset occurs.

17.5.2 POWER-ON RESET (POR)

The RTCCON1 and ALRMRPT registers are reset only on a POR. Once the device exits the POR state, the clock registers should be reloaded with the desired values.

The timer prescaler values can be reset only by writing to the SECONDS register. No device Reset can affect the prescalers.

17.6 Register Maps

Table 17-5, Table 17-6 and Table 17-7 summarize the registers associated with the RTCC module.

TABLE 17-5: RTCC CONTROL REGISTERS

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTCCON1	RTCEN	—	RTCWREN	RTCSYNC	HALFSEC	RTCOE	RTCPTR1	RTCPTR0
RTCCAL	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0
RTCCON2	PWCEN	PWCPOL	PWCCPRE	PWCSPRE	RTCCLKSEL1	RTCCLKSEL0	RTCSECSEL1	RTCSECSEL
ALRMCFG	ALRMEN	CHIME	AMASK3	AMASK2	AMASK1	AMASK0	ALRMPTR1	ALRMPTR0
ALRMRPT	ARPT7	ARPT6	ARPT5	ARPT4	ARPT3	ARPT2	ARPT1	ARPT0
PMD3	DSMMD	CTMUMD	ADCMD	RTCCMD	LCDMD	PSPMD	REFO1MD	REFO2MD

Legend: — = unimplemented, read as '0'. Reset values are shown in hexadecimal for 80-pin devices.

TABLE 17-6: RTCC VALUE REGISTERS

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTCVALH	RTCC Value High Register Window based on RTCPTR<1:0>							
RTCVALL	RTCC Value Low Register Window based on RTCPTR<1:0>							

TABLE 17-7: ALARM VALUE REGISTERS

File Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ALRMVALH	Alarm Value High Register Window based on ALRMPTR<1:0>							
ALRMVALL	Alarm Value Low Register Window based on ALRMPTR<1:0>							

18.0 ENHANCED CAPTURE/ COMPARE/PWM (ECCP) MODULE

PIC18FXXJ94 devices have three Enhanced Capture/ Compare/PWM (ECCP) modules: ECCP1, ECCP2 and ECCP3. These modules contain a 16-bit register, which can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register. These ECCP modules are upward compatible with CCP

Note:	Throughout this section, generic references are used for register and bit names that are the same, except for an 'x' variable that indicates the item's association with the CCP1, CCP2 or CCP3 module. For example, the control register is named CCPxCON and refers to CCP1CON, CCP2CON and CCP3CON.

ECCP1, ECCP2 and ECCP3 are implemented as standard CCP modules with enhanced PWM capabilities. These include:

- Provision for two or four output channels
- Output Steering modes
- Programmable polarity
- Programmable dead-band control
- Automatic shutdown and restart

The enhanced features are discussed in detail in **Section 18.4 "PWM (Enhanced Mode)"**.

The ECCP1, ECCP2 and ECCP3 modules use the ECCP Control registers, CCP1CON, CCP2CON and CCP3CON. The control registers, CCP4CON through CCP10CON, are for the modules, CCP4 through CCP10.

REGISTER 18-1: CCPxCON: ENHANCED CAPTURE/COMPARE/PWM x CONTROL

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxM1	PxM0	DCxB1	DCxB0	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **PxM<1:0>:** Enhanced PWM Output Configuration bits

If CCPxM<3:2> = 00, 01, 10:

xx =PxA is assigned as the capture/compare input/output; PxB, PxC and PxD are assigned as port pins

If CCPxM<3:2> = 11:

00 =Single output: PxA, PxB, PxC and PxD are controlled by steering (see **Section 18.4.7 "Pulse Steering Mode"**)

01 =Full-bridge output forward: PxD is modulated; PxA is active; PxB, PxC are inactive

10 =Half-bridge output: PxA, PxB are modulated with dead-band control; PxC and PxD are assigned as port pins

11 =Full-bridge output reverse: PxB is modulated; PxC is active; PxA and PxD are inactive

bit 5-4 **DCxB<1:0>:** PWM Duty Cycle bit

Capture mode:

Unused.

Compare mode:

Unused.

PWM mode:

These bits are the two LSbs of the 10-bit PWM duty cycle. The eight MSbs of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM<3:0>**: CCPx Mode Select bits

0000 = Capture/Compare/PWM off (resets ECCPx module)
0001 = Reserved
0010 = Compare mode: Toggle output on match
0011 = Reserved
0100 = Capture mode: Every falling edge
0101 = Capture mode: Every rising edge
0110 = Capture mode: Every fourth rising edge
0111 = Capture mode: Every 16th rising edge
1000 = Compare mode: Initialize ECCPx pin low, set output on compare match (set CCPxIF)
1001 = Compare mode: Initialize ECCPx pin high, clear output on compare match (set CCPxIF)
1010 = Compare mode: Generate software interrupt only, ECCPx pin reverts to I/O state
1011 = Compare mode: Trigger special event (ECCPx resets TMR1 or TMR3, starts A/D conversion, sets CCPxIF bit)
1100 = PWM mode: PxA and PxC are active-high; PxB and PxD are active-high
1101 = PWM mode: PxA and PxC are active-high; PxB and PxD are active-low
1110 = PWM mode: PxA and PxC are active-low; PxB and PxD are active-high
1111 = PWM mode: PxA and PxC are active-low; PxB and PxD are active-low

REGISTER 18-2: CCPTMRS0: CCP TIMER SELECT 0 REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C3TSEL1	C3TSEL0	C2TSEL2	C2TSEL1	C2TSEL0	C1TSEL2	C1TSEL1	C1TSEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **C3TSEL<1:0>:** CCP3 Timer Selection bits
00 = CCP3 is based off of TMR1/TMR2
01 = CCP3 is based off of TMR3/TMR4
10 = CCP3 is based off of TMR3/TMR6
11 = CCP3 is based off of TMR3/TMR8

bit 5-3 **C2TSEL<2:0>:** CCP2 Timer Selection bits
000 = CCP2 is based off of TMR1/TMR2
001 = CCP2 is based off of TMR3/TMR4
010 = CCP2 is based off of TMR3/TMR6
011 = CCP2 is based off of TMR3/TMR8
100 = Reserved; do not use
101 = Reserved; do not use
110 = Reserved; do not use
111 = Reserved; do not use

bit 2-0 **C1TSEL<2:0>:** CCP1 Timer Selection bits
000 = CCP1 is based off of TMR1/TMR2
001 = CCP1 is based off of TMR3/TMR4
010 = CCP1 is based off of TMR3/TMR6
011 = CCP1 is based off of TMR3/TMR8
100 = Reserved; do not use
101 = Reserved; do not use
110 = Reserved; do not use
111 = Reserved; do not use

In addition to the expanded range of modes available through the CCPxCON, the ECCP modules have three additional registers associated with Enhanced PWM operation, Pulse Steering Control and auto-shutdown features. They are:

- ECCPxDEL – Enhanced PWM x Control
- PSTRxCON – Pulse Steering x Control
- ECCPxAS – Auto-Shutdown x Control

18.1 ECCP Outputs and Configuration

The Enhanced CCP module may have up to four PWM outputs, depending on the selected operating mode.

These outputs, designated as PxA through PxD, are routed through the PPS-Lite module. Therefore, individual functions can be mapped to any of the remappable I/O pins (RPn). The outputs that are active depend on the ECCP operating mode selected. The pin assignments are summarized in Table 18-3.

To configure the I/O pins as PWM outputs, the proper PWM mode must be selected by setting the PxM<1:0> and CCPxM<3:0> bits. The appropriate TRIS direction bits for the port pins must also be set as outputs Table 18-3.

18.1.1 ECCP MODULE AND TIMER RESOURCES

The ECCP modules use Timers, 1, 2, 3, 4, 6 or 8, depending on the mode selected. These timers are available to CCP modules in Capture, Compare or PWM modes, as shown in Table 18-1.

TABLE 18-1: ECCP MODE – TIMER RESOURCE

ECCP Mode	Timer Resource
Capture	Timer1 or Timer3
Compare	Timer1 or Timer3
PWM	Timer2, Timer4, Timer6 or Timer8

The assignment of a particular timer to a module is determined by the timer to ECCP enable bits in the CCPTMRS0 register (Register 18-2). The interactions between the two modules are depicted in Figure 18-1. Capture operations are designed to be used when the timer is configured for Synchronous Counter mode. Capture operations may not work as expected if the associated timer is configured for Asynchronous Counter mode.

18.2 Capture Mode

In Capture mode, the CCPRxH:CCPRxL register pair captures the 16-bit value of the TMR1 or TMR3 registers when an event occurs on the corresponding ECCPx pin. An event is defined as one of the following:

- Every falling edge
- Every rising edge
- Every fourth rising edge
- Every 16th rising edge

The event is selected by the mode select bits, CCPxM<3:0> (CCPxCON<3:0>). When a capture is made, the interrupt request flag bit, CCPxIF, is set (see Table 18-2). The flag must be cleared by software. If another capture occurs before the value in the CCPRxH/L register is read, the old captured value is overwritten by the new captured value.

TABLE 18-2: ECCP1/2/3 INTERRUPT FLAG BITS

ECCP Module	Flag Bit
1	PIR3<1>
2	PIR3<2>
3	PIR4<0>

18.2.1 ECCP PIN CONFIGURATION

In Capture mode, the appropriate ECCPx pin should be configured as an input by setting the corresponding TRIS direction bit.

Note: If the ECCPx pin is configured as an output, a write to the port can cause a capture condition.

18.2.2 TIMER1/2/3/4/5/6/8 MODE SELECTION

The timers that are to be used with the capture feature (Timer1/2/3/4/5/6 or 8) must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation will not work. The timer to be used with each ECCP module is selected in the CCPTMRS0 register (Register 18-2).

18.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCPxIE interrupt enable bit clear to avoid false interrupts. The interrupt flag bit, CCPxIF, should also be cleared following any such change in operating mode.

18.2.4 ECCP PRESCALER

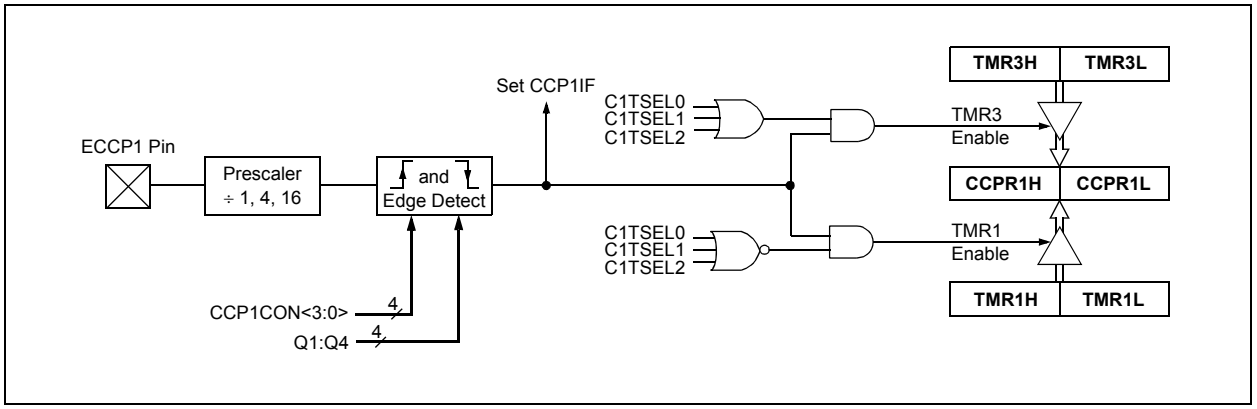
There are four prescaler settings in Capture mode; they are specified as part of the operating mode selected by the mode select bits (CCPxM<3:0>). Whenever the ECCP module is turned off, or Capture mode is disabled, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Also, the prescaler counter will not be cleared; therefore, the first capture may be from a non-zero prescaler. Example 18-1 provides the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 18-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF   CCP1CON      ; Turn ECCP module off
MOVLW  NEW_CAPT_PS  ; Load WREG with the
                    ; new prescaler mode
                    ; value and ECCP ON
MOVWF  CCP1CON      ; Load CCP1CON with
                    ; this value
```

FIGURE 18-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



18.3 Compare Mode

In Compare mode, the 16-bit CCPRx register value is constantly compared against the Timer register pair value selected in the CCPTMR0 register. When a match occurs, the ECCPx pin can be:

- Driven high
- Driven low
- Toggled (high-to-low or low-to-high)
- Unchanged (that is, reflecting the state of the I/O latch)

The action on the pin is based on the value of the mode select bits (CCPxM<3:0>). At the same time, the interrupt flag bit, CCPxIF, is set.

18.3.1 ECCPx PIN CONFIGURATION

Users must configure the ECCPx pin as an output by clearing the appropriate TRIS bit.

Note:	Clearing the CCPxCON register will force the ECCPx compare output latch (depending on device configuration) to the default low level. This is not the PORTx I/O data latch.

18.3.2 TIMER1/2/3/4/5/6/8 MODE SELECTION

Timer1/2/3/4, 6 or 8, must be running in Timer mode or Synchronized Counter mode if the ECCP module is using the compare feature. In Asynchronous Counter mode, the compare operation will not work reliably.

18.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCPxM<3:0> = 1010), the ECCPx pin is not affected; only the CCPxIF interrupt flag is affected.

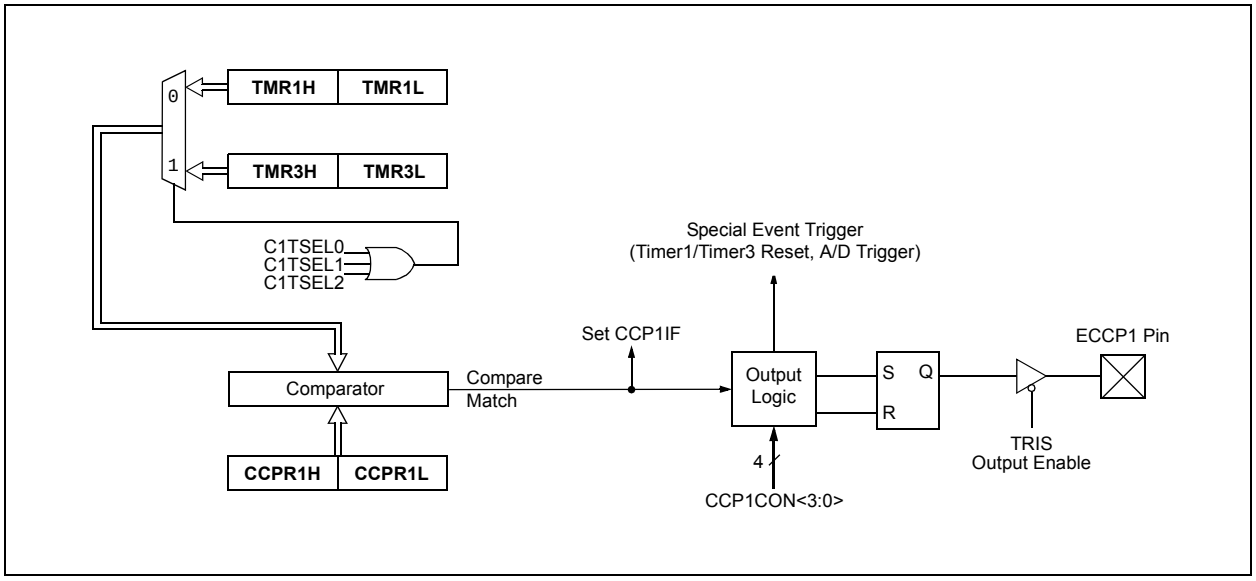
18.3.4 SPECIAL EVENT TRIGGER

The ECCP module is equipped with a Special Event Trigger. This is an internal hardware signal generated in Compare mode to trigger actions by other modules. The Special Event Trigger is enabled by selecting the Compare Special Event Trigger mode (CCPxM<3:0> = 1011).

The Special Event Trigger resets the Timer register pair for whichever timer resource is currently assigned as the module's time base. This allows the CCPRx registers to serve as a programmable period register for either timer.

The Special Event Trigger can also start an A/D conversion. In order to do this, the A/D Converter must already be enabled.

FIGURE 18-2: COMPARE MODE OPERATION BLOCK DIAGRAM



18.4 PWM (Enhanced Mode)

The Enhanced PWM mode can generate a PWM signal on up to four different output pins, with up to 10 bits of resolution. It can do this through four different PWM Output modes:

- Single PWM
- Half-Bridge PWM
- Full-Bridge PWM, Forward mode
- Full-Bridge PWM, Reverse mode

To select an Enhanced PWM mode, the PxM bits of the CCPxCON register must be set appropriately.

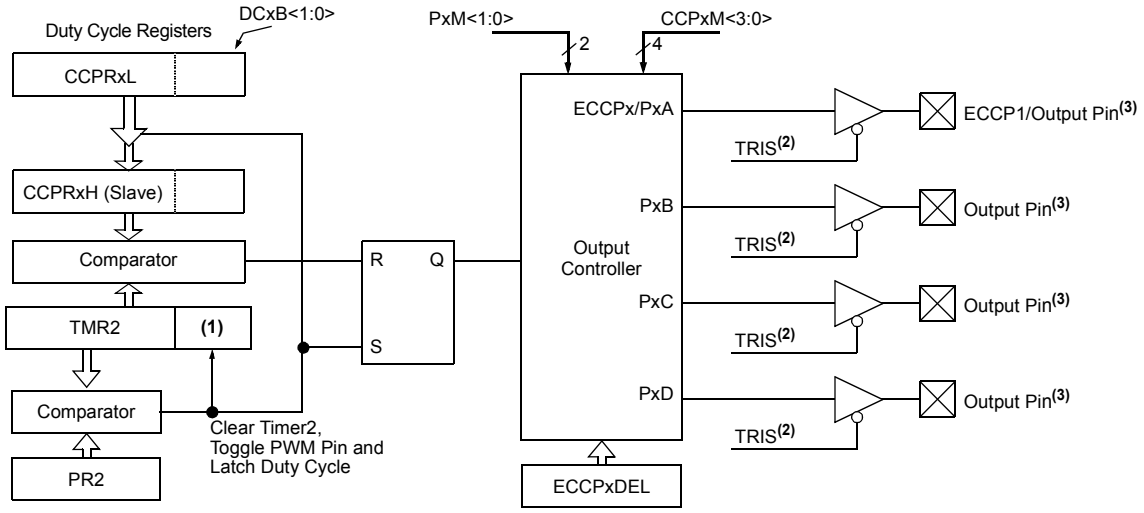
The PWM outputs are multiplexed with I/O pins and are designated: PxA, PxB, PxC and PxD. The polarity of the PWM pins is configurable and is selected by setting the CCPxM bits in the CCPxCON register appropriately.

Table 18-1 provides the pin assignments for each Enhanced PWM mode.

Figure 18-3 provides an example of a simplified block diagram of the Enhanced PWM module.

Note:	To prevent the generation of an incomplete waveform when the PWM is first enabled, the ECCP module waits until the start of a new PWM period before generating a PWM signal.

FIGURE 18-3: EXAMPLE SIMPLIFIED BLOCK DIAGRAM OF THE ENHANCED PWM MODE



Note 1: The 8-bit TMR2 register is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

2: The TRIS register value for each PWM output must be configured appropriately.

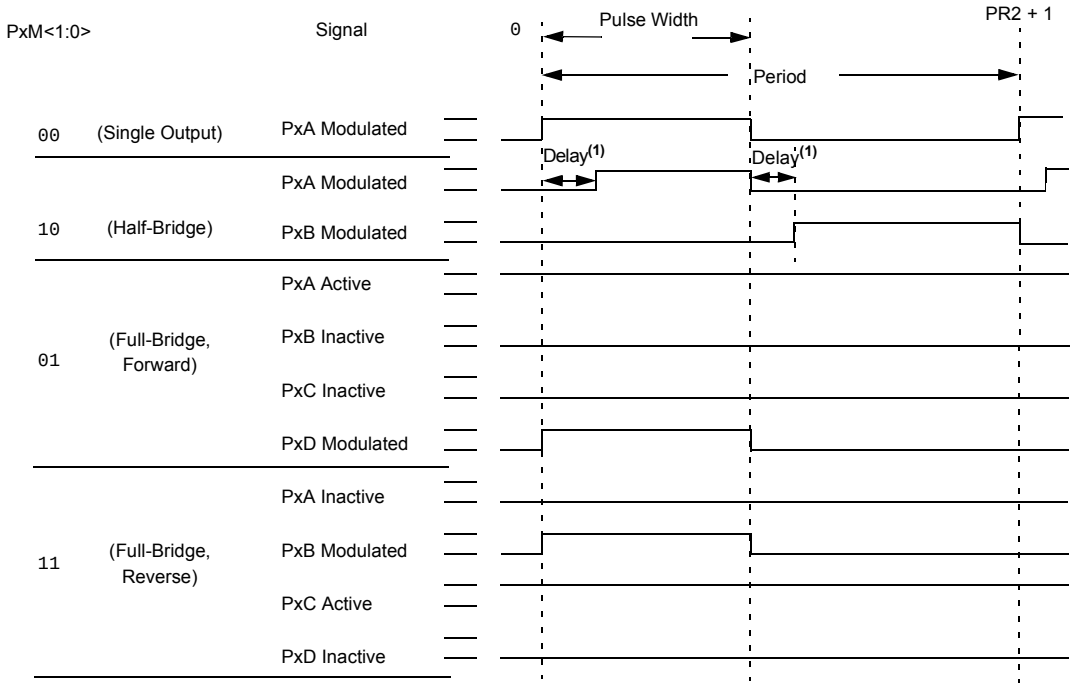
3: Any pin not used by an Enhanced PWM mode is available for alternate pin functions.

TABLE 18-3: EXAMPLE PIN ASSIGNMENTS FOR VARIOUS PWM ENHANCED MODES

ECCP Mode	PxM<1:0>	PxA	PxB	PxC	PxD
Single	00	Yes(1)	Yes(1)	Yes(1)	Yes(1)
Half-Bridge	10	Yes	Yes	No	No
Full-Bridge, Forward	01	Yes	Yes	Yes	Yes
Full-Bridge, Reverse	11	Yes	Yes	Yes	Yes

Note 1: Outputs are enabled by pulse steering in Single mode (see Register 18-5).

FIGURE 18-4: EXAMPLE PWM (ENHANCED MODE) OUTPUT RELATIONSHIPS (ACTIVE-HIGH STATE)

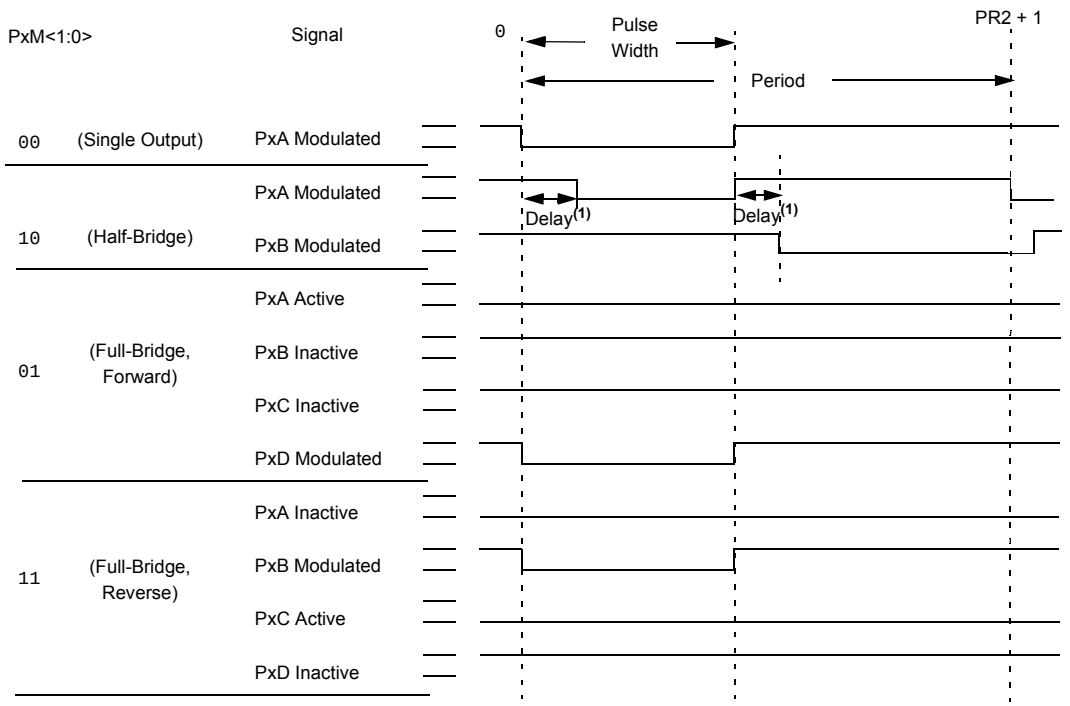


Relationships:

- Period = 4 * Tosc * (PR2 + 1) * (TMR2 Prescale Value)
- Pulse Width = Tosc * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * Tosc * (ECCPxDEL<6:0>)

Note 1: Dead-band delay is programmed using the ECCPxDEL register (**Section 18.4.6 "Programmable Dead-Band Delay Mode"**).

FIGURE 18-5: EXAMPLE ENHANCED PWM OUTPUT RELATIONSHIPS (ACTIVE-LOW STATE)



Relationships:

- Period = 4 * Tosc * (PR2 + 1) * (TMR2 Prescale Value)
- Pulse Width = Tosc * (CCPRxL<7:0>:CCPxCON<5:4>) * (TMR2 Prescale Value)
- Delay = 4 * Tosc * (ECCPxDEL<6:0>)

Note 1: Dead-band delay is programmed using the ECCPxDEL register (**Section 18.4.6 "Programmable Dead-Band Delay Mode"**).

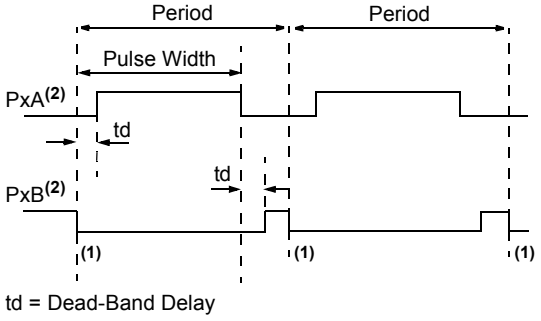
18.4.1 HALF-BRIDGE MODE

In Half-Bridge mode, two pins are used as outputs to drive push-pull loads. The PWM output signal is output on the PxA pin, while the complementary PWM output signal is output on the PxB pin (see Figure 18-6). This mode can be used for half-bridge applications, as shown in Figure 18-7, or for full-bridge applications, where four power switches are being modulated with two PWM signals.

In Half-Bridge mode, the programmable dead-band delay can be used to prevent shoot-through current in half-bridge power devices. The value of the PxDC<6:0> bits of the ECCPxDEL register sets the number of instruction cycles before the output is driven active. If the value is greater than the duty cycle, the corresponding output remains inactive during the entire cycle. For more details on the dead-band delay operations, see **Section 18.4.6 "Programmable Dead-Band Delay Mode"**.

Since the PxA and PxB outputs are multiplexed with the PORT data latches, the associated TRIS bits must be cleared to configure PxA and PxB as outputs.

FIGURE 18-6: EXAMPLE OF HALF-BRIDGE PWM OUTPUT



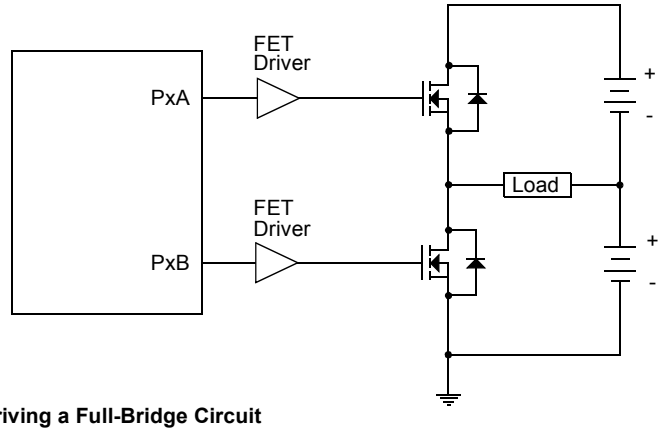
td = Dead-Band Delay

Note 1: At this time, the TMR2 register is equal to the PR2 register.

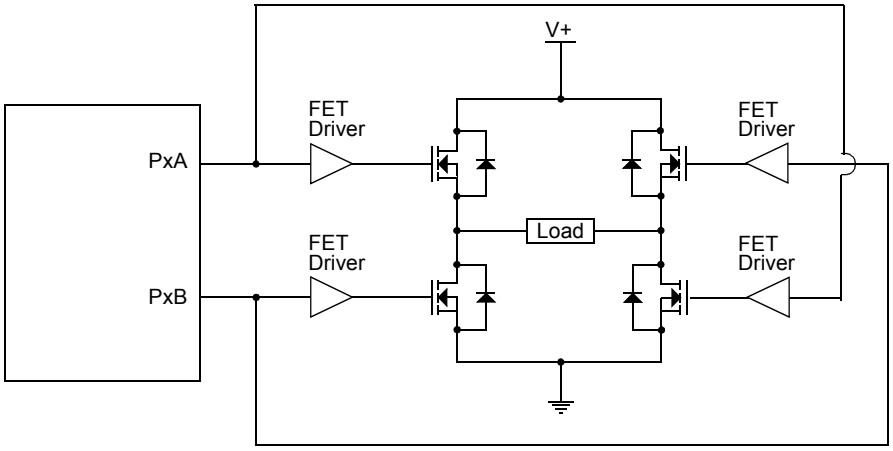
2: Output signals are shown as active-high.

FIGURE 18-7: EXAMPLE OF HALF-BRIDGE APPLICATIONS

Standard Half-Bridge Circuit ("Push-Pull")



Half-Bridge Output Driving a Full-Bridge Circuit



18.4.2 FULL-BRIDGE MODE

In Full-Bridge mode, all four pins are used as outputs. An example of a full-bridge application is provided in Figure 18-8.

In the Forward mode, the PxA pin is driven to its active state and the PxD pin is modulated, while the PxB and PxC pins are driven to their inactive state, as provided in Figure 18-9.

In the Reverse mode, the PxC pin is driven to its active state and the PxB pin is modulated, while the PxA and PxD pins are driven to their inactive state, as provided in Figure 18-9.

The PxA, PxB, PxC and PxD outputs are multiplexed with the port data latches. The associated TRIS bits must be cleared to configure the PxA, PxB, PxC and PxD pins as outputs.

FIGURE 18-8: EXAMPLE OF FULL-BRIDGE APPLICATION

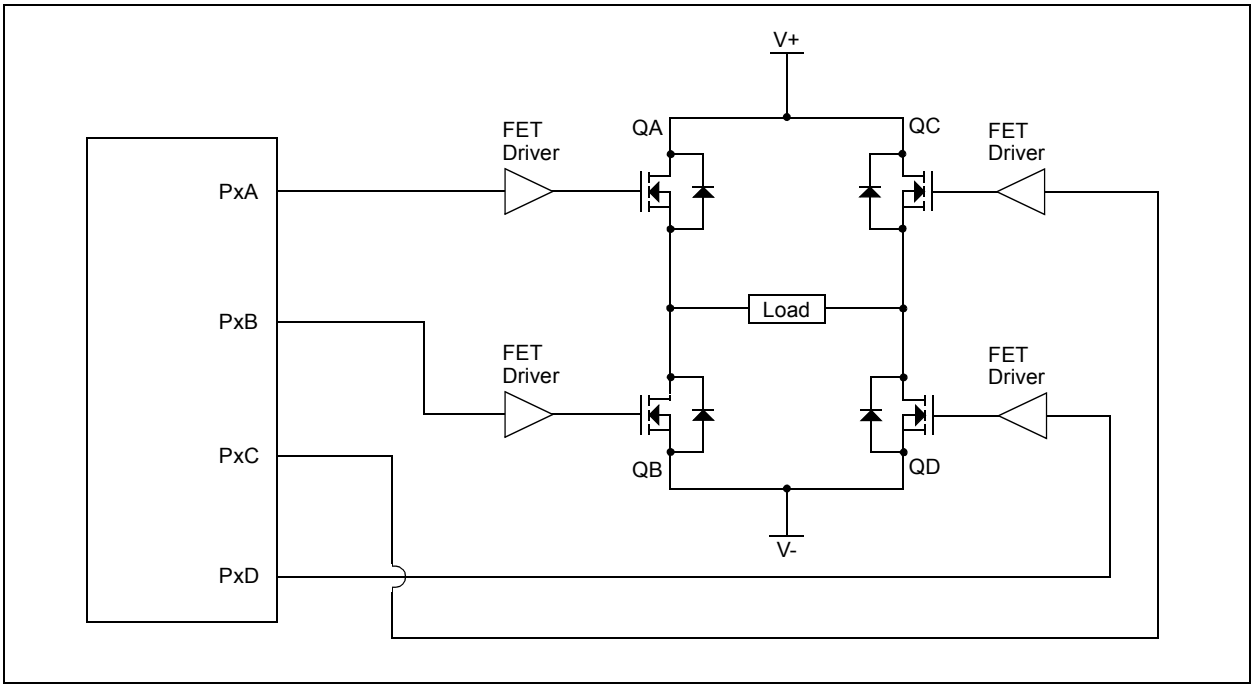
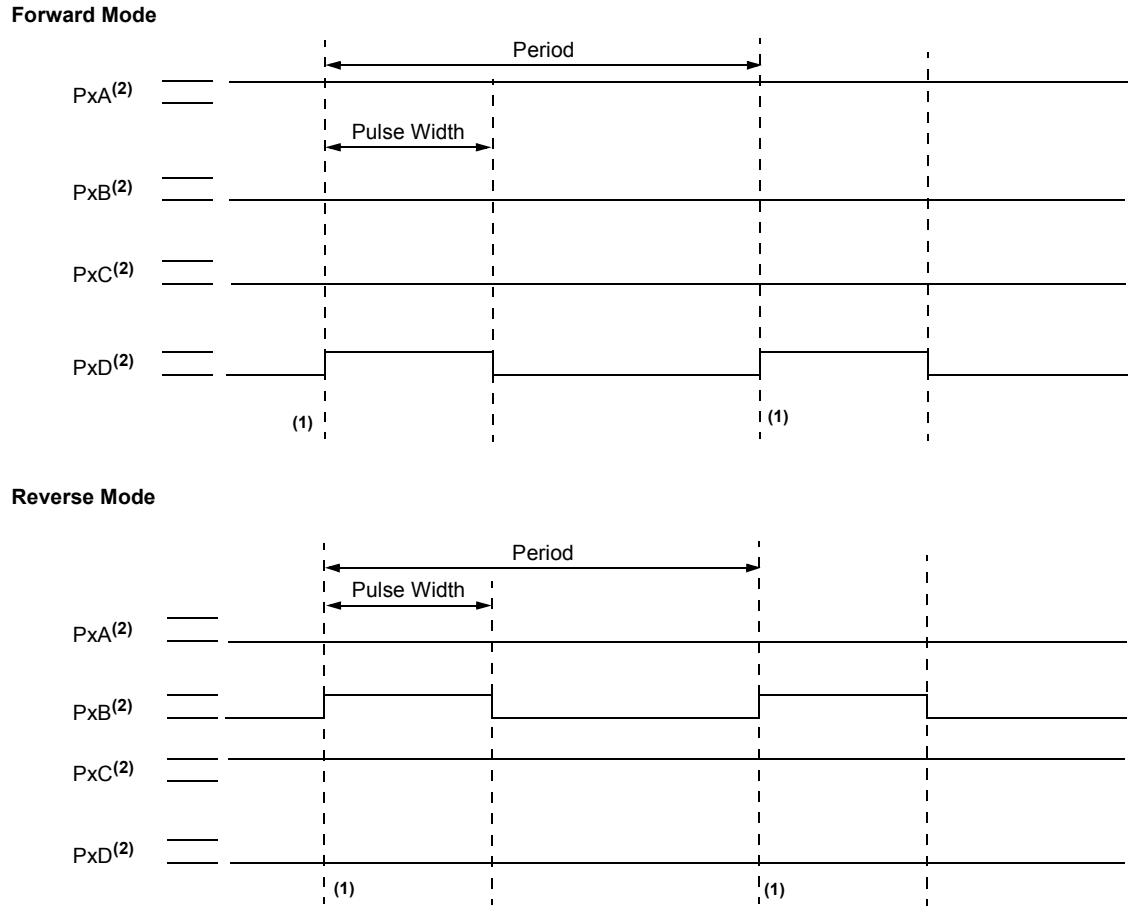


FIGURE 18-9: EXAMPLE OF FULL-BRIDGE PWM OUTPUT



Note 1: At this time, the TMR2 register is equal to the PR2 register.

2: The output signal is shown as active-high.

18.4.2.1 Direction Change in Full-Bridge Mode

In Full-Bridge mode, the PxM1 bit in the CCPxCON register allows users to control the forward/reverse direction. When the application firmware changes this direction control bit, the module will change to the new direction on the next PWM cycle.

A direction change is initiated in software by changing the PxM1 bit of the CCPxCON register. The following sequence occurs prior to the end of the current PWM period:

- The modulated outputs (PxB and PxD) are placed in their inactive state.
- The associated unmodulated outputs (PxA and PxC) are switched to drive in the opposite direction.
- PWM modulation resumes at the beginning of the next period.

For an illustration of this sequence, see Figure 18-10.

The Full-Bridge mode does not provide a dead-band delay. As one output is modulated at a time, a dead-band delay is generally not required. There is a situation where a dead-band delay is required. This situation occurs when both of the following conditions are true:

- The direction of the PWM output changes when the duty cycle of the output is at or near 100%.
- The turn-off time of the power switch, including the power device and driver circuit, is greater than the turn-on time.

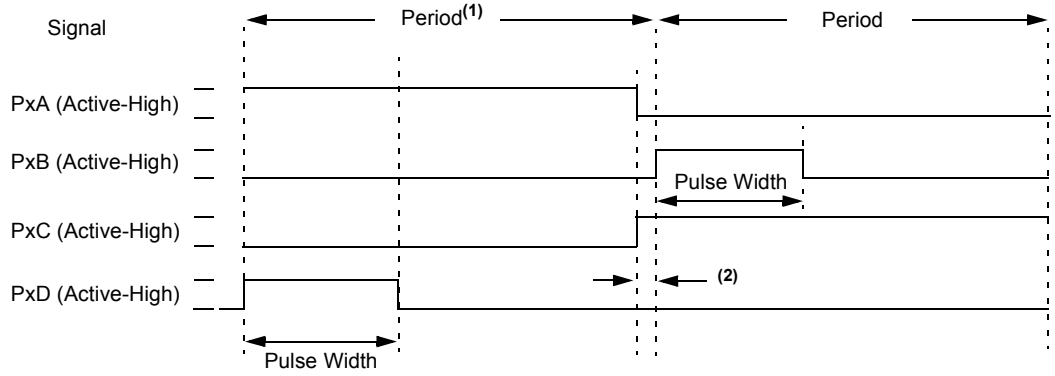
Figure 18-11 shows an example of the PWM direction changing from forward to reverse, at a near 100% duty cycle. In this example, at time, t1, the PxA and PxD outputs become inactive, while the PxC output becomes active. Since the turn-off time of the power devices is longer than the turn-on time, a shoot-through current will flow through power devices, QC and QD (see Figure 18-8), for the duration of 't'. The same phenomenon will occur to power devices, QA and QB, for PWM direction change from reverse to forward.

If changing PWM direction at high duty cycle is required for an application, two possible solutions for eliminating the shoot-through current are:

- Reduce PWM duty cycle for one PWM period before changing directions.
- Use switch drivers that can drive the switches off faster than they can drive them on.

Other options to prevent shoot-through current may exist.

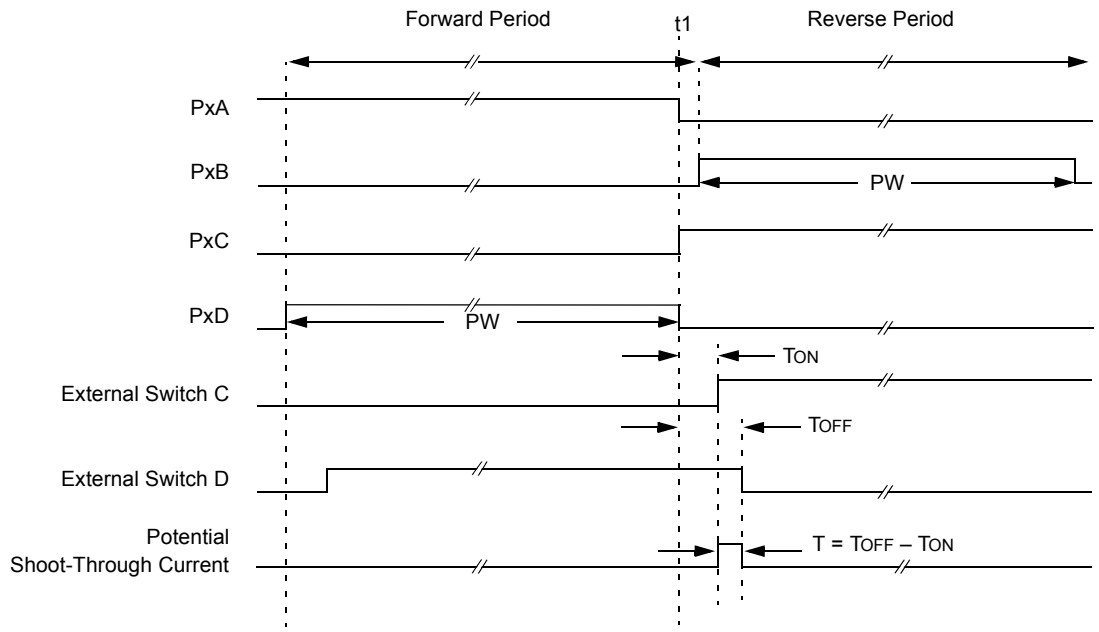
FIGURE 18-10: EXAMPLE OF PWM DIRECTION CHANGE



Note 1: The direction bit, PxM1 of the CCPxCON register, is written any time during the PWM cycle.

2: When changing directions, the PxA and PxC signals switch before the end of the current PWM cycle. The modulated PxB and PxD signals are inactive at this time. The length of this time is: (1/FOSC) • TMR2 Prescale Value.

FIGURE 18-11: EXAMPLE OF PWM DIRECTION CHANGE AT NEAR 100% DUTY CYCLE



Note 1: All signals are shown as active-high.

2: TON is the turn-on delay of power switch QC and its driver.

3: TOFF is the turn-off delay of power switch QD and its driver.

18.4.3 START-UP CONSIDERATIONS

When any PWM mode is used, the application hardware must use the proper external pull-up and/or pull-down resistors on the PWM output pins.

Note:	When the microcontroller is released from Reset, all of the I/O pins are in the high-impedance state. The external circuits must keep the power switch devices in the OFF state until the microcontroller drives the I/O pins with the proper signal levels or activates the PWM output(s).

The CCPxM<1:0> bits of the CCPxCON register allow the user to choose whether the PWM output signals are active-high or active-low for each pair of PWM output pins (PxA/PxC and PxB/PxD). The PWM output polarities must be selected before the PWM pin output drivers are enabled. Changing the polarity configuration while the PWM pin output drivers are enabled is not recommended since it may result in damage to the application circuits.

The PxA, PxB, PxC and PxD output latches may not be in the proper states when the PWM module is initialized. Enabling the PWM pin output drivers, at the same time as the Enhanced PWM modes, may cause damage to the application circuit. The Enhanced PWM modes must be enabled in the proper Output mode and complete a full PWM cycle before enabling the PWM pin output drivers. The completion of a full PWM cycle is indicated by the TMR2IF or TMR4IF bit of the PIR1 or PIR5 register being set as the second PWM period begins.

18.4.4 ENHANCED PWM AUTO-SHUTDOWN MODE

The PWM mode supports an Auto-Shutdown mode that will disable the PWM outputs when an external shutdown event occurs. Auto-Shutdown mode places the PWM output pins into a predetermined state. This mode is used to help prevent the PWM from damaging the application.

The auto-shutdown sources are selected using the ECCPxAS<2:0> bits (ECCPxAS<6:4>). A shutdown event may be generated by:

- A logic '0' on the pin that is assigned the FLT0 input function
- Comparator C1
- Comparator C2
- Setting the ECCPxASE bit in firmware

A shutdown condition is indicated by the ECCPxASE (Auto-Shutdown Event Status) bit (ECCPxAS<7>). If the bit is a '0', the PWM pins are operating normally. If the bit is a '1', the PWM outputs are in the shutdown state.

When a shutdown event occurs, two things happen:

- The ECCPxASE bit is set to '1'. The ECCPxASE will remain set until cleared in firmware or an auto-restart occurs. (See **Section 18.4.5 "Auto-Restart Mode"**.)
- The enabled PWM pins are asynchronously placed in their shutdown states. The PWM output pins are grouped into pairs (PxA/PxC and PxB/PxD). The state of each pin pair is determined by the PSSxAC and PSSxBD bits (ECCPxAS<3:2> and <1:0>, respectively).

Each pin pair may be placed into one of three states:

- Drive logic '1'
- Drive logic '0'
- Tri-state (high-impedance)

REGISTER 18-3: ECCPxAS: ECCPx AUTO-SHUTDOWN CONTROL REGISTER(1,2,3)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ECCPxASE	ECCPxAS2	ECCPxAS1	ECCPxAS0	PSSxAC1	PSSxAC0	PSSxBD1	PSSxBD0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ECCPxASE:** ECCP Auto-Shutdown Event Status bit
1 = A shutdown event has occurred; ECCP outputs are in a shutdown state
0 = ECCP outputs are operating

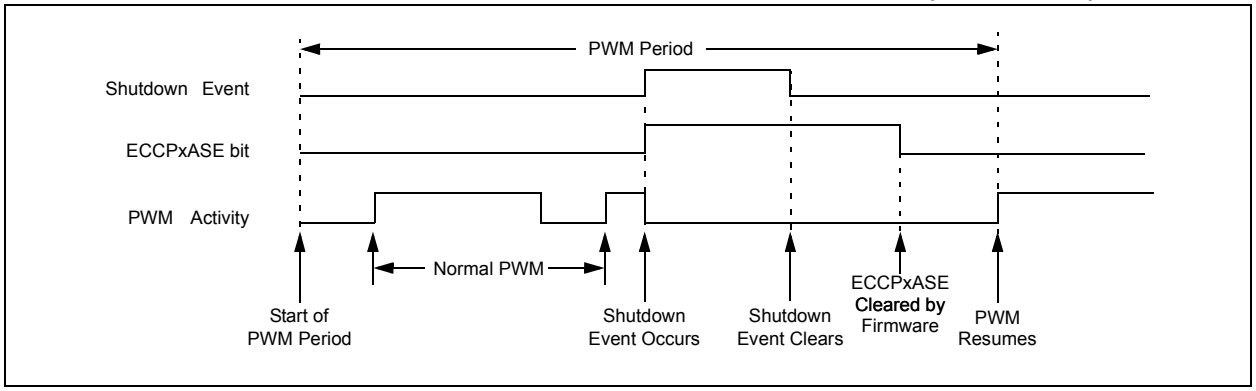
bit 6-4 **ECCPxAS<2:0>:** ECCP Auto-Shutdown Source Select bits
000 = Auto-shutdown is disabled
001 = Comparator C1OUT output is high
010 = Comparator C2OUT output is high
011 = Either Comparator C1OUT or C2OUT is high
100 = VIL on FLT0 pin
101 = VIL on FLT0 pin or Comparator C1OUT output is high
110 = VIL on FLT0 pin or Comparator C2OUT output is high
111 = VIL on FLT0 pin or Comparator C1OUT or Comparator C2OUT is high

bit 3-2 **PSSxAC<1:0>:** PxA and PxC Pins Shutdown State Control bits
00 = Drive pins: PxA and PxC to '0'
01 = Drive pins: PxA and PxC to '1'
1x = PxA and PxC pins tri-state

bit 1-0 **PSSxBD<1:0>:** Pins PxB and PxD Shutdown State Control bits
00 = Drive pins: PxB and PxD to '0'
01 = Drive pins: PxB and PxD to '1'
1x = PxB and PxD pins tri-state

Note 1: The auto-shutdown condition is a level-based signal, not an edge-based signal. As long as the level is present, the auto-shutdown will persist.
2: Writing to the ECCPxASE bit is disabled while an auto-shutdown condition persists.
3: Once the auto-shutdown condition has been removed and the PWM restarted (either through firmware or auto-restart), the PWM signal will always restart at the beginning of the next PWM period.

FIGURE 18-12: PWM AUTO-SHUTDOWN WITH FIRMWARE RESTART (PxRSEN = 0)



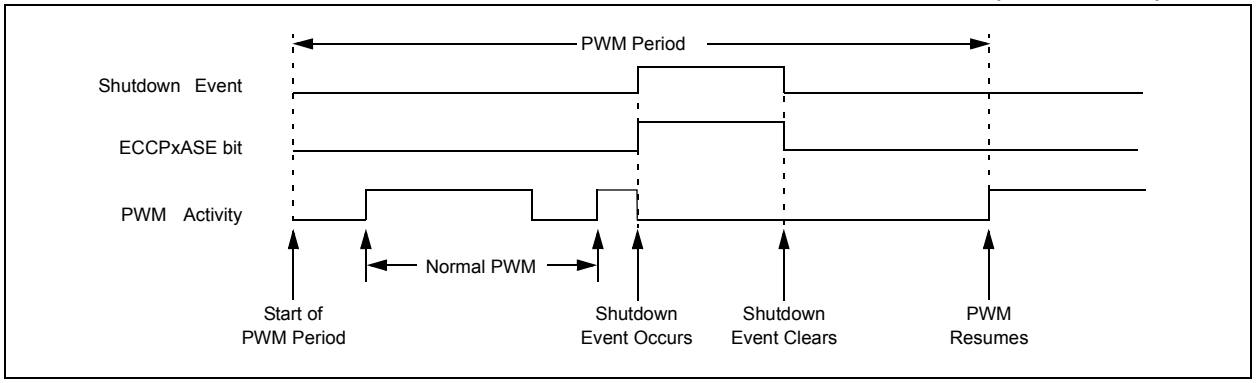
18.4.5 AUTO-RESTART MODE

The Enhanced PWM can be configured to automatically restart the PWM signal once the auto-shutdown condition has been removed. Auto-restart is enabled by setting the PxRSEN bit (ECCPxDEL<7>).

If auto-restart is enabled, the ECCPxASE bit will remain set as long as the auto-shutdown condition is active. When the auto-shutdown condition is removed, the ECCPxASE bit will be cleared via hardware and normal operation will resume.

The module will wait until the next PWM period begins, however, before re-enabling the output pin. This behavior allows the auto-shutdown with auto-restart features to be used in applications based on current mode of PWM control.

FIGURE 18-13: PWM AUTO-SHUTDOWN WITH AUTO-RESTART ENABLED (PxRSEN = 1)

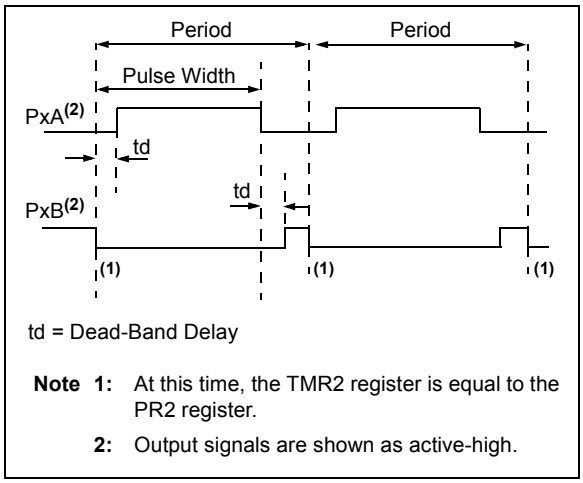


18.4.6 PROGRAMMABLE DEAD-BAND DELAY MODE

In half-bridge applications, where all power switches are modulated at the PWM frequency, the power switches normally require more time to turn off than to turn on. If both the upper and lower power switches are switched at the same time (one turned on and the other turned off), both switches may be on for a short period until one switch completely turns off. During this brief interval, a very high current (shoot-through current) will flow through both power switches, shorting the bridge supply. To avoid this potentially destructive shoot-through current from flowing during switching, turning on either of the power switches is normally delayed to allow the other switch to completely turn off.

In Half-Bridge mode, a digitally programmable dead-band delay is available to avoid shoot-through current from destroying the bridge power switches. The delay occurs at the signal transition from the non-active state to the active state. For an illustration, see Figure 18-14. The lower seven bits of the associated ECCPxDEL register (Register 18-4) set the delay period in terms of microcontroller instruction cycles (TCY or 4 TOSC).

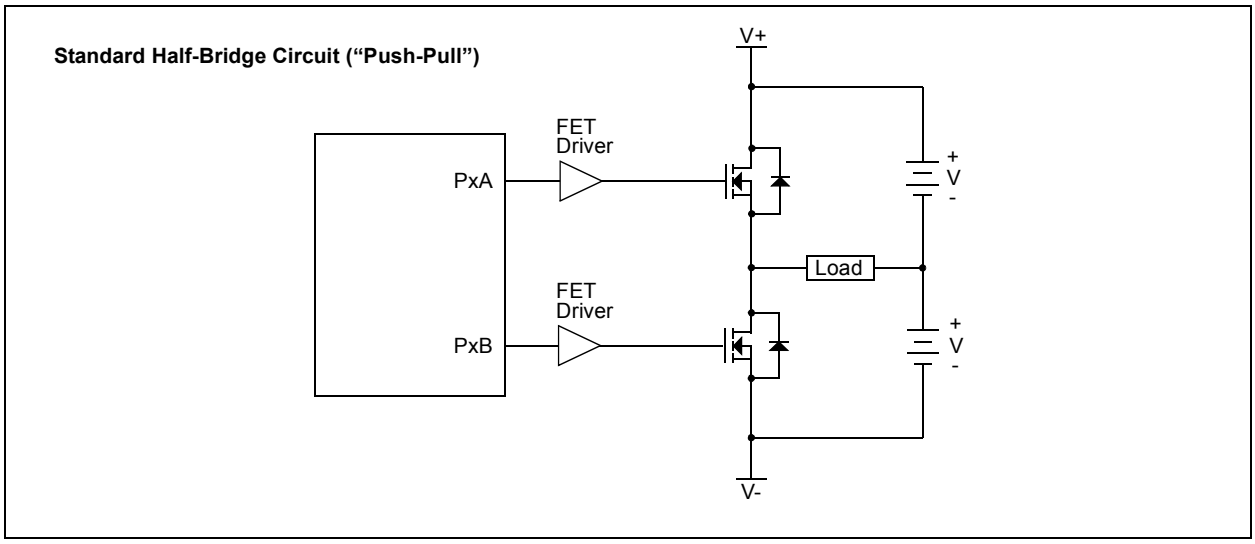
FIGURE 18-14: EXAMPLE OF HALF-BRIDGE PWM OUTPUT



Note 1: At this time, the TMR2 register is equal to the PR2 register.

2: Output signals are shown as active-high.

FIGURE 18-15: EXAMPLE OF HALF-BRIDGE APPLICATIONS



REGISTER 18-4: ECCPxDEL: ENHANCED PWM CONTROL REGISTER x

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PxRSEN	PxDC6	PxDC5	PxDC4	PxDC3	PxDC2	PxDC1	PxDC0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **PxRSEN:** PWM Restart Enable bit

1 = Upon auto-shutdown, the ECCPxASE bit clears automatically once the shutdown event goes away; the PWM restarts automatically

0 = Upon auto-shutdown, ECCPxASE must be cleared by software to restart the PWM

bit 6-0 **PxDC<6:0>:** PWM Delay Count bits

PxDCn=Number of FOSC/4 (4 * TOSC) cycles between the scheduled time when a PWM signal **should** transition active and the **actual** time it does transition active.

18.4.7 PULSE STEERING MODE

In Single Output mode, pulse steering allows any of the PWM pins to be the modulated signal. Additionally, the same PWM signal can simultaneously be available on multiple pins.

Once the Single Output mode is selected (CCPxM<3:2> = 11 and PxM<1:0> = 00 of the CCPxCON register), the user firmware can bring out the same PWM signal to one, two, three or four output pins by setting the appropriate STR<D:A> bits (PSTRxCON<3:0>), as provided in Table 18-3.

Note:	The associated TRIS bits must be set to output ('0') to enable the pin output driver in order to see the PWM signal on the pin.

While the PWM Steering mode is active, the CCPxM<1:0> bits (CCPxCON<1:0>) select the PWM output polarity for the Px<D:A> pins.

The PWM auto-shutdown operation also applies to the PWM Steering mode, as described in **Section 18.4.4 "Enhanced PWM Auto-shutdown mode"**. An auto-shutdown event will only affect pins that have PWM outputs enabled.

REGISTER 18-5: PSTRxCON: PULSE STEERING CONTROL(1)

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1
CMPL1	CMPL0	—	STRSYNC	STRD	STRC	STRB	STRA
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **CMPL<1:0>:** Complementary Mode Output Assignment Steering Sync bits
00 =See STR<D:A>
01 =PA and PB are selected as the complementary output pair
10 =PA and PC are selected as the complementary output pair
11 =PA and PD are selected as the complementary output pair

bit 5 **Unimplemented:** Read as '0'

bit 4 **STRSYNC:** Steering Sync bit
1 = Output steering update occurs on the next PWM period
0 = Output steering update occurs at the beginning of the instruction cycle boundary

bit 3 **STRD:** Steering Enable bit D
1 = PxD pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxD pin is assigned to port pin

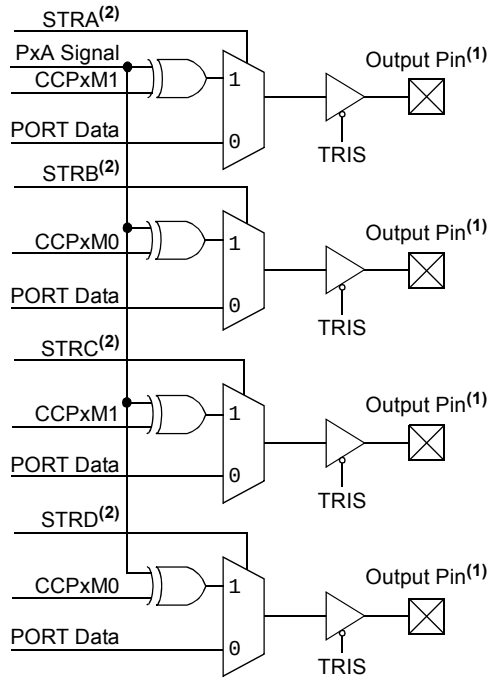
bit 2 **STRC:** Steering Enable bit C
1 = PxC pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxC pin is assigned to port pin

bit 1 **STRB:** Steering Enable bit B
1 = PxB pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxB pin is assigned to port pin

bit 0 **STRA:** Steering Enable bit A
1 = PxA pin has the PWM waveform with polarity control from CCPxM<1:0>
0 = PxA pin is assigned to port pin

Note 1: The PWM Steering mode is available only when the CCPxCON register bits, CCPxM<3:2> = 11 and PxM<1:0> = 00.

FIGURE 18-16: SIMPLIFIED STEERING BLOCK DIAGRAM



Note 1: Port outputs are configured as displayed when the CCPxCON register bits, PxM<1:0> = 00 and CCPxM<3:2> = 11.

2: Single PWM output requires setting at least one of the STRx bits.

18.4.7.1 Steering Synchronization

The STRSYNC bit of the PSTRxCON register gives the user two choices for when the steering event will happen. When the STRSYNC bit is '0', the steering event will happen at the end of the instruction that writes to the PSTRxCON register. In this case, the output signal at the Px<D:A> pins may be an incomplete PWM waveform. This operation is useful when the user firmware needs to immediately remove a PWM signal from the pin.

When the STRSYNC bit is '1', the effective steering update will happen at the beginning of the next PWM period. In this case, steering on/off the PWM output will always produce a complete PWM waveform.

Figures 18-17 and 18-18 illustrate the timing diagrams of the PWM steering depending on the STRSYNC setting.

FIGURE 18-17: EXAMPLE OF STEERING EVENT AT END OF INSTRUCTION (STRSYNC = 0)

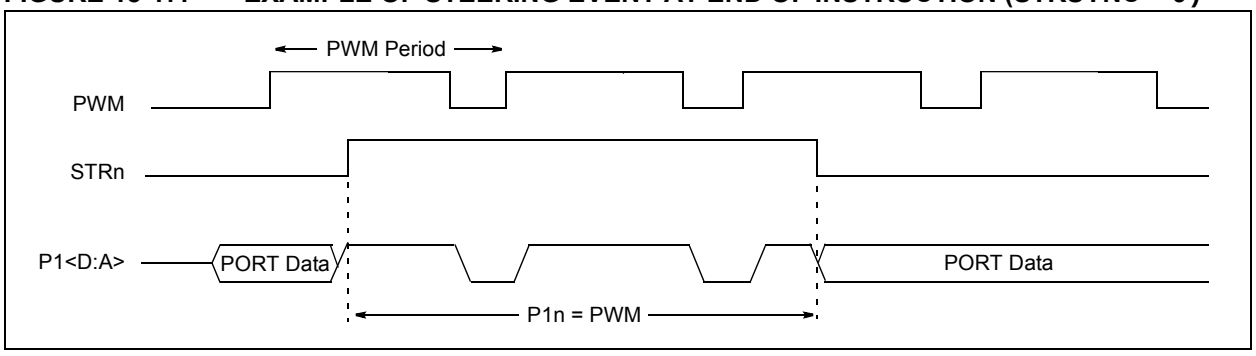
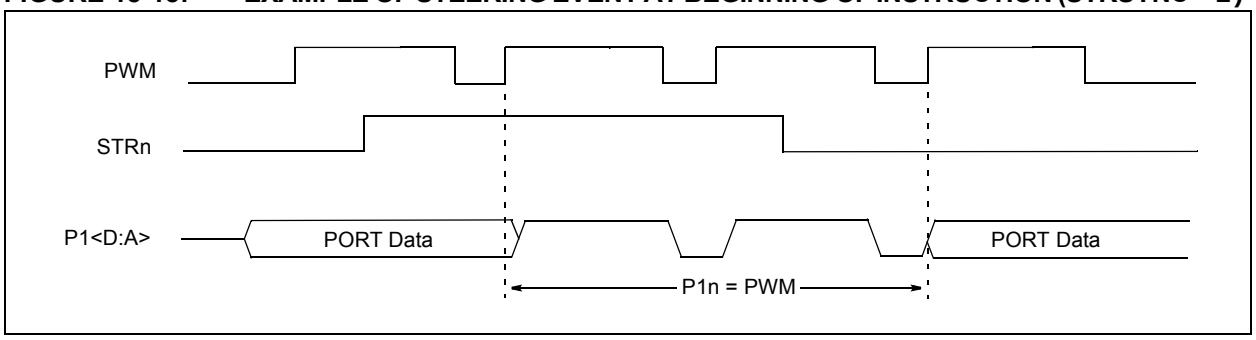


FIGURE 18-18: EXAMPLE OF STEERING EVENT AT BEGINNING OF INSTRUCTION (STRSYNC = 1)



18.4.8 OPERATION IN POWER-MANAGED MODES

In Sleep mode, all clock sources are disabled. Timer2/4/6/8 will not increment and the state of the module will not change. If the ECCPx pin is driving a value, it will continue to drive that value. When the device wakes up, it will continue from this state. If Two-Speed Start-ups are enabled, the initial start-up frequency from HF-INTOSC and the postscaler may not be stable immediately.

In PRI_IDLE mode, the primary clock will continue to clock the ECCPx module without change.

18.4.8.1 Operation with Fail-Safe Clock Monitor (FSCM)

If the Fail-Safe Clock Monitor (FSCM) is enabled, a clock failure will force the device into the power-managed RC_RUN mode and the OSCFIF bit of the PIR2 register will be set. The ECCPx will then be clocked from the internal oscillator clock source, which may have a different clock frequency than the primary clock.

18.4.9 EFFECTS OF A RESET

Both Power-on Reset and subsequent Resets will force all ports to Input mode and the ECCP registers to their Reset states.

This forces the ECCP module to reset to a state compatible with previous, non-enhanced CCP modules used on other PIC18 and PIC16 devices.

19.0 CAPTURE/COMPARE/PWM (CCP) MODULES

PIC18FXXJ94 devices have seven CCP (Capture/Compare/PWM) modules, designated CCP4 through CCP10. All the modules implement standard Capture, Compare and Pulse-Width Modulation (PWM) modes.

Note: Throughout this section, generic references are used for register and bit names that are the same, except for an 'x' variable that indicates the item's association with the specific CCP module. For example, the control register is named CCPxCON and refers to CCP4CON through CCP10CON.

Each CCP module contains a 16-bit register that can operate as a 16-bit Capture register, a 16-bit Compare register or a PWM Master/Slave Duty Cycle register. For the sake of clarity, all CCP module operation in the following sections is described with respect to CCP4, but is equally applicable to CCP5 through CCP10.

REGISTER 19-1: CCPxCON: CCPx CONTROL REGISTER (CCP4-CCP10 MODULES)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	DCxB1	DCxB0	CCPxM3(1)	CCPxM2(1)	CCPxM1(1)	CCPxM0(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **DCxB<1:0>:** PWM Duty Cycle bit 1 and bit 0 for CCPx module bits

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two Least Significant bits (bit 1 and bit 0) of the 10-bit PWM duty cycle. The eight Most Significant bits (DCx<9:2>) of the duty cycle are found in CCPRxL.

bit 3-0 **CCPxM<3:0>**: CCPx Module Mode Select bits(1)

0000 =Capture/Compare/PWM is disabled (resets CCPx module)
0001 =Reserved
0010 =Compare mode, toggles output on match (CCPxIF bit is set)
0011 =Reserved
0100 =Capture mode: Every falling edge
0101 =Capture mode: Every rising edge
0110 =Capture mode: Every 4th rising edge
0111 =Capture mode: Every 16th rising edge
1000 =Compare mode: Initialize CCPx pin low; on compare match, force CCPx pin high (CCPxIF bit is set)
1001 =Compare mode: Initialize CCPx pin high; on compare match, force CCPx pin low (CCPxIF bit is set)
1010 =Compare mode: Generate software interrupt on compare match (CCPxIF bit is set, CCPx pin reflects I/O state)
1011 =Compare mode: Special Event Trigger; reset timer on CCPx match (CCPxIF bit is set)
11xx =PWM mode

Note 1: CCPxM<3:0> = 1011 will only reset the timer and not start an A/D conversion on a CCPx match.

REGISTER 19-2: CCPTMRS1: CCP TIMER SELECT REGISTER 1

R/W-0	R/W-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
C7TSEL1	C7TSEL0	—	C6TSEL0	—	C5TSEL0	C4TSEL1	C4TSEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-6 **C7TSEL<1:0>:** CCP7 Timer Selection bits
00 =CCP7 is based off of TMR1/TMR2
01 =CCP7 is based off of TMR5/TMR4
10 =CCP7 is based off of TMR5/TMR6
11 =CCP7 is based off of TMR5/TMR8

bit 5 **Unimplemented:** Read as ‘0’

bit 4 **C6TSEL0:** CCP6 Timer Selection bit
0 = CCP6 is based off of TMR1/TMR2
1 = CCP6 is based off of TMR5/TMR2

bit 3 **Unimplemented:** Read as ‘0’

bit 2 **C5TSEL0:** CCP5 Timer Selection bit
0 = CCP5 is based off of TMR1/TMR2
1 = CCP5 is based off of TMR5/TMR4

bit 1-0 **C4TSEL<1:0>:** CCP4 Timer Selection bits
00 =CCP4 is based off of TMR1/TMR2
01 =CCP4 is based off of TMR3/TMR4
10 =CCP4 is based off of TMR3/TMR6
11 =Reserved; do not use

REGISTER 19-3: CCPTMRS2: CCP TIMER SELECT REGISTER 2

U-0	U-0	U-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	C10TSEL0	—	C9TSEL0	C8TSEL1	C8TSEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **C10TSEL0:** CCP10 Timer Selection bit
0 = CCP10 is based off of TMR1/TMR2
1 = CCP10 is based off of TMR5/TMR2

bit 3 **Unimplemented:** Read as '0'

bit 2 **C9TSEL0:** CCP9 Timer Selection bit
0 = CCP9 is based off of TMR1/TMR2
1 = CCP9 is based off of TMR5/TMR4

bit 1-0 **C8TSEL<1:0>:** CCP8 Timer Selection bits
00 =CCP8 is based off of TMR1/TMR2
01 =CCP8 is based off of TMR3/TMR4
10 =CCP8 is based off of TMR3/TMR6
11 =Reserved; do not use

REGISTER 19-4: CCPRxL: CCPx PERIOD LOW BYTE REGISTER

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
CCPRxL7	CCPRxL6	CCPRxL5	CCPRxL4	CCPRxL3	CCPRxL2	CCPRxL1	CCPRxL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **CCPRxL<7:0>:** CCPx Period Register Low Byte bits
Capture mode: Capture Register Low Byte
Compare mode: Compare Register Low Byte
PWM mode: Duty Cycle Register

REGISTER 19-5: CCPRxH: CCPx PERIOD HIGH BYTE REGISTER

R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
CCPRxH7	CCPRxH6	CCPRxH5	CCPRxH4	CCPRxH3	CCPRxH2	CCPRxH1	CCPRxH0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-0 **CCPRxH<7:0>:** CCPx Period Register High Byte bits
Capture mode: Capture Register High Byte
Compare mode: Compare Register High Byte
PWM mode: Duty Cycle Buffer Register

19.1 CCP Module Configuration

Each Capture/Compare/PWM module is associated with a control register (generically, CCPxCON) and a data register (CCPRx). The data register, in turn, is comprised of two 8-bit registers: CCPRxL (low byte) and CCPRxH (high byte). All registers are both readable and writable.

19.1.1 CCP MODULES AND TIMER RESOURCES

The CCP modules utilize Timers, 1 through 8, that vary with the selected mode. Various timers are available to the CCP modules in Capture, Compare or PWM modes, as shown in Table 19-1.

TABLE 19-1: CCP MODE – TIMER RESOURCE

CCP Mode	Timer Resource
Capture	Timer1, Timer3 or Timer 5
Compare	
PWM	Timer2, Timer4, Timer 6 or Timer8

The assignment of a particular timer to a module is determined by the timer to CCP enable bits in the CCPTMRSx registers. (See Register 19-2 and Register 19-3.) All of the modules may be active at once and may share the same timer resource if they are configured to operate in the same mode (Capture/Compare or PWM) at the same time.

The CCPTMRS1 register selects the timers for CCP modules, 7, 6, 5 and 4, and the CCPTMRS2 register selects the timers for CCP modules, 10, 9 and 8. The possible configurations are shown in Table 19-2 and Table 19-3.

TABLE 19-2: TIMER ASSIGNMENTS FOR CCP MODULES 4, 5, 6 AND 7

CCPTMRS1 Register											
CCP4			CCP5			CCP6			CCP7		
C4TSEL <1:0>	Capture/ Compare Mode	PWM Mode	C5TSEL0	Capture/ Compare Mode	PWM Mode	C6TSEL0	Capture/ Compare Mode	PWM Mode	C7TSEL <1:0>	Capture/ Compare Mode	PWM Mode
0 0	TMR1	TMR2	0	TMR1	TMR2	0	TMR1	TMR2	0 0	TMR1	TMR2
0 1	TMR3	TMR4	1	TMR5	TMR4	1	TMR5	TMR2	0 1	TMR5	TMR4
1 0	TMR3	TMR6							1 0	TMR5	TMR6
1 1	Reserved(1)								1 1	TMR5	TMR8

Note 1: Do not use the reserved bits.

TABLE 19-3: TIMER ASSIGNMENTS FOR CCP MODULES 8, 9 AND 10

CCPTMRS2 Register											
CCP8			CCP8 Devices with 32 Kbytes			CCP9			CCP10		
C8TSEL <1:0>	Capture/ Compare Mode	PWM Mode	C8TSEL <1:0>	Capture/ Compare Mode	PWM Mode	C9TSEL0	Capture/ Compare Mode	PWM Mode	C10TSEL0	Capture/ Compare Mode	PWM Mode
0 0	TMR1	TMR2	0 0	TMR1	TMR2	0	TMR1	TMR2	0	TMR1	TMR2
0 1	TMR5	TMR4	0 1	TMR1	TMR4	1	TMR5	TMR4	1	TMR5	TMR2
1 0	TMR5	TMR6	1 0	TMR1	TMR6						
1 1	Reserved(1)		1 1	Reserved(1)							

Note 1: Do not use the reserved bits.

19.1.2 OPEN-DRAIN OUTPUT OPTION

When operating in Output mode (the Compare or PWM modes), the drivers for the CCPx pins can be optionally configured as open-drain outputs. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor and allows the output to communicate with external circuits without the need for additional level shifters.

The open-drain output option is controlled by the CCPxOD bits (ODCON2<7:1>). Setting the appropriate bit configures the pin for the corresponding module for open-drain operation.

19.2 Capture Mode

In Capture mode, the CCPR4H:CCPR4L register pair captures the 16-bit value of the Timer register selected in the CCPTMRS1 when an event occurs on the CCP4 pin. An event is defined as one of the following:

- Every falling edge
- Every rising edge
- Every 4th rising edge
- Every 16th rising edge

The event is selected by the mode select bits, CCP4M<3:0> (CCP4CON<3:0>). When a capture is made, the interrupt request flag bit, CCP4IF (PIR4<1>), is set. (It must be cleared in software.) If another capture occurs before the value in CCPR4 is read, the old captured value is overwritten by the new captured value.

Figure 19-1 shows the Capture mode block diagram.

19.2.1 CCP PIN CONFIGURATION

In Capture mode, the appropriate CCPx pin should be configured as an input by setting the corresponding TRIS direction bit.

Note:	If the CCPx pin is configured as an output, a write to the port can cause a capture condition.

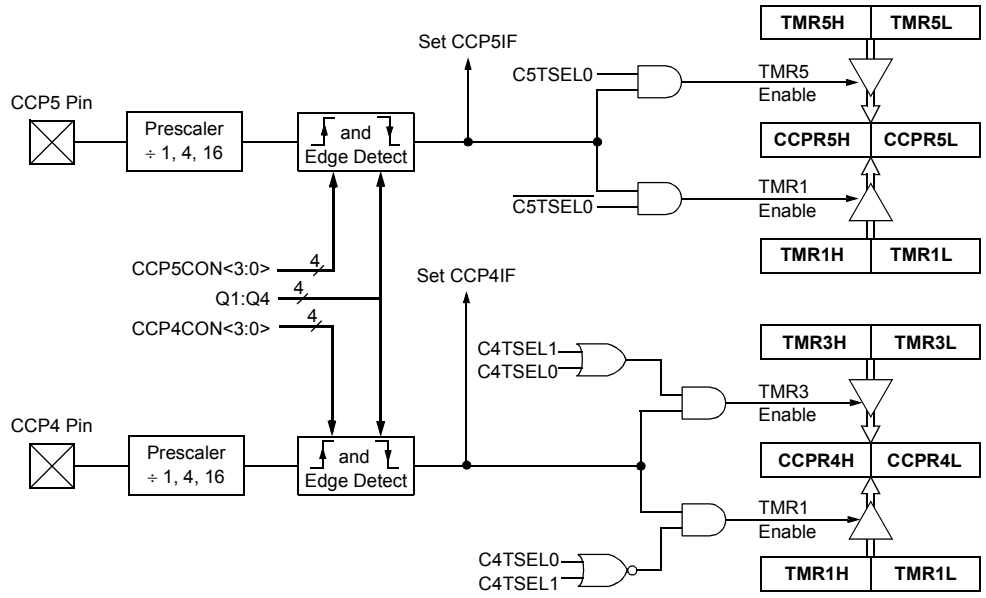
19.2.2 TIMER1/3/5/7 MODE SELECTION

For the available timers (1/3/5) to be used for the capture feature, the used timers must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the capture operation will not work.

The timer to be used with each CCP module is selected in the CCPTMRSx registers. (See **Section 19.1.1 "CCP Modules and Timer Resources"**.)

Details of the timer assignments for the CCP modules are given in Table 19-2 and Table 19-3.

FIGURE 19-1: CAPTURE MODE OPERATION BLOCK DIAGRAM



Note: This block diagram uses CCP4 and CCP5, and their appropriate timers as an example. For details on all of the CCP modules and their timer assignments, see Table 19-2 and Table 19-3.

19.2.3 SOFTWARE INTERRUPT

When the Capture mode is changed, a false capture interrupt may be generated. The user should keep the CCP4IE bit (PIE4<1>) clear to avoid false interrupts and should clear the flag bit, CCP4IF, following any such change in operating mode.

19.2.4 CCP PRESCALER

There are four prescaler settings in Capture mode. They are specified as part of the operating mode selected by the mode select bits (CCP4M<3:0>). Whenever the CCP module is turned off, or the CCP module is not in Capture mode, the prescaler counter is cleared. This means that any Reset will clear the prescaler counter.

Switching from one capture prescaler to another may generate an interrupt. Doing that will also not clear the prescaler counter – meaning the first capture may be from a non-zero prescaler.

Example 19-1 shows the recommended method for switching between capture prescalers. This example also clears the prescaler counter and will not generate the “false” interrupt.

EXAMPLE 19-1: CHANGING BETWEEN CAPTURE PRESCALERS

```
CLRF   CCP4CON      ; Turn CCP module off
MOVLW NEW_CAPT_PS ; Load WREG with the
                    ; new prescaler mode
                    ; value and CCP ON
MOVWF CCP4CON      ; Load CCP4CON with
                    ; this value
```

19.3 Compare Mode

In Compare mode, the 16-bit CCPR4 register value is constantly compared against the Timer register pair value selected in the CCPTMR1 register. When a match occurs, the CCP4 pin can be:

- Driven high
- Driven low
- Toggled (high-to-low or low-to-high)
- Unchanged (that is, reflecting the state of the I/O latch)

The action on the pin is based on the value of the mode select bits (CCP4M<3:0>). At the same time, the interrupt flag bit, CCP4IF, is set.

Figure 19-2 gives the Compare mode block diagram

19.3.1 CCP PIN CONFIGURATION

The user must configure the CCPx pin as an output by clearing the appropriate TRIS bit.

Note:	Clearing the CCPxCON register will force the CCPx compare output latch (depending on device configuration) to the default low level. This is not the PORTx I/O data latch.

19.3.2 TIMER1/3/5 MODE SELECTION

If the CCP module is using the compare feature in conjunction with any of the Timer1/3/5 timers, the timers must be running in Timer mode or Synchronized Counter mode. In Asynchronous Counter mode, the compare operation will not work.

Note:	Details of the timer assignments for the CCP modules are given in Table 19-2 and Table 19-3.

19.3.3 SOFTWARE INTERRUPT MODE

When the Generate Software Interrupt mode is chosen (CCP4M<3:0> = 1010), the CCP4 pin is not affected. Only a CCP interrupt is generated, if enabled, and the CCP4IE bit is set.

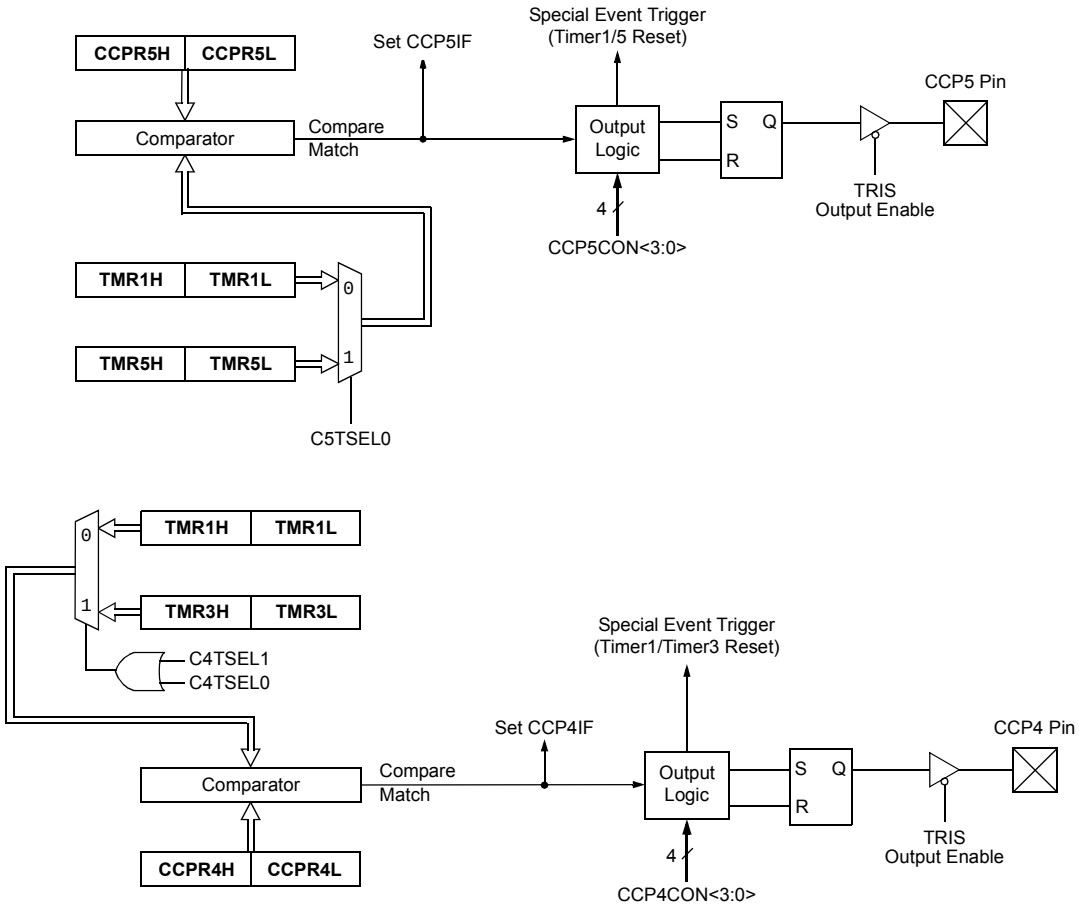
19.3.4 SPECIAL EVENT TRIGGER

Both CCP modules are equipped with a Special Event Trigger. This is an internal hardware signal, generated in Compare mode, to trigger actions by other modules. The Special Event Trigger is enabled by selecting the Compare Special Event Trigger mode (CCP4M<3:0> = 1011).

For either CCP module, the Special Event Trigger resets the Timer register pair for whichever timer resource is currently assigned as the module's time base. This allows the CCPRx registers to serve as a Programmable Period register for either timer.

The Special Event Trigger for CCP4 cannot start an A/D conversion.

FIGURE 19-2: COMPARE MODE OPERATION BLOCK DIAGRAM



Note: This block diagram uses CCP4 and CCP5 and their appropriate timers as an example. For details on all of the CCP modules and their timer assignments, see Table 19-2 and Table 19-3.

19.4 PWM Mode

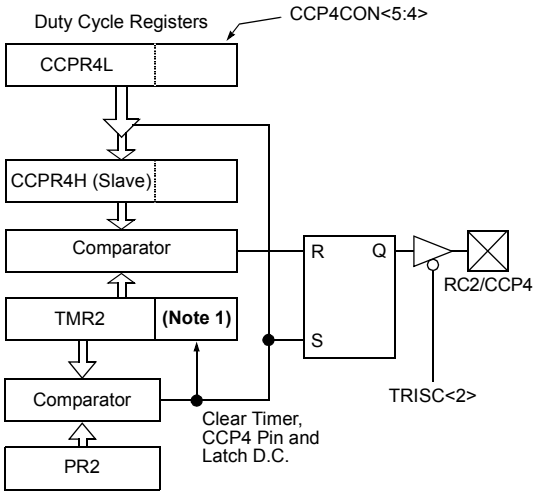
In Pulse-Width Modulation (PWM) mode, the CCP4 pin produces up to a 10-bit resolution PWM output. Since the CCP4 pin is multiplexed with a PORTC or PORTE data latch, the appropriate TRIS bit must be cleared to make the CCP4 pin an output.

Note:	Clearing the CCPxCON register will force the CCPx compare output latch (depending on device configuration) to the default low level. This is not the PORTx I/O data latch.

Figure 19-3 shows a simplified block diagram of the CCP4 module in PWM mode.

For a step-by-step procedure on how to set up the CCP module for PWM operation, see **Section 19.4.3 "Setup for PWM Operation"**.

FIGURE 19-3: SIMPLIFIED PWM BLOCK DIAGRAM

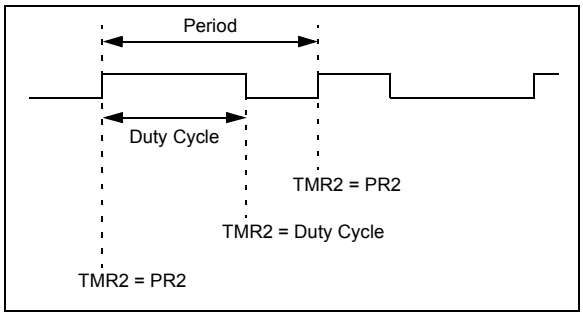


Note 1: The 8-bit TMR2 value is concatenated with the 2-bit internal Q clock, or 2 bits of the prescaler, to create the 10-bit time base.

2: CCP4 and its appropriate timers are used as an example. For details on all of the CCP modules and their timer assignments, see Table 19-2 and Table 19-3.

A PWM output (Figure 19-4) has a time base (period) and a time that the output stays high (duty cycle). The frequency of the PWM is the inverse of the period (1/period).

FIGURE 19-4: PWM OUTPUT



19.4.1 PWM PERIOD

The PWM period is specified by writing to the PR2 register. The PWM period can be calculated using the following formula:

EQUATION 19-1: PWM PERIOD CALCULATION

$$\text{PWM Period} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot (\text{TMR2 Prescale Value})$$

PWM frequency is defined as 1/[PWM period].

When TMR2 is equal to PR2, the following three events occur on the next increment cycle:

- TMR2 is cleared
- The CCP4 pin is set

  (An exception: If PWM Duty Cycle = 0%, the CCP4 pin will not be set)
- The PWM duty cycle is latched from CCPR4L into CCPR4H

Note:	The Timer2 postscalers (see **Section 16.0 "Timer2/4/6/8 Modules"**) are not used in the determination of the PWM frequency. The postscaler could be used to have a servo update rate at a different frequency than the PWM output.

19.4.2 PWM DUTY CYCLE

The PWM duty cycle is specified, to use CCP4 as an example, by writing to the CCPR4L register and to the CCP4CON<5:4> bits. Up to 10-bit resolution is available. The CCPR4L contains the eight MSbs and the CCP4CON<5:4> contains the two LSbs. This 10-bit value is represented by CCPR4L:CCP4CON<5:4>. The following equation is used to calculate the PWM duty cycle in time:

EQUATION 19-2: PWM DUTY CYCLE (IN TIME)

PWM Duty Cycle = (CCPR4L:CCP4CON<5:4>) • TOSC • (TMR2 Prescale Value)

CCPR4L and CCP4CON<5:4> can be written to at any time, but the duty cycle value is not latched into CCPR4H until after a match between PR2 and TMR2 occurs (that is, the period is complete). In PWM mode, CCPR4H is a read-only register.

The CCPR4H register and a two-bit internal latch are used to double-buffer the PWM duty cycle. This double-buffering is essential for glitchless PWM operation.

When the CCPR4H and two-bit latch match TMR2, concatenated with an internal two-bit Q clock or two bits of the TMR2 prescaler, the CCP4 pin is cleared.

The maximum PWM resolution (bits) for a given PWM frequency is given by the equation:

EQUATION 19-3: PWM RESOLUTION

$$\text{PWM Resolution (max)} = \frac{\log\left(\frac{FOSC}{FPWM}\right)}{\log(2)} \text{bits}$$

Note: If the PWM duty cycle value is longer than the PWM period, the CCP4 pin will not be cleared.

TABLE 19-4: EXAMPLE PWM FREQUENCIES AND RESOLUTIONS AT 40 MHz

PWM Frequency	2.44 kHz	9.77 kHz	39.06 kHz	156.25 kHz	312.50 kHz	416.67 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	FFh	FFh	FFh	3Fh	1Fh	17h
Maximum Resolution (bits)	10	10	10	8	7	6.58

19.4.3 SETUP FOR PWM OPERATION

To configure the CCP module for PWM operation using CCP4 as an example:

1. Set the PWM period by writing to the PR2 register.
2. Set the PWM duty cycle by writing to the CCPR4L register and CCP4CON<5:4> bits.
3. Make the CCP4 pin an output by clearing the appropriate TRIS bit.
4. Set the TMR2 prescale value, then enable Timer2 by writing to T2CON.
5. Configure the CCP4 module for PWM operation.

20.0 MASTER SYNCHRONOUS SERIAL PORT (MSSP) MODULE

20.1 Master SSP (MSSP) Module Overview

The Master Synchronous Serial Port (MSSP) module is a serial interface, useful for communicating with other peripheral or microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D Converters, etc. The MSSP module can operate in one of two modes:

- Serial Peripheral Interface (SPI)
- Inter-Integrated Circuit™ (I^2C)
  - Full Master mode
  - Slave mode (with general address call)

The I^2C interface supports the following modes in hardware:

- Master mode
- Multi-Master mode
- Slave mode with 5-bit and 7-bit address masking (with address masking for both 10-bit and 7-bit addressing)

All members of the PIC18FXXJ94 have two MSSP modules, designated as MSSP1 and MSSP2. Each module operates independently of the other.

Note:	Throughout this section, generic references to an MSSP module in any of its operating modes may be interpreted as being equally applicable to MSSP1 or MSSP2. Register names and module I/O signals use the generic designator 'x' to indicate the use of a numeral to distinguish a particular module when required. Control bit names are not individuated.

20.2 Control Registers

Each MSSP module has four associated control registers. These include a STATUS register (SSPxSTAT) and three control registers (SSPxCON1, SSPxCON2, and SSPxCON3). The use of these registers and their individual Configuration bits differ significantly depending on whether the MSSP module is operated in SPI or I^2C mode.

Additional details are provided under the individual sections. On all PIC18F97J94 family devices, the SPI DMA capability can only be used in conjunction with MSSP1. The SPI DMA feature is described in **Section 20.4 "SPI DMA Module"**.

Note:	In devices with more than one MSSP module, it is very important to pay close attention to SSPxCON register names. SSP1CON1 and SSP1CON2 control different operational aspects of the same module, while SSP1CON1 and SSP2CON1 control the same features for two different modules.

Note:	The SSPxBUF register cannot be used with read-modify-write instructions, such as BCF, COMF, etc. To avoid lost data in Master mode, a read of the SSPxBUF must be performed to clear the Buffer Full (BF) detect bit (SSPSTAT<0>) between each transmission.

20.3 SPI Mode

The SPI mode allows 8 bits of data to be synchronously transmitted and received simultaneously. All four modes of SPI are supported. To accomplish communication, three pins are typically used. These pins must be assigned through the PPS-Lite Configuration registers before use.

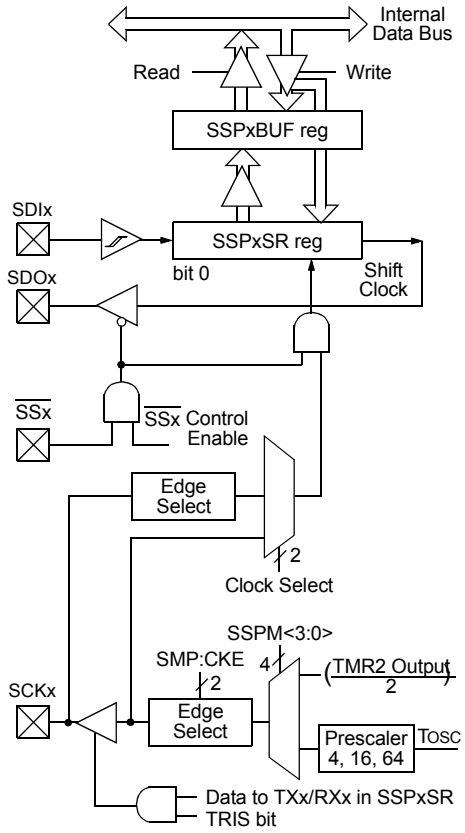
- Serial Data Out (SDOx) – Mapped to pin using PPS-Lite Peripheral Output registers
- Serial Data In (SDIx) – Mapped to pin using PPS-Lite Peripheral Input registers
- Serial Clock (SCKx) – Mapped to pin using PPS-Lite Peripheral Input registers (for Slave mode) or Peripheral Output registers (for Master mode).

Additionally, a fourth pin may be used when in a Slave mode of operation:

- Slave Select ($\overline{SSx}$) – Mapped through PPS-Lite Peripheral Input registers

Figure 20-1 shows the block diagram of the MSSPx module when operating in SPI mode.

FIGURE 20-1: MSSPx BLOCK DIAGRAM (SPI MODE)



Note: PPS-Lite signal names are used in this diagram for the sake of brevity. Refer to the text for a full list of multiplexed functions.

20.3.1 REGISTERS

Each MSSP module has four registers for SPI mode operation. These are:

- MSSPx Control Register 1 (SSPxCON1)
- MSSPx STATUS Register (SSPxSTAT)
- MSSPx Control Register 3 (SSPxCON3)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSPx Shift Register (SSPxSR) – Not directly accessible

SSPxCON1, SSPxCON3 and SSPxSTAT are the control and STATUS registers in SPI mode operation. The SSPxCON1 and SSPxCON3 registers are readable and writable. The lower 6 bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

SSPxSR is the shift register used for shifting data in or out. SSPxBUF is the buffer register to which data bytes are written to or read from.

In receive operations, SSPxSR and SSPxBUF together, create a double-buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not double-buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

REGISTER 20-1: SSPxSTAT: MSSPx STATUS REGISTER (SPI MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE(1)	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **SMP:** Sample bit
SPI Master mode:
1 = Input data is sampled at the end of data output time
0 = Input data is sampled at the middle of data output time
SPI Slave mode:
SMP must be cleared when SPI is used in Slave mode.

bit 6 **CKE:** SPI Clock Select bit(1)
1 = Transmit occurs on the transition from active to Idle clock state
0 = Transmit occurs on the transition from Idle to active clock state

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit
Used in I^2C mode only.

bit 4 **P:** Stop bit
Used in I^2C mode only. This bit is cleared when the MSSPx module is disabled; SSPEN is cleared.

bit 3 **S:** Start bit
Used in I^2C mode only.

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ Information bit
Used in I^2C mode only.

bit 1 **UA:** Update Address bit
Used in I^2C mode only.

bit 0 **BF:** Buffer Full Status bit (Receive mode only)
1 = Receive is complete, SSPxBUF is full
0 = Receive is not complete, SSPxBUF is empty

Note 1: Polarity of clock state is set by the CKP bit (SSPxCON1<4>).

REGISTER 20-2: SSPxCON1: MSSPx CONTROL REGISTER 1 (SPI MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV(1)	SSPEN(2)	CKP	SSPM3(4)	SSPM2(4)	SSPM1(4)	SSPM0(4)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **WCOL:** Write Collision Detect bit
- 1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
- 0 = No collision

bit 6 **SSPOV:** Receive Overflow Indicator bit(1)

SPI Slave mode:
- 1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. The user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).
- 0 = No overflow

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit(2)
- 1 = Enables serial port and configures SCKx, SDOx, SDIx and $\overline{SSx}$ as serial port pins
- 0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP:** Clock Polarity Select bit
- 1 = Idle state for the clock is a high level
- 0 = Idle state for the clock is a low level

bit 3-0 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits(4)
- 1010 = SPI Master mode: Clock = FOSC/(4 * (SSPxADD + 1)(3)
- 0101 = SPI Slave mode: Clock = SCKx pin; $\overline{SSx}$ pin control is disabled; $\overline{SSx}$ can be used as I/O pin
- 0100 = SPI Slave mode: Clock = SCKx pin; $\overline{SSx}$ pin control is enabled
- 0011 = SPI Master mode: Clock = TMR2 output/2
- 0010 = SPI Master mode: Clock = FOSC/64
- 0001 = SPI Master mode: Clock = FOSC/16
- 0000 = SPI Master mode: Clock = FOSC/4

Note 1: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.

2: When enabled, these pins must be properly configured as inputs or outputs.

3: SSPxADD = 0 is not supported.

4: Bit combinations not specifically listed here are either reserved or implemented in I^2C mode only.

REGISTER 20-3: SSPxCON3: MSSP CONTROL REGISTER 3 (SPI MODE)

R/HS/HC-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ACKTIM:** Acknowledge Time Status bit
Unused in SPI.

bit 6 **PCIE:** Stop Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Stop condition
0 = Stop detection interrupts are disabled

bit 5 **SCIE:** Start Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Start or Restart conditions
0 = Start detection interrupts are disabled

bit 4 **BOEN:** Buffer Overwrite Enable bit(2)
1 = SSPBUF updates every time a new data byte is shifted in, ignoring the BF bit
0 = If a new byte is received with BF bit already set, SSPOV is set, and the buffer is not updated

bit 3 **SDAHT:** SDA Hold Time Selection bit
Unused in SPI.

bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit
Unused in SPI.

bit 1 **AHEN:** Address Hold Enable bit
Unused in SPI.

bit 0 **DHEN:** Data Hold Enable bit
Unused in SPI.

Note 1: This bit has no effect in Slave modes that Start and Stop condition detection is explicitly listed as enabled.

2: For daisy-chained SPI operation; allows the user to ignore all but the last received byte. SSPOV is still set when a new byte is received and BF = 1, but hardware continues to write the most recent byte to SSPxBUF.

20.3.2 OPERATION

When initializing the SPI, several options need to be specified. This is done by programming the appropriate control bits (SSPxCON1<5:0> and SSPxSTAT<7:6>). These control bits allow the following to be specified:

- I/O pins must be mapped to the SPI peripheral in order to function. See **Section 11.15 "PPS-Lite"** for an explanation of the PPS-Lite mapping feature.
- Master mode (SCKx is the clock output)
- Slave mode (SCKx is the clock input)
- Clock Polarity (Idle state of SCKx)
- Data Input Sample Phase (middle or end of data output time)
- Clock Edge (output data on rising/falling edge of SCKx)
- Clock Rate (Master mode only)
- Slave Select mode (Slave mode only)

Each MSSPx module consists of a Transmit/Receive Shift register (SSPxSR) and a Buffer register (SSPxBUF). The SSPxSR shifts the data in and out of the device, MSb first. The SSPxBUF holds the data that was written to the SSPxSR until the received data is ready. Once the 8 bits of data have been received, that byte is moved to the SSPxBUF register. Then, the Buffer Full detect bit, BF (SSPxSTAT<0>), and the interrupt flag bit, SSPxIF, are set. This double-buffering of the received data (SSPxBUF) allows the next byte to start reception before reading the data that was just received. Any write to the SSPxBUF register during transmission/reception of data will be ignored and the Write Collision Detect bit, WCOL (SSPxCON1<7>), will be set. User software must clear the WCOL bit so that it can be determined if the following write(s) to the SSPxBUF register completed successfully.

When the application software is expecting to receive valid data, the SSPxBUF should be read before the next byte of data to transfer is written to the SSPxBUF. The Buffer Full bit, BF (SSPxSTAT<0>), indicates when SSPxBUF has been loaded with the received data (transmission is complete). When the SSPxBUF is read, the BF bit is cleared. This data may be irrelevant if the SPI is only a transmitter. Generally, the MSSPx interrupt is used to determine when the transmission/reception has completed. If the interrupt method is not going to be used, then software polling can be done to ensure that a write collision does not occur. Example 20-1 shows the loading of the SSPxBUF (SSPxSR) for data transmission.

The SSPxSR is not directly readable or writable and can only be accessed by addressing the SSPxBUF register. Additionally, the SSPxSTAT register indicates the various status conditions.

20.3.3 OPEN-DRAIN OUTPUT OPTION

The drivers for the SDOx output and SCKx clock pins can be optionally configured as open-drain outputs. This feature allows the voltage level on the pin to be pulled to a higher level through an external pull-up resistor, and allows the output to communicate with external circuits without the need for additional level shifters. For more information, see **Section 11.1.3 "Open-Drain Outputs"**.

The open-drain output option is controlled by the SSPxOD bits (ODCON1<1:0>). Setting an SSPxOD bit configures the SDOx and SCKx pins for the corresponding module for open-drain operation.

Note:	To avoid lost data in Master mode, a read of the SSPxBUF must be performed to clear the Buffer Full (BF) detect bit (SSPxSTAT<0>) between each transmission.

EXAMPLE 20-1: LOADING THE SSP1BUF (SSP1SR) REGISTER

```
LOOP    BTFSS   SSP1STAT, BF    ;Has data been received (transmit complete)?
        BRA     LOOP            ;No
        MOVF    SSP1BUF, W      ;WREG reg = contents of SSP1BUF
        MOVWF   RXDATA          ;Save in user RAM, if data is meaningful
        MOVF    TXDATA, W       ;W reg = contents of TXDATA
        MOVWF   SSP1BUF         ;New data to xmit
```

20.3.4 ENABLING SPI I/O

To enable the serial port, the peripheral must first be mapped to I/O pins using the PPS-Lite feature. To enable the SPI peripheral, the MSSPx Enable bit, SSPEN (SSPxCON1<5>) must be set. To reset or reconfigure SPI mode, clear the SSPEN bit, re-initialize the SSPxCON registers and then set the SSPEN bit. This configures the SDIx, SDOx, SCKx and $\overline{SSx}$ pins as serial port pins. For the pins to behave as the serial port function, some must have their data direction bits (in the TRIS register) appropriately programmed as follows:

- SDIx is automatically controlled by the SPI module
- SDOx must have the TRIS bit cleared for the corresponding RPn pin.
- SCKx (Master mode) must have the TRIS bit cleared for the corresponding RPn pin
- SCKx (Slave mode) must have the TRIS bit set for the corresponding RPn pin
- $\overline{SSx}$ must have the TRIS bit set for the corresponding RPn pin.

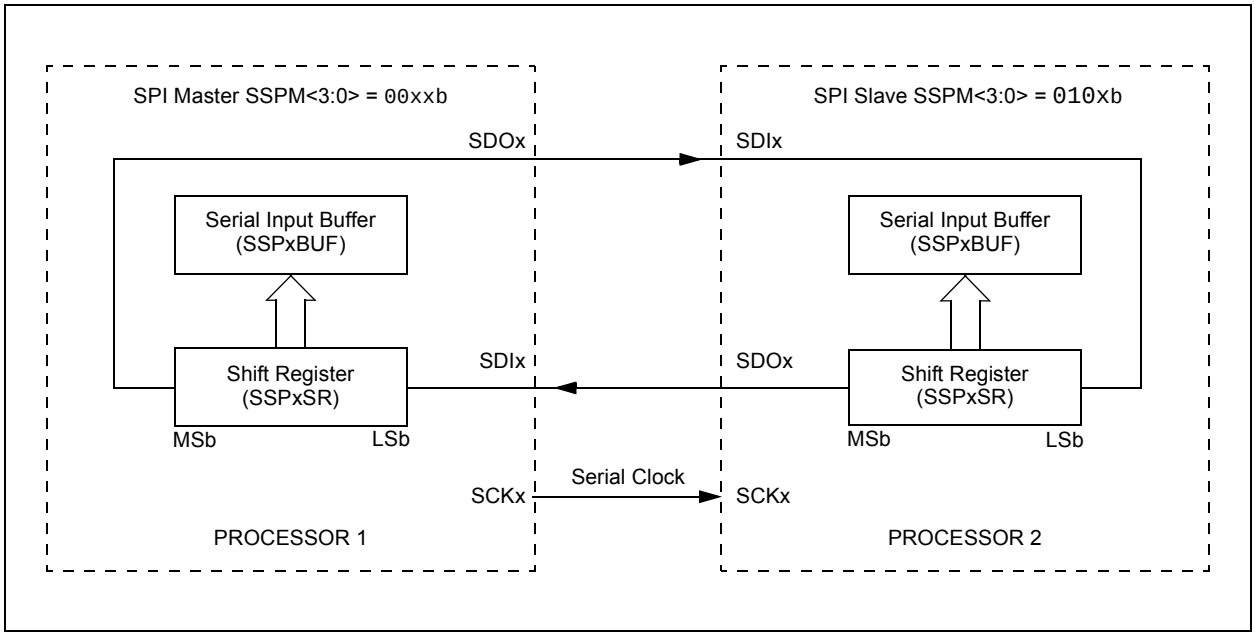
Any serial port function that is not desired may be overridden by programming the corresponding Data Direction (TRIS) register to the opposite value.

20.3.5 TYPICAL CONNECTION

Figure 20-2 shows a typical connection between two microcontrollers. The master controller (Processor 1) initiates the data transfer by sending the SCKx signal. Data is shifted out of both shift registers on their programmed clock edge and latched on the opposite edge of the clock. Both processors should be programmed to the same Clock Polarity (CKP), then both controllers would send and receive data at the same time. Whether the data is meaningful (or dummy data) depends on the application software. This leads to three scenarios for data transmission:

- Master sends data – Slave sends dummy data
- Master sends data – Slave sends data
- Master sends dummy data – Slave sends data

FIGURE 20-2: SPI MASTER/SLAVE CONNECTION



20.3.6 MASTER MODE

The master can initiate the data transfer at any time because it controls the SCKx signal. The master determines when the slave (Processor 2, Figure 20-2) is to broadcast data by the software protocol.

In Master mode, the data is transmitted/received as soon as the SSPxBUF register is written to. If the SPI is only going to receive, the SDOx output could be disabled (programmed as an input). The SSPxSR register will continue to shift in the signal present on the SDIx pin at the programmed clock rate. As each byte is received, it will be loaded into the SSPxBUF register as if a normal received byte (interrupts and Status bits appropriately set). This could be useful in receiver applications as a “Line Activity Monitor” mode.

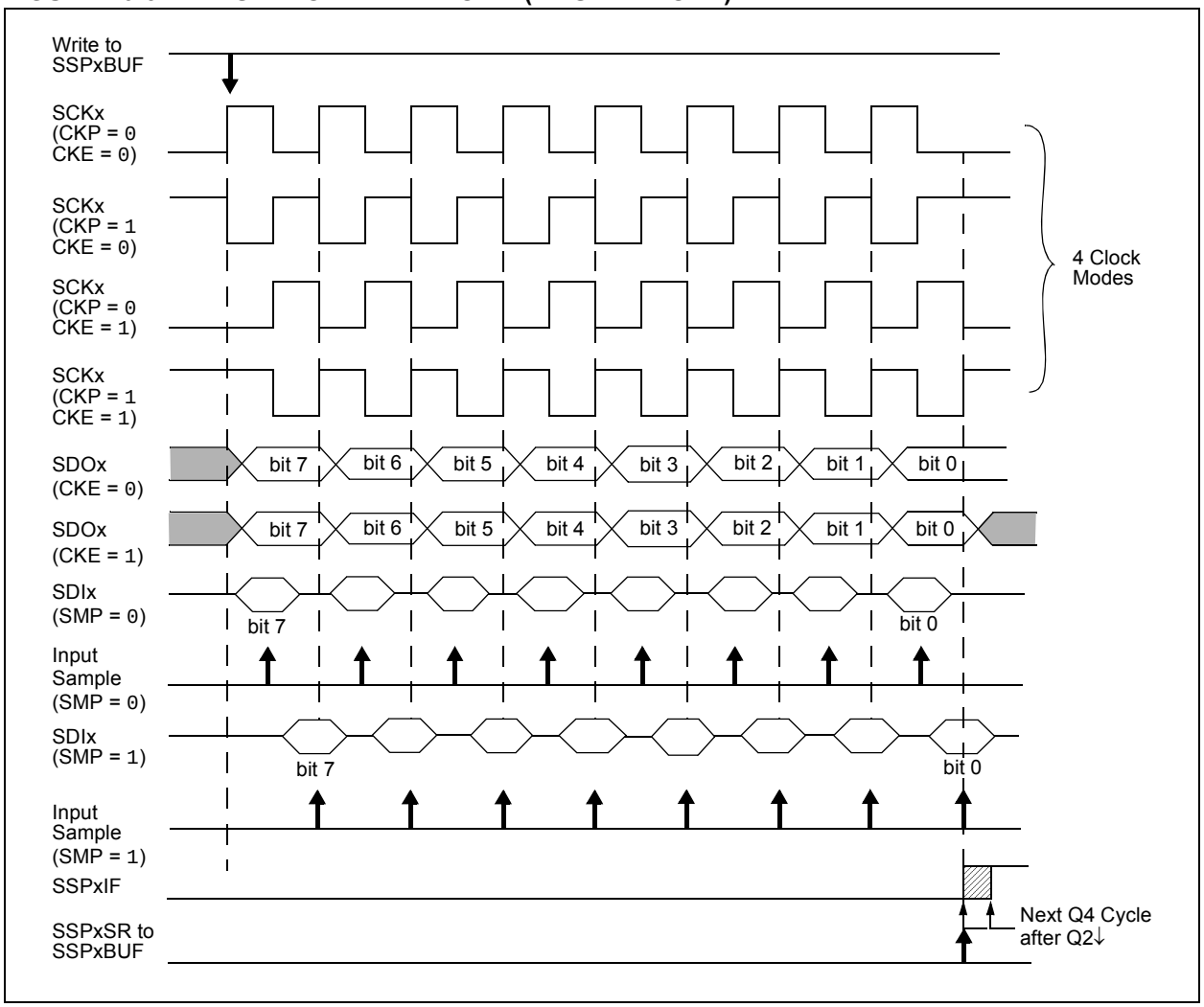
The clock polarity is selected by appropriately programming the CKP bit (SSPxCON1<4>). This, then, would give waveforms for SPI communication, as shown in Figure 20-3, Figure 20-5 and Figure 20-6, where the MSB is transmitted first. In Master mode, the SPI clock rate (bit rate) is user-programmable to be one of the following:

- FOSC/4 (or TCY)
- FOSC/(4 * (SSPxADD + 1)
- FOSC/16 (or 4 • TCY)
- FOSC/64 (or 16 • TCY)
- Timer2 output/2

This allows a maximum data rate (at 64 MHz) of 16.00 Mbps.

Figure 20-3 shows the waveforms for Master mode. When the CKE bit is set, the SDOx data is valid before there is a clock edge on SCKx. The change of the input sample is shown based on the state of the SMP bit. The time when the SSPxBUF is loaded with the received data is shown.

FIGURE 20-3: SPI MODE WAVEFORM (MASTER MODE)



20.3.7 SLAVE MODE

In Slave mode, the data is transmitted and received as the external clock pulses appear on SCKx. When the last bit is latched, the SSPxIF interrupt flag bit is set.

While in Slave mode, the external clock is supplied by the external clock source on the SCKx pin. This external clock must meet the minimum high and low times as specified in the electrical specifications.

While in Sleep mode, the slave can transmit/receive data. When a byte is received, the device can be configured to wake-up from Sleep.

20.3.8 SLAVE SELECT SYNCHRONIZATION

The $\overline{SSx}$ pin allows a Synchronous Slave mode. The SPI must be in Slave mode with the $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 04h). When the $\overline{SSx}$ pin is low, transmission and reception are enabled and the SDOx pin is driven. When the $\overline{SSx}$ pin goes high, the SDOx pin is no longer driven, even if in the middle of a transmitted byte and becomes a floating output. External pull-up/pull-down resistors may be desirable depending on the application.

Note:	When the SPI is in Slave mode with $\overline{SSx}$ pin control enabled (SSPxCON1<3:0> = 0100), the SPI module will reset if the $\overline{SSx}$ pin is set to VDD. If the SPI is used in Slave mode with CKE set, then the $\overline{SSx}$ pin control must be enabled.

When the SPI module resets, the bit counter is forced to '0'. This can be done by either forcing the $\overline{SSx}$ pin to a high level or clearing the SSPEN bit.

To emulate two-wire communication, the SDOx pin can be connected to the SDIx pin. When the SPI needs to operate as a receiver, the SDOx pin can be configured as an input. This disables transmissions from the SDOx. The SDIx can always be left as an input (SDIx function) since it cannot create a bus conflict.

FIGURE 20-4: SLAVE SYNCHRONIZATION WAVEFORM

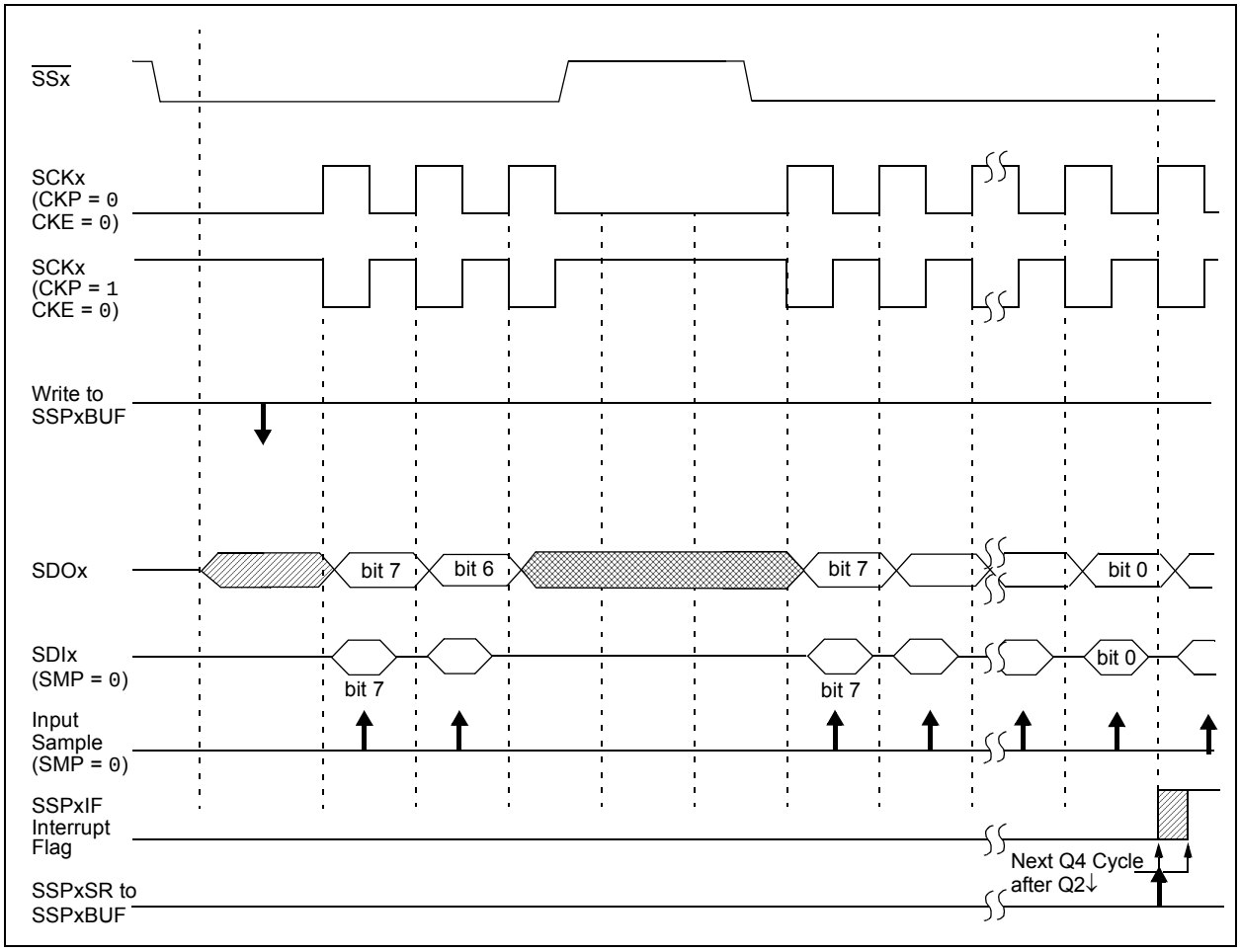


FIGURE 20-5: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 0)

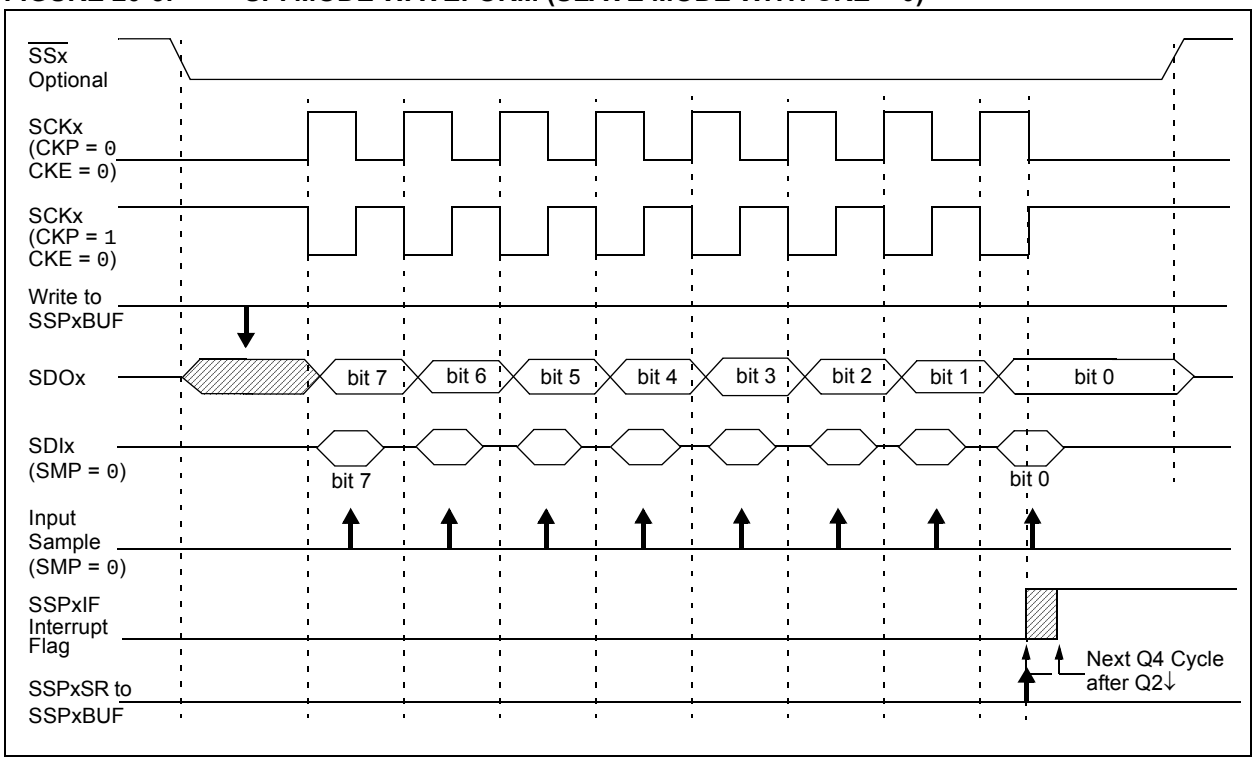
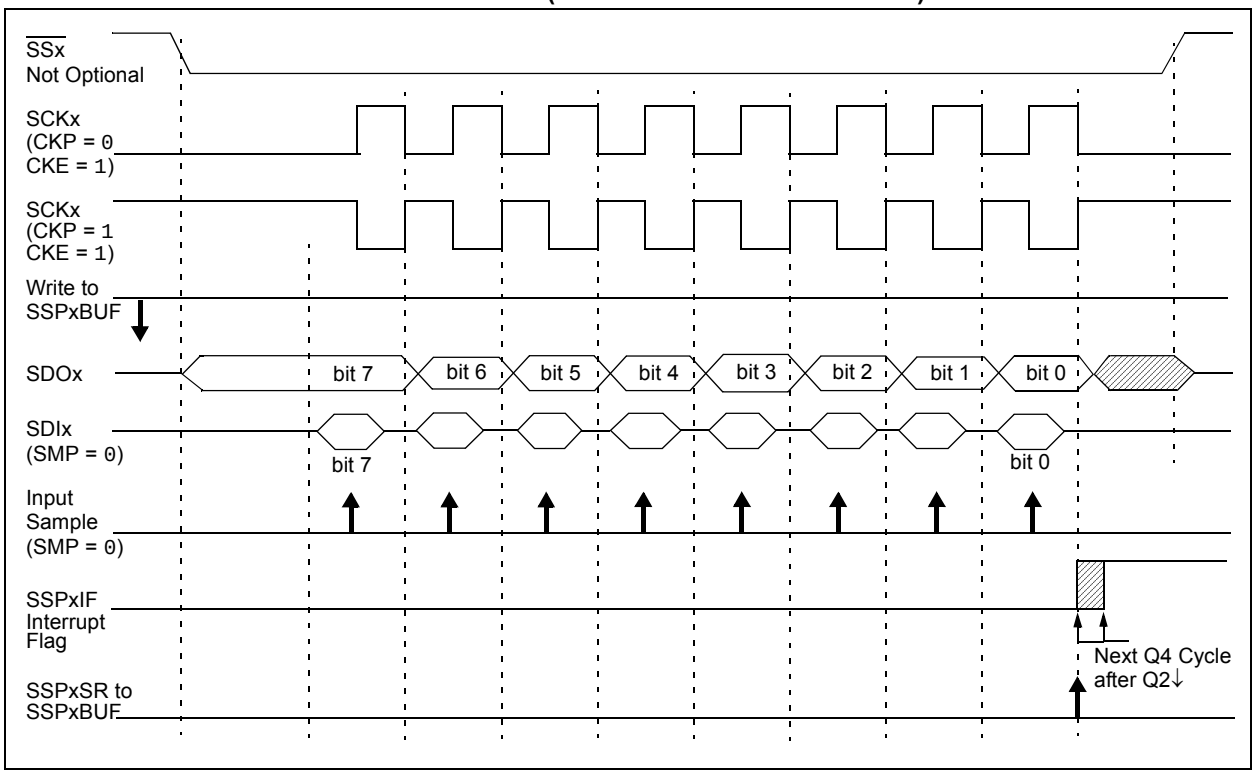


FIGURE 20-6: SPI MODE WAVEFORM (SLAVE MODE WITH CKE = 1)



20.3.9 OPERATION IN POWER-MANAGED MODES

In SPI Master mode, module clocks may be operating at a different speed than when in full-power mode. In the case of Sleep mode, all clocks are halted.

In Idle modes, a clock is provided to the peripherals. That clock can be from the primary clock source, the secondary clock (SOSC Oscillator) or the INTOSC source.

In most cases, the speed that the master clocks SPI data is not important; however, this should be evaluated for each system.

If MSSPx interrupts are enabled, they can wake the controller from Sleep mode, or one of the Idle modes, when the master completes sending data. If an exit from Sleep or Idle mode is not desired, MSSPx interrupts should be disabled.

If the Sleep mode is selected, all module clocks are halted and the transmission/reception will remain in that state until the device wakes. After the device returns to Run mode, the module will resume transmitting and receiving data.

In SPI Slave mode, the SPI Transmit/Receive Shift register operates asynchronously to the device. This allows the device to be placed in any power-managed mode and data to be shifted into the SPI Transmit/Receive Shift register. When all 8 bits have been received, the MSSPx interrupt flag bit will be set, and if enabled, will wake the device.

20.3.10 EFFECTS OF A RESET

A Reset disables the MSSPx module and terminates the current transfer.

20.3.11 BUS MODE COMPATIBILITY

Table 20-1 shows the compatibility between the standard SPI modes and the states of the CKP and CKE control bits.

TABLE 20-1: SPI BUS MODES

Standard SPI Mode Terminology	Control Bits State	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

There is also an SMP bit which controls when the data is sampled.

20.3.12 SPI CLOCK SPEED AND MODULE INTERACTIONS

Because MSSP1 and MSSP2 are independent modules, they can operate simultaneously at different data rates. Setting the SSPM<3:0> bits of the SSPxCON1 register determines the rate for the corresponding module.

An exception is when both modules use Timer2 as a time base in Master mode. In this instance, any changes to the Timer2 module's operation will affect both MSSPx modules equally. If different bit rates are required for each module, the user should select one of the other three time base options for one of the modules.

20.4 SPI DMA MODULE

The SPI DMA module contains control logic to allow the MSSP1 module to perform SPI Direct Memory Access transfers. This enables the module to quickly transmit or receive large amounts of data with relatively little CPU intervention. When the SPI DMA module is used, MSSP1 can directly read and write to general purpose SRAM. When the SPI DMA module is not enabled, MSSP1 functions normally, but without DMA capability.

The SPI DMA module is composed of control logic, a Destination Receive Address Pointer, a Transmit Source Address Pointer, an interrupt manager and a Byte Count register for setting the size of each DMA transfer. The DMA module may be used with all SPI Master and Slave modes, and supports both half-duplex and full-duplex transfers.

20.4.1 I/O PIN CONSIDERATIONS

When enabled, the SPI DMA module uses the MSSP1 module. All SPI input and output signals, related to MSSP1, are routed through the Peripheral Pin Select (PPS) module. The appropriate initialization procedure, as described in **Section 20.4.6 "Using the SPI DMA Module"**, will need to be followed prior to using the SPI DMA module. The output pins assigned to the SDO and SCK functions can optionally be configured as open-drain outputs, such as for level shifting operations mentioned in the same section.

20.4.2 RAM TO RAM COPY OPERATIONS

Although the SPI DMA module is primarily intended to be used for SPI communication purposes, the module can also be used to perform RAM to RAM copy operations. To do this, configure the module for Full-Duplex Master mode operation, but assign the SDO output and SDI input functions onto the same RPn pin in the PPS-Lite module. Also assign SCK out and SCK in onto the same RPn pin (a different pin than used for SDO and SDI). This will allow the module to operate in Loopback mode, providing RAM copy capability.

20.4.3 IDLE AND SLEEP CONSIDERATIONS

The SPI DMA module remains fully functional when the microcontroller is in Idle mode.

During normal Sleep, the SPI DMA module is not functional and should not be used. To avoid corrupting a transfer, user firmware should be careful to make certain that pending DMA operations are complete by polling the DMAEN bit in the DMACON1 register, prior to putting the microcontroller into Sleep.

In SPI Slave modes, the MSSP1 module is capable of transmitting and/or receiving one byte of data while in Sleep mode. This allows the SSP1IF flag in the PIR1 register to be used as a wake-up source. When the DMAEN bit is cleared, the SPI DMA module is effectively disabled, and the MSSP1 module functions normally, but without DMA capabilities. If the DMAEN bit is clear prior to entering Sleep, it is still possible to use the SSP1IF as a wake-up source without any data loss.

Neither MSSP1 nor the SPI DMA module will provide any functionality in Deep Sleep. Upon exiting from Deep Sleep, all of the I/O pins, MSSP1 and SPI DMA related registers will need to be fully re-initialized before the SPI DMA module can be used again.

20.4.4 REGISTERS

The SPI DMA engine is enabled and controlled by the following Special Function Registers:

- DMACON1
- DMACON2
- TXADDRH
- TXADDRL
- RXADDRH
- RXADDRL
- DMABCH
- DMABCL

20.4.4.1 DMACON1

The DMACON1 register is used to select the main operating mode of the SPI DMA module. The SSCON1 and SSCON0 bits are used to control the slave select pin.

When MSSP1 is used in SPI Master mode with the SPI DMA module, SSDMA can be controlled by the DMA module as an output pin. If MSSP1 will be used to communicate with an SPI slave device that needs the SSx pin to be toggled periodically, the SPI DMA hardware can automatically be used to de-assert SSx between each byte, every two bytes or every four bytes.

Alternatively, user firmware can manually generate slave select signals with normal general purpose I/O pins, if required by the slave device(s).

When the TXINC bit is set, the TXADDR register will automatically increment after each transmitted byte. Automatic transmit address increment can be disabled by clearing the TXINC bit. If the automatic transmit address increment is disabled, each byte which is output on SDO will be the same (the contents of the SRAM pointed to by the TXADDR register) for the entire DMA transaction.

When the RXINC bit is set, the RXADDR register will automatically increment after each received byte. Automatic receive address increment can be disabled by clearing the RXINC bit. If RXINC is disabled in Full-Duplex or Half-Duplex Receive modes, all incoming data bytes on SDI will overwrite the same memory location pointed to by the RXADDR register. After the SPI DMA transaction has completed, the last received byte will reside in the memory location pointed to by the RXADDR register.

The SPI DMA module can be used for either half-duplex receive only communication, half-duplex transmit only communication or full-duplex simultaneous transmit and receive operations. All modes are available for both SPI master and SPI slave configurations. The DUPLEX0 and DUPLEX1 bits can be used to select the desired operating mode.

The behavior of the DLYINTEN bit varies greatly depending on the SPI operating mode. For example behavior for each of the modes, see Figure 20-3 through Figure 20-6.

SPI Slave mode, DLYINTEN = 1: In this mode, an SSP1IF interrupt will be generated during a transfer if the time between successful byte transmission events is longer than the value set by the DLYCYC<3:0> bits in the DMACON2 register. This interrupt allows slave firmware to know that the master device is taking an unusually large amount of time between byte transmissions. For example, this information may be useful for implementing application defined communication protocols, involving time-outs if the bus remains Idle for too long. When DLYINTEN = 1, the DLYLVL<3:0> interrupts occur normally according to the selected setting.

SPI Slave mode, DLYINTEN = 0: In this mode, the time-out based interrupt is disabled. No additional SSP1IF interrupt events will be generated by the SPI DMA module, other than those indicated by the INTLVL<3:0> bits in the DMACON2 register. In this mode, always set DLYCYC<3:0> = 0000.

SPI Master mode, DLYINTEN = 0: The DLYCYC<3:0> bits in the DMACON2 register determine the amount of additional inter-byte delay, which is added by the SPI DMA module during a transfer; the Master mode SS1 output feature may be used.

SPI Master mode, DLYINTEN = 1: The amount of hardware overhead is slightly reduced in this mode, and the minimum inter-byte delay is 8 TCY for FOSC/4, 9 TCY for FOSC/16 and 15 TCY for FOSC/64. This mode can potentially be used to obtain slightly higher effective SPI bandwidth. In this mode, the SS1 control feature cannot be used and should always be disabled (DMACON1<7:6> = 00). Additionally, the interrupt generating hardware (used in Slave mode) remains active. To avoid extraneous SSP1IF interrupt events, set the DMACON2 Delay bits, DLYCYC<3:0> = 1111, and ensure that the SPI serial clock rate is no slower than FOSC/64.

In SPI Master modes, the DMAEN bit is used to enable the SPI DMA module and to initiate an SPI DMA transaction. After user firmware sets the DMAEN bit, the DMA hardware will begin transmitting and/or receiving data bytes according to the configuration used. In SPI Slave modes, setting the DMAEN bit will finish the initialization steps needed to prepare the SPI DMA module for communication (which must still be initiated by the master device).

To avoid possible data corruption, once the DMAEN bit is set, user firmware should not attempt to modify any of the MSSP2 or SPI DMA related registers, with the exception of the INTLVLx bits in the DMACON2 register.

If user firmware wants to halt an ongoing DMA transaction, the DMAEN bit can be manually cleared by the firmware. Clearing the DMAEN bit while a byte is currently being transmitted will not immediately halt the byte in progress. Instead, any byte currently in progress will be completed before the MSSP1 and SPI DMA modules go back to their Idle conditions. If user firmware clears the DMAEN bit, the TXADDR, RXADDR and DMABC registers will no longer update, and the DMA module will no longer make any additional read or writes to SRAM; therefore, state information can be lost.

REGISTER 20-4: DMACON1: DMA CONTROL REGISTER 1

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
SSCON1	SSCON0	TXINC	RXINC	DUPLEX1	DUPLEX0	DLYINTEN	DMAEN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **SSCON<1:0>:** $\overline{\text{SSDMA}}$ Output Control bits (Master modes only)
11 = $\overline{\text{SSDMA}}$ is asserted for the duration of 4 bytes; DLYINTEN is always reset low
01 = $\overline{\text{SSDMA}}$ is asserted for the duration of 2 bytes; DLYINTEN is always reset low
10 = $\overline{\text{SSDMA}}$ is asserted for the duration of 1 byte; DLYINTEN is always reset low
00 = $\overline{\text{SSDMA}}$ is not controlled by the DMA module; DLYINTEN bit is software programmable

bit 5 **TXINC:** Transmit Address Increment Enable bit
Allows the transmit address to increment as the transfer progresses.
1 = The transmit address is to be incremented from the initial value of TXADDR<11:0>
0 = The transmit address is always set to the initial value of TXADDR<11:0>

bit 4 **RXINC:** Receive Address Increment Enable bit
Allows the receive address to increment as the transfer progresses.
1 = The received address is to be incremented from the initial value of RXADDR<11:0>
0 = The received address is always set to the initial value of RXADDR<11:0>

bit 3-2 **DUPLEX<1:0>:** Transmit/Receive Operating Mode Select bits
10 = SPI DMA operates in Full-Duplex mode, data is simultaneously transmitted and received
01 = DMA operates in Half-Duplex mode, data is transmitted only
00 = DMA operates in Half-Duplex mode, data is received only

bit 1 **DLYINTEN:** Delay Interrupt Enable bit
Enables the interrupt to be invoked after the number of TCY cycles, specified in DLYCYC<3:0>, has elapsed from the latest completed transfer.
1 = The interrupt is enabled, SSCON<1:0> must be set to '00'
0 = The interrupt is disabled

bit 0 **DMAEN:** DMA Operation Start/Stop bit
This bit is set by the users' software to start the DMA operation. It is reset back to zero by the DMA engine when the DMA operation is completed or aborted.
1 = DMA is in session
0 = DMA is not in session

20.4.4.2 DMACON2

The DMACON2 register contains control bits for controlling interrupt generation and inter-byte delay behavior. The INTLVL<3:0> bits are used to select when an SSP1IF interrupt should be generated. The function of the DLYCYC<3:0> bits depends on the SPI operating mode (Master/Slave), as well as the DLYINTEN setting. In SPI Master mode, the DLYCYC<3:0> bits can be used to control how much time the module will Idle between bytes in a transfer. By default, the hardware requires a minimum delay of 8 TCY for FOSC/4, 9 TCY for FOSC/16 and 15 TCY for FOSC/64. An additional delay can be added with the DLYCYCx bits. In SPI Slave modes, the DLYCYC<3:0> bits may optionally be used to trigger an additional time-out based interrupt.

REGISTER 20-5: DMACON2: DMA CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
DLYCYC3	DLYCYC2	DLYCYC1	DLYCYC0	INTLVL3	INTLVL2	INTLVL1	INTLVL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-4 **DLYCYC<3:0>:** Delay Cycle Selection bits

When DLYINTEN = 0, these bits specify the additional delay (above the base overhead of the hardware), in number of TCY cycles, before the SSP2BUF register is written again for the next transfer.

When DLYINTEN = 1, these bits specify the delay in number of TCY cycles from the latest completed transfer before an interrupt to the CPU is invoked. In this case, the additional delay before the SSP2BUF register is written again is 1 TCY + (base overhead of hardware).

1111 = Delay time in number of instruction cycles is 2,048 cycles
1110 = Delay time in number of instruction cycles is 1,024 cycles
1101 = Delay time in number of instruction cycles is 896 cycles
1100 = Delay time in number of instruction cycles is 768 cycles
1011 = Delay time in number of instruction cycles is 640 cycles
1010 = Delay time in number of instruction cycles is 512 cycles
1001 = Delay time in number of instruction cycles is 384 cycles
1000 = Delay time in number of instruction cycles is 256 cycles
0111 = Delay time in number of instruction cycles is 128 cycles
0110 = Delay time in number of instruction cycles is 64 cycles
0101 = Delay time in number of instruction cycles is 32 cycles
0100 = Delay time in number of instruction cycles is 16 cycles
0011 = Delay time in number of instruction cycles is 8 cycles
0010 = Delay time in number of instruction cycles is 4 cycles
0001 = Delay time in number of instruction cycles is 2 cycles
0000 = Delay time in number of instruction cycles is 1 cycle

REGISTER 20-5: DMACON2: DMA CONTROL REGISTER 2 (CONTINUED)

bit 3-0 **INTLVL<3:0>:** Watermark Interrupt Enable bits
These bits specify the amount of remaining data yet to be transferred (transmitted and/or received) upon which an interrupt is generated.

1111 = Amount of remaining data to be transferred is 576 bytes
1110 = Amount of remaining data to be transferred is 512 bytes
1101 = Amount of remaining data to be transferred is 448 bytes
1100 = Amount of remaining data to be transferred is 384 bytes
1011 = Amount of remaining data to be transferred is 320 bytes
1010 = Amount of remaining data to be transferred is 256 bytes
1001 = Amount of remaining data to be transferred is 192 bytes
1000 = Amount of remaining data to be transferred is 128 bytes
0111 = Amount of remaining data to be transferred is 67 bytes
0110 = Amount of remaining data to be transferred is 32 bytes
0101 = Amount of remaining data to be transferred is 16 bytes
0100 = Amount of remaining data to be transferred is 8 bytes
0011 = Amount of remaining data to be transferred is 4 bytes
0010 = Amount of remaining data to be transferred is 2 bytes
0001 = Amount of remaining data to be transferred is 1 byte
0000 = Transfer complete

20.4.4.3 DMABCH and DMABCL

The DMABCH and DMABCL register pair forms a 10-bit Byte Count register, which is used by the SPI DMA module to send/receive up to 1,024 bytes for each DMA transaction. When the DMA module is actively running (DMAEN = 1), the DMA Byte Count register decrements after each byte is transmitted/received. The DMA transaction will halt and the DMAEN bit will be automatically cleared by hardware after the last byte has completed. After a DMA transaction is complete, the DMABC register will read 0x000.

Prior to initiating a DMA transaction by setting the DMAEN bit, user firmware should load the appropriate value into the DMABCH/DMABCL registers. The DMABC is a “base zero” counter, so the actual number of bytes which will be transmitted follows in Equation 20-1.

For example, if user firmware wants to transmit 7 bytes in one transaction, DMABC should be loaded with 006h. Similarly, if user firmware wishes to transmit 1,024 bytes, DMABC should be loaded with 3FFh.

EQUATION 20-1: BYTES TRANSMITTED FOR A GIVEN DMABC

$$Bytes_{XMIT} = (DMABC + 1)$$

20.4.4.4 TXADDRH and TXADDRL

The TXADDRH and TXADDRL registers pair together to form a 12-bit Transmit Source Address Pointer register. In modes that use TXADDR (Full-Duplex and Half-Duplex Transmit), the TXADDR will be incremented after each byte is transmitted. Transmitted data bytes will be taken from the memory location pointed to by the TXADDR register. The contents of the memory locations pointed to by TXADDR will not be modified by the DMA module during a transmission.

The SPI DMA module can read from, and transmit data from, all general purpose memory on the device, including memory used for USB endpoint buffers. The SPI DMA module cannot be used to read from the Special Function Registers (SFRs) contained in Banks 14 and 15.

20.4.4.5 RXADDRH and RXADDRL

The RXADDRH and RXADDRL registers pair together to form a 12-bit Receive Destination Address Pointer. In modes that use RXADDR (Full-Duplex and Half-Duplex Receive), the RXADDR register will be incremented after each byte is received. Received data bytes will be stored at the memory location pointed to by the RXADDR register.

The SPI DMA module can write received data to all general purpose memory on the device, including memory used for USB endpoint buffers. The SPI DMA module cannot be used to modify the Special Function Registers contained in Banks 14 and 15.

20.4.5 INTERRUPTS

The SPI DMA module alters the behavior of the SSP1IF interrupt flag. In normal non-DMA modes, the SSP1IF is set once after every single byte is transmitted/received through the MSSP1 module. When MSSP1 is used with the SPI DMA module, the SSP1IF interrupt flag will be set according to the user-selected INTLVL<3:0> value specified in the DMACON2 register. The SSP1IF interrupt condition will also be generated once the SPI DMA transaction has fully completed and the DMAEN bit has been cleared by hardware.

The SSP1IF flag becomes set once the DMA byte count value indicates that the specified INTLVLx has been reached. For example, if DMACON2<3:0> = 0101 (16 bytes remaining), the SSP1IF interrupt flag will become set once DMABC reaches 00Fh. If user firmware then clears the SSP1IF interrupt flag, the flag will not be set again by the hardware until after all bytes have been fully transmitted and the DMA transaction is complete.

Note:	User firmware may modify the INTLVLx bits while a DMA transaction is in progress (DMAEN = 1). If an INTLVLx value is selected which is higher than the actual remaining number of bytes (indicated by DMABC + 1), the SSP1IF interrupt flag will immediately become set.

For example, if DMABC = 00Fh (implying 16 bytes are remaining) and user firmware writes '1111' to INTLVL<3:0> (interrupt when 576 bytes are remaining), the SSP1IF interrupt flag will immediately become set. If user firmware clears this interrupt flag, a new interrupt condition will not be generated until either: user firmware again writes INTLVLx with an interrupt level higher than the actual remaining level, or the DMA transaction completes and the DMAEN bit is cleared.

Note:	If the INTLVLx bits are modified while a DMA transaction is in progress, care should be taken to avoid inadvertently changing the DLYCYC<3:0> value.

20.4.6 USING THE SPI DMA MODULE

The following steps would typically be taken to enable and use the SPI DMA module:

1. Configure the I/O pins, which will be used by MSSP2:
   a) Assign SCK1, SDO1, SDI1 and $\overline{SS1}$ to the RPn pins, as appropriate for the SPI mode which will be used. Only functions which will be used need to be assigned to a pin.
   b) Initialize the associated LATx registers for the desired Idle SPI bus state.
   c) If Open-Drain Output mode on SDO1 and SCK1 (Master mode) is desired, set ODCON1<1>.
   d) Configure the corresponding TRISx bits for each I/O pin used.
2. Configure and enable MSSP1 for the desired SPI operating mode:
   a) Select the desired operating mode (Master or Slave, SPI Mode 0, 1, 2 and 3) and configure the module by writing to the SSP1STAT and SSP1CON1 registers.
   b) Enable MSSP1 by setting SSP1CON1<5> = 1.
3. Configure the SPI DMA engine:
   a) Select the desired operating mode by writing the appropriate values to DMACON2 and DMACON1.
   b) Initialize the TXADDRH/TXADDRL Pointer (Full-Duplex or Half-Duplex Transmit Only mode).
   c) Initialize the RXADDRH/RXADDRL Pointer (Full-Duplex or Half-Duplex Receive Only mode).
   d) Initialize the DMABCH/DMABCL Byte Count register with the number of bytes to be transferred in the next SPI DMA operation.
   e) Set the DMAEN bit (DMACON1<0>).

   In SPI Master modes, this will initiate a DMA transaction. In SPI Slave modes, this will complete the initialization process, and the module will now be ready to begin receiving and/or transmitting data to the master device once the master starts the transaction.
4. Detect the SSP1IF interrupt condition (PIR1<3):
   a) If the interrupt was configured to occur at the completion of the SPI DMA transaction, the DMAEN bit (DMACON1<0>) will be clear. User firmware may prepare the module for another transaction by repeating Steps 3.b through 3.e.
   b) If the interrupt was configured to occur prior to the completion of the SPI DMA transaction, the DMAEN bit may still be set, indicating the transaction is still in progress. User firmware would typically use this interrupt condition to begin preparing new data for the next DMA transaction. Firmware should not repeat Steps 3.b. through 3.e. until the DMAEN bit is cleared by the hardware, indicating the transaction is complete.

Example 20-3 provides example code, demonstrating the initialization process and the steps needed to use the SPI DMA module to perform a 512-byte Full-Duplex Master mode transfer.

EXAMPLE 20-2: 512-BYTE SPI MASTER MODE INIT AND TRANSFER

```
                                    ;For this example, let's use RP3(RA3) for SCK1,
                                    ;RP1(RA1) for SDO1, and RP0(RA0) for SDI1

                                    ;Let's use SPI master mode, CKE = 0, CKP = 0,
                                    ;without using slave select signalling.

InitSPIPins:
    movlb       0x0E                ;Select bank 14, for access to ODCON1 register
    bcf         ODCON1, SSP1_OD     ;Let's not use open drain outputs in this example

    bcf         LATA, RA3           ;Initialize our (to be) SCK1 pin low (idle).
    bcf         LATA, RA1           ;Initialize our (to be) SDO1 pin to an idle state
    bcf         TRISA, RA1          ;Make SDO1 output, and drive low
    bcf         TRISA, RA3          ;Make SCK1 output, and drive low (idle state)
    bsf         TRISA, RA0          ;SDI2 is an input, make sure it is tri-stated

                                    ;Now we should unlock the PPS-Lite registers, so we can
                                    ;assign the MSSP2 functions to our desired I/O pins.

    movlb       0x0F                ;Select bank 15 for access to PPS-Lite registers
    bcf         INTCON, GIE         ;I/O Pin unlock sequence will not work if CPU
                                    ;services an interrupt during the sequence
    movlw       0x55                ;Unlock sequence consists of writing 0x55
    movwf       EECON2              ;and 0xAA to the EECON2 register.
    movlw       0xAA
    movwf       EECON2
    bcf         OSCCON2, IOLOCK     ;We may now write to RPINRx and RPORx registers
    bsf         INTCON, GIE         ;May now turn back on interrupts if desired

    movlw       0x00                ;RP0 will be SDI1
    movwf       RPINR8-9            ;Assign the SDI1 function to pin RP0

    movlw       0x30                ;Let's assign SCK1 output to pin RP3
    movwf       RPOR2_3             ;RPOR2_3 maps output signals to RP3 pin
    movlw       0x00                ;SCK1 also needs to be configured as an input on the same pin
    movwf       RPINR8_9            ;SCK1 input function taken from RP3 pin
    movlw       0x40                ;0x40 is SDO1 output
    movwf       RPOR0_1             ;Assign SDO1 output signal to the RP1 (RA1) pin
    movlb       0x0F                ;Done with PPS-Lite registers, bank 15 has other SFRs

InitMSSP2:
    clrf        SSP1STAT            ;CKE = 0, SMP = 0 (sampled at middle of bit)
    movlw       b'00000000'         ;CKP = 0, SPI Master mode, Fosc/4
    movwf       SSP1CON1            ;MSSP2 initialized
    bsf         SSP1CON1, SSPEN     ;Enable the MSSP2 module

InitSPIDMA:
    movlw       b'00111010'         ;Full duplex, RX/TXINC enabled, no SSCON
    movwf       DMACON1             ;DLYINTEN is set, so DLYCYC3:DLYCYC0 = 1111
    movlw       b'11110000'         ;Minimum delay between bytes, interrupt
    movwf       DMACON2             ;only once when the transaction is complete

                                    ;Somewhere else in our project, lets assume we have
                                    ;allocated some RAM for use as SPI receive and
                                    ;transmit buffers.
```

EXAMPLE 20-2: 512-BYTE SPI MASTER MODE INIT AND TRANSFER (CONTINUED)

```
;           udata   0x500
;DestBuf    res     0x200               ;Let's reserve 0x500-0x6FF for use as our SPI
;                                       ;receive data buffer in this example
;SrcBuf     res     0x200               ;Lets reserve 0x700-0x8FF for use as our SPI
;                                       ;transmit data buffer in this example

PrepareTransfer:
            movlw   HIGH(DestBuf)       ;Get high byte of DestBuf address (0x05)
            movwf   RXADDRH             ;Load upper four bits of the RXADDR register
            movlw   LOW(DestBuf)        ;Get low byte of the DestBuf address (0x00)
            movwf   RXADDRL             ;Load lower eight bits of the RXADDR register

            movlw   HIGH(SrcBuf)        ;Get high byte of SrcBuf address (0x07)
            movwf   TXADDRH             ;Load upper four bits of the TXADDR register
            movlw   LOW(SrcBuf)         ;Get low byte of the SrcBuf address (0x00)
            movwf   TXADDRL             ;Load lower eight bits of the TXADDR register

            movlw   0x01                ;Lets move 0x200 (512) bytes in one DMA xfer
            movwf   DMABCH              ;Load the upper two bits of DMABC register
            movlw   0xFF                ;Actual bytes transferred is (DMABC + 1), so
            movwf   DMABCL              ;we load 0x01FF into DMABC to xfer 0x200 bytes

BeginXfer:
            bsf     DMACON1, DMAEN      ;The SPI DMA module will now begin transferring
                                        ;the data taken from SrcBuf, and will store
                                        ;received bytes into DestBuf.

    ;Execute whatever                   ;CPU is now free to do whatever it wants to
                                        ;and the DMA operation will continue without
                                        ;intervention, until it completes.

                                        ;When the transfer is complete, the SSP2IF flag in
                                        ;the PIR3 register will become set, and the DMAEN bit
                                        ;is automatically cleared by the hardware.
                                        ;The DestBuf (0x500-0x7FF) will contain the received
                                        ;data.  To start another transfer, firmware will need
                                        ;to reinitialize RXADDR, TXADDR, DMABC and then
                                        ;set the DMAEN bit.
```

20.5 I²C Mode

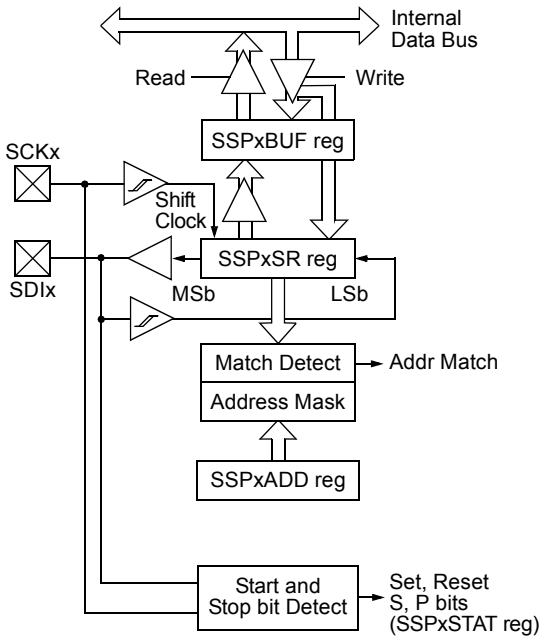
The MSSPx module in I²C mode fully implements all master and slave functions (including general call support), and provides interrupts on Start and Stop bits in hardware to determine a free bus (multi-master function). The MSSPx module implements the standard mode specifications, as well as 7-bit and 10-bit addressing.

Two pins are used for data transfer:

- Serial Clock (SCLx) – RC3/SCL1 or RD6/SCL2
- Serial Data (SDAx) – RC4/SDA1 or RD5/SDA2

The user must configure these pins as inputs by setting the associated TRIS bits.

FIGURE 20-7: MSSPx BLOCK DIAGRAM (I²C MODE)



Note: Only port I/O names are used in this diagram for the sake of brevity. Refer to the text for a full list of multiplexed functions.

20.5.1 REGISTERS

The MSSPx module has seven registers for I²C operation. These are:

- MSSPx Control Register 1 (SSPxCON1)
- MSSPx Control Register 2 (SSPxCON2)
- MSSPx Control Register 3 (SSPxCON3)
- MSSPx STATUS Register (SSPxSTAT)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSPx Shift Register (SSPxSR) – Not directly accessible
- MSSPx Address Register (SSPxADD)
- I²C Slave Address Mask Register (SSPxMSK)

SSPxCON1, SSPxCON2, SSPxCON3 and SSPxSTAT are the control and STATUS registers in I²C mode operation. The SSPxCON1, SSPxCON2, and SSPxCON3 registers are readable and writable. The lower 6 bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

SSPxSR is the shift register used for shifting data in or out. SSPxBUF is the buffer register to which data bytes are written to or read from.

SSPxADD contains the slave device address when the MSSPx is configured in I²C Slave mode. When the MSSPx is configured in Master mode, the lower seven bits of SSPxADD act as the Baud Rate Generator reload value.

SSPxMSK holds the slave address mask value when the module is configured for 7-Bit Address Masking mode. While it is a separate register, it shares the same SFR address as SSPxADD; it is only accessible when the SSPM<3:0> bits are specifically set to permit access. Additional details are provided in **Section 20.5.4.3 "7-Bit Address Masking Mode"**.

In receive operations, SSPxSR and SSPxBUF together, create a double-buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not double-buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

REGISTER 20-6: SSPxSTAT: MSSPx STATUS REGISTER (I^2C MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/$\overline{A}$	P(1)	S(1)	R/$\overline{W}$(2,3)	UA	BF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **SMP:** Slew Rate Control bit

In Master or Slave mode:
1 = Slew rate control is disabled for Standard Speed mode (100 kHz and 1 MHz)
0 = Slew rate control is enabled for High-Speed mode (400 kHz)

bit 6 **CKE:** SMBus Select bit

In Master or Slave mode:
1 = Enables SMBus-specific inputs
0 = Disables SMBus-specific inputs

bit 5 **D/$\overline{A}$:** Data/$\overline{\text{Address}}$ bit

In Master mode:
Reserved.

In Slave mode:
1 = Indicates that the last byte received or transmitted was data
0 = Indicates that the last byte received or transmitted was address

bit 4 **P:** Stop bit(1)

1 = Indicates that a Stop bit has been detected last
0 = Stop bit was not detected last

bit 3 **S:** Start bit(1)

1 = Indicates that a Start bit has been detected last
0 = Start bit was not detected last

bit 2 **R/$\overline{W}$:** Read/$\overline{\text{Write}}$ Information bit(2,3)

In Slave mode:
1 = Read
0 = Write

In Master mode:
1 = Transmit is in progress
0 = Transmit is not in progress

bit 1 **UA:** Update Address bit (10-Bit Slave mode only)

1 = Indicates that the user needs to update the address in the SSPxADD register
0 = Address does not need to be updated

bit 0 **BF:** Buffer Full Status bit

In Transmit mode:
1 = SSPxBUF is full
0 = SSPxBUF is empty

In Receive mode:
1 = SSPxBUF is full (does not include the $\overline{\text{ACK}}$ and Stop bits)
0 = SSPxBUF is empty (does not include the $\overline{\text{ACK}}$ and Stop bits)

Note 1: This bit is cleared on Reset and when SSPEN is cleared.

2: This bit holds the R/$\overline{W}$ bit information following the last address match. This bit is only valid from the address match to the next Start bit, Stop bit or not $\overline{\text{ACK}}$ bit.

3: ORing this bit with SEN, RSEN, PEN, RCEN or ACKEN will indicate if the MSSPx is in Active mode.

REGISTER 20-7: SSPxCON1: MSSPx CONTROL REGISTER 1 (I^2C MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN(1)	CKP	SSPM3(2)	SSPM2(2)	SSPM1(2)	SSPM0(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **WCOL:** Write Collision Detect bit

In Master Transmit mode:
1 = A write to the SSPxBUF register was attempted while the I^2C conditions were not valid for a transmission to be started (must be cleared in software)
0 = No collision

In Slave Transmit mode:
1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
0 = No collision

In Receive mode (Master or Slave modes):
This is a "don't care" bit.

bit 6 **SSPOV:** Receive Overflow Indicator bit

In Receive mode:
1 = A byte is received while the SSPxBUF register is still holding the previous byte (must be cleared in software)
0 = No overflow

In Transmit mode:
This is a "don't care" bit in Transmit mode.

bit 5 **SSPEN:** Master Synchronous Serial Port Enable bit(1)
1 = Enables the serial port and configures the SDAx and SCLx pins as the serial port pins
0 = Disables serial port and configures these pins as I/O port pins

bit 4 **CKP:** SCKx Release Control bit

In Slave mode:
1 = Releases clock
0 = Holds clock low (clock stretch), used to ensure data setup time

In Master mode:
Unused in this mode.

bit 3-0 **SSPM<3:0>:** Master Synchronous Serial Port Mode Select bits(2)
1111 = I^2C Slave mode: 10-bit address with Start and Stop bit interrupts enabled
1110 = I^2C Slave mode: 7-bit address with Start and Stop bit interrupts enabled
1011 = I^2C Firmware Controlled Master mode (slave Idle)
1001 = Load SSPxMSK register at SSPxADD SFR address(3,4)
1000 = I^2C Master mode: Clock = FOSC/(4 * (SSPxADD + 1))
0111 = I^2C Slave mode: 10-bit address(3,4)
0110 = I^2C Slave mode: 7-bit address

Note 1: When enabled, the SDAx and SCLx pins must be configured as inputs.
2: Bit combinations not specifically listed here are either reserved or implemented in SPI mode only.
3: When SSPM<3:0> = 1001, any reads or writes to the SSPxADD SFR address actually accesses the SSPxMSK register.
4: This mode is only available when 7-Bit Address Masking mode is selected (MSSPMSK Configuration bit is '1').

REGISTER 20-8: SSPxCON2: MSSPx CONTROL REGISTER 2 (I^2C MASTER MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT(1)	ACKEN(2)	RCEN(2)	PEN(2)	RSEN(2)	SEN(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **GCEN:** General Call Enable bit
Unused in Master mode.

bit 6 **ACKSTAT:** Acknowledge Status bit (Master Transmit mode only)
1 = Acknowledge was not received from slave
0 = Acknowledge was received from slave

bit 5 **ACKDT:** Acknowledge Data bit (Master Receive mode only)(1)
1 = Not Acknowledge
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit(2)
1 = Initiates Acknowledge sequence on SDAx and SCLx pins and transmits ACKDT data bit; automatically cleared by hardware
0 = Acknowledge sequence is Idle

bit 3 **RCEN:** Receive Enable bit (Master Receive mode only)(2)
1 = Enables Receive mode for I^2C
0 = Receive is Idle

bit 2 **PEN:** Stop Condition Enable bit(2)
1 = Initiates Stop condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Stop condition is Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit(2)
1 = Initiates Repeated Start condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Repeated Start condition is Idle

bit 0 **SEN:** Start Condition Enable bit(2)
1 = Initiates Start condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Start condition is Idle

Note 1: The value that will be transmitted when the user initiates an Acknowledge sequence at the end of a receive.

2: If the I^2C module is active, these bits may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

REGISTER 20-9: SSPxCON3: MSSP CONTROL REGISTER 3 (I^2C MASTER MODE)

R/HS/HC-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ACKTIM:** Acknowledge Time Status bit
Unused in Master mode.

bit 6 **PCIE:** Stop Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Stop condition
0 = Stop detection interrupts are disabled

bit 5 **SCIE:** Start Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Start or Restart conditions
0 = Start detection interrupts are disabled

bit 4 **BOEN:** Buffer Overwrite Enable bit
1 = SSPBUF is updated every time a new data byte is available, ignoring the SSPOV effect on updating the buffer
0 = SSPBUF is only updated when SSPOV is clear

bit 3 **SDAHT:** SDA Hold Time Selection bit
1 = Minimum of 300ns hold time on SDA after the falling edge of SCL
0 = Minimum of 100ns hold time on SDA after the falling edge of SCL

bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit
Unused in Master mode.

bit 1 **AHEN:** Address Hold Enable bit
Unused in Master mode.

bit 0 **DHEN:** Data Hold Enable bit
Unused in Master mode.

Note 1: This bit has no effect in Slave modes that Start and Stop condition detection is explicitly listed as enabled.

REGISTER 20-10: SSPxCON2: MSSPx CONTROL REGISTER 2 (I^2C SLAVE MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT(1)	ACKEN(1)	RCEN(1)	PEN(1)	RSEN(1)	SEN(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **GCEN:** General Call Enable bit
1 = Enables interrupt when a general call address (0000h) is received in the SSPxSR
0 = General call address is disabled

bit 6 **ACKSTAT:** Acknowledge Status bit
Unused in Slave mode.

bit 5 **ACKDT:** Acknowledge Data bit (Master Receive mode only)(1)
1 = Not Acknowledge
0 = Acknowledge

bit 4 **ACKEN:** Acknowledge Sequence Enable bit(1)
1 = Initiates Acknowledge sequence on SDAx and SCLx pins and transmits ACKDT data bit; automatically cleared by hardware
0 = Acknowledge sequence is Idle

bit 3 **RCEN:** Receive Enable bit (Master Receive mode only)(1)
1 = Enables Receive mode for I^2C
0 = Receive is Idle

bit 2 **PEN:** Stop Condition Enable bit(1)
1 = Initiates Stop condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Stop condition is Idle

bit 1 **RSEN:** Repeated Start Condition Enable bit(1)
1 = Initiates Repeated Start condition on SDAx and SCLx pins; automatically cleared by hardware
0 = Repeated Start condition is Idle

bit 0 **SEN:** Stretch Enable bit(1)
1 = Clock stretching is enabled for both slave transmit and slave receive (stretch enabled)
0 = Clock stretching is disabled

Note 1: If the I^2C module is active, this bit may not be set (no spooling) and the SSPxBUF may not be written (or writes to the SSPxBUF are disabled).

REGISTER 20-11: SSPxCON3: MSSP CONTROL REGISTER 3 (I^2C SLAVE MODE)

R/HS/HC-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ACKTIM	PCIE	SCIE	BOEN	SDAHT	SBCDE	AHEN	DHEN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ACKTIM:** Acknowledge Time Status bit
1 = Indicates the I^2C bus is in an Acknowledge sequence, set on 8th falling edge of SCL clock
0 = Not an Acknowledge sequence, cleared on 9th rising edge of SCL clock

bit 6 **PCIE:** Stop Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Stop condition
0 = Stop detection interrupts are disabled

bit 5 **SCIE:** Start Condition Interrupt Enable bit(1)
1 = Enable interrupt on detection of Start or Restart conditions
0 = Start detection interrupts are disabled

bit 4 **BOEN:** Buffer Overwrite Enable bit
1 = SSPBUF is updated every time a new data byte is available, ignoring the SSPOV effect on updating the buffer
0 = SSPBUF is only updated when SSPOV is clear

bit 3 **SDAHT:** SDA Hold Time Selection bit
1 = Minimum of 300ns hold time on SDA after the falling edge of SCL
0 = Minimum of 100ns hold time on SDA after the falling edge of SCL

bit 2 **SBCDE:** Slave Mode Bus Collision Detect Enable bit
If, on the rising edge of SCL, SDA is sampled low when the module is outputting a high state, the BCLIF bit is set, and bus goes Idle.
1 = Enable slave bus collision interrupts
0 = Slave bus collision interrupts are disabled

bit 1 **AHEN:** Address Hold Enable bit
1 = Following the 8th falling edge of SCL for a matching received address byte; CKP bit of SSPxCON1 will be cleared and the SCL will be held low.
0 = Address holding is disabled

bit 0 **DHEN:** Data Hold Enable bit
1 = Following the 8th falling edge of SCL for a received data byte; slave hardware clears the CKP bit of SSPCON register and SCL is held low.
0 = Data holding is disabled

Note 1: This bit has no effect in Slave modes that Start and Stop condition detection is explicitly listed as enabled.

REGISTER 20-12: SSPxMSK: MSSPx I²C SLAVE ADDRESS MASK REGISTER (7-BIT MASKING MODE)(1)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0(2)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **MSK<7:0>:** Slave Address Mask Select bits

1 = Masking of corresponding bit of SSPxADD is enabled
0 = Masking of corresponding bit of SSPxADD is disabled

Note 1: This register shares the same SFR address as SSPxADD and is only addressable in select MSSPx operating modes. See **Section 20.5.4.3 "7-Bit Address Masking Mode"** for more details.

2: MSK0 is not used as a mask bit in 7-bit addressing.

20.5.2 OPERATION

The MSSPx module functions are enabled by setting the MSSPx Enable bit, SSPEN (SSPxCON1<5>).

The SSPxCON1 register allows control of the I^2C operation. Four mode selection bits (SSPxCON1<3:0>) allow one of the following I^2C modes to be selected:

- I^2C Master mode, clock
- I^2C Slave mode (7-bit address)
- I^2C Slave mode (10-bit address)
- I^2C Slave mode (7-bit address) with Start and Stop bit interrupts enabled
- I^2C Slave mode (10-bit address) with Start and Stop bit interrupts enabled
- I^2C Firmware Controlled Master mode, slave is Idle

Selection of any I^2C mode, with the SSPEN bit set, forces the SCLx and SDAx pins to be open-drain, provided these pins are programmed as inputs by setting the appropriate TRISC or TRISD bits. To ensure proper operation of the module, pull-up resistors must be provided externally to the SCLx and SDAx pins.

20.5.3 SLAVE MODE

In Slave mode, the SCLx and SDAx pins must be configured as inputs (TRISC<4:3> set). The MSSPx module will override the input state with the output data when required (slave-transmitter).

The I^2C Slave mode hardware will always generate an interrupt on an address match. Address masking will allow the hardware to generate an interrupt for more than one address (up to 31 in 7-bit addressing and up to 63 in 10-bit addressing). Through the mode select bits, the user can also choose to interrupt on Start and Stop bits.

When an address is matched, or the data transfer after an address match is received, the hardware automatically will generate the Acknowledge ($\overline{ACK}$) pulse and load the SSPxBUF register with the received value currently in the SSPxSR register.

Any combination of the following conditions will cause the MSSPx module not to give this $\overline{ACK}$ pulse:

- The Buffer Full bit, BF (SSPxSTAT<0>), was set before the transfer was received.
- The overflow bit, SSPOV (SSPxCON1<6>), was set before the transfer was received.

In this case, the SSPxSR register value is not loaded into the SSPxBUF, but bit, SSPxIF, is set. The BF bit is cleared by reading the SSPxBUF register, while bit, SSPOV, is cleared through software.

The SCLx clock input must have a minimum high and low for proper operation. The high and low times of the I^2C specification, as well as the requirement of the MSSPx module, are shown in timing Parameter 100 and Parameter 101.

20.5.4 ADDRESSING

Once the MSSPx module has been enabled, it waits for a Start condition to occur. Following the Start condition, the 8 bits are shifted into the SSPxSR register. All incoming bits are sampled with the rising edge of the clock (SCLx) line. The value of register, SSPxSR<7:1>, is compared to the value of the SSPxADD register. The address is compared on the falling edge of the eighth clock (SCLx) pulse. If the addresses match, and the BF and SSPOV bits are clear, the following events occur:

1. The SSPxSR register value is loaded into the SSPxBUF register.
2. The Buffer Full bit, BF, is set.
3. An $\overline{ACK}$ pulse is generated.
4. The MSSPx Interrupt Flag bit, SSPxIF, is set (and interrupt is generated if enabled) on the falling edge of the ninth SCLx pulse.

In 10-Bit Addressing mode, two address bytes need to be received by the slave. The five Most Significant bits (MSbs) of the first address byte specify if this is a 10-bit address. The R/$\overline{W}$ (SSPxSTAT<2>) bit must specify a write so the slave device will receive the second address byte. For a 10-bit address, the first byte would equal '11110 A9 A8 0', where 'A9' and 'A8' are the two MSbs of the address. The sequence of events for 10-bit addressing is as follows, with Steps 7 through 9 for the slave-transmitter:

1. Receive first (high) byte of address (bits, SSPxIF, BF and UA, are set on address match).
2. Update the SSPxADD register with second (low) byte of address (clears bit, UA, and releases the SCLx line).
3. Read the SSPxBUF register (clears bit, BF) and clear flag bit, SSPxIF.
4. Receive second (low) byte of address (bits, SSPxIF, BF and UA, are set).
5. Update the SSPxADD register with the first (high) byte of address. If match releases SCLx line, this will clear bit, UA.
6. Read the SSPxBUF register (clears bit, BF) and clear flag bit SSPxIF.
7. Receive Repeated Start condition.
8. Receive first (high) byte of address (bits, SSPxIF and BF, are set).
9. Read the SSPxBUF register (clears bit, BF) and clear flag bit, SSPxIF.

20.5.4.1 Address Masking Modes

Masking an address bit causes that bit to become a "don't care". When one address bit is masked, two addresses will be Acknowledged and cause an interrupt. It is possible to mask more than one address bit at a time, which greatly expands the number of addresses Acknowledged.

The I^2C slave behaves the same way, whether address masking is used or not. However, when address masking is used, the I^2C slave can Acknowledge multiple addresses and cause interrupts. When this occurs, it is necessary to determine which address caused the interrupt by checking the SSPxBUF.

The PIC18FXXJ94 of devices is capable of using two different Address Masking modes in I^2C slave operation: 5-Bit Address Masking and 7-Bit Address Masking. The Masking mode is selected at device configuration using the MSSPMSK<2:1> Configuration bits. The default device configuration is 7-Bit Address Masking.

Both Masking modes, in turn, support address masking of 7-bit and 10-bit addresses. The combination of Masking modes and addresses provides different ranges of Acknowledgable addresses for each combination.

While both Masking modes function in roughly the same manner, the way they use address masks are different.

20.5.4.2 5-Bit Address Masking Mode

As the name implies, 5-Bit Address Masking mode uses an address mask of up to 5 bits to create a range of addresses to be Acknowledged, using bits, 5 through 1, of the incoming address. This allows the module to Acknowledge up to 31 addresses when using 7-bit addressing, or 63 addresses with 10-bit addressing (see Example 20-3). This Masking mode is selected when the MSSPMSK<2:1> Configuration bits are programmed ('00').

The address mask in this mode is stored in the SSPxCON2 register, which stops functioning as a control register in I^2C Slave mode (Register 20-10). In 7-Bit Address Masking mode, Address Mask bits, MSK<5:1> (SSPxMSK<5:1>), mask the corresponding address bits in the SSPxADD register. For any MSK bits that are set (MSK<n> = 1), the corresponding address bit is ignored (SSPxADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

In 10-Bit Address Masking mode, the MSK<5:2> bits mask the corresponding address bits in the SSPxADD register. In addition, MSK1 simultaneously masks the two LSbs of the address (SSPxADD<1:0>). For any MSKx bits that are active (MSK<n> = 1), the corresponding address bit is ignored (SPxADD<n> = x). Also note that although in 10-Bit Address Masking mode, the upper address bits re-use part of the SSPxADD register bits. The address mask bits do not interact with those bits; they only affect the lower address bits.

Note 1: MSK1 masks the two Least Significant bits of the address.

2: The two Most Significant bits of the address are not affected by address masking.

EXAMPLE 20-3: ADDRESS MASKING EXAMPLES IN 5-BIT MASKING MODE

7-Bit Addressing:

SSPxADD<7:1>= A0h (1010000) (SSPxADD<0> is assumed to be '0')

MSK<5:1>= 00111

Addresses Acknowledged: A0h, A2h, A4h, A6h, A8h, AAh, ACh, AEh

10-Bit Addressing:

SSPxADD<7:0> = A0h (10100000) (The two MSb of the address are ignored in this example, since they are not affected by masking.)

MSK<5:1> = 00111

Addresses Acknowledged: A0h, A1h, A2h, A3h, A4h, A5h, A6h, A7h, A8h, A9h, AAh, ABh, ACh, ADh, AEh, AFh

20.5.4.3 7-Bit Address Masking Mode

Unlike 5-bit masking, 7-Bit Address Masking mode uses a mask of up to 8 bits (in 10-bit addressing) to define a range of addresses that can be Acknowledged, using the lowest bits of the incoming address. This allows the module to Acknowledge up to 127 different addresses with 7-bit addressing, or 255 with 10-bit addressing (see Example 20-4). This mode is the default configuration of the module, which is selected when MSSPMSK<2:1> are unprogrammed ('1').

The address mask for 7-Bit Address Masking mode is stored in the SSPxMSK register, instead of the SSPxCON2 register. SSPxMSK is a separate hardware register within the module, but it is not directly addressable. Instead, it shares an address in the SFR space with the SSPxADD register. To access the SSPxMSK register, it is necessary to select MSSP mode, '1001' (SSPxCON1<3:0> = 1001) and then read or write to the location of SSPxADD.

To use 7-Bit Address Masking mode, it is necessary to initialize SSPxMSK with a value before selecting the I^2C Slave Addressing mode. Thus, the required sequence of events is:

1. Select SSPxMSK Access mode (SSPxCON2<3:0> = 1001).
2. Write the mask value to the appropriate SSPxADD register address (FC8h for MSSP1, F6Eh for MSSP2).
3. Set the appropriate I^2C Slave mode (SSPxCON2<3:0> = 0111 for 10-bit addressing, 0110 for 7-bit addressing).

Setting or clearing mask bits in SSPxMSK behaves in the opposite manner of the MSKx bits in 5-Bit Address Masking mode. That is, clearing a bit in SSPxMSK causes the corresponding address bit to be masked; setting the bit requires a match in that position. SSPxMSK resets to all '1's upon any Reset condition, and therefore, has no effect on the standard MSSP operation until written with a mask value.

With 7-bit addressing, SSPxMSK<7:1> bits mask the corresponding address bits in the SSPxADD register. For any SSPxMSK bits that are active (SSPxMSK<n> = 0), the corresponding SSPxADD address bit is ignored (SSPxADD<n> = x). For the module to issue an address Acknowledge, it is sufficient to match only on addresses that do not have an active address mask.

With 10-bit addressing, SSPxMSK<7:0> bits mask the corresponding address bits in the SSPxADD register. For any SSPxMSK bits that are active (= 0), the corresponding SSPxADD address bit is ignored (SSPxADD<n> = x).

Note:	The two Most Significant bits of the address are not affected by address masking.

EXAMPLE 20-4: ADDRESS MASKING EXAMPLES IN 7-BIT MASKING MODE

7-Bit Addressing:

SSPxADD<7:1> = 1010 000

SSPxMSK<7:1> = 1111 001

Addresses Acknowledged = ACh, A8h, A4h, A0h

10-Bit Addressing:

SSPxADD<7:0> = 1010 0000 (The two MSb are ignored in this example since they are not affected)

SSPxMSK<5:1> = 1111 0011

Addresses Acknowledged = ACh, A8h, A4h, A0h

20.5.5 RECEPTION

When the R/$\overline{W}$ bit of the address byte is clear and an address match occurs, the R/$\overline{W}$ bit of the SSPxSTAT register is cleared. The received address is loaded into the SSPxBUF register and the SDAx line is held low ($\overline{ACK}$).

When the address byte overflow condition exists, then the no Acknowledge ($\overline{ACK}$) pulse is given. An overflow condition is defined if either bit, BF (SSPxSTAT<0>), is set or bit, SSPOV (SSPxCON1<6>), is set.

An MSSPx interrupt is generated for each data transfer byte. The interrupt flag bit, SSPxIF, must be cleared in software. The SSPxSTAT register is used to determine the status of the byte.

If SEN is enabled (SSPxCON2<0> = 1), SCLx will be held low (clock stretch) following each data transfer. The clock must be released by setting bit, CKP (SSPxCON1<4>). See **Section 20.5.7 "Clock Stretching"** for more details.

20.5.6 TRANSMISSION

When the R/$\overline{W}$ bit of the incoming address byte is set and an address match occurs, the R/$\overline{W}$ bit of the SSPxSTAT register is set. The received address is loaded into the SSPxBUF register. The $\overline{ACK}$ pulse will be sent on the ninth bit and pin, SCLx, is held low regardless of SEN (see **Section 20.5.7 "Clock Stretching"** for more details). By stretching the clock, the master will be unable to assert another clock pulse until the slave is done preparing the transmit data. The transmit data must be loaded into the SSPxBUF register which also loads the SSPxSR register. Then, pin, SCLx, should be enabled by setting bit, CKP (SSPxCON1<4>). The eight data bits are shifted out on the falling edge of the SCLx input. This ensures that the SDAx signal is valid during the SCLx high time (Figure 20-10).

The $\overline{ACK}$ pulse from the master-receiver is latched on the rising edge of the ninth SCLx input pulse. If the SDAx line is high (not $\overline{ACK}$), then the data transfer is complete. In this case, when the $\overline{ACK}$ is latched by the slave, the slave logic is reset and the slave monitors for another occurrence of the Start bit. If the SDAx line was low ($\overline{ACK}$), the next transmit data must be loaded into the SSPxBUF register. Again, pin SCLx must be enabled by setting bit, CKP.

An MSSPx interrupt is generated for each data transfer byte. The SSPxIF bit must be cleared in software and the SSPxSTAT register is used to determine the status of the byte. The SSPxIF bit is set on the falling edge of the ninth clock pulse.

FIGURE 20-8: I^2C SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 7-BIT ADDRESS)

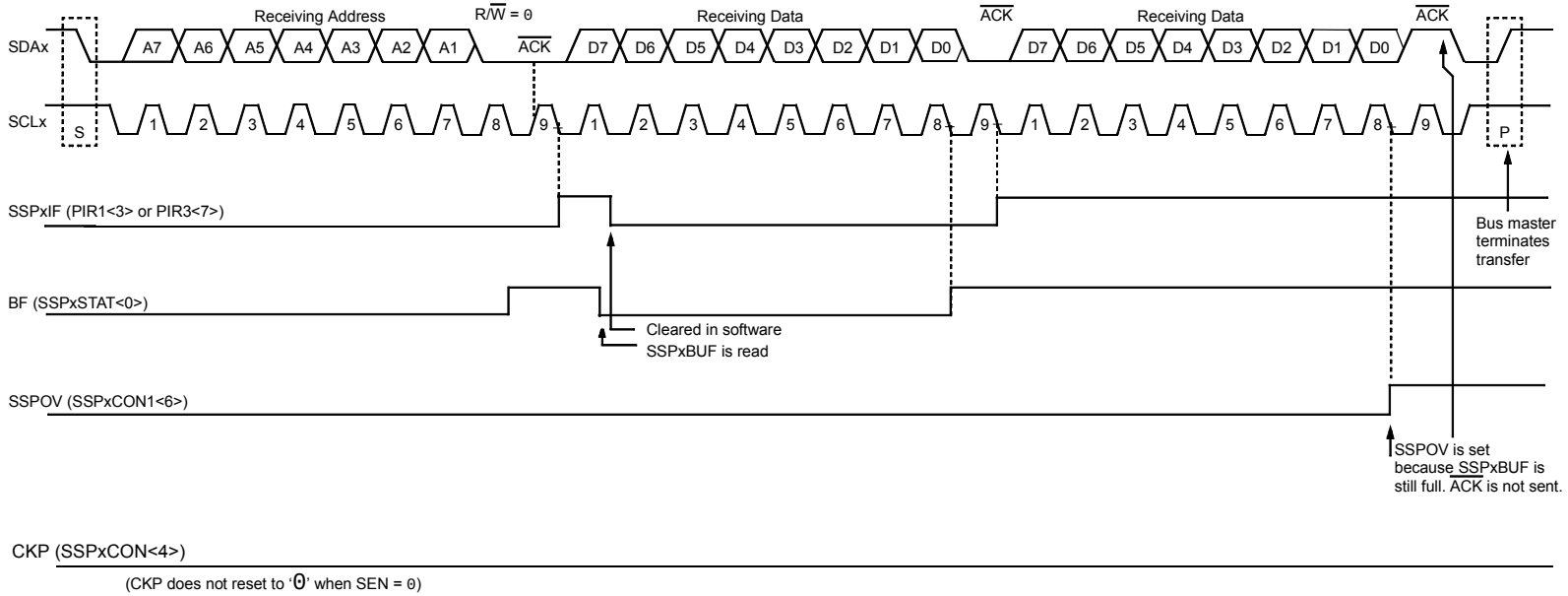
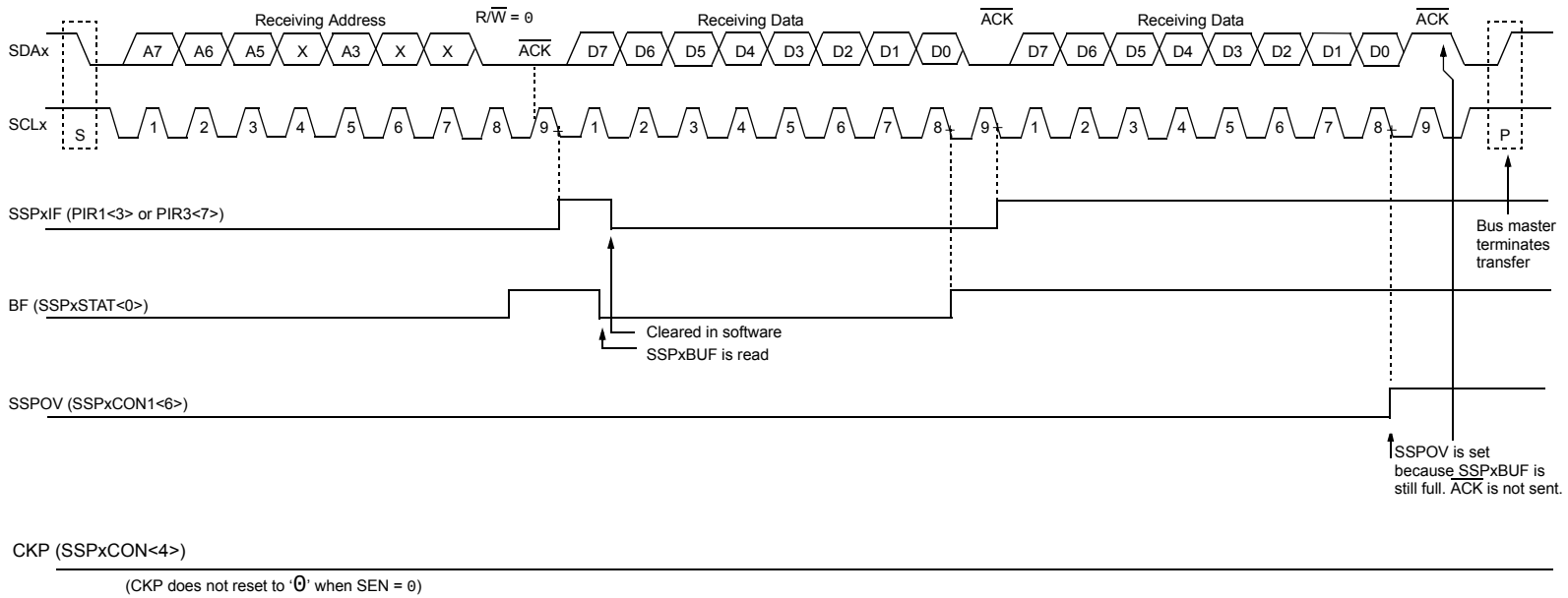


FIGURE 20-9: I²C SLAVE MODE TIMING WITH SEN = 0 AND MSK<5:1> = 01011 (RECEPTION, 7-BIT ADDRESS)



Note 1: x = Don't care (i.e., address bit can either be a '1' or a '0').

2: In this example, an address equal to A7.A6.A5.X.A3.X.X will be Acknowledged and cause an interrupt.

FIGURE 20-10: I²C SLAVE MODE TIMING (TRANSMISSION, 7-BIT ADDRESS)

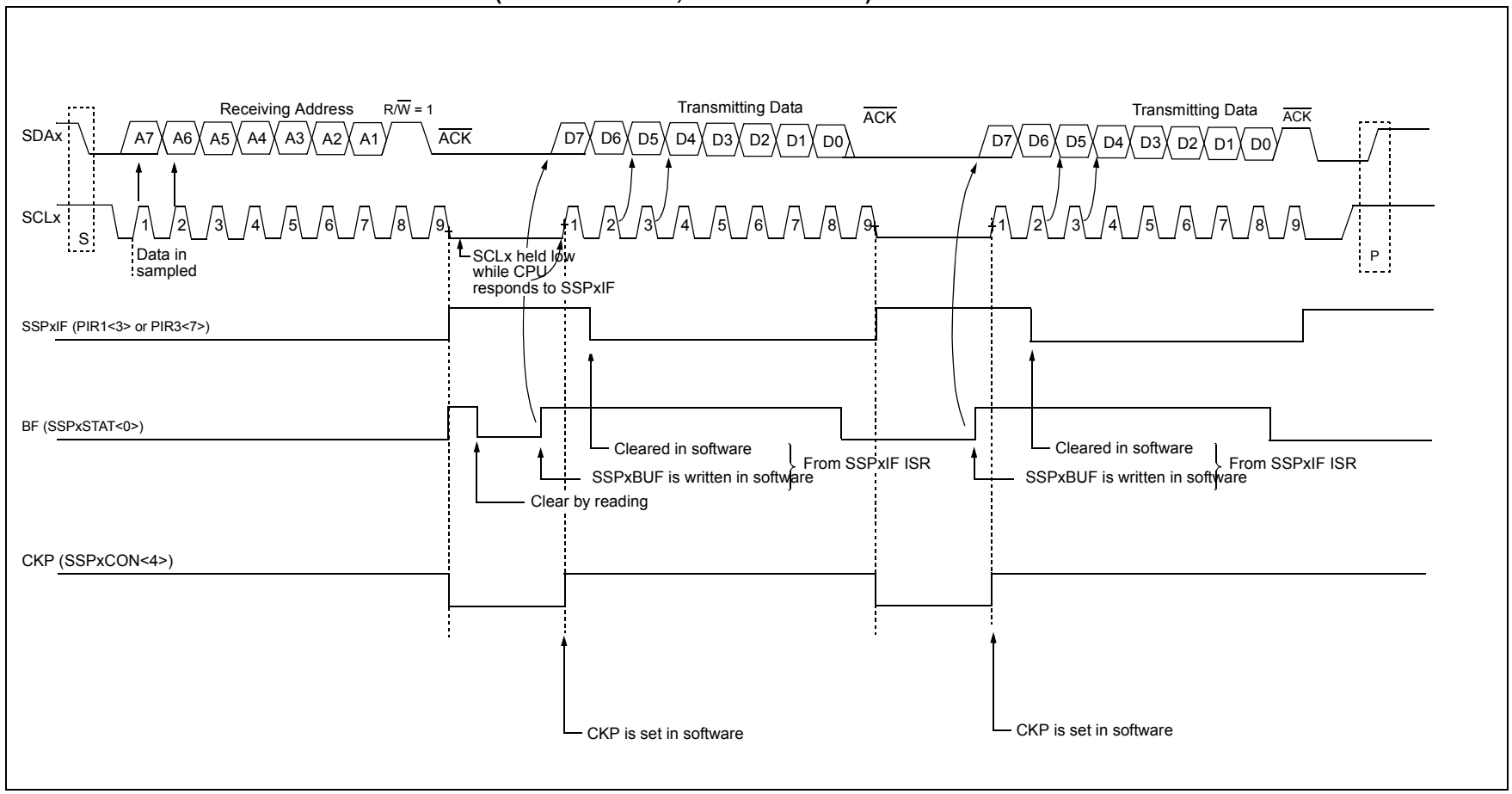
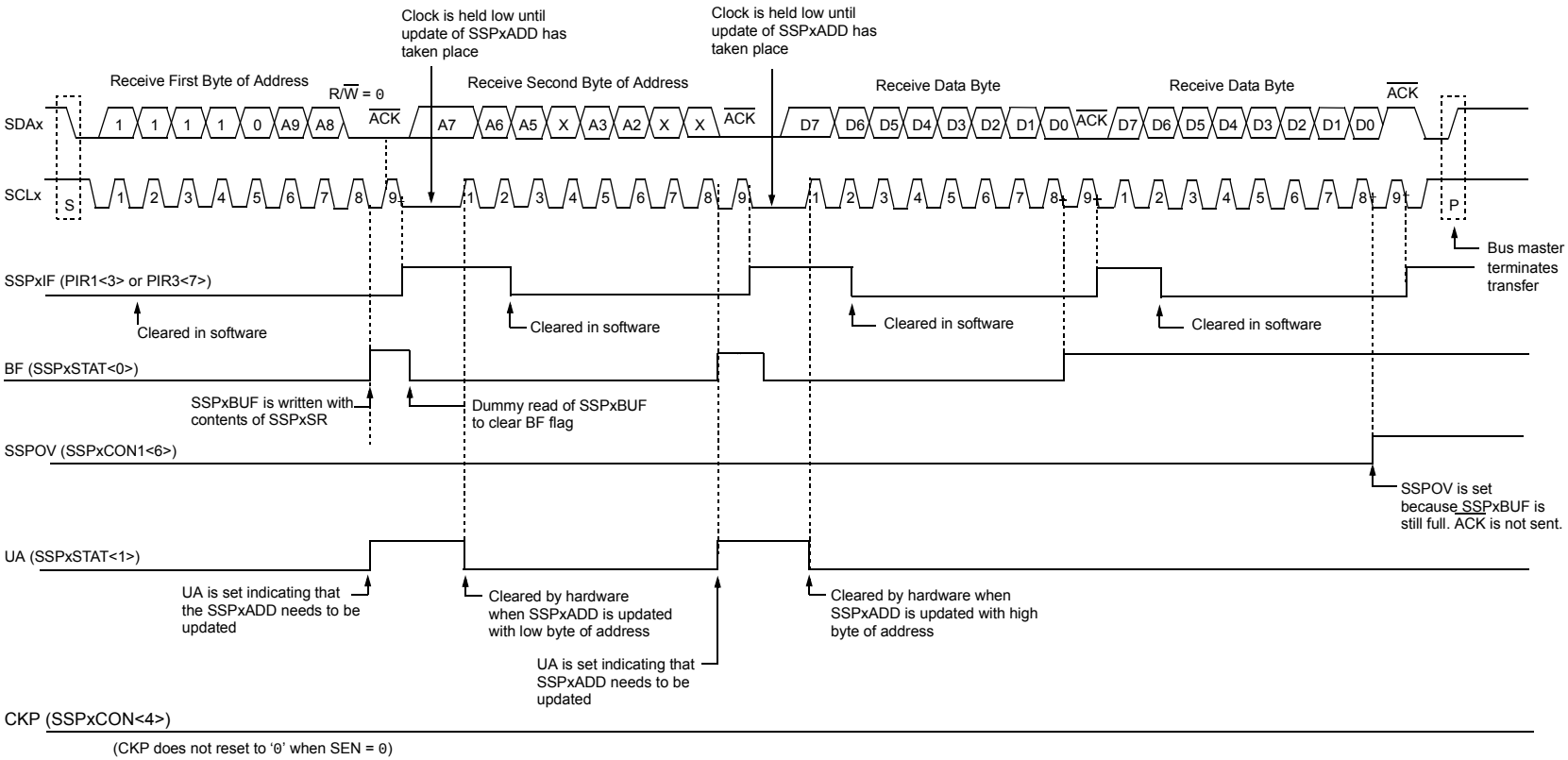


FIGURE 20-11: I²C SLAVE MODE TIMING WITH SEN = 0 AND MSK<5:1> = 01001 (RECEPTION, 10-BIT ADDRESS)



Note 1: x = Don't care (i.e., address bit can either be a '1' or a '0').

2: In this example, an address equal to A9.A8.A7.A6.A5.X.A3.A2.X.X will be Acknowledged and cause an interrupt.

3: Note that the Most Significant bits of the address are not affected by the bit masking.

FIGURE 20-12: I²C SLAVE MODE TIMING WITH SEN = 0 (RECEPTION, 10-BIT ADDRESS)

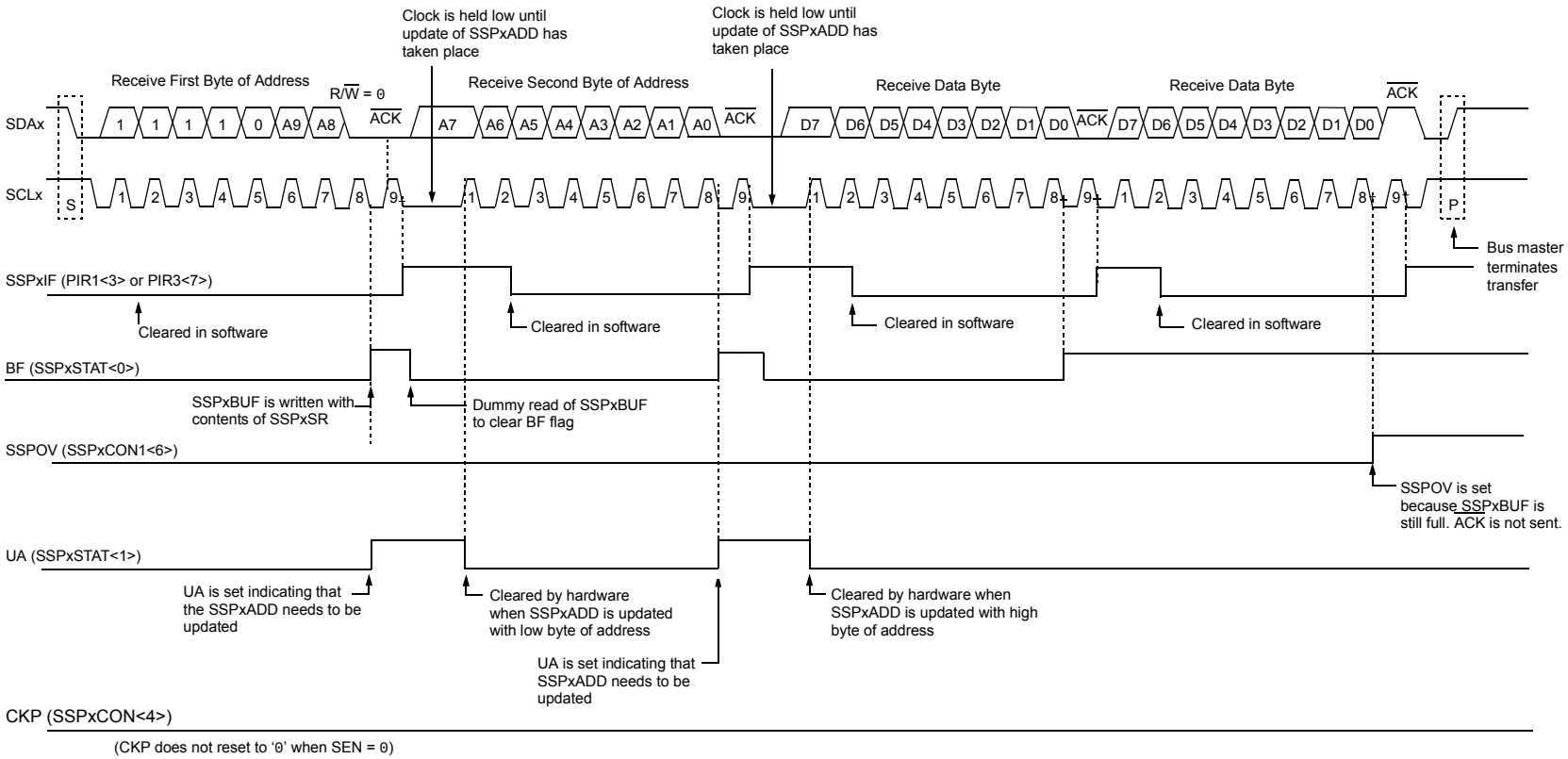
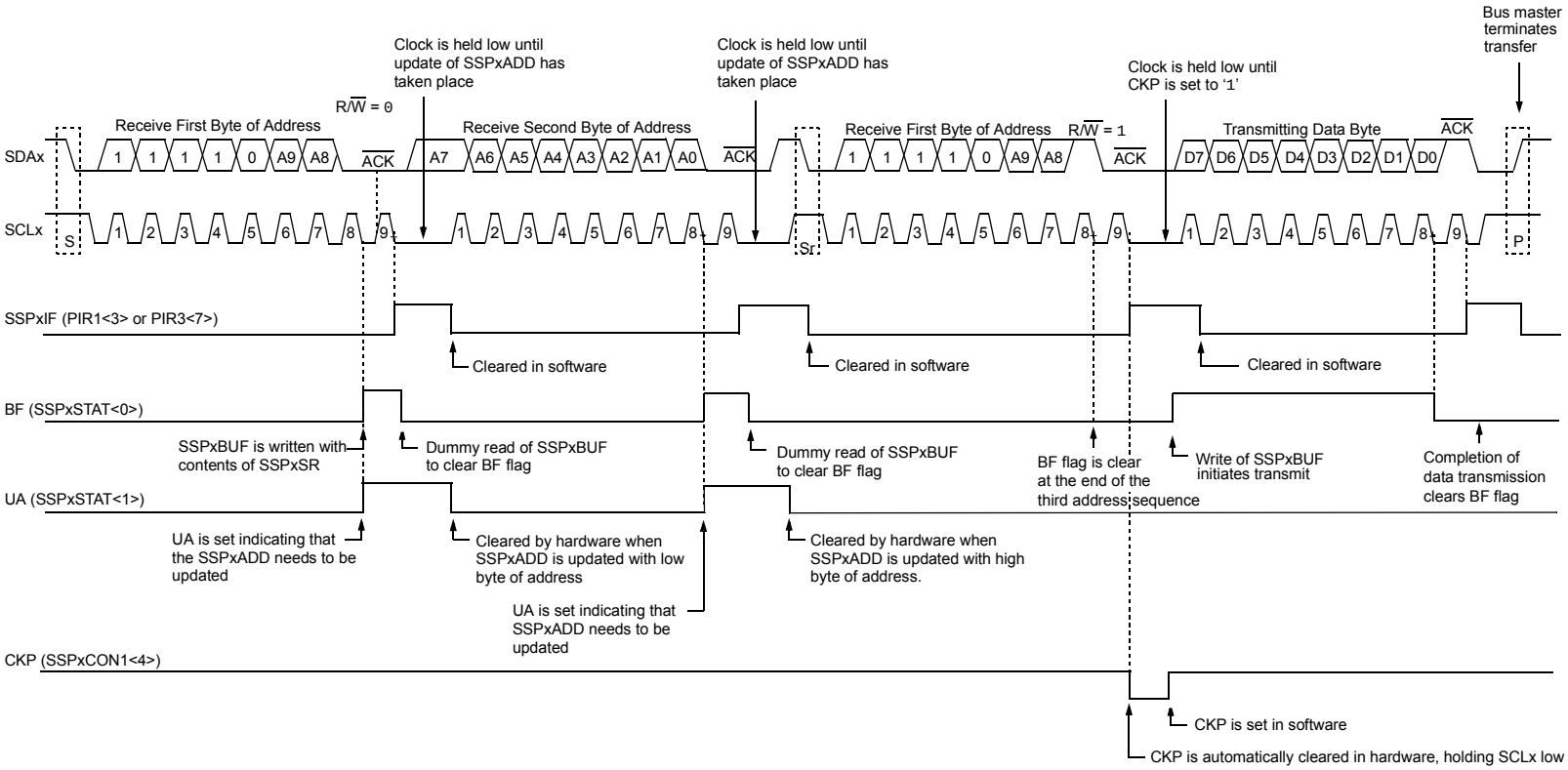


FIGURE 20-13: I^2C SLAVE MODE TIMING (TRANSMISSION, 10-BIT ADDRESS)



20.5.7 CLOCK STRETCHING

Both 7-Bit and 10-Bit Slave modes implement automatic clock stretching during a transmit sequence.

The SEN bit (SSPxCON2<0>) allows clock stretching to be enabled during receives. Setting SEN will cause the SCLx pin to be held low at the end of each data receive sequence.

20.5.7.1 Clock Stretching for 7-Bit Slave Receive Mode (SEN = 1)

In 7-Bit Slave Receive mode, on the falling edge of the ninth clock at the end of the $\overline{ACK}$ sequence, if the BF bit is set, the CKP bit in the SSPxCON1 register is automatically cleared, forcing the SCLx output to be held low. The CKP bit, being cleared to '0', will assert the SCLx line low. The CKP bit must be set in the user's ISR before reception is allowed to continue. By holding the SCLx line low, the user has time to service the ISR and read the contents of the SSPxBUF before the master device can initiate another receive sequence. This will prevent buffer overruns from occurring (see Figure 20-15).

Note 1: If the user reads the contents of the SSPxBUF before the falling edge of the ninth clock, thus clearing the BF bit, the CKP bit will not be cleared and clock stretching will not occur.

2: The CKP bit can be set in software regardless of the state of the BF bit. The user should be careful to clear the BF bit in the ISR before the next receive sequence in order to prevent an overflow condition.

20.5.7.2 Clock Stretching for 10-Bit Slave Receive Mode (SEN = 1)

In 10-Bit Slave Receive mode, during the address sequence, clock stretching automatically takes place but CKP is not cleared. During this time, if the UA bit is set after the ninth clock, clock stretching is initiated. The UA bit is set after receiving the upper byte of the 10-bit address, and following the receive of the second byte of the 10-bit address, with the R/$\overline{W}$ bit cleared to '0'. The release of the clock line occurs upon updating SSPxADD. Clock stretching will occur on each data receive sequence as described in 7-bit mode.

Note: If the user polls the UA bit and clears it by updating the SSPxADD register before the falling edge of the ninth clock occurs, and if the user hasn't cleared the BF bit by reading the SSPxBUF register before that time, then the CKP bit will still NOT be asserted low. Clock stretching, on the basis of the state of the BF bit, only occurs during a data sequence, not an address sequence.

20.5.7.3 Clock Stretching for 7-Bit Slave Transmit Mode

The 7-Bit Slave Transmit mode implements clock stretching by clearing the CKP bit after the falling edge of the ninth clock if the BF bit is clear. This occurs regardless of the state of the SEN bit.

The user's ISR must set the CKP bit before transmission is allowed to continue. By holding the SCLx line low, the user has time to service the ISR and load the contents of the SSPxBUF before the master device can initiate another transmit sequence (see Figure 20-10).

Note 1: If the user loads the contents of SSPxBUF, setting the BF bit before the falling edge of the ninth clock, the CKP bit will not be cleared and clock stretching will not occur.

2: The CKP bit can be set in software, regardless of the state of the BF bit.

20.5.7.4 Clock Stretching for 10-Bit Slave Transmit Mode

In 10-Bit Slave Transmit mode, clock stretching is controlled during the first two address sequences by the state of the UA bit, just as it is in 10-Bit Slave Receive mode. The first two addresses are followed by a third address sequence, which contains the high-order bits of the 10-bit address and the R/$\overline{W}$ bit set to '1'. After the third address sequence is performed, the UA bit is not set, the module is now configured in Transmit mode and clock stretching is controlled by the BF flag as in 7-Bit Slave Transmit mode (see Figure 20-13).

20.5.7.5 Clock Synchronization and the CKP bit

When the CKP bit is cleared, the SCLx output is forced to '0'. However, clearing the CKP bit will not assert the SCLx output low until the SCLx output is already sampled low. Therefore, the CKP bit will not assert the SCLx line until an external I^2C master device has already asserted the SCLx line. The SCLx output will remain low until the CKP bit is set and all other devices on the I^2C bus have deasserted SCLx. This ensures that a write to the CKP bit will not violate the minimum high time requirement for SCLx (see Figure 20-14).

FIGURE 20-14: CLOCK SYNCHRONIZATION TIMING

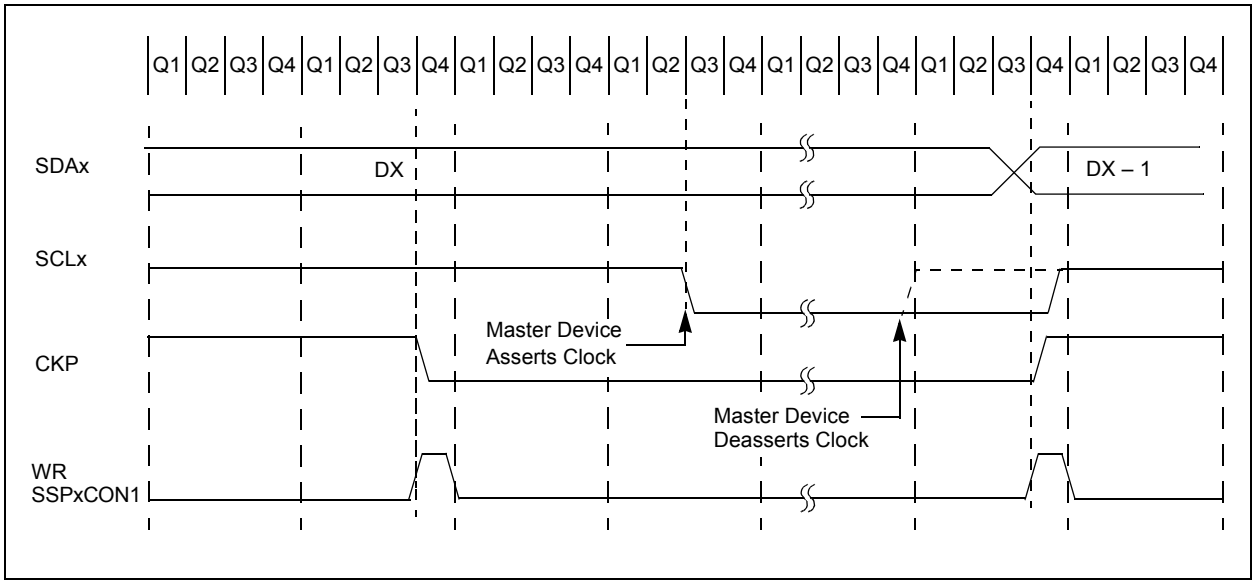


FIGURE 20-15: I^2C SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 7-BIT ADDRESS)

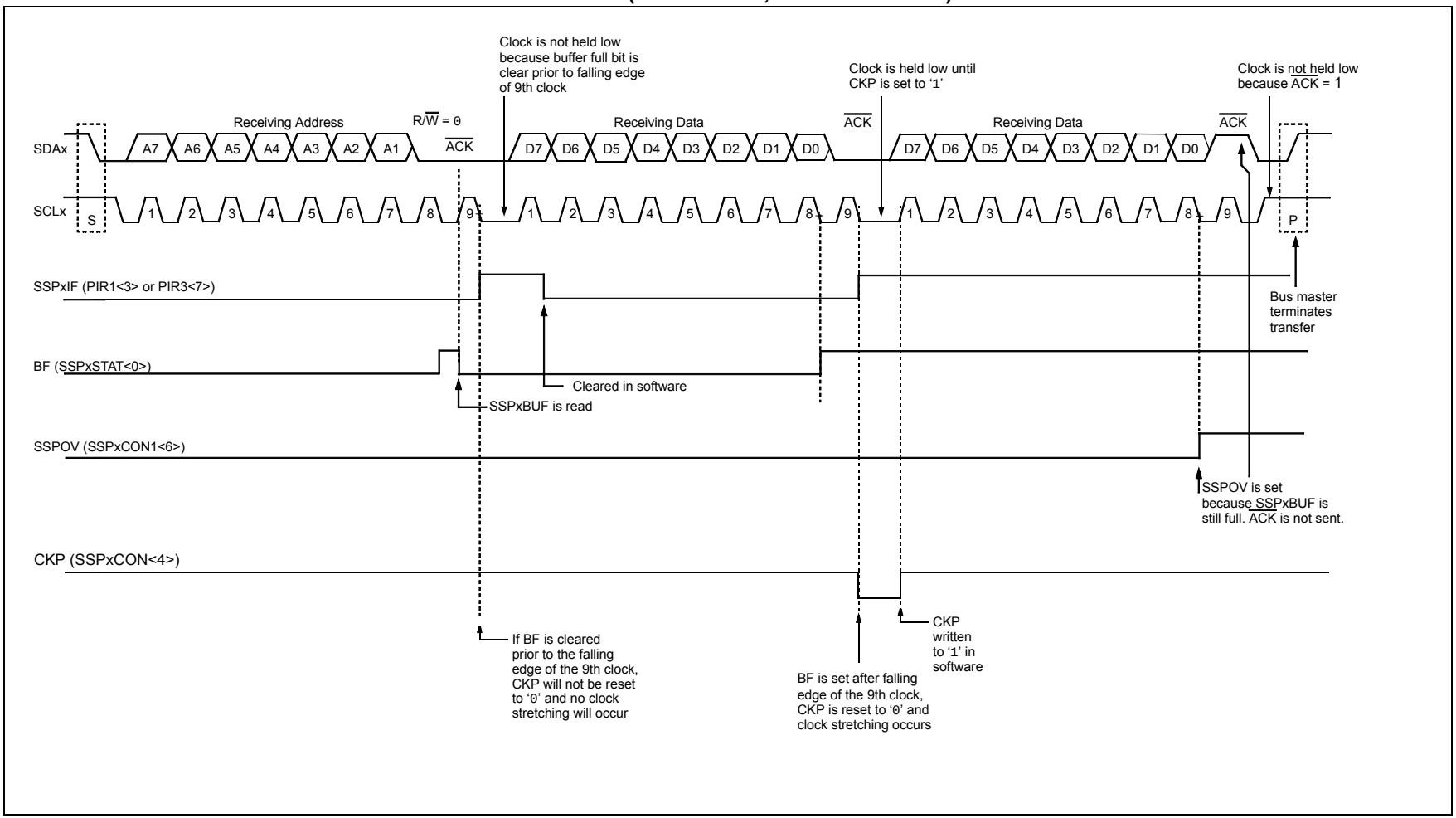
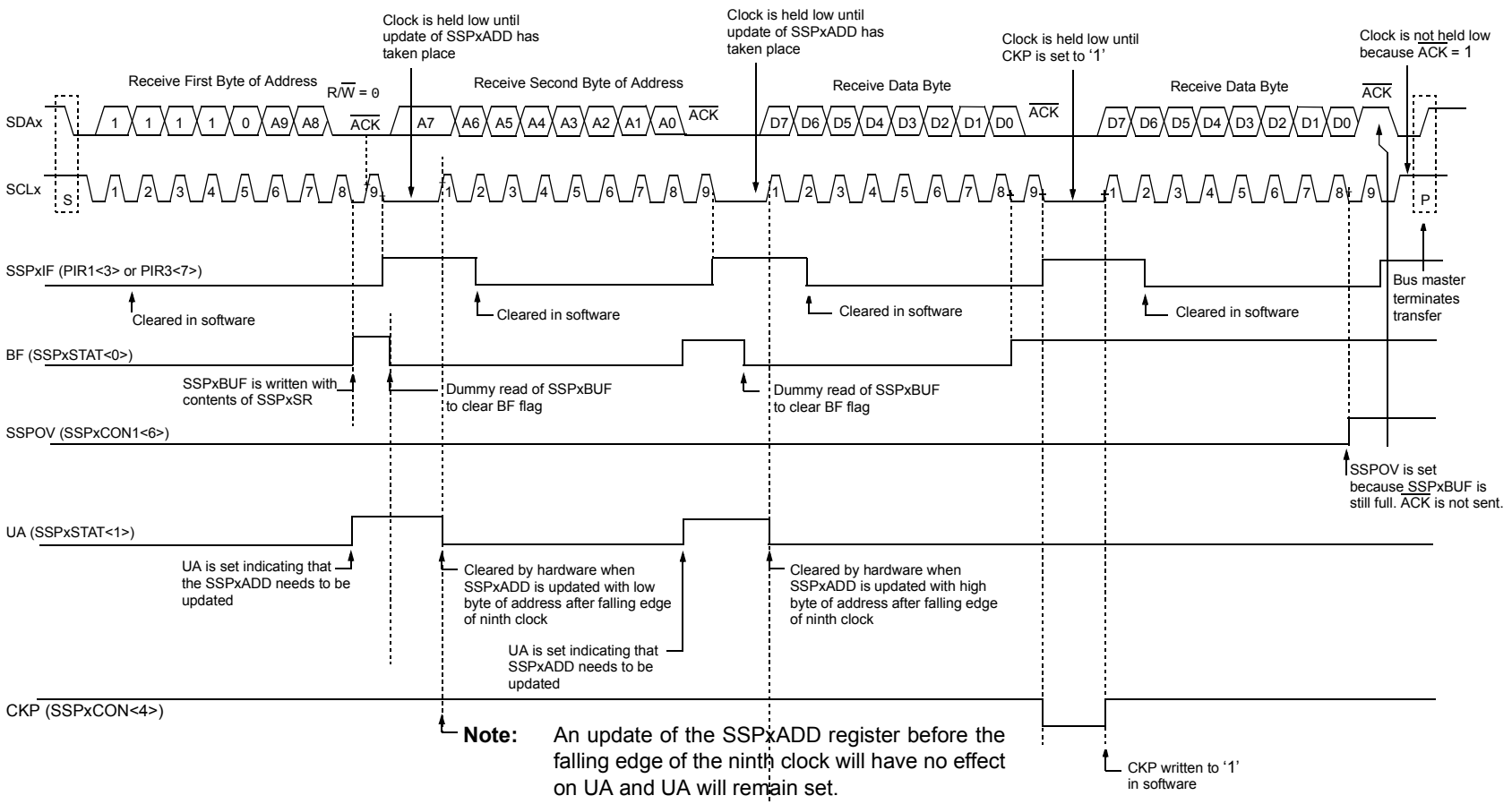


FIGURE 20-16: I²C SLAVE MODE TIMING WITH SEN = 1 (RECEPTION, 10-BIT ADDRESS)



20.5.8 GENERAL CALL ADDRESS SUPPORT

The addressing procedure for the I^2C bus is such that the first byte after the Start condition usually determines which device will be the slave addressed by the master. The exception is the general call address which can address all devices. When this address is used, all devices should, in theory, respond with an Acknowledge.

The general call address is one of eight addresses reserved for specific purposes by the I^2C protocol. It consists of all '0's with R/$\overline{W}$ = 0.

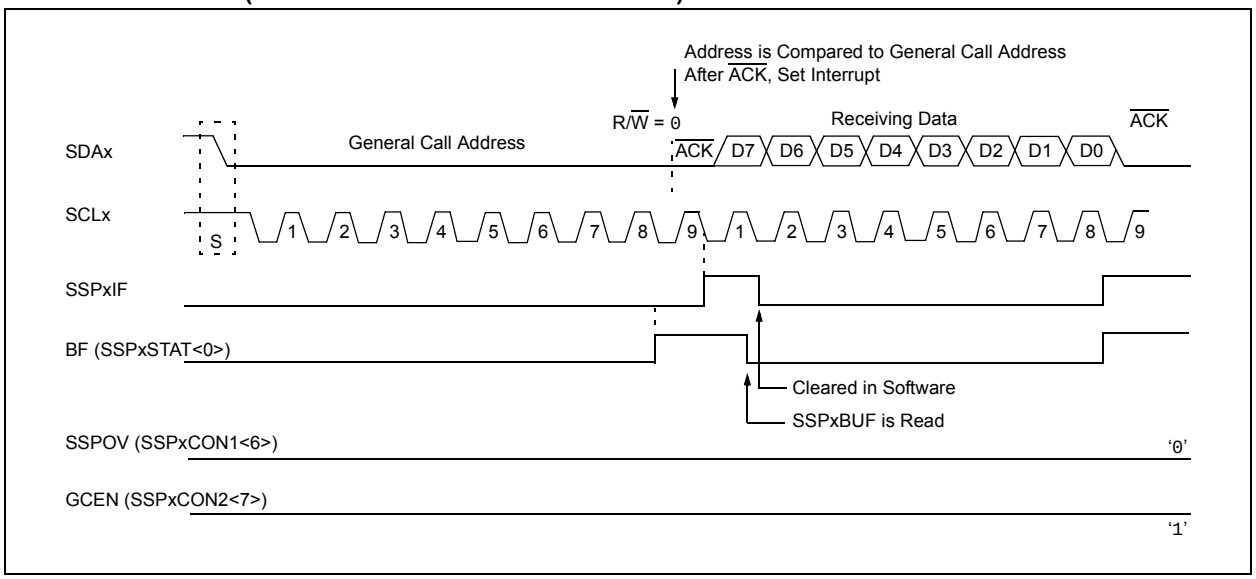
The general call address is recognized when the General Call Enable bit, GCEN, is enabled (SSPxCON2<7> set). Following a Start bit detect, eight bits are shifted into the SSPxSR and the address is compared against the SSPxADD. It is also compared to the general call address and fixed in hardware.

If the general call address matches, the SSPxSR is transferred to the SSPxBUF, the BF flag bit is set (eighth bit), and on the falling edge of the ninth bit ($\overline{ACK}$ bit), the SSPxIF interrupt flag bit is set.

When the interrupt is serviced, the source for the interrupt can be checked by reading the contents of the SSPxBUF. The value can be used to determine if the address was device-specific or a general call address.

In 10-Bit Addressing mode, the SSPxADD is required to be updated for the second half of the address to match and the UA bit is set (SSPxSTAT<1>). If the general call address is sampled when the GCEN bit is set, while the slave is configured in 10-Bit Addressing mode, then the second half of the address is not necessary, the UA bit will not be set and the slave will begin receiving data after the Acknowledge (Figure 20-17).

FIGURE 20-17: SLAVE MODE GENERAL CALL ADDRESS SEQUENCE (7 OR 10-BIT ADDRESSING MODE)



20.5.9 MASTER MODE

Master mode is enabled by setting and clearing the appropriate SSPMx bits in SSPxCON1, and by setting the SSPEN bit. In Master mode, the SCLx and SDAx lines are manipulated by the MSSPx hardware if the TRIS bits are set.

The Master mode of operation is supported by interrupt generation on the detection of the Start and Stop conditions. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSPx module is disabled. Control of the I^2C bus may be taken when the P bit is set, or the bus is Idle, with both the S and P bits clear.

In Firmware Controlled Master mode, user code conducts all I^2C bus operations based on Start and Stop bit conditions.

Once Master mode is enabled, the user has six options.

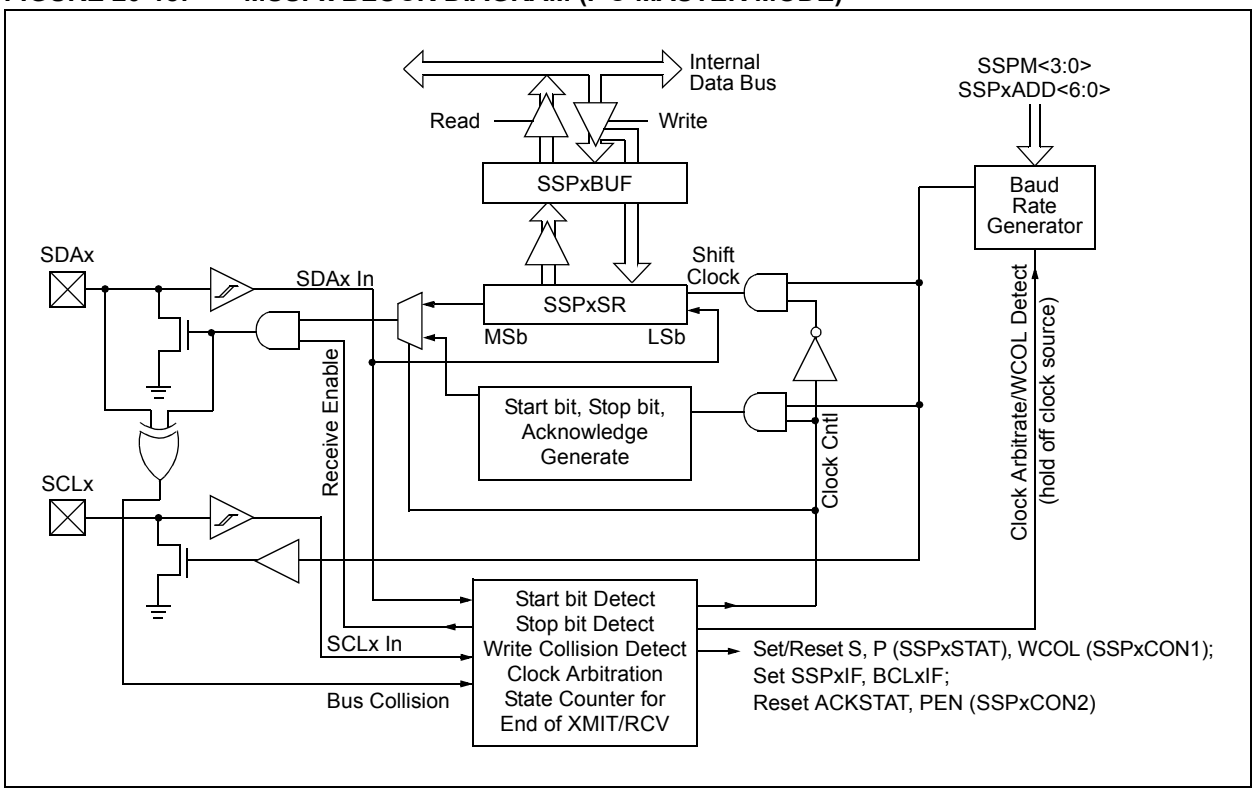
1. Assert a Start condition on SDAx and SCLx.
2. Assert a Repeated Start condition on SDAx and SCLx.
3. Write to the SSPxBUF register initiating transmission of data/address.
4. Configure the I^2C port to receive data.
5. Generate an Acknowledge condition at the end of a received byte of data.
6. Generate a Stop condition on SDAx and SCLx.

Note: The MSSPx module, when configured in I^2C Master mode, does not allow queueing of events. For instance, the user is not allowed to initiate a Start condition and immediately write the SSPxBUF register to initiate transmission before the Start condition is complete. In this case, the SSPxBUF will not be written to and the WCOL bit will be set, indicating that a write to the SSPxBUF did not occur.

The following events will cause the MSSPx Interrupt Flag bit, SSPxIF, to be set (and MSSPx interrupt if enabled):

- Start condition
- Stop condition
- Data transfer byte transmitted/received
- Acknowledge transmitted
- Repeated Start

FIGURE 20-18: MSSPx BLOCK DIAGRAM (I^2C MASTER MODE)



20.5.9.1 I²C Master Mode Operation

The master device generates all of the serial clock pulses and the Start and Stop conditions. A transfer is ended with a Stop condition or with a Repeated Start condition. Since the Repeated Start condition is also the beginning of the next serial transfer, the I²C bus will not be released.

In Master Transmitter mode, serial data is output through SDAx while SCLx outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7 bits) and the Read/Write (R/$\overline{W}$) bit. In this case, the R/$\overline{W}$ bit will be logic '0'. Serial data is transmitted, 8 bits at a time. After each byte is transmitted, an Acknowledge bit is received. Start and Stop conditions are output to indicate the beginning and the end of a serial transfer.

In Master Receive mode, the first byte transmitted contains the slave address of the transmitting device (7 bits) and the R/$\overline{W}$ bit. In this case, the R/$\overline{W}$ bit will be logic '1'. Thus, the first byte transmitted is a 7-bit slave address, followed by a '1' to indicate the receive bit. Serial data is received via SDAx, while SCLx outputs the serial clock. Serial data is received, 8 bits at a time. After each byte is received, an Acknowledge bit is transmitted. Start and Stop conditions indicate the beginning and end of transmission.

The Baud Rate Generator, used for the SPI mode operation, is used to set the SCLx clock frequency for either 100 kHz, 400 kHz or 1 MHz I²C operation. See **Section 20.5.10 "Baud Rate"** for more details.

A typical transmit sequence would go as follows:

1. The user generates a Start condition by setting the Start Enable bit, SEN (SSPxCON2<0>).
2. SSPxIF is set. The MSSPx module will wait the required start time before any other operation takes place.
3. The user loads the SSPxBUF with the slave address to transmit.
4. Address is shifted out the SDAx pin until all 8 bits are transmitted.
5. The MSSPx module shifts in the $\overline{ACK}$ bit from the slave device and writes its value into the SSPxCON2 register (SSPxCON2<6>).
6. The MSSPx module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
7. The user loads the SSPxBUF with eight bits of data.
8. Data is shifted out the SDAx pin until all 8 bits are transmitted.
9. The MSSPx module shifts in the $\overline{ACK}$ bit from the slave device and writes its value into the SSPxCON2 register (SSPxCON2<6>).
10. The MSSPx module generates an interrupt at the end of the ninth clock cycle by setting the SSPxIF bit.
11. The user generates a Stop condition by setting the Stop Enable bit, PEN (SSPxCON2<2>).
12. Interrupt is generated once the Stop condition is complete.

20.5.10 BAUD RATE

In I^2C Master mode, the Baud Rate Generator (BRG) reload value is placed in the lower 7 bits of the SSPxADD register (Figure 20-19). When a write occurs to SSPxBUF, the Baud Rate Generator will automatically begin counting. The BRG counts down to 0 and stops until another reload has taken place. The BRG count is decremented, twice per instruction cycle (TCY), on the Q2 and Q4 clocks. In I^2C Master mode, the BRG is reloaded automatically.

Once the given operation is complete (i.e., transmission of the last data bit is followed by $\overline{ACK}$), the internal clock will automatically stop counting and the SCLx pin will remain in its last state.

Table 20-2 demonstrates clock rates based on instruction cycles and the BRG value loaded into SSPxADD. The SSPxADD BRG value of '0x00' is not supported.

20.5.10.1 Baud Rate and Module Interdependence

Because MSSP1 and MSSP2 are independent, they can operate simultaneously in I^2C Master mode at different baud rates. This is done by using different BRG reload values for each module.

Because this mode derives its basic clock source from the system clock, any changes to the clock will affect both modules in the same proportion. It may be possible to change one or both baud rates back to a previous value by changing the BRG reload value.

FIGURE 20-19: BAUD RATE GENERATOR BLOCK DIAGRAM

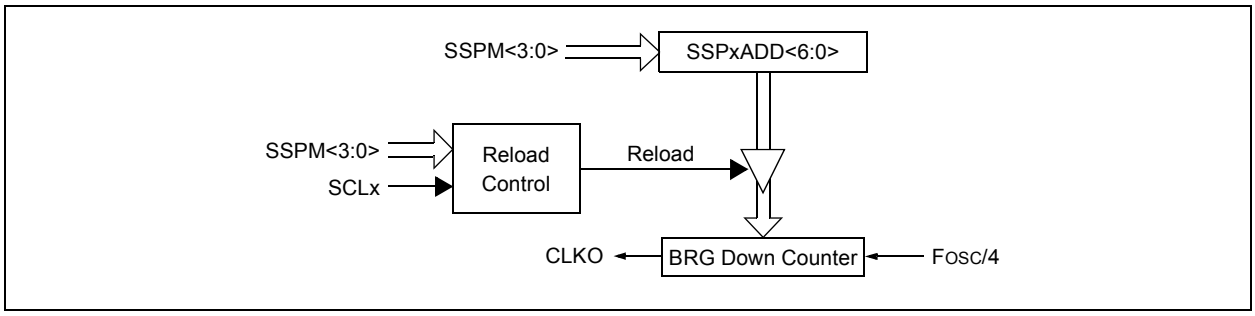


TABLE 20-2: I^2C CLOCK RATE w/BRG

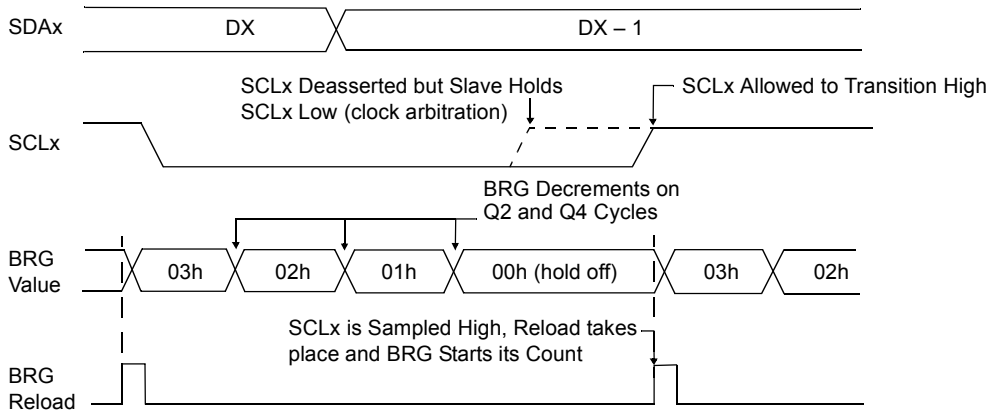
FOSC	FCY	FCY * 2	BRG Value	FSCL (2 Rollovers of BRG)
64 MHz	16 MHz	32 MHz	27h	400 kHz(1)
64 MHz	16 MHz	32 MHz	32h	313.72 kHz
64 MHz	16 MHz	32 MHz	9Fh	100 kHz
16 MHz	4 MHz	8 MHz	09h	400 kHz(1)
16 MHz	4 MHz	8 MHz	0Ch	308 kHz
16 MHz	4 MHz	8 MHz	27h	100 kHz
4 MHz	1 MHz	2 MHz	02h	333 kHz(1)
4 MHz	1 MHz	2 MHz	09h	100 kHz
16 MHz	4 MHz	8 MHz	03h	1 MHz(1,1)

Note 1: A minimum of 16 MHz FOSC is required to get 1 MHz I^2C.

20.5.10.2 Clock Arbitration

Clock arbitration occurs when the master, during any receive, transmit or Repeated Start/Stop condition, deasserts the SCLx pin (SCLx allowed to float high). When the SCLx pin is allowed to float high, the Baud Rate Generator (BRG) is suspended from counting until the SCLx pin is actually sampled high. When the SCLx pin is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and begins counting. This ensures that the SCLx high time will always be at least one BRG rollover count in the event that the clock is held low by an external device (Figure 20-20).

FIGURE 20-20: BAUD RATE GENERATOR TIMING WITH CLOCK ARBITRATION



20.5.11 I²C MASTER MODE START CONDITION TIMING

To initiate a Start condition, the user sets the Start Enable bit, SEN (SSPxCON2<0>). If the SDAx and SCLx pins are sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and starts its count. If SCLx and SDAx are both sampled high when the Baud Rate Generator times out (TBRG), the SDAx pin is driven low. The action of the SDAx being driven low while SCLx is high is the Start condition and causes the S bit (SSPxSTAT<3>) to be set. Following this, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and resumes its count. When the Baud Rate Generator times out (TBRG), the SEN bit (SSPxCON2<0>) will be automatically cleared by hardware. The Baud Rate Generator is suspended, leaving the SDAx line held low and the Start condition is complete.

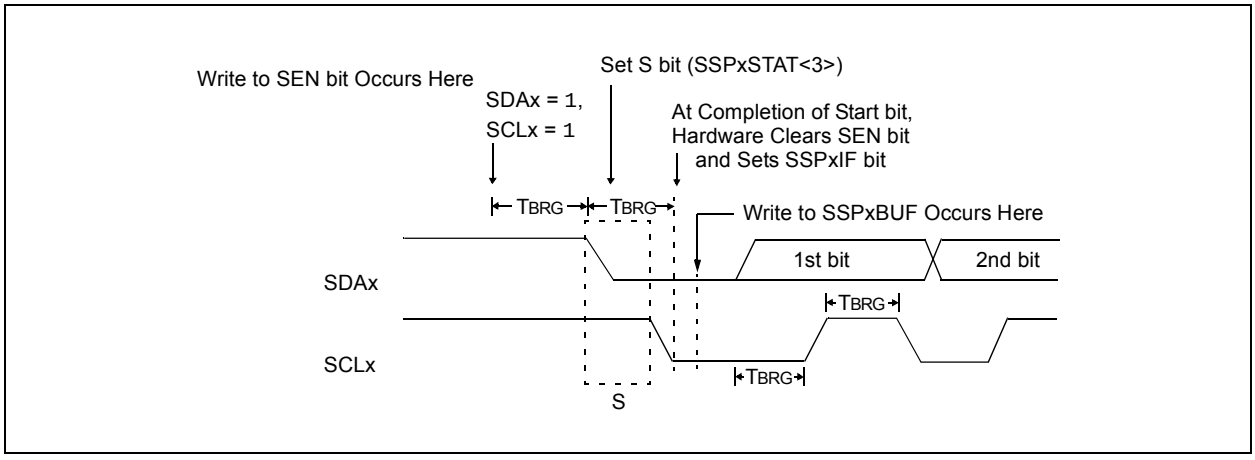
Note:	If, at the beginning of the Start condition, the SDAx and SCLx pins are already sampled low, or if during the Start condition, the SCLx line is sampled low before the SDAx line is driven low, a bus collision occurs, the Bus Collision Interrupt Flag, BCLxIF, is set, the Start condition is aborted and the I²C module is reset into its Idle state.

20.5.11.1 WCOL Status Flag

If the user writes the SSPxBUF when a Start sequence is in progress, the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

Note:	Because queueing of events is not allowed, writing to the lower 5 bits of SSPxCON2 is disabled until the Start condition is complete.

FIGURE 20-21: FIRST START BIT TIMING



20.5.12 I²C MASTER MODE REPEATED START CONDITION TIMING

A Repeated Start condition occurs when the RSEN bit (SSPxCON2<1>) is programmed high and the I²C logic module is in the Idle state. When the RSEN bit is set, the SCLx pin is asserted low. When the SCLx pin is sampled low, the Baud Rate Generator is loaded with the contents of SSPxADD<5:0> and begins counting. The SDAx pin is released (brought high) for one Baud Rate Generator count (TBRG). When the Baud Rate Generator times out, and if SDAx is sampled high, the SCLx pin will be deasserted (brought high). When SCLx is sampled high, the Baud Rate Generator is reloaded with the contents of SSPxADD<6:0> and begins counting. SDAx and SCLx must be sampled high for one TBRG. This action is then followed by assertion of the SDAx pin (SDAx = 0) for one TBRG while SCLx is high. Following this, the RSEN bit (SSPxCON2<1>) will be automatically cleared and the Baud Rate Generator will not be reloaded, leaving the SDAx pin held low. As soon as a Start condition is detected on the SDAx and SCLx pins, the S bit (SSPxSTAT<3>) will be set. The SSPxIF bit will not be set until the Baud Rate Generator has timed out.

Note 1: If RSEN is programmed while any other event is in progress, it will not take effect.

2: A bus collision during the Repeated Start condition occurs if:

- SDAx is sampled low when SCLx goes from low-to-high.
- SCLx goes low before SDAx is asserted low. This may indicate that another master is attempting to transmit a data '1'.

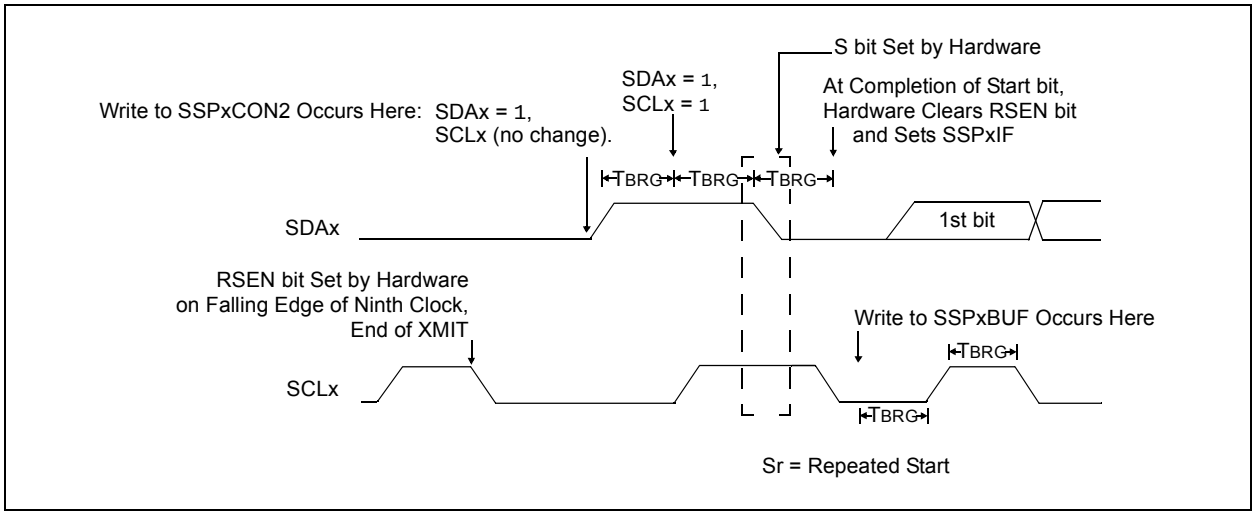
Immediately following the SSPxIF bit getting set, the user may write the SSPxBUF with the 7-bit address in 7-bit mode or the default first address in 10-bit mode. After the first eight bits are transmitted and an $\overline{ACK}$ is received, the user may then transmit an additional eight bits of address (10-bit mode) or eight bits of data (7-bit mode).

20.5.12.1 WCOL Status Flag

If the user writes the SSPxBUF when a Repeated Start sequence is in progress, the WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

Note: Because queueing of events is not allowed, writing of the lower 5 bits of SSPxCON2 is disabled until the Repeated Start condition is complete.

FIGURE 20-22: REPEATED START CONDITION WAVEFORM



20.5.13 I²C MASTER MODE TRANSMISSION

Transmission of a data byte, a 7-bit address or the other half of a 10-bit address, is accomplished by simply writing a value to the SSPxBUF register. This action will set the Buffer Full flag bit, BF, and allow the Baud Rate Generator to begin counting and start the next transmission. Each bit of address/data will be shifted out onto the SDAx pin after the falling edge of SCLx is asserted (see data hold time specification Parameter 106). SCLx is held low for one Baud Rate Generator rollover count (TBRG). Data should be valid before SCLx is released high (see data setup time specification Parameter 107). When the SCLx pin is released high, it is held that way for TBRG. The data on the SDAx pin must remain stable for that duration and some hold time after the next falling edge of SCLx. After the eighth bit is shifted out (the falling edge of the eighth clock), the BF flag is cleared and the master releases SDAx. This allows the slave device being addressed to respond with an ACK bit during the ninth bit time if an address match occurred, or if data was received properly. The status of ACK is written into the ACKDT bit on the falling edge of the ninth clock. If the master receives an Acknowledge, the Acknowledge Status bit, ACKSTAT, is cleared; if not, the bit is set. After the ninth clock, the SSPxIF bit is set and the master clock (Baud Rate Generator) is suspended until the next data byte is loaded into the SSPxBUF, leaving SCLx low and SDAx unchanged (Figure 20-23).

After the write to the SSPxBUF, each bit of the address will be shifted out on the falling edge of SCLx until all seven address bits and the R/W bit are completed. On the falling edge of the eighth clock, the master will deassert the SDAx pin, allowing the slave to respond with an Acknowledge. On the falling edge of the ninth clock, the master will sample the SDAx pin to see if the address was recognized by a slave. The status of the ACK bit is loaded into the ACKSTAT Status bit (SSPxCON2<6>). Following the falling edge of the ninth clock transmission of the address, the SSPxIF flag is set, the BF flag is cleared and the Baud Rate Generator is turned off until another write to the SSPxBUF takes place, holding SCLx low and allowing SDAx to float.

20.5.13.1 BF Status Flag

In Transmit mode, the BF bit (SSPxSTAT<0>) is set when the CPU writes to SSPxBUF and is cleared when all 8 bits are shifted out.

20.5.13.2 WCOL Status Flag

If the user writes the SSPxBUF when a transmit is already in progress (i.e., SSPxSR is still shifting out a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur) after 2 TCY after the SSPxBUF write. If SSPxBUF is rewritten within 2 TCY, the WCOL bit is set and SSPxBUF is updated. This may result in a corrupted transfer.

The user should verify that the WCOL bit is clear after each write to SSPxBUF to ensure the transfer is correct. In all cases, WCOL must be cleared in software.

20.5.13.3 ACKSTAT Status Flag

In Transmit mode, the ACKSTAT bit (SSPxCON2<6>) is cleared when the slave has sent an Acknowledge (ACK = 0) and is set when the slave does not Acknowledge (ACK = 1). A slave sends an Acknowledge when it has recognized its address (including a general call), or when the slave has properly received its data.

20.5.14 I²C MASTER MODE RECEPTION

Master mode reception is enabled by programming the Receive Enable bit, RCEN (SSPxCON2<3>).

Note:	The MSSPx module must be in an inactive state before the RCEN bit is set or the RCEN bit will be disregarded.

The Baud Rate Generator begins counting, and on each rollover, the state of the SCLx pin changes (high-to-low/low-to-high) and data is shifted into the SSPxSR. After the falling edge of the eighth clock, the receive enable flag is automatically cleared, the contents of the SSPxSR are loaded into the SSPxBUF, the BF flag bit is set, the SSPxIF flag bit is set and the Baud Rate Generator is suspended from counting, holding SCLx low. The MSSPx is now in Idle state awaiting the next command. When the buffer is read by the CPU, the BF flag bit is automatically cleared. The user can then send an Acknowledge bit at the end of reception by setting the Acknowledge Sequence Enable bit, ACKEN (SSPxCON2<4>).

20.5.14.1 BF Status Flag

In receive operation, the BF bit is set when an address or data byte is loaded into SSPxBUF from SSPxSR. It is cleared when the SSPxBUF register is read.

20.5.14.2 SSPOV Status Flag

In receive operation, the SSPOV bit is set when 8 bits are received into the SSPxSR and the BF flag bit is already set from a previous reception.

20.5.14.3 WCOL Status Flag

If the user writes the SSPxBUF when a receive is already in progress (i.e., SSPxSR is still shifting in a data byte), the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 20-23: I²C MASTER MODE WAVEFORM (TRANSMISSION, 7 OR 10-BIT ADDRESS)

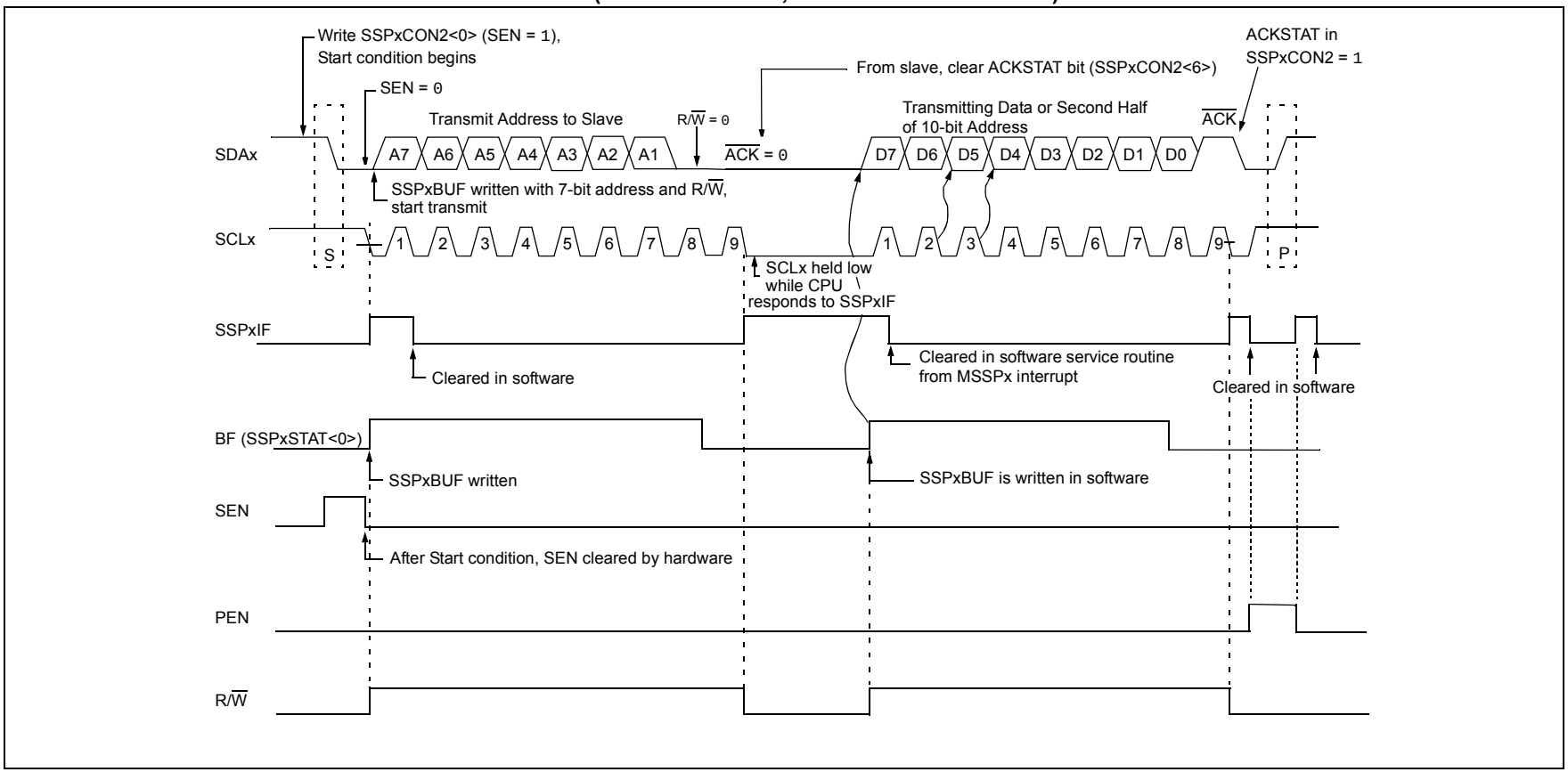
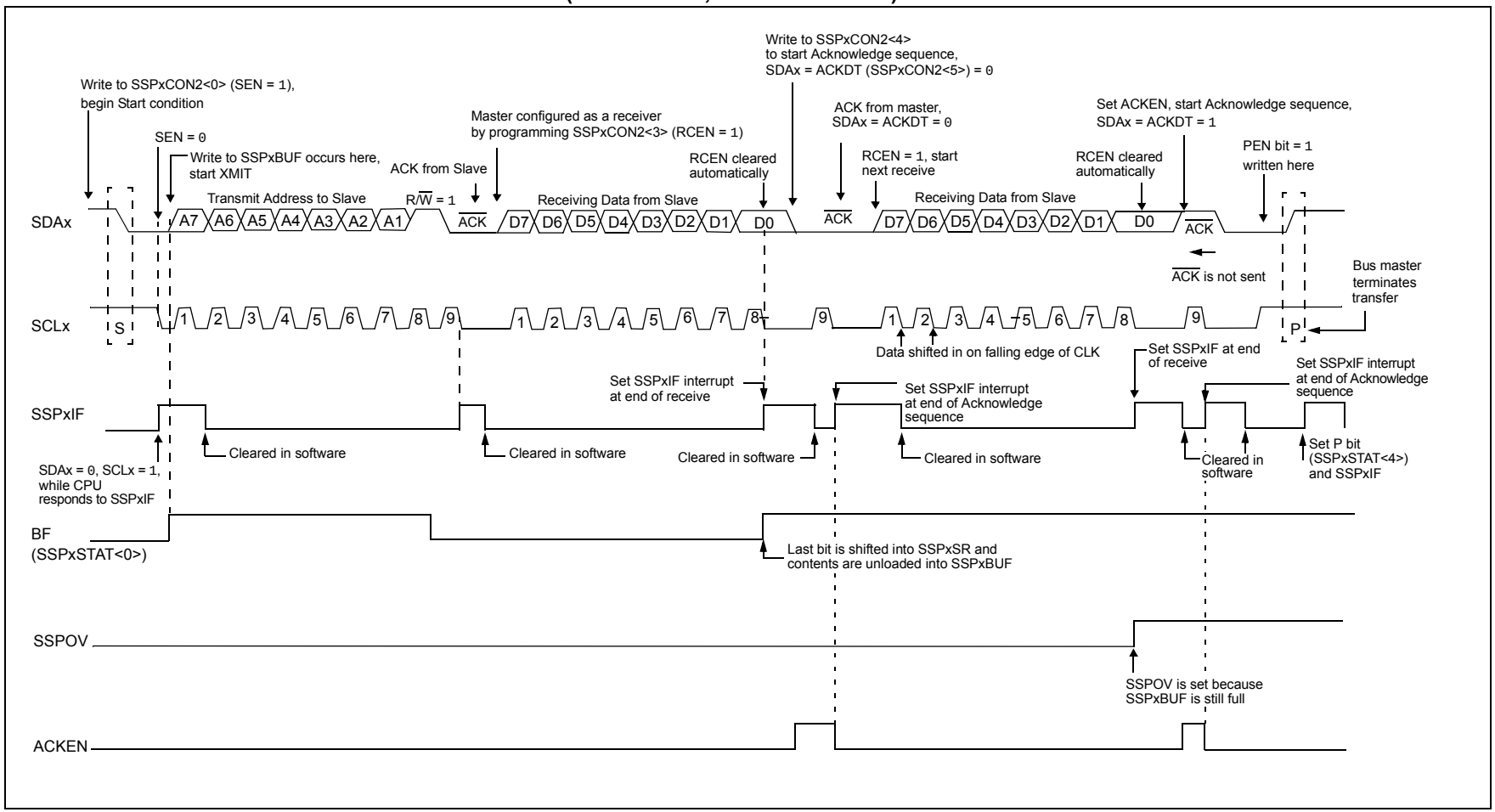


FIGURE 20-24: I²C MASTER MODE WAVEFORM (RECEPTION, 7-BIT ADDRESS)



20.5.15 ACKNOWLEDGE SEQUENCE TIMING

An Acknowledge sequence is enabled by setting the Acknowledge Sequence Enable bit, ACKEN (SSPxCON2<4>). When this bit is set, the SCLx pin is pulled low and the contents of the Acknowledge data bit are presented on the SDAx pin. If the user wishes to generate an Acknowledge, then the ACKDT bit should be cleared. If not, the user should set the ACKDT bit before starting an Acknowledge sequence. The Baud Rate Generator then counts for one rollover period (TBRG) and the SCLx pin is deasserted (pulled high). When the SCLx pin is sampled high (clock arbitration), the Baud Rate Generator counts for TBRG; the SCLx pin is then pulled low. Following this, the ACKEN bit is automatically cleared, the Baud Rate Generator is turned off and the MSSPx module then goes into an inactive state (Figure 20-25).

20.5.15.1 WCOL Status Flag

If the user writes the SSPxBUF when an Acknowledge sequence is in progress, then WCOL is set and the contents of the buffer are unchanged (the write doesn't occur).

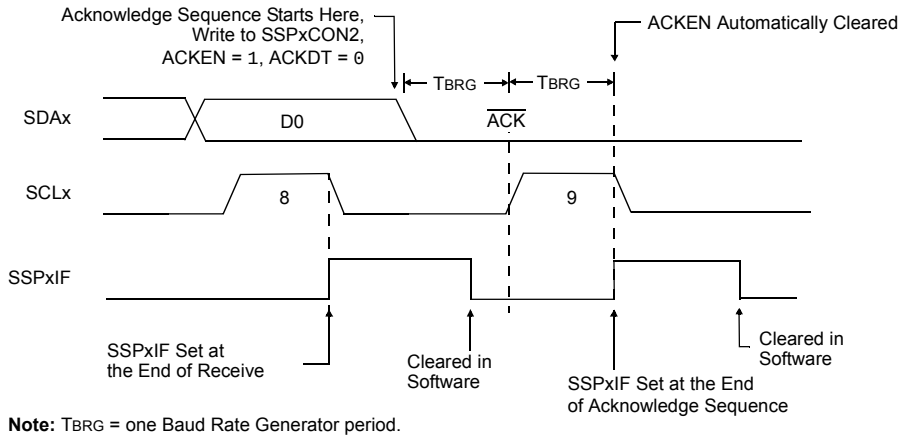
20.5.16 STOP CONDITION TIMING

A Stop bit is asserted on the SDAx pin at the end of a receive/transmit by setting the Stop Sequence Enable bit, PEN (SSPxCON2<2>). At the end of a receive/transmit, the SCLx line is held low after the falling edge of the ninth clock. When the PEN bit is set, the master will assert the SDAx line low. When the SDAx line is sampled low, the Baud Rate Generator is reloaded and counts down to 0. When the Baud Rate Generator times out, the SCLx pin will be brought high and one TBRG (Baud Rate Generator rollover count) later, the SDAx pin will be deasserted. When the SDAx pin is sampled high while SCLx is high, the P bit (SSPxSTAT<4>) is set. A TBRG later, the PEN bit is cleared and the SSPxIF bit is set (Figure 20-26).

20.5.16.1 WCOL Status Flag

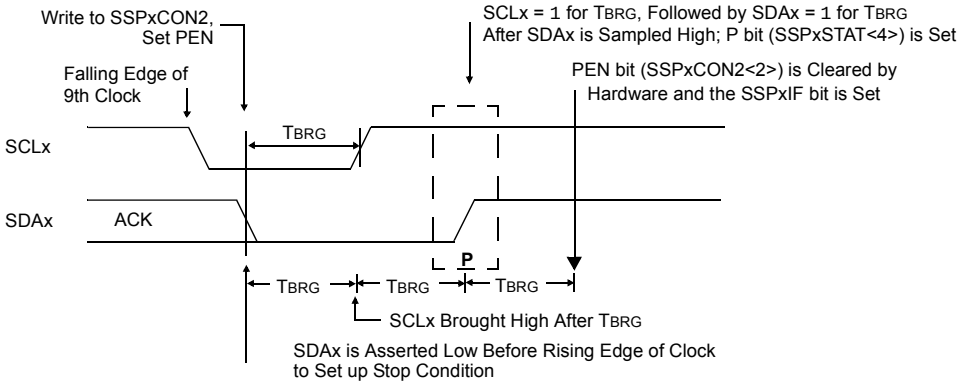
If the user writes the SSPxBUF when a Stop sequence is in progress, then the WCOL bit is set and the contents of the buffer are unchanged (the write doesn't occur).

FIGURE 20-25: ACKNOWLEDGE SEQUENCE WAVEFORM



Note: TBRG = one Baud Rate Generator period.

FIGURE 20-26: STOP CONDITION RECEIVE OR TRANSMIT MODE



Note: TBRG = one Baud Rate Generator period.

20.5.17 SLEEP OPERATION

While in Sleep mode, the I^2C module can receive addresses or data and when an address match or complete byte transfer occurs, wake the processor from Sleep (if the MSSPx interrupt is enabled).

20.5.18 EFFECTS OF A RESET

A Reset disables the MSSPx module and terminates the current transfer.

20.5.19 MULTI-MASTER MODE

In Multi-Master mode, the interrupt generation on the detection of the Start and Stop conditions allows the determination of when the bus is free. The Stop (P) and Start (S) bits are cleared from a Reset or when the MSSPx module is disabled. Control of the I^2C bus may be taken when the P bit (SSPxSTAT<4>) is set, or the bus is Idle, with both the S and P bits clear. When the bus is busy, enabling the MSSPx interrupt will generate the interrupt when the Stop condition occurs.

In multi-master operation, the SDAx line must be monitored for arbitration to see if the signal level is the expected output level. This check is performed in hardware with the result placed in the BCLxIF bit.

The states where arbitration can be lost are:

- Address Transfer
- Data Transfer
- A Start Condition
- A Repeated Start Condition
- An Acknowledge Condition

20.5.20 MULTI -MASTER COMMUNICATION, BUS COLLISION AND BUS ARBITRATION

Multi-Master mode support is achieved by bus arbitration. When the master outputs address/data bits onto the SDAx pin, arbitration takes place when the master outputs a '1' on SDAx, by letting SDAx float high, and another master asserts a '0'. When the SCLx pin floats high, data should be stable. If the expected data on SDAx is a '1' and the data sampled on the SDAx pin = 0, then a bus collision has taken place. The master will set the Bus Collision Interrupt Flag, BCLxIF, and reset the I^2C port to its Idle state (Figure 20-27).

If a transmit was in progress when the bus collision occurred, the transmission is halted, the BF flag is cleared, the SDAx and SCLx lines are deasserted and the SSPxBUF can be written to. When the user services the bus collision Interrupt Service Routine and if the I^2C bus is free, the user can resume communication by asserting a Start condition.

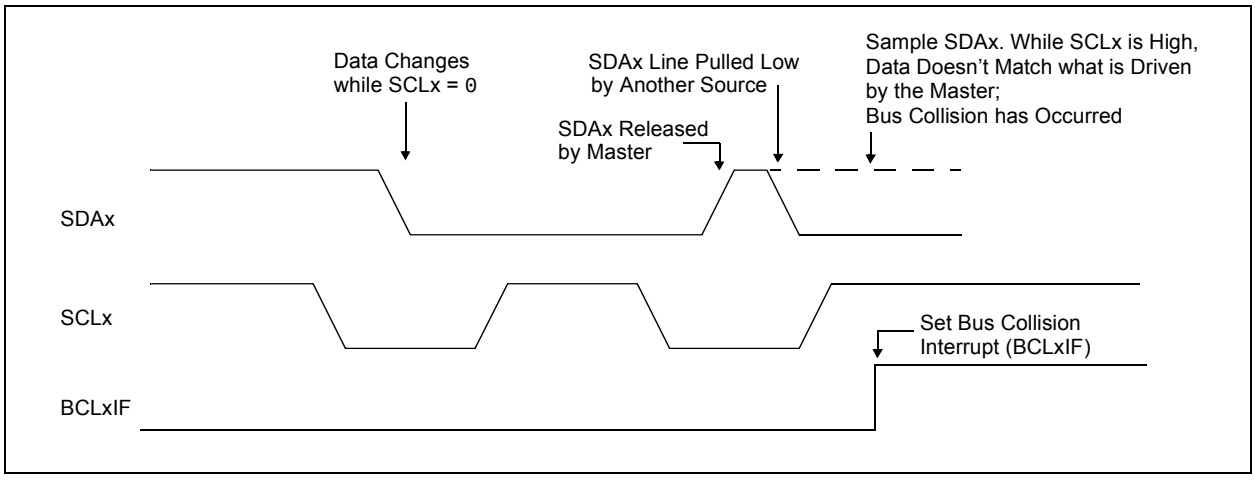
If a Start, Repeated Start, Stop or Acknowledge condition was in progress when the bus collision occurred, the condition is aborted, the SDAx and SCLx lines are deasserted and the respective control bits in the SSPxCON2 register are cleared. When the user services the bus collision Interrupt Service Routine (ISR), and if the I^2C bus is free, the user can resume communication by asserting a Start condition.

The master will continue to monitor the SDAx and SCLx pins. If a Stop condition occurs, the SSPxIF bit will be set.

A write to the SSPxBUF will start the transmission of data at the first data bit regardless of where the transmitter left off when the bus collision occurred.

In Multi-Master mode, the interrupt generation on the detection of Start and Stop conditions allows the determination of when the bus is free. Control of the I^2C bus can be taken when the P bit is set in the SSPxSTAT register, or the bus is Idle and the S and P bits are cleared.

FIGURE 20-27: BUS COLLISION TIMING FOR TRANSMIT AND ACKNOWLEDGE



20.5.20.1 Bus Collision During a Start Condition

During a Start condition, a bus collision occurs if:

a) SDAx or SCLx is sampled low at the beginning of the Start condition (Figure 20-28).

b) SCLx is sampled low before SDAx is asserted low (Figure 20-29).

During a Start condition, both the SDAx and the SCLx pins are monitored.

If the SDAx pin is already low, or the SCLx pin is already low, then all of the following occur:

- the Start condition is aborted,
- the BCLxIF flag is set and
- the MSSPx module is reset to its inactive state (Figure 20-28)

The Start condition begins with the SDAx and SCLx pins deasserted. When the SDAx pin is sampled high, the Baud Rate Generator is loaded from SSPxADD<6:0> and counts down to 0. If the SCLx pin is sampled low while SDAx is high, a bus collision occurs because it is assumed that another master is attempting to drive a data '1' during the Start condition.

If the SDAx pin is sampled low during this count, the BRG is reset and the SDAx line is asserted early (Figure 20-30). If, however, a '1' is sampled on the SDAx pin, the SDAx pin is asserted low at the end of the BRG count. The Baud Rate Generator is then reloaded and counts down to 0. If the SCLx pin is sampled as '0' during this time, a bus collision does not occur. At the end of the BRG count, the SCLx pin is asserted low.

Note:	The reason that bus collision is not a factor during a Start condition is that no two bus masters can assert a Start condition at the exact same time. Therefore, one master will always assert SDAx before the other. This condition does not cause a bus collision because the two masters must be allowed to arbitrate the first address following the Start condition. If the address is the same, arbitration must be allowed to continue into the data portion, Repeated Start or Stop conditions.

FIGURE 20-28: BUS COLLISION DURING START CONDITION (SDAx ONLY)

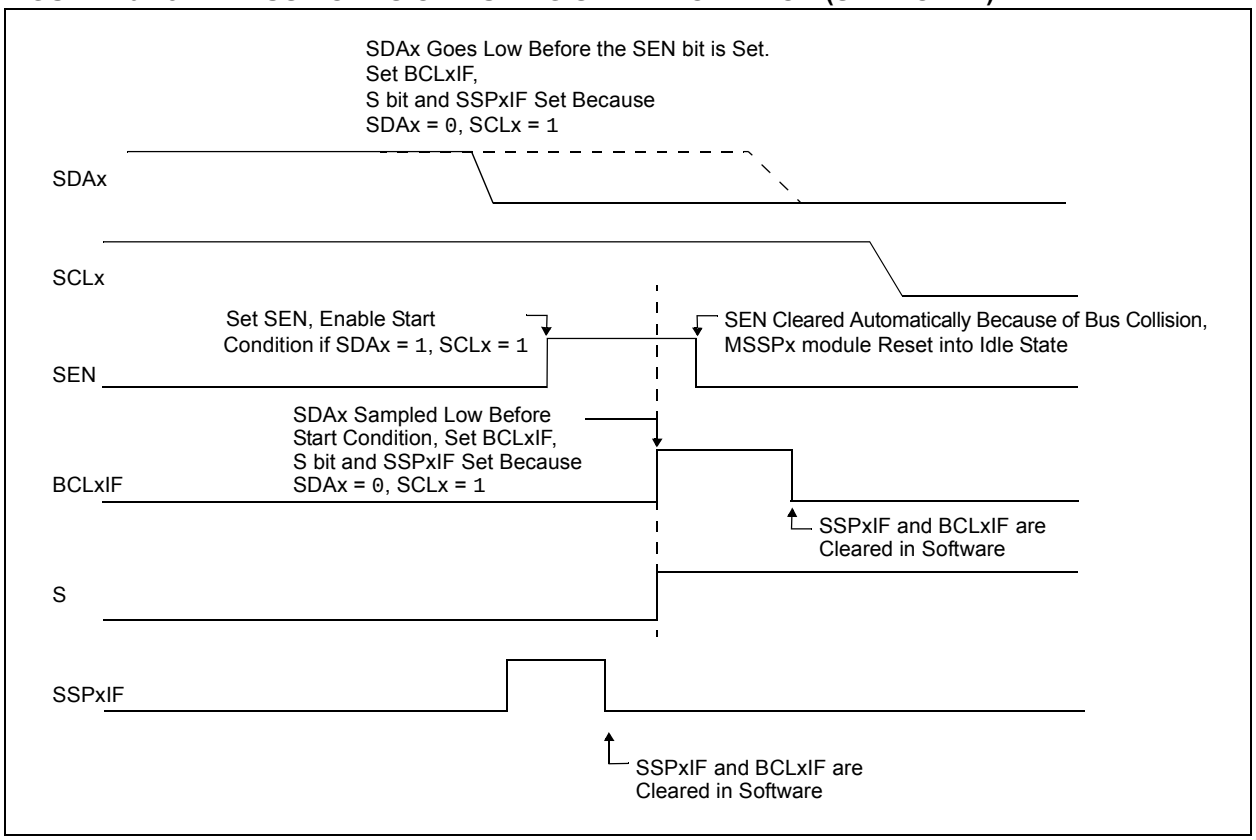


FIGURE 20-29: BUS COLLISION DURING START CONDITION (SCLx = 0)

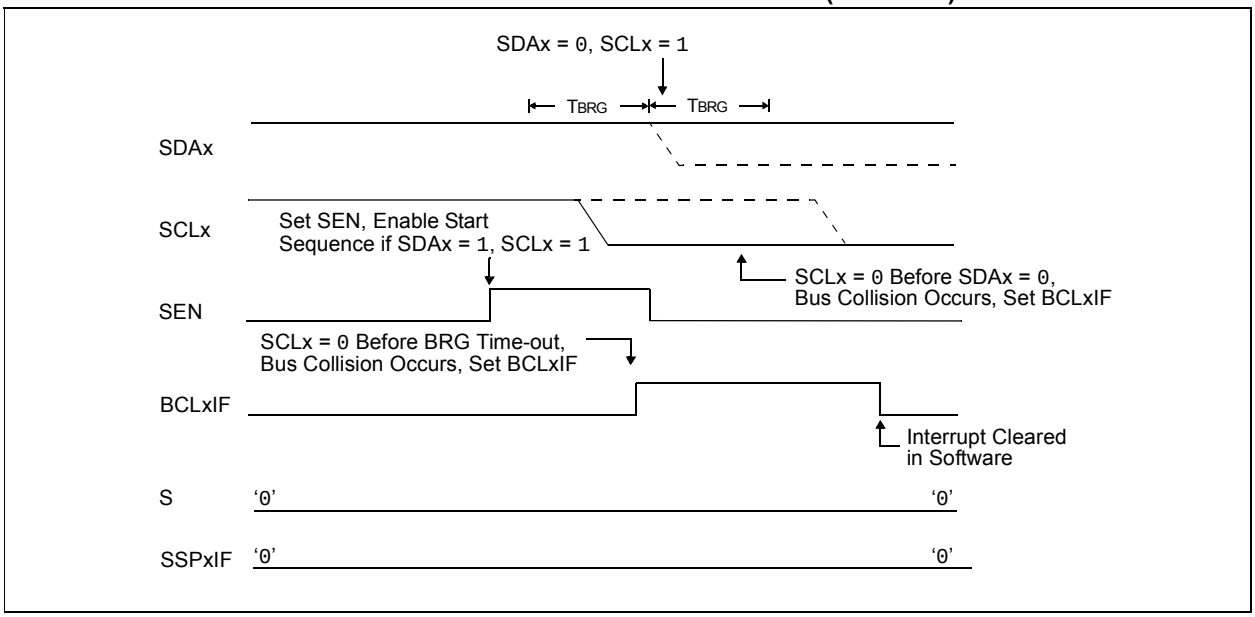
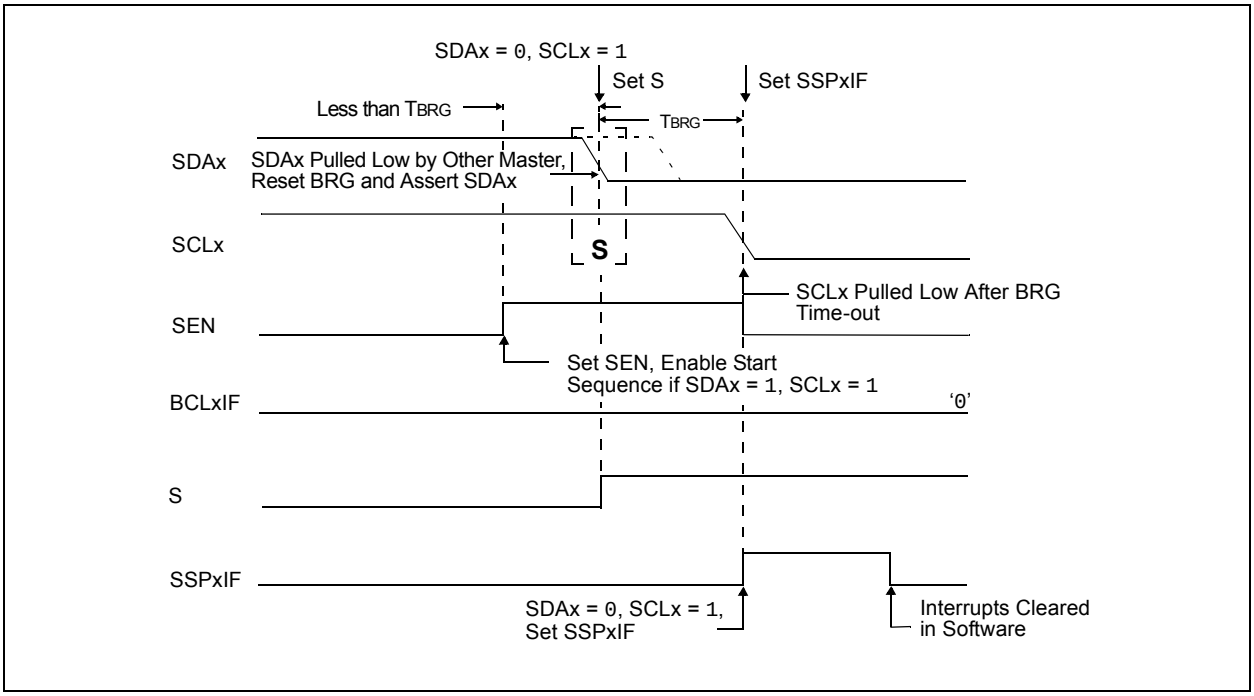


FIGURE 20-30: BRG RESET DUE TO SDAx ARBITRATION DURING START CONDITION



20.5.20.2 Bus Collision During a Repeated Start Condition

During a Repeated Start condition, a bus collision occurs if:

a) A low level is sampled on SDAx when SCLx goes from a low level to a high level.

b) SCLx goes low before SDAx is asserted low, indicating that another master is attempting to transmit a data '1'.

When the user deasserts SDAx and the pin is allowed to float high, the BRG is loaded with SSPxADD<6:0> and counts down to 0. The SCLx pin is then deasserted and when sampled high, the SDAx pin is sampled.

If SDAx is low, a bus collision has occurred (i.e., another master is attempting to transmit a data '0', Figure 20-31). If SDAx is sampled high, the BRG is reloaded and begins counting. If SDAx goes from high-to-low before the BRG times out, no bus collision occurs because no two masters can assert SDAx at exactly the same time.

If SCLx goes from high-to-low before the BRG times out and SDAx has not already been asserted, a bus collision occurs. In this case, another master is attempting to transmit a data '1' during the Repeated Start condition (see Figure 20-32).

If, at the end of the BRG time-out, both SCLx and SDAx are still high, the SDAx pin is driven low and the BRG is reloaded and begins counting. At the end of the count, regardless of the status of the SCLx pin, the SCLx pin is driven low and the Repeated Start condition is complete.

FIGURE 20-31: BUS COLLISION DURING A REPEATED START CONDITION (CASE 1)

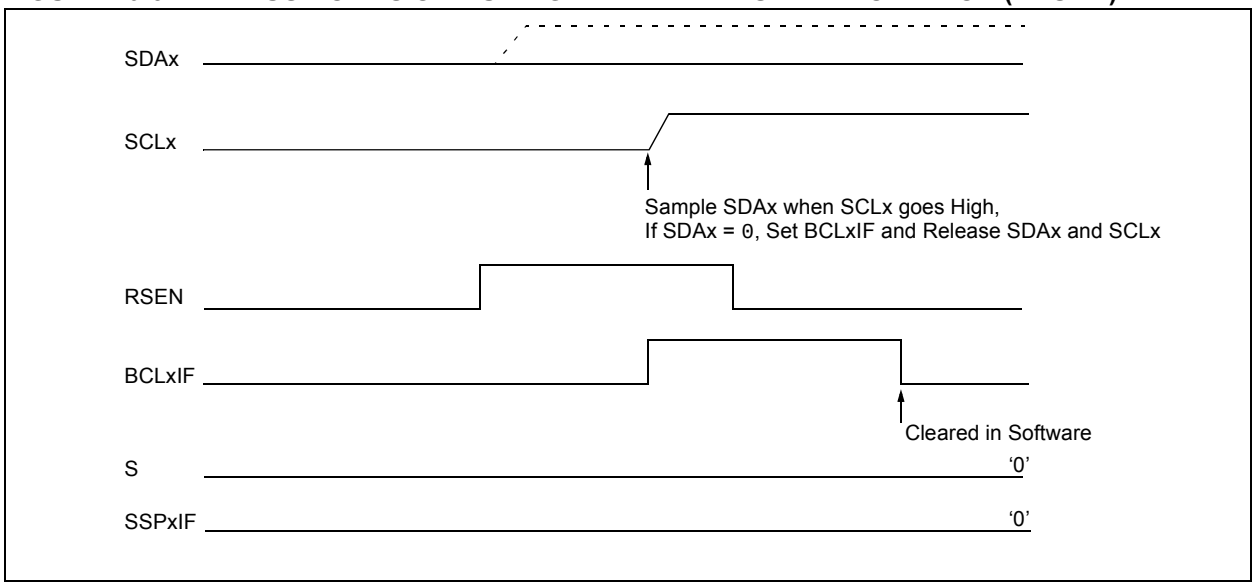
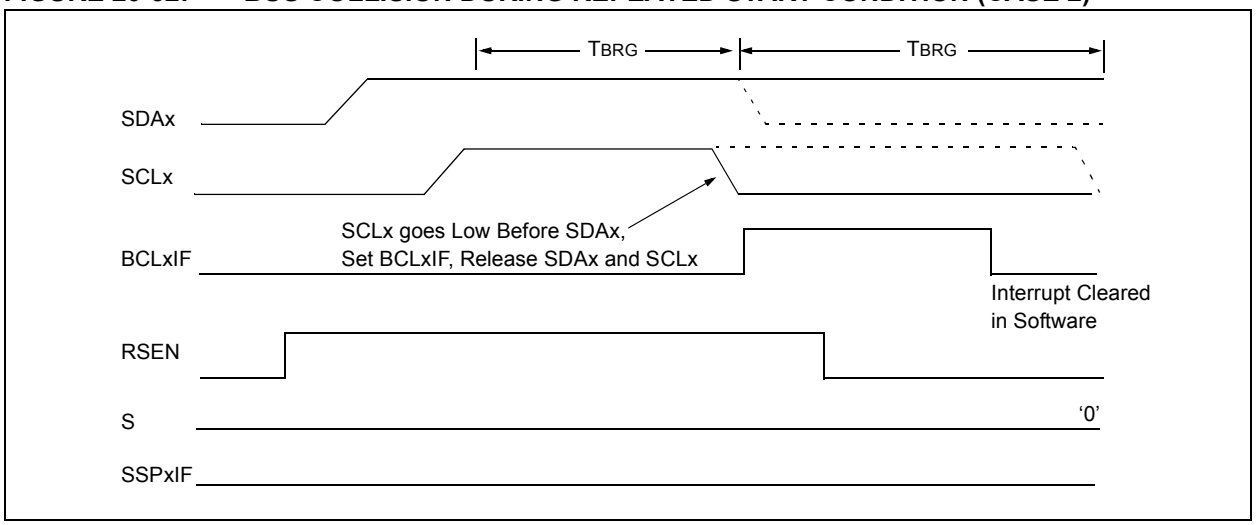


FIGURE 20-32: BUS COLLISION DURING REPEATED START CONDITION (CASE 2)



20.5.20.3 Bus Collision During a Stop Condition

Bus collision occurs during a Stop condition if:

a) After the SDAx pin has been deasserted and allowed to float high, SDAx is sampled low after the BRG has timed out.

b) After the SCLx pin is deasserted, SCLx is sampled low before SDAx goes high.

The Stop condition begins with SDAx asserted low. When SDAx is sampled low, the SCLx pin is allowed to float. When the pin is sampled high (clock arbitration), the Baud Rate Generator is loaded with SSPxADD<6:0> and counts down to 0. After the BRG times out, SDAx is sampled. If SDAx is sampled low, a bus collision has occurred. This is due to another master attempting to drive a data '0' (Figure 20-33). If the SCLx pin is sampled low before SDAx is allowed to float high, a bus collision occurs. This is another case of another master attempting to drive a data '0' (Figure 20-34).

FIGURE 20-33: BUS COLLISION DURING A STOP CONDITION (CASE 1)

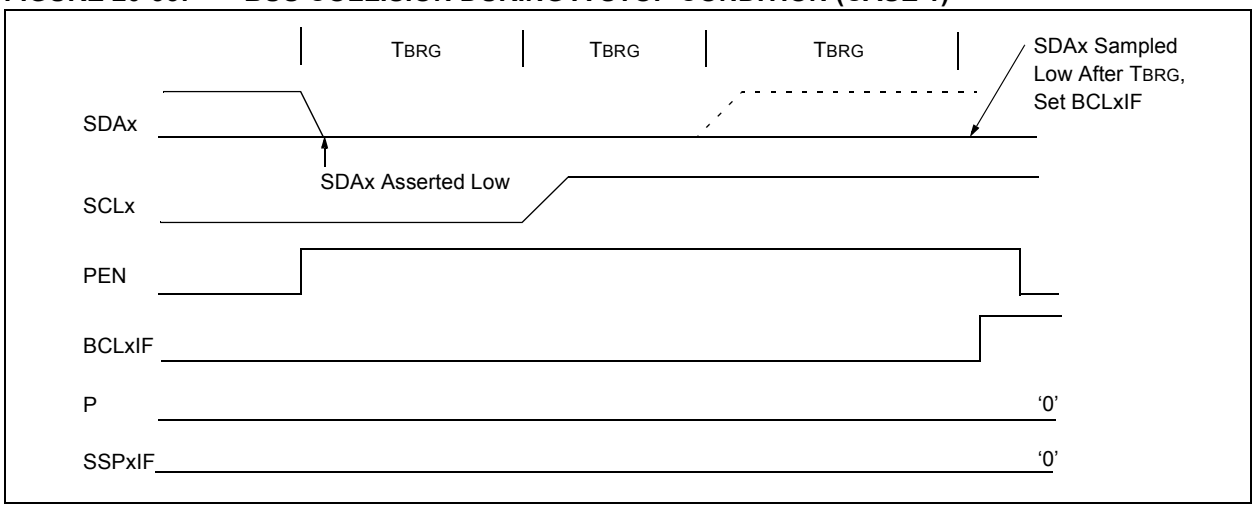
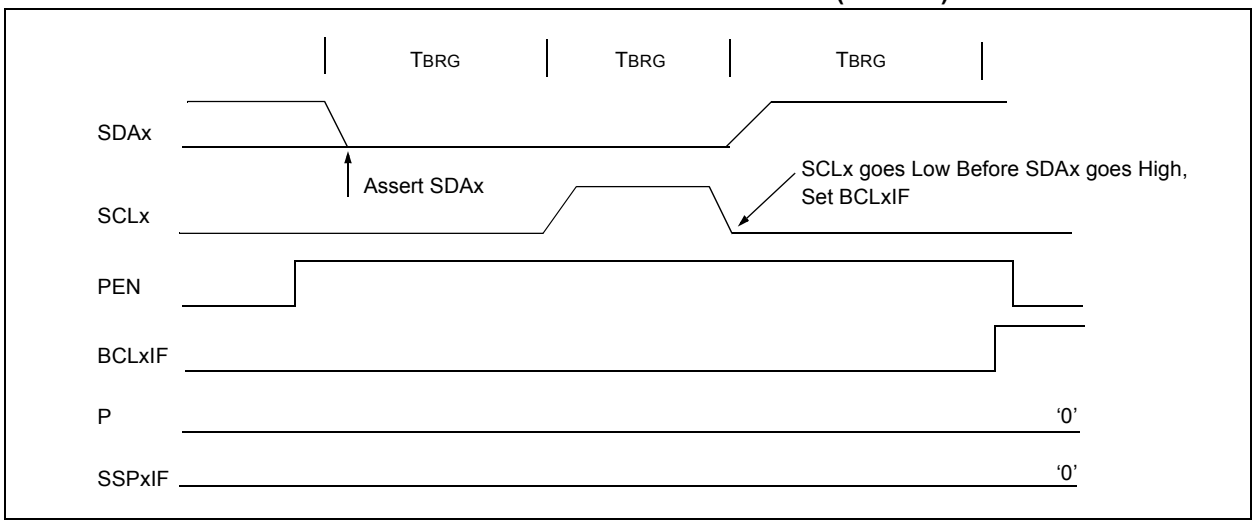


FIGURE 20-34: BUS COLLISION DURING A STOP CONDITION (CASE 2)



21.0 ENHANCED UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (EUSART)

The Enhanced Universal Synchronous Asynchronous Receiver Transmitter (EUSART) module is one of four serial I/O modules. (Generically, the EUSART is also known as a Serial Communications Interface or SCI.) The EUSART can be configured as a full-duplex, asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers. It can also be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The Enhanced USART module implements additional features, including automatic baud rate detection and calibration, automatic wake-up on Sync Break reception and 12-bit Break character transmit. These make it ideally suited for use in Local Interconnect Network bus (LIN/J2602 bus) systems.

All members of the PIC18FXXJ94 are equipped with four independent EUSART modules, referred to as EUSART1, EUSART2, EUSART3 and EUSART4. They can be configured in the following modes:

- Asynchronous (full duplex) with:
  - Auto-wake-up on character reception
  - Auto-baud calibration
  - 12-bit Break character transmission
- Synchronous – Master (half duplex) with selectable clock polarity
- Synchronous – Slave (half duplex) with selectable clock polarity

The Enhanced USART module has the following enhancements over the AUSART module:

- Selectable 16-Bit Baud Rate Generator mode
- Interrupt on Sync Break character received; allows an asynchronous wake-up from Sleep
- 12-Bit Break character transmit
- Auto-baud calibration on Sync character
- Clock polarity select for Synchronous mode
- Transmit and receive polarity select for Asynchronous mode
- Receive Shift register empty Status bit
- Local Interconnect Network (LIN/J2602) protocol standard

The Enhanced USART module has the following IrDA® related enhancements over previous Enhanced USART modules:

- 16x Baud Clock output for IrDA support
- IrDA encoder and decoder logic

The pins of EUSART1 through EUSART4 are multiplexed with functions using PPS-Lite. The TXx and RXx pins of each EUSART can be individually controlled using PPS-Lite registers, respectively.

Refer to **Section 11.15 "PPS-Lite"** for setting up EUSART1 through EUSART4.

Note:	The EUSART control will automatically reconfigure the pin from input to output as needed.

The operation of each Enhanced USART module is controlled through seven registers:

- RCSTAx – EUSARTx Receive Status and Control Register
- TXSTAx – EUSARTx Transmit Status and Control Register
- BAUDCONx – Baud Rate Control Register
- SPBRGx – Baud Rate Generator Register
- SPBRGHx – Baud Rate Generator High Register
- RCREGx – EUSARTx Receive Data Register
- TXREGx – EUSARTx Transmit Data Register

These are detailed on the following pages in Register 21-1, Register 21-2 and Register 21-3, respectively.

Note:	Throughout this section, references to register and bit names that may be associated with a specific EUSART module are referred to generically by the use of 'x' in place of the specific module number. Thus, "RCSTAx" might refer to the Receive STATUS register for either EUSART1, EUSART2, EUSART3 or EUSART4.

REGISTER 21-1: TXSTAx: EUSARTx TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN(1)	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CSRC:** Clock Source Select bit

Asynchronous mode:
Don't care.

Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9:** 9-Bit Transmit Enable bit

1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN:** Transmit Enable bit(1)

1 = Transmit is enabled
0 = Transmit is disabled

bit 4 **SYNC:** EUSARTx Mode Select bit

1 = Synchronous mode
0 = Asynchronous mode

bit 3 **SENDB:** Send Break Character bit

Asynchronous mode:
1 = Send Sync Break on next transmission (cleared by hardware upon completion)
0 = Sync Break transmission has completed

Synchronous mode:
Don't care.

bit 2 **BRGH:** High Baud Rate Select bit

Asynchronous mode:
1 = High speed
0 = Low speed

Synchronous mode:
Unused in this mode.

bit 1 **TRMT:** Transmit Shift Register Status bit

1 = TSR is empty
0 = TSR is full

bit 0 **TX9D:** 9th bit of Transmit Data

Can be address/data bit or a parity bit.

Note 1: SREN/CREN overrides TXEN in Sync mode.

REGISTER 21-2: RCSTAx: EUSARTx RECEIVE STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **SPEN:** Serial Port Enable bit
1 = Serial port is enabled
0 = Serial port is disabled (held in Reset)

bit 6 **RX9:** 9-Bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN:** Single Receive Enable bit
Asynchronous mode:
Don't care.
Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode – Slave:
Don't care.

bit 4 **CREN:** Continuous Receive Enable bit
Asynchronous mode:
1 = Enables receiver
0 = Disables receiver
Synchronous mode:
1 = Enables continuous receive until enable bit, CREN, is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-Bit (RX9 = 1):
1 = Enables address detection, enables interrupt and loads the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and the ninth bit can be used as a parity bit
Asynchronous mode 9-Bit (RX9 = 0):
Don't care.

bit 2 **FERR:** Framing Error bit
1 = Framing error (can be cleared by reading the RCREGx register and receiving the next valid byte)
0 = No framing error

bit 1 **OERR:** Overrun Error bit
1 = Overrun error (can be cleared by clearing bit, CREN)
0 = No overrun error

bit 0 **RX9D:** 9th bit of Received Data
This can be an address/data bit or a parity bit and must be calculated by user firmware.

REGISTER 21-3: BAUDCONx: BAUD RATE CONTROL REGISTER x

R/W-0	R-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ABDOVF	RCIDL	RXDTP	TXCKP	BRG16	IREN	WUE	ABDEN
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ABDOVF:** Auto-Baud Acquisition Rollover Status bit
1 = A BRG rollover has occurred during Auto-Baud Rate Detect mode (must be cleared in software)
0 = No BRG rollover has occurred

bit 6 **RCIDL:** Receive Operation Idle Status bit
1 = Receive operation is Idle
0 = Receive operation is active

bit 5 **RXDTP:** Data/Receive Polarity Select bit
Asynchronous mode:
1 = Receive data (RXx) is inverted (active-low)
0 = Receive data (RXx) is not inverted (active-high)
Asynchronous IrDA mode:
No effect on operation
Synchronous mode:
1 = Data (DTx) is inverted (active-low)
0 = Data (DTx) is not inverted (active-high)

bit 4 **TXCKP:** Synchronous Clock Polarity Select bit
Asynchronous mode:
1 = Idle state for transmit (TXx) is a low level
0 = Idle state for transmit (TXx) is a high level
Asynchronous IrDA mode:
1 = Idle state for IrDA transmit (TX) is a high level ('1')
0 = Idle state for IrDA transmit (TX) is a low level ('0')
Synchronous mode:
1 = Idle state for clock (CKx) is a high level
0 = Idle state for clock (CKx) is a low level

bit 3 **BRG16:** 16-Bit Baud Rate Register Enable bit
1 = 16-bit Baud Rate Generator – SPBRGHx and SPBRGx
0 = 8-bit Baud Rate Generator – SPBRGx only (Compatible mode), SPBRGHx value is ignored

bit 2 **IREN:** IrDA® Encoder and Decoder Enable bit
Asynchronous mode:
1 = IrDA encoder and decoder are enabled (Asynchronous IrDA mode is active)
0 = IrDA encoder and decoder are disabled
Synchronous mode:(1)
No effect on operation.

bit 1 **WUE:** Wake-up Enable bit
Asynchronous mode:
1 = EUSARTx will continue to sample the RXx pin – interrupt is generated on the falling edge; bit is cleared in hardware on following rising edge
0 = RXx pin is not monitored or rising edge detected
Synchronous mode:
Unused in this mode.

Note 1: This feature is only available in Asynchronous mode with the 16x clock preset. The 16x clock is present for both the x16 and x64 BRG configurations.

bit 0 **ABDEN:** Auto-Baud Detect Enable bit

Asynchronous mode:

1 = Enables baud rate measurement on the next character, requires reception of a Sync field (55h); cleared in hardware upon completion

0 = Baud rate measurement is disabled or has completed

Synchronous mode:

Unused in this mode.

Note 1: This feature is only available in Asynchronous mode with the 16x clock preset. The 16x clock is present for both the x16 and x64 BRG configurations.

21.1 Baud Rate Generator (BRG)

The BRG is a dedicated, 8-bit or 16-bit generator that supports both the Asynchronous and Synchronous modes of the EUSARTx. By default, the BRG operates in 8-bit mode; setting the BRG16 bit (BAUDCONx<3>) selects 16-bit mode.

The SPBRGHx:SPBRGx register pair controls the period of a free-running timer. In Asynchronous mode, bits, BRGH (TXSTAx<2>) and BRG16 (BAUDCONx<3>), also control the baud rate. In Synchronous mode, BRGH is ignored. Table 21-1 shows the formula for computation of the baud rate for different EUSARTx modes which only apply in Master mode (internally generated clock).

Given the desired baud rate and FOSC, the nearest integer value for the SPBRGHx:SPBRGx registers can be calculated using the formulas in Table 21-1. From this, the error in baud rate can be determined. An example calculation is shown in Example 21-1. Typical baud rates and error values for the various Asynchronous modes are shown in Table 21-2. It may be advantageous to use the high baud rate (BRGH = 1) or the 16-bit BRG to reduce the baud rate error, or achieve a slow baud rate for a fast oscillator frequency.

Writing a new value to the SPBRGHx:SPBRGx registers causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate. When operated in the Synchronous mode, SPBRGH:SPBRG values of 0000h and 0001h are not supported. In the Asynchronous mode, all BRG values may be used.

21.1.1 OPERATION IN POWER-MANAGED MODES

The device clock is used to generate the desired baud rate. When one of the power-managed modes is entered, the new clock source may be operating at a different frequency. This may require an adjustment to the value in the SPBRGx register pair.

21.1.2 SAMPLING

The data on the RXx pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RXx pin.

TABLE 21-1: BAUD RATE FORMULAS

Configuration Bits			BRG/EUSART Mode	Baud Rate Formula
SYNC	BRG16	BRGH		
0	0	0	8-bit/Asynchronous	FOSC/[64 (n + 1)]
0	0	1	8-bit/Asynchronous	FOSC/[16 (n + 1)]
0	1	0	16-bit/Asynchronous	
0	1	1	16-bit/Asynchronous	FOSC/[4 (n + 1)]
1	0	x	8-bit/Synchronous	
1	1	x	16-bit/Synchronous	

Legend: x = Don't care, n = value of SPBRGHx:SPBRGx register pair

EXAMPLE 21-1: CALCULATING BAUD RATE ERROR

For a device with FOSC of 16 MHz, desired baud rate of 9600, Asynchronous mode, and 8-bit BRG:

Desired Baud Rate = FOSC/(64 ([SPBRGHx:SPBRGx] + 1))

Solving for SPBRGHx:SPBRGx:

X = ((FOSC/Desired Baud Rate)/64) – 1
= ((16000000/9600)/64) – 1
= [25.042] = 25

Calculated Baud Rate = 16000000/(64 (25 + 1))
= 9615

Error = (Calculated Baud Rate – Desired Baud Rate)/Desired Baud Rate
= (9615 – 9600)/9600 = 0.16%

TABLE 21-2: BAUD RATES FOR ASYNCHRONOUS MODES

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	1.221	1.73	255	1.202	0.16	129	1.201	-0.16	103
2.4	2.441	1.73	255	2.404	0.16	129	2.404	0.16	64	2.403	-0.16	51
9.6	9.615	0.16	64	9.766	1.73	31	9.766	1.73	15	9.615	-0.16	12
19.2	19.531	1.73	31	19.531	1.73	15	19.531	1.73	7	—	—	—
57.6	56.818	-1.36	10	62.500	8.51	4	52.083	-9.58	2	—	—	—
115.2	125.000	8.51	4	104.167	-9.58	2	78.125	-32.18	1	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.16	207	0.300	-0.16	103	0.300	-0.16	51
1.2	1.202	0.16	51	1.201	-0.16	25	1.201	-0.16	12
2.4	2.404	0.16	25	2.403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0											
	FOSC = 40.000 MHz			FOSC = 20.000 MHz			FOSC = 10.000 MHz			FOSC = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	—	—	—	—	—	—
1.2	—	—	—	—	—	—	—	—	—	—	—	—
2.4	—	—	—	—	—	—	2.441	1.73	255	2.403	-0.16	207
9.6	9.766	1.73	255	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 0								
	FOSC = 4.000 MHz			FOSC = 2.000 MHz			FOSC = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	—	—	—	—	—	—	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

TABLE 21-2: BAUD RATES FOR ASYNCHRONOUS MODES (CONTINUED)

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	8332	0.300	0.02	4165	0.300	0.02	2082	0.300	-0.04	1665
1.2	1.200	0.02	2082	1.200	-0.03	1041	1.200	-0.03	520	1.201	-0.16	415
2.4	2.402	0.06	1040	2.399	-0.03	520	2.404	0.16	259	2.403	-0.16	207
9.6	9.615	0.16	259	9.615	0.16	129	9.615	0.16	64	9.615	-0.16	51
19.2	19.231	0.16	129	19.231	0.16	64	19.531	1.73	31	19.230	-0.16	25
57.6	58.140	0.94	42	56.818	-1.36	21	56.818	-1.36	10	55.555	3.55	8
115.2	113.636	-1.36	21	113.636	-1.36	10	125.000	8.51	4	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.04	832	0.300	-0.16	415	0.300	-0.16	207
1.2	1.202	0.16	207	1.201	-0.16	103	1.201	-0.16	51
2.4	2.404	0.16	103	2.403	-0.16	51	2.403	-0.16	25
9.6	9.615	0.16	25	9.615	-0.16	12	—	—	—
19.2	19.231	0.16	12	—	—	—	—	—	—
57.6	62.500	8.51	3	—	—	—	—	—	—
115.2	125.000	8.51	1	—	—	—	—	—	—

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1											
	Fosc = 40.000 MHz			Fosc = 20.000 MHz			Fosc = 10.000 MHz			Fosc = 8.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.00	33332	0.300	0.00	16665	0.300	0.00	8332	0.300	-0.01	6665
1.2	1.200	0.00	8332	1.200	0.02	4165	1.200	0.02	2082	1.200	-0.04	1665
2.4	2.400	0.02	4165	2.400	0.02	2082	2.402	0.06	1040	2.400	-0.04	832
9.6	9.606	0.06	1040	9.596	-0.03	520	9.615	0.16	259	9.615	-0.16	207
19.2	19.193	-0.03	520	19.231	0.16	259	19.231	0.16	129	19.230	-0.16	103
57.6	57.803	0.35	172	57.471	-0.22	86	58.140	0.94	42	57.142	0.79	34
115.2	114.943	-0.22	86	116.279	0.94	42	113.636	-1.36	21	117.647	-2.12	16

BAUD RATE (K)	SYNC = 0, BRGH = 1, BRG16 = 1 or SYNC = 1, BRG16 = 1								
	Fosc = 4.000 MHz			Fosc = 2.000 MHz			Fosc = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.01	3332	0.300	-0.04	1665	0.300	-0.04	832
1.2	1.200	0.04	832	1.201	-0.16	415	1.201	-0.16	207
2.4	2.404	0.16	415	2.403	-0.16	207	2.403	-0.16	103
9.6	9.615	0.16	103	9.615	-0.16	51	9.615	-0.16	25
19.2	19.231	0.16	51	19.230	-0.16	25	19.230	-0.16	12
57.6	58.824	2.12	16	55.555	3.55	8	—	—	—
115.2	111.111	-3.55	8	—	—	—	—	—	—

21.1.3 AUTO-BAUD RATE DETECT

The Enhanced USART module supports the automatic detection and calibration of baud rate. This feature is active only in Asynchronous mode and while the WUE bit is clear.

The automatic baud rate measurement sequence (Figure 21-1) begins whenever a Start bit is received and the ABDEN bit is set. The calculation is self-averaging.

In the Auto-Baud Rate Detect (ABD) mode, the clock to the BRG is reversed. Rather than the BRG clocking the incoming RXx signal, the RXx signal is timing the BRG. In ABD mode, the internal Baud Rate Generator is used as a counter to time the bit period of the incoming serial byte stream.

Once the ABDEN bit is set, the state machine will clear the BRG and look for a Start bit. The Auto-Baud Rate Detect must receive a byte with the value, 55h (ASCII "U", which is also the LIN/J2602 bus Sync character), in order to calculate the proper bit rate. The measurement is taken over both a low and a high bit time in order to minimize any effects caused by asymmetry of the incoming signal. After a Start bit, the SPBRGx begins counting up, using the preselected clock source on the first rising edge of RXx. After eight bits on the RXx pin or the fifth rising edge, an accumulated value totalling the proper BRG period is left in the SPBRGHx:SPBRGx register pair. Once the 5th edge is seen (this should correspond to the Stop bit), the ABDEN bit is automatically cleared.

If a rollover of the BRG occurs (an overflow from FFFFh to 0000h), the event is trapped by the ABDOVF Status bit (BAUDCONx<7>). It is set in hardware by BRG rollovers and can be set or cleared by the user in software. ABD mode remains active after rollover events and the ABDEN bit remains set (Figure 21-2).

While calibrating the baud rate period, the BRG registers are clocked at 1/8th the preconfigured clock rate. The BRG clock will be configured by the BRG16 and BRGH bits. The BRG16 bit must be set to use both SPBRG1 and SPBRGH1 as a 16-bit counter. This allows the user to verify that no carry occurred for 8-bit modes by checking for 00h in the SPBRGHx register. Refer to Table 21-3 for counter clock rates to the BRG.

While the ABD sequence takes place, the EUSARTx state machine is held in Idle. The RCxIF interrupt is set once the fifth rising edge on RXx is detected. The value in the RCREGx needs to be read to clear the RCxIF interrupt. The contents of RCREGx should be discarded.

Note 1: If the WUE bit is set with the ABDEN bit, Auto-Baud Rate Detection will occur on the byte *following* the Break character.

2: It is up to the user to determine that the incoming character baud rate is within the range of the selected BRG clock source. Some combinations of oscillator frequency and EUSARTx baud rates are not possible due to bit error rates. Overall system timing and communication baud rates must be taken into consideration when using the Auto-Baud Rate Detection feature.

3: To maximize baud rate range, if that feature is used it is recommended that the BRG16 bit (BAUDCONx<3>) be set.

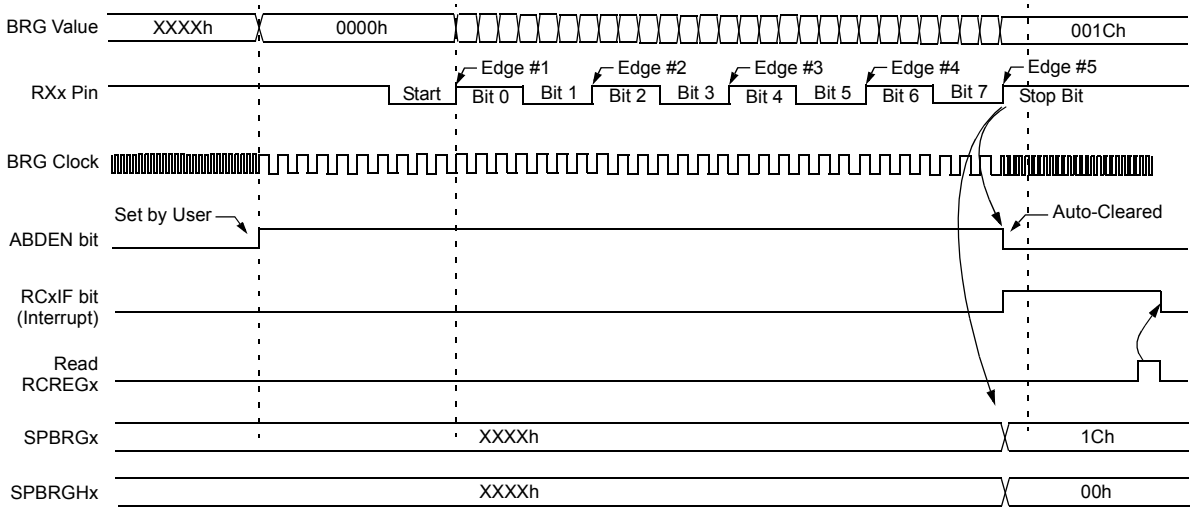
TABLE 21-3: BRG COUNTER CLOCK RATES

BRG16	BRGH	BRG Counter Clock
0	0	FOSC/512
0	1	FOSC/128
1	0	FOSC/128
1	1	FOSC/32

21.1.3.1 ABD and EUSARTx Transmission

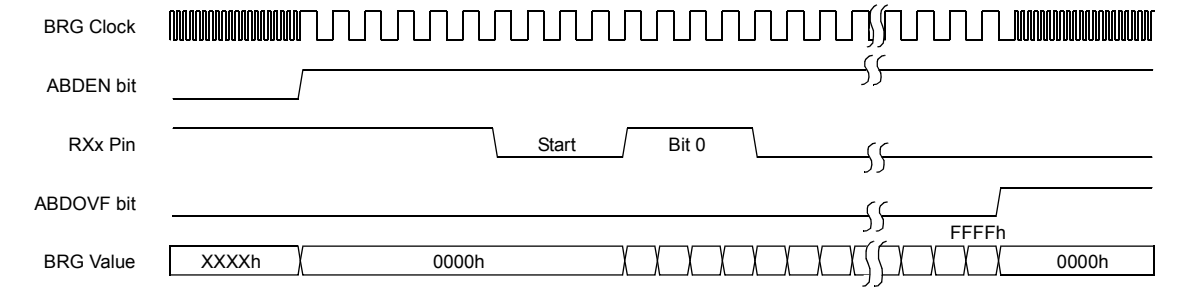
Since the BRG clock is reversed during ABD acquisition, the EUSARTx transmitter cannot be used during ABD. This means that whenever the ABDEN bit is set, TXREGx cannot be written to. Users should also ensure that ABDEN does not become set during a transmit sequence. Failing to do this may result in unpredictable EUSARTx operation.

FIGURE 21-1: AUTOMATIC BAUD RATE CALCULATION



Note: The ABD sequence requires the EUSARTx module to be configured in Asynchronous mode and WUE = 0.

FIGURE 21-2: BRG OVERFLOW SEQUENCE



21.2 EUSARTx Asynchronous Mode

The Asynchronous mode of operation is selected by clearing the SYNC bit (TXSTAx<4>). In this mode, the EUSARTx uses standard Non-Return-to-Zero (NRZ) format (one Start bit, eight or nine data bits and one Stop bit). The most common data format is 8 bits. An on-chip, dedicated 8-bit/16-bit Baud Rate Generator can be used to derive standard baud rate frequencies from the oscillator.

The EUSARTx transmits and receives the LSb first. The EUSARTx's transmitter and receiver are functionally independent but use the same data format and baud rate. The Baud Rate Generator produces a clock, either x16 or x64 of the bit shift rate, depending on the BRGH and BRG16 bits (TXSTAx<2> and BAUDCONx<3>). Parity is not supported by the hardware but can be implemented in software and stored as the 9th data bit.

When operating in Asynchronous mode, the EUSARTx module consists of the following important elements:

- Baud Rate Generator
- Sampling Circuit
- Asynchronous Transmitter
- Asynchronous Receiver
- Auto-Wake-up on Sync Break Character
- 12-Bit Break Character Transmit
- Auto-Baud Rate Detection

21.2.1 EUSARTx ASYNCHRONOUS TRANSMITTER

The EUSARTx transmitter block diagram is shown in Figure 21-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The Shift register obtains its data from the Read/Write Transmit Buffer register, TXREGx. The TXREGx register is loaded with data in software. The TSR register is not loaded until the Stop bit has been transmitted from the previous load. As soon as the Stop bit is transmitted, the TSR is loaded with new data from the TXREGx register (if available).

Once the TXREGx register transfers the data to the TSR register (occurs in one TCY), the TXREGx register is empty and the TXxIF flag bit is set. This interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXxIE. TXxIF will be set regardless of the state of TXxIE; it cannot be cleared in software. TXxIF is also not cleared immediately upon loading TXREGx, but becomes valid in the second instruction cycle following the load instruction. Polling TXxIF immediately following a load of TXREGx will return invalid results.

While TXxIF indicates the status of the TXREGx register, another bit, TRMT (TXSTAx<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR register is empty. No interrupt logic is tied to this bit so the user has to poll this bit in order to determine if the TSR register is empty.

Note 1:	The TSR register is not mapped in data memory, so it is not available to the user.
2:	Flag bit, TXxIF, is set when enable bit, TXEN, is set.

To set up an Asynchronous Transmission:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, TXxIE.
4. If 9-bit transmission is desired, set transmit bit, TX9. Can be used as address/data bit.
5. Enable the transmission by setting bit, TXEN, which will also set bit, TXxIF.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Load data to the TXREGx register (starts transmission).
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 21-3: EUSARTx TRANSMIT BLOCK DIAGRAM

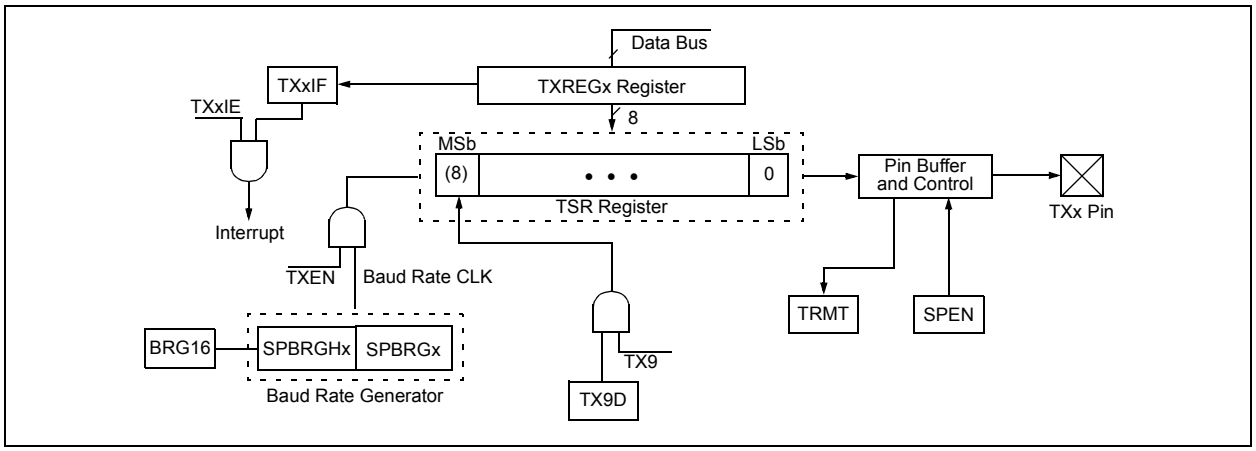


FIGURE 21-4: ASYNCHRONOUS TRANSMISSION

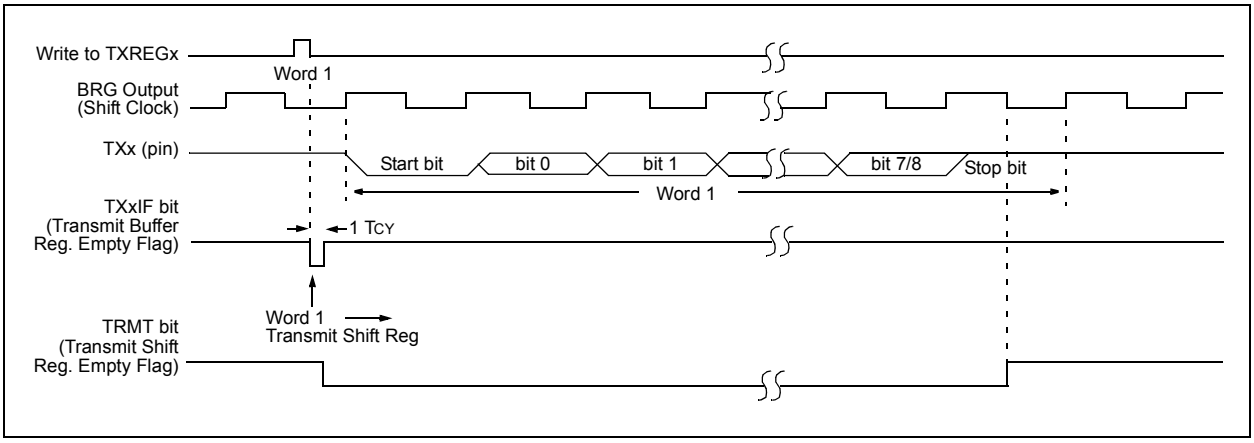
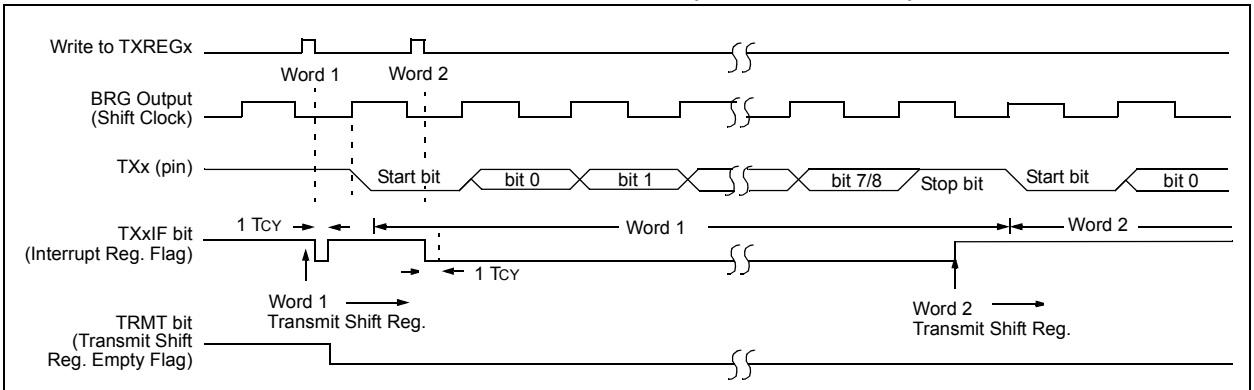


FIGURE 21-5: ASYNCHRONOUS TRANSMISSION (BACK-TO-BACK)



Note: This timing diagram shows two consecutive transmissions.

21.2.2 EUSARTx ASYNCHRONOUS RECEIVER

The receiver block diagram is shown in Figure 21-6. The data is received on the RXx pin and drives the data recovery block. The data recovery block is actually a high-speed shifter operating at x16 times the baud rate, whereas the main receive serial shifter operates at the bit rate or at FOSC. This mode would typically be used in RS-232 systems.

To set up an Asynchronous Reception:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing bit, SYNC, and setting bit, SPEN.
3. If interrupts are desired, set enable bit, RCxIE.
4. If 9-bit reception is desired, set bit, RX9.
5. Enable the reception by setting bit, CREN.
6. Flag bit, RCxIF, will be set when reception is complete and an interrupt will be generated if enable bit, RCxIE, was set.
7. Read the RCSTAx register to get the 9th bit (if enabled) and determine if any error occurred during reception.
8. Read the 8-bit received data by reading the RCREGx register.
9. If any error occurred, clear the error by clearing enable bit, CREN.
10. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

21.2.3 SETTING UP 9-BIT MODE WITH ADDRESS DETECT

This mode would typically be used in RS-485 systems. To set up an Asynchronous Reception with Address Detect Enable:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRGH and BRG16 bits, as required, to achieve the desired baud rate.
2. Enable the asynchronous serial port by clearing the SYNC bit and setting the SPEN bit.
3. If interrupts are required, set the RCEN bit and select the desired priority level with the RCxIP bit.
4. Set the RX9 bit to enable 9-bit reception.
5. Set the ADDEN bit to enable address detect.
6. Enable reception by setting the CREN bit.
7. The RCxIF bit will be set when reception is complete. The interrupt will be Acknowledged if the RCxIE and GIE bits are set.
8. Read the RCSTAx register to determine if any error occurred during reception, as well as read bit 9 of data (if applicable).
9. Read RCREGx to determine if the device is being addressed.
10. If any error occurred, clear the CREN bit.
11. If the device has been addressed, clear the ADDEN bit to allow all received data into the receive buffer and interrupt the CPU.

FIGURE 21-6: EUSARTx RECEIVE BLOCK DIAGRAM

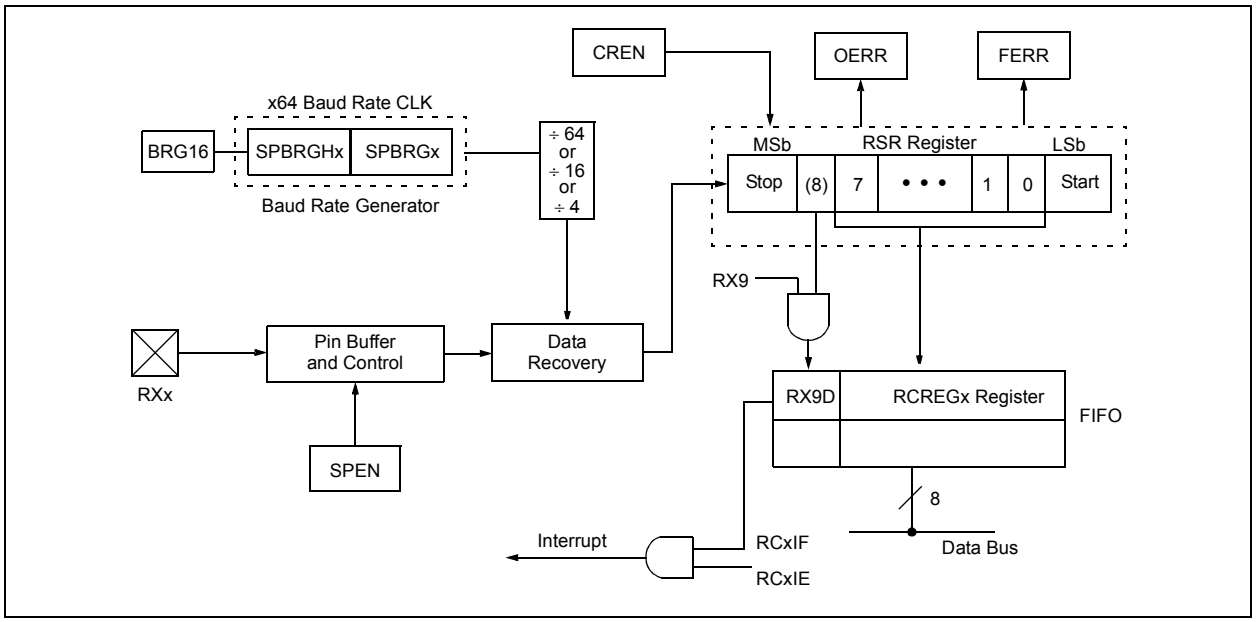
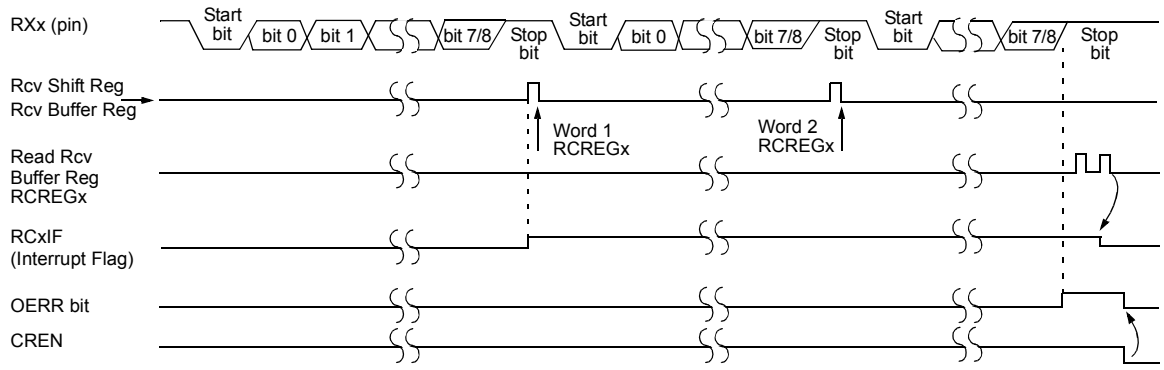


FIGURE 21-7: ASYNCHRONOUS RECEPTION



Note: This timing diagram shows three words appearing on the RXx input. The RCREGx (Receive Buffer) is read after the third word, causing the OERR (Overrun) bit to be set.

21.2.4 AUTO-WAKE-UP ON SYNC BREAK CHARACTER

During Sleep mode, all clocks to the EUSARTx are suspended. Because of this, the Baud Rate Generator is inactive and a proper byte reception cannot be performed. The auto-wake-up feature allows the controller to wake-up due to activity on the RXx/DTx line while the EUSARTx is operating in Asynchronous mode.

The auto-wake-up feature is enabled by setting the WUE bit (BAUDCONx<1>). Once set, the typical receive sequence on RXx/DTx is disabled and the EUSARTx remains in an Idle state, monitoring for a wake-up event independent of the CPU mode. A wake-up event consists of a high-to-low transition on the RXx/DTx line. (This coincides with the start of a Sync Break or a Wake-up Signal character for the LIN/J2602 protocol.)

Following a wake-up event, the module generates an RCxIF interrupt. The interrupt is generated synchronously to the Q clocks in normal operating modes (Figure 21-8) and asynchronously if the device is in Sleep mode (Figure 21-9). The interrupt condition is cleared by reading the RCREGx register.

The WUE bit is automatically cleared once a low-to-high transition is observed on the RXx line following the wake-up event. At this point, the EUSARTx module is in Idle mode and returns to normal operation. This signals to the user that the Sync Break event is over.

21.2.4.1 Special Considerations Using Auto-Wake-up

Since auto-wake-up functions by sensing rising edge transitions on RXx/DTx, information with any state changes before the Stop bit may signal a false End-of-Character (EOC) and cause data or framing errors. To work properly, therefore, the initial character in the transmission must be all '0's. This can be 00h (8 bits) for standard RS-232 devices or 000h (12 bits) for the LIN/J2602 bus.

Oscillator start-up time must also be considered, especially in applications using oscillators with longer start-up intervals (i.e., HS or HSPLL mode). The Sync Break (or Wake-up Signal) character must be of sufficient length and be followed by a sufficient interval to allow enough time for the selected oscillator to start and provide proper initialization of the EUSARTx.

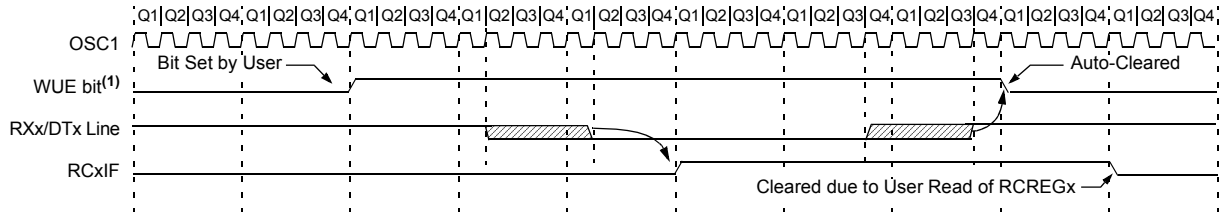
21.2.4.2 Special Considerations Using the WUE Bit

The timing of WUE and RCxIF events may cause some confusion when it comes to determining the validity of received data. As noted, setting the WUE bit places the EUSARTx in an Idle mode. The wake-up event causes a receive interrupt by setting the RCxIF bit. The WUE bit is cleared after this when a rising edge is seen on RXx/DTx. The interrupt condition is then cleared by reading the RCREGx register. Ordinarily, the data in RCREGx will be dummy data and should be discarded.

The fact that the WUE bit has been cleared (or is still set), and the RCxIF flag is set, should not be used as an indicator of the integrity of the data in RCREGx. Users should consider implementing a parallel method in firmware to verify received data integrity.

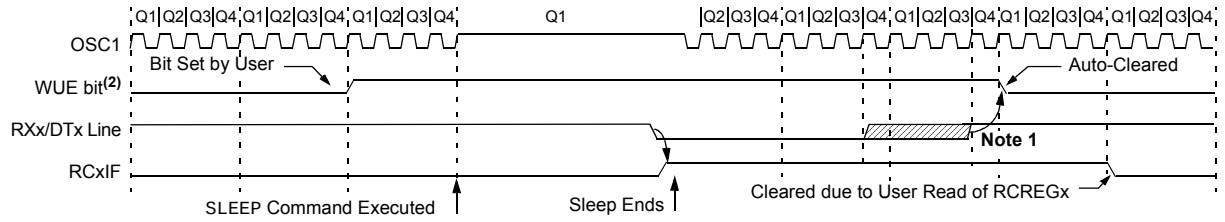
To assure that no actual data is lost, check the RCIDL bit to verify that a receive operation is not in process. If a receive operation is not occurring, the WUE bit may then be set just prior to entering the Sleep mode.

FIGURE 21-8: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING NORMAL OPERATION



Note 1: The EUSARTx remains in Idle while the WUE bit is set.

FIGURE 21-9: AUTO-WAKE-UP BIT (WUE) TIMINGS DURING SLEEP



Note 1: If the wake-up event requires long oscillator warm-up time, the auto-clear of the WUE bit can occur before the oscillator is ready. This sequence should not depend on the presence of Q clocks.

2: The EUSARTx remains in Idle while the WUE bit is set.

21.2.5 BREAK CHARACTER SEQUENCE

The EUSARTx module has the capability of sending the special Break character sequences that are required by the LIN/J2602 bus standard. The Break character transmit consists of a Start bit, followed by twelve '0' bits and a Stop bit. The Frame Break character is sent whenever the SENDB and TXEN bits (TXSTAx<3> and TXSTAx<5>, respectively) are set while the Transmit Shift Register is loaded with data. Note that the value of data written to TXREGx will be ignored and all '0's will be transmitted.

The SENDB bit is automatically reset by hardware after the corresponding Stop bit is sent. This allows the user to preload the transmit FIFO with the next transmit byte following the Break character (typically, the Sync character in the LIN/J2602 specification).

Note that the data value written to the TXREGx for the Break character is ignored. The write simply serves the purpose of initiating the proper sequence.

The TRMT bit indicates when the transmit operation is active or Idle, just as it does during normal transmission. See Figure 21-10 for the timing of the Break character sequence.

21.2.5.1 Break and Sync Transmit Sequence

The following sequence will send a message frame header made up of a Break, followed by an Auto-Baud Sync byte. This sequence is typical of a LIN/J2602 bus master.

1. Configure the EUSARTx for the desired mode.
2. Set the TXEN and SENDB bits to set up the Break character.
3. Load the TXREGx with a dummy character to initiate transmission (the value is ignored).
4. Write '55h' to TXREGx to load the Sync character into the transmit FIFO buffer.
5. After the Break has been sent, the SENDB bit is reset by hardware. The Sync character now transmits in the preconfigured mode.

When the TXREGx becomes empty, as indicated by the TXxIF, the next data byte can be written to TXREGx.

21.2.6 RECEIVING A BREAK CHARACTER

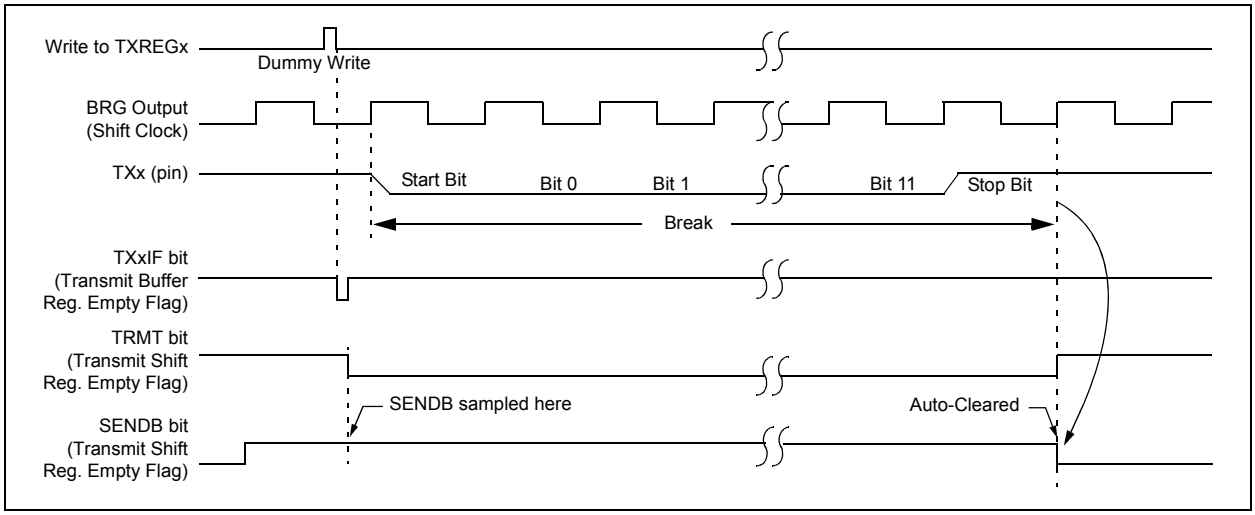
The Enhanced USART module can receive a Break character in two ways.

The first method forces configuration of the baud rate at a frequency of 9/13 the typical speed. This allows for the Stop bit transition to be at the correct sampling location (13 bits for Break versus Start bit and 8 data bits for typical data).

The second method uses the auto-wake-up feature described in **Section 21.2.4 "Auto-Wake-up on Sync Break Character"**. By enabling this feature, the EUSARTx will sample the next two transitions on RXx/DTx, cause an RCxIF interrupt and receive the next data byte followed by another interrupt.

Note that following a Break character, the user will typically want to enable the Auto-Baud Rate Detect feature. For both methods, the user can set the ABDEN bit once the TXxIF interrupt is observed.

FIGURE 21-10: SEND BREAK CHARACTER SEQUENCE



21.3 EUSARTx Synchronous Master Mode

The Synchronous Master mode is entered by setting the CSRC bit (TXSTAx<7>). In this mode, the data is transmitted in a half-duplex manner (i.e., transmission and reception do not occur at the same time). When transmitting data, the reception is inhibited and vice versa. Synchronous mode is entered by setting bit, SYNC (TXSTAx<4>). In addition, enable bit, SPEN (RCSTAx<7>), is set in order to configure the TXx and RXx pins to CKx (clock) and DTx (data) lines, respectively.

The Master mode indicates that the processor transmits the master clock on the CKx line. Clock polarity is selected with the TXCKP bit (BAUDCONx<4>). Setting TXCKP sets the Idle state on CKx as high, while clearing the bit sets the Idle state as low. This option is provided to support Microwire devices with this module.

21.3.1 EUSARTx SYNCHRONOUS MASTER TRANSMISSION

The EUSARTx transmitter block diagram is shown in Figure 21-3. The heart of the transmitter is the Transmit (Serial) Shift Register (TSR). The shift register obtains its data from the Read/Write Transmit Buffer register, TXREGx. The TXREGx register is loaded with data in software. The TSR register is not loaded until the last bit has been transmitted from the previous load. As soon as the last bit is transmitted, the TSR is loaded with new data from the TXREGx (if available).

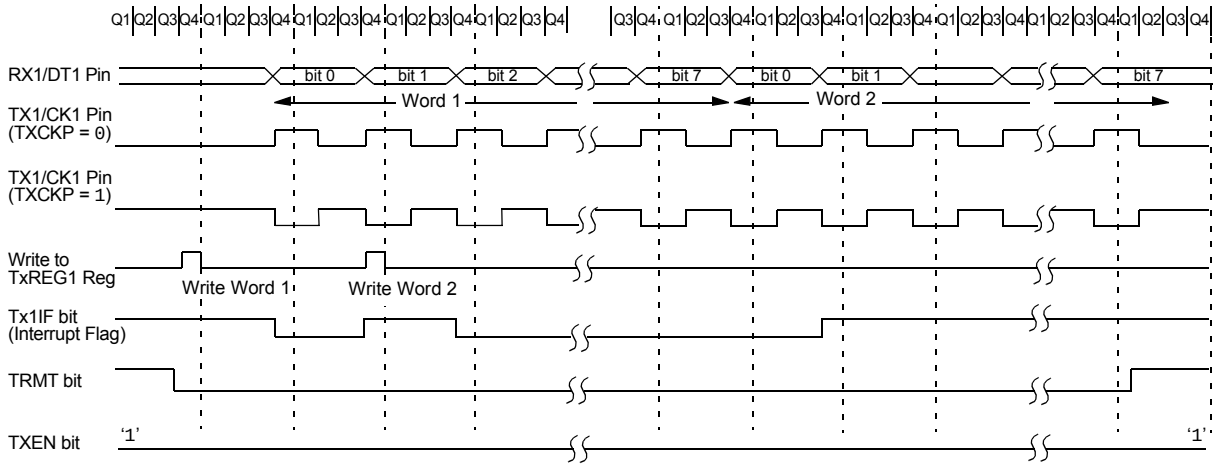
Once the TXREGx register transfers the data to the TSR register (occurs in one TCY), the TXREGx is empty and the TXxIF flag bit is set. The interrupt can be enabled or disabled by setting or clearing the interrupt enable bit, TXxIE. TXxIF is set regardless of the state of enable bit, TXxIE; it cannot be cleared in software. It will reset only when new data is loaded into the TXREGx register.

While flag bit, TXxIF, indicates the status of the TXREGx register, another bit, TRMT (TXSTAx<1>), shows the status of the TSR register. TRMT is a read-only bit which is set when the TSR is empty. No interrupt logic is tied to this bit, so the user must poll this bit in order to determine if the TSR register is empty. The TSR is not mapped in data memory so it is not available to the user.

To set up a Synchronous Master Transmission:

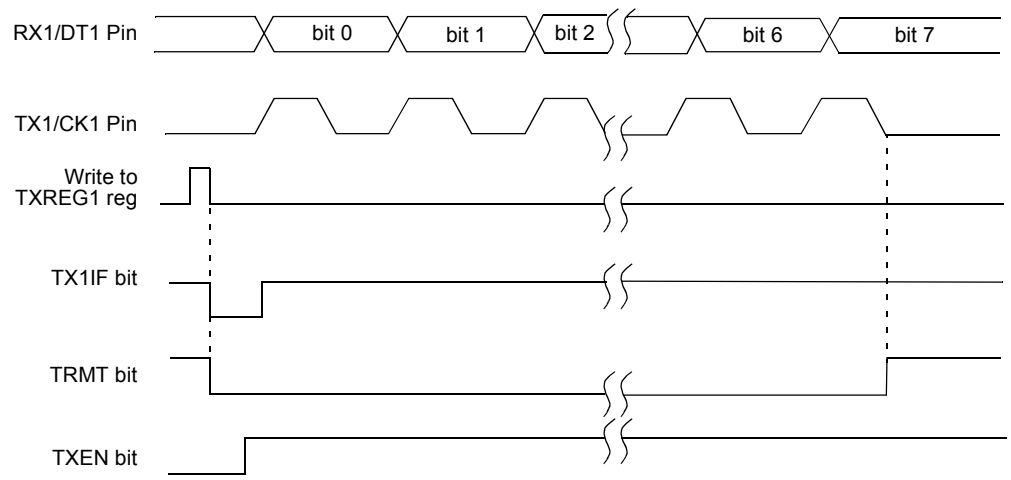
1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the Master Synchronous Serial Port by setting bits, SYNC, SPEN and CSRC.
3. If interrupts are desired, set enable bit, TXxIE.
4. If 9-bit transmission is desired, set bit, TX9.
5. Enable the transmission by setting bit, TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Start transmission by loading data to the TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 21-11: SYNCHRONOUS TRANSMISSION



Note: Sync Master mode, SPBRGx = 0, continuous transmission of two 8-bit words. This example is equally applicable to EUSART2 (TX2/CK2 and RX2/DT2).

FIGURE 21-12: SYNCHRONOUS TRANSMISSION (THROUGH TXEN)



Note: This example is equally applicable to EUSART2 (TX2/CK2 and RX2/DT2).

21.3.2 EUSARTx SYNCHRONOUS MASTER RECEPTION

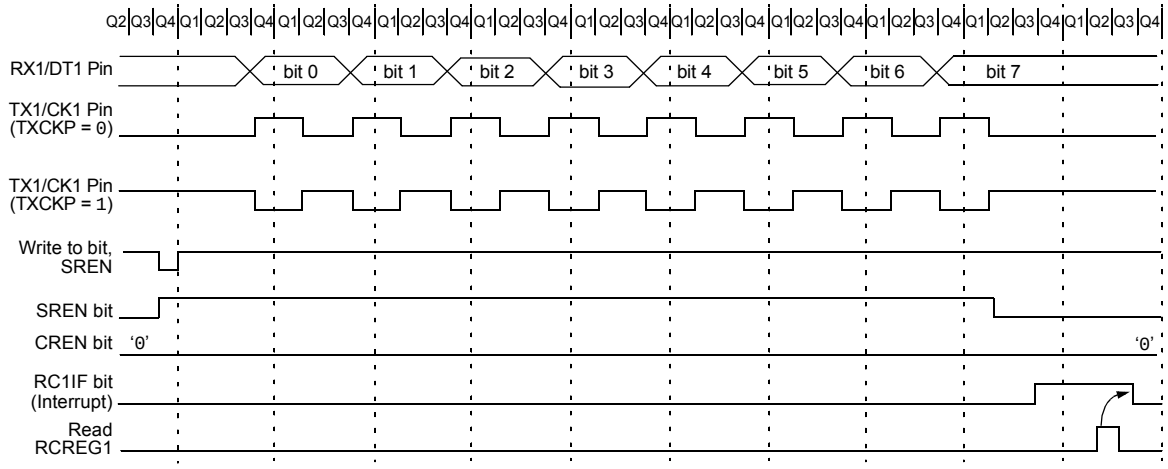
Once Synchronous mode is selected, reception is enabled by setting either the Single Receive Enable bit, SREN (RCSTAx<5>) or the Continuous Receive Enable bit, CREN (RCSTAx<4>). Data is sampled on the RXx pin on the falling edge of the clock.

If enable bit, SREN, is set, only a single word is received. If enable bit, CREN, is set, the reception is continuous until CREN is cleared. If both bits are set, then CREN takes precedence.

To set up a Synchronous Master Reception:

1. Initialize the SPBRGHx:SPBRGx registers for the appropriate baud rate. Set or clear the BRG16 bit, as required, to achieve the desired baud rate.
2. Enable the Master Synchronous Serial Port by setting bits, SYNC, SPEN and CSRC.
3. Ensure bits, CREN and SREN, are clear.
4. If interrupts are desired, set enable bit, RCxIE.
5. If 9-bit reception is desired, set bit, RX9.
6. If a single reception is required, set bit, SREN. For continuous reception, set bit, CREN.
7. Interrupt flag bit, RCxIF, will be set when reception is complete and an interrupt will be generated if the enable bit, RCxIE, was set.
8. Read the RCSTAx register to get the 9th bit (if enabled) and determine if any error occurred during reception.
9. Read the 8-bit received data by reading the RCREGx register.
10. If any error occurred, clear the error by clearing bit CREN.
11. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

FIGURE 21-13: SYNCHRONOUS RECEPTION (MASTER MODE, SREN)



Note: Timing diagram demonstrates Sync Master mode with bit, SREN = 1, and bit, BRGH = 0. This example is equally applicable to EUSART2 (TX2/CK2 and RX2/DT2).

21.4 EUSARTx Synchronous Slave Mode

Synchronous Slave mode is entered by clearing bit, CSRC (TXSTAx<7>). This mode differs from the Synchronous Master mode in that the shift clock is supplied externally at the CKx pin (instead of being supplied internally in Master mode). This allows the device to transfer or receive data while in any low-power mode.

21.4.1 EUSARTx SYNCHRONOUS SLAVE TRANSMISSION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep mode.

If two words are written to the TXREGx and then the SLEEP instruction is executed, the following will occur:

a) The first word will immediately transfer to the TSR register and transmit.
b) The second word will remain in the TXREGx register.
c) Flag bit, TXxIF, will not be set.
d) When the first word has been shifted out of TSR, the TXREGx register will transfer the second word to the TSR and flag bit, TXxIF, will now be set.
e) If enable bit, TXxIE, is set, the interrupt will wake the chip from Sleep. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Transmission:

1. Enable the synchronous slave serial port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. Clear bits, CREN and SREN.
3. If interrupts are desired, set enable bit, TXxIE.
4. If 9-bit transmission is desired, set bit, TX9.
5. Enable the transmission by setting enable bit, TXEN.
6. If 9-bit transmission is selected, the ninth bit should be loaded in bit, TX9D.
7. Start transmission by loading data to the TXREGx register.
8. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

21.4.2 EUSARTx SYNCHRONOUS SLAVE RECEPTION

The operation of the Synchronous Master and Slave modes is identical, except in the case of Sleep, or any Idle mode and bit, SREN, which is a “don’t care” in Slave mode.

If receive is enabled by setting the CREN bit prior to entering Sleep or any Idle mode, then a word may be received while in this Low-Power mode. Once the word is received, the RSR register will transfer the data to the RCREGx register. If the RCxIE enable bit is set, the interrupt generated will wake the chip from the Low-Power mode. If the global interrupt is enabled, the program will branch to the interrupt vector.

To set up a Synchronous Slave Reception:

1. Enable the Master Synchronous Serial Port by setting bits, SYNC and SPEN, and clearing bit, CSRC.
2. If interrupts are desired, set enable bit, RCxIE.
3. If 9-bit reception is desired, set bit, RX9.
4. To enable reception, set enable bit, CREN.
5. Flag bit, RCxIF, will be set when reception is complete. An interrupt will be generated if enable bit, RCxIE, was set.
6. Read the RCSTAx register to get the 9th bit (if enabled) and determine if any error occurred during reception.
7. Read the 8-bit received data by reading the RCREGx register.
8. If any error occurred, clear the error by clearing bit, CREN.
9. If using interrupts, ensure that the GIE and PEIE bits in the INTCON register (INTCON<7:6>) are set.

21.5 Infrared Support

This module provides support for two types of infrared USART port implementations:

- IrDA clock output to support an external IrDA encoder/decoder device
- Full implementation of the IrDa encoder and decoder as part of the USART logic

Since the 16x clock is required to perform the IrDA encoding, both by this module and the external transmitter, this feature only works in the 16x Baud Rate mode and is not available in the 4x mode.

21.5.1 EXTERNAL IrDA SUPPORT – IRDA CLOCK OUTPUT

The 16x Baud Clock is provided on the BCLK (Baud Clock) pin if the EUSARTx is enabled (SPEN = 1); it is configured for Asynchronous mode (SYNC = 0) when Clock Source Select is active (CSRC = 1). Note that the BCLK can be active in regular or IrDA mode (IREN bit is ignored).

21.5.1.1 BCLK Output

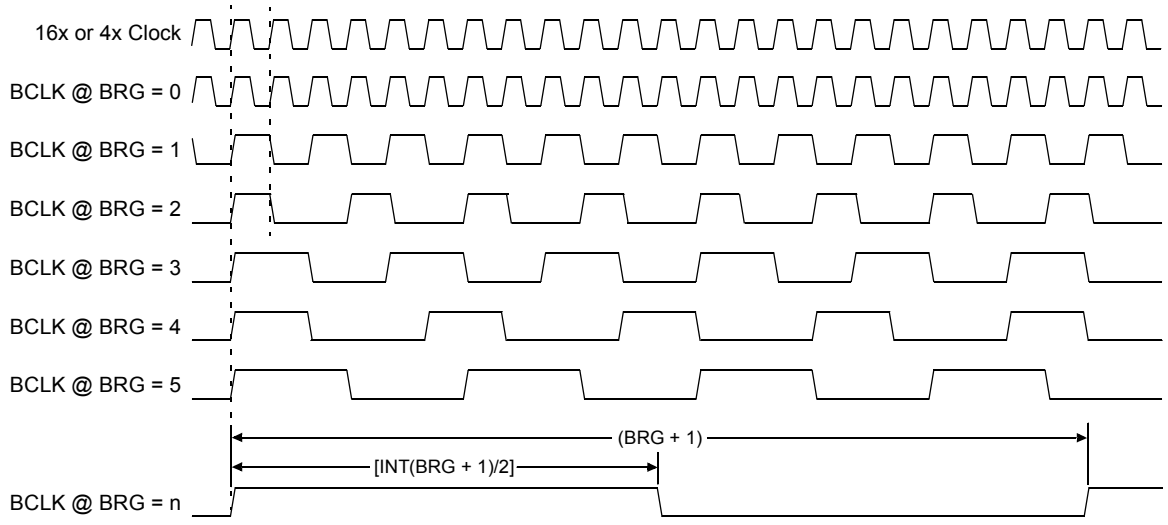
The timing of the Baud Clock (BCLK) output is independent of the 16x or 4x Baud Rate mode, resulting in the same output for a particular BRG value (since the 4x mode is four times faster, but has four times less pulses per period).

When the BCLK pin mode is active, the RXx Baud Rate Generator will be turned on, independent of a TXx or RXx operation. This will cause the RXx stream to synchronize to the already running RXx Baud Clock. This is acceptable only when BCLK is enabled for use.

The BCLK output goes inactive and stays low during Sleep mode.

The BCLK pin is taken over by the EUSARTx module and forced as an output, irrespective of port latch and TRIS latch bits. BCLK remains an output as long as USART is kept enabled in this mode.

FIGURE 21-14: BCLK OUTPUT vs. BRG PROGRAMMING



Note: The BCLK has 50% duty cycle only for odd BRG values. This is due to having all BCLK edges synchronous to the rising edge of the 16x/4x clock.

21.5.2 BUILT-IN IrDA ENCODER AND DECODER

The built-in IrDA encoder and decoder functionality is enabled using the IREN bit in the BAUDCONx register while the module is in Asynchronous mode (SYNC = 0). When enabled (IREN = 1), the Receive pin (RXx) acts as the input from the infrared receiver. The Transmit pin (TXx) acts as the output to the infrared transmitter. The 16x clock must be available for this feature to work properly.

The IrDA feature cannot be enabled for Synchronous modes (SYNC = 1).

21.5.2.1 IrDA Encoder Function

The encoder works by taking the serial data from the USART and replacing it as follows:

- Transmit bit data of '1' gets encoded as '0' for the entire 16 periods of the 16x Baud Clock
- Transmit bit data of '0' gets encoded as '0' for the first 7 periods of the 16x Baud Clock, then as '1' for the next 3 periods, and as '0' for the remaining 6 periods

See Figure 21-15 and Figure 21-17 for details.

21.5.2.2 IrDA Transmit Polarity

The IrDA transmit polarity is selected using the TXCKP bit. This bit only affects the transmit encoder and does not affect the receiver.

When TXCKP = 0, the Idle state of the TXx line is '0' (see Figure 21-15). When TXCKP = 1, the Idle state of the TXx line is '1' (see Figure 21-16).

FIGURE 21-15: IrDA® ENCODING SCHEME

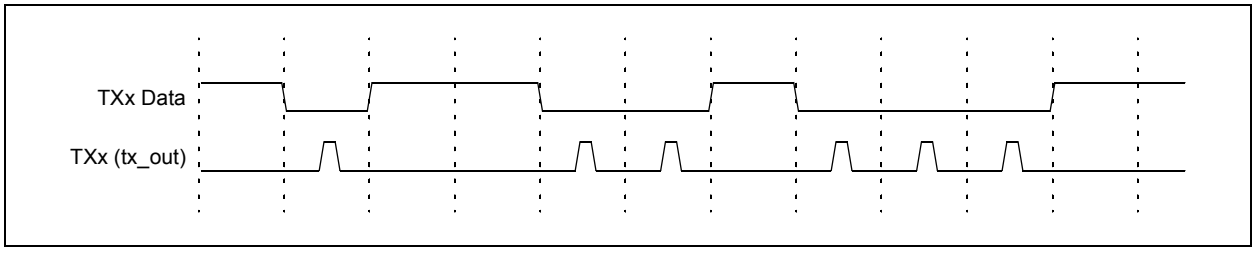


FIGURE 21-16: INVERTED IrDA® ENCODING (TXCKP = 1)

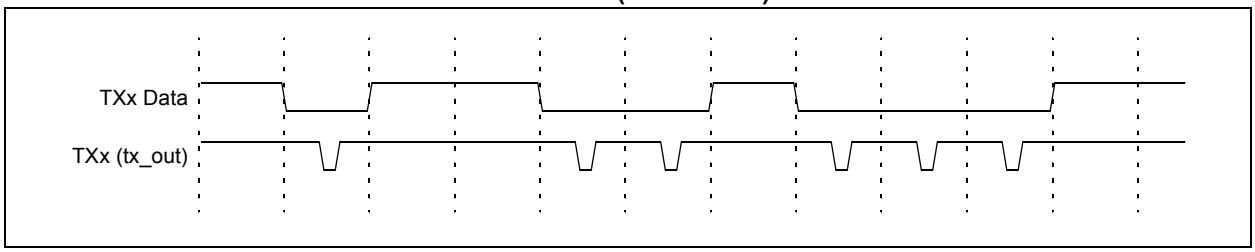
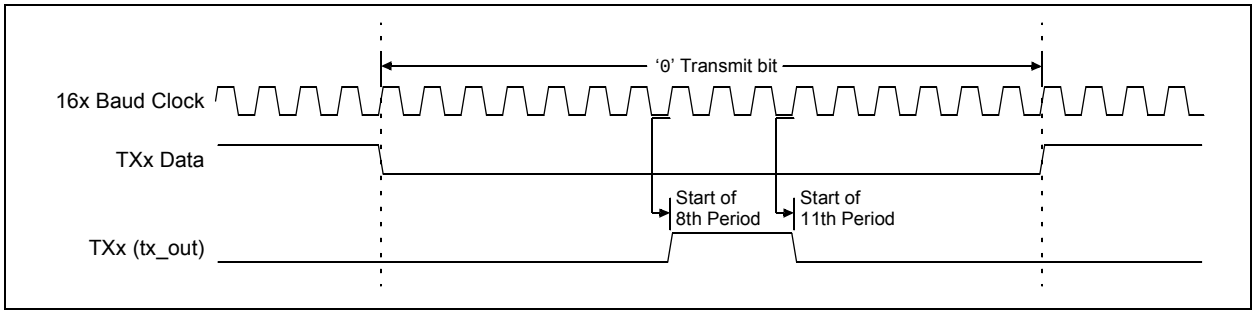


FIGURE 21-17: '0' BIT DATA IrDA® ENCODING SCHEME



21.5.2.3 IrDA Decoder Function

The decoder works by taking the serial data from the RXx pin and replacing it with the decoded data stream. The stream is decoded based on falling edge detection of the RXx input.

Each falling edge of RXx causes the decoded data to be driven low for 16 periods of the 16x Baud Clock. If another falling edge has been detected by the time the 16 periods expire, the decoded data remains low for another 16 periods. If no falling edge was detected, the decoded data is driven high.

Note that the data stream into the device is shifted anywhere from 7 to 8 periods of the 16x Baud Clock from the actual message source. The one clock uncertainty is due to the clock edge resolution. See Figure 21-18 for details.

21.5.2.4 IrDA Receive Polarity

The IrDA receive polarity is selected using the RXDTP bit. This bit only affects the receive encoder and does not affect the transmitter.

When RXDTP = 0, the Idle state of the RXx line is '1' (see Figure 21-18). When RXDTP = 1, the Idle state of the RXx line is '0' (see Figure 21-19).

21.5.2.5 Clock Jitter

Due to jitter or slight frequency differences between devices, it is possible for the next falling bit edge to be missed for one of the 16x periods. In that case, one clock-wide pulse appears on the decoded data stream. Since the EUSARTx performs a majority detect around the bit center, this does not cause erroneous data. See Figure 21-20 for details.

FIGURE 21-18: MACRO VIEW OF IrDA® DECODING SCHEME (RXDTP = 0)

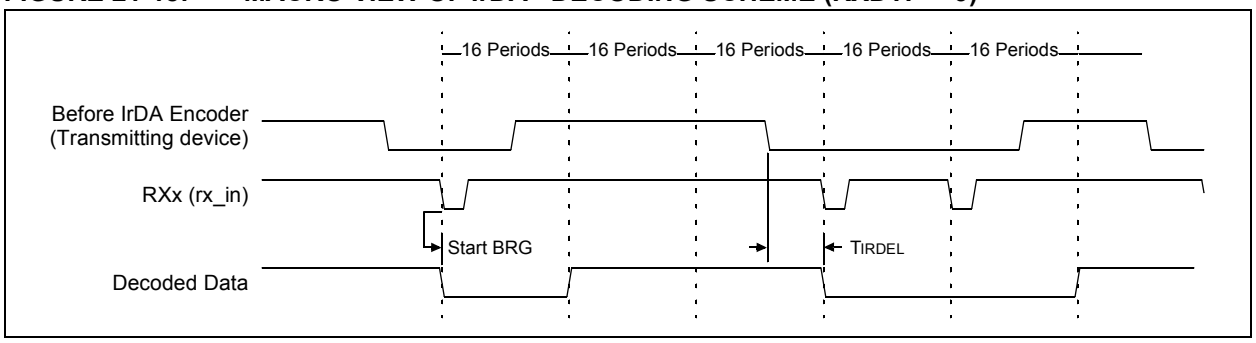


FIGURE 21-19: INVERTED POLARITY DECODING RESULTS (RXDTP = 1)

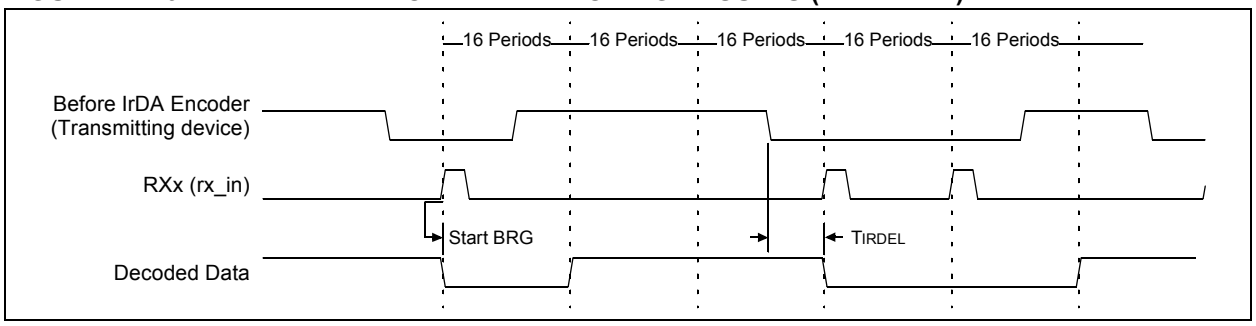
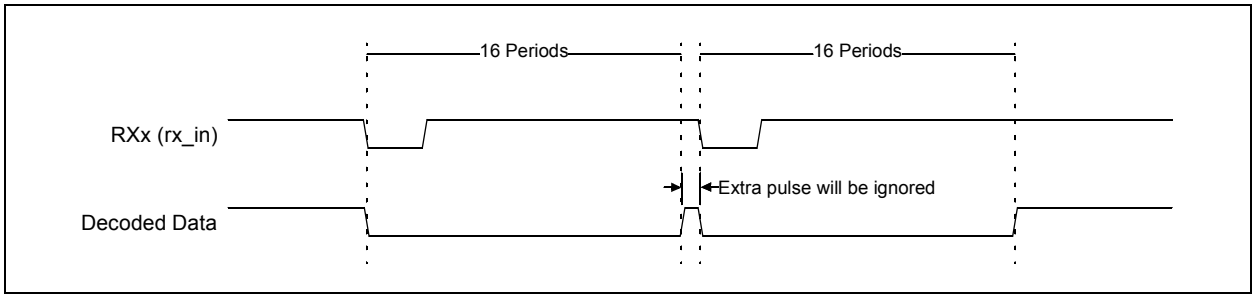


FIGURE 21-20: CLOCK JITTER CAUSING A PULSE BETWEEN CONSECUTIVE ZEROS



22.0 12-BIT A/D CONVERTER WITH THRESHOLD SCAN

The 12-bit A/D Converter has the following key features:

- Successive Approximation Register (SAR) Conversion
- Conversion Speeds of up to 200 ksps at 12 bits and 500 ksps at 10 bits
- Up to 32 Analog Input Channels (internal and external)
- Selectable 10-Bit or 12-Bit (default) Conversion
- Resolution
- Multiple Internal Reference Input Channels
- External Voltage Reference Input Pins
- Unipolar Differential Sample-and-Hold (S/H) Amplifier
- Automated Threshold Scan and Compare Operation to Pre-Evaluate up to 26 Conversion Results
- Selectable Conversion Trigger Source
- Fixed Length (one word per channel), Configurable Conversion Result Buffer
- Four Options for Results Alignment
- Configurable Interrupt Generation
- Operation During CPU Sleep and Idle modes

The 12-bit A/D Converter module is an enhanced version of the 10-bit module offered in some PIC18 devices. Both modules are Successive Approximation Register (SAR) Converters at their cores, surrounded by a range of hardware features for flexible configuration. This version of the module extends functionality by providing 12-bit resolution, a wider range of automatic sampling options, tighter integration with other analog modules, such as the CTMU, and a configurable results buffer. This module also includes a unique Threshold Detect feature that allows the module itself to make simple decisions based on the conversion results.

As before, an internal Sample-and-Hold (S/H) amplifier acquires a sample of an input signal, then holds that value constant during the conversion process. A combination of input multiplexers selects the signal to be converted from up to 32 analog inputs, both external (analog input pins) and internal (e.g., on-chip voltage references and other analog modules). The whole multiplexer path includes provisions for differential analog input, although, with a limited number of negative input pins. The sampled voltage is held and converted to a digital value, which strictly speaking, represents the ratio of that input voltage to a reference voltage. Configuration choices allow connection of an external reference or use of the device power and ground (AV_{DD} and AV_{SS}). Reference and input signal pins are assigned differently depending on the particular device.

An array of timing and control selections allow the user to create flexible scanning sequences. Conversions can be started individually by program control, continuously free-running or triggered by selected hardware events. A single channel may be repeatedly converted. Alternate conversions may be performed on two channels, or any or all of the channels may be sequentially scanned and converted according to a user-defined bit map. The resulting conversion output is a 12-bit digital number, which can be signed or unsigned, left or right justified. (In some devices, a user-selectable resolution of ten bits is available; in other devices, 10-bit resolution is the only option available.)

Conversions are automatically stored in a dedicated buffer, allowing for multiple successive readings to be taken before software service is needed. The buffer can be configured to function as a FIFO buffer or as a channel indexed buffer. In FIFO mode, the buffer can be split into two equal sections for simultaneous conversion and read operations. In Indexed mode, the buffer can use the Threshold Scan feature to determine if a conversion meets specific, user-defined criteria, storing or discarding the converted value as appropriate, and then set semaphore flags to indicate the event. This allows conversions to occur in low-power modes when the CPU is inactive, waking the device only when specific conditions have occurred.

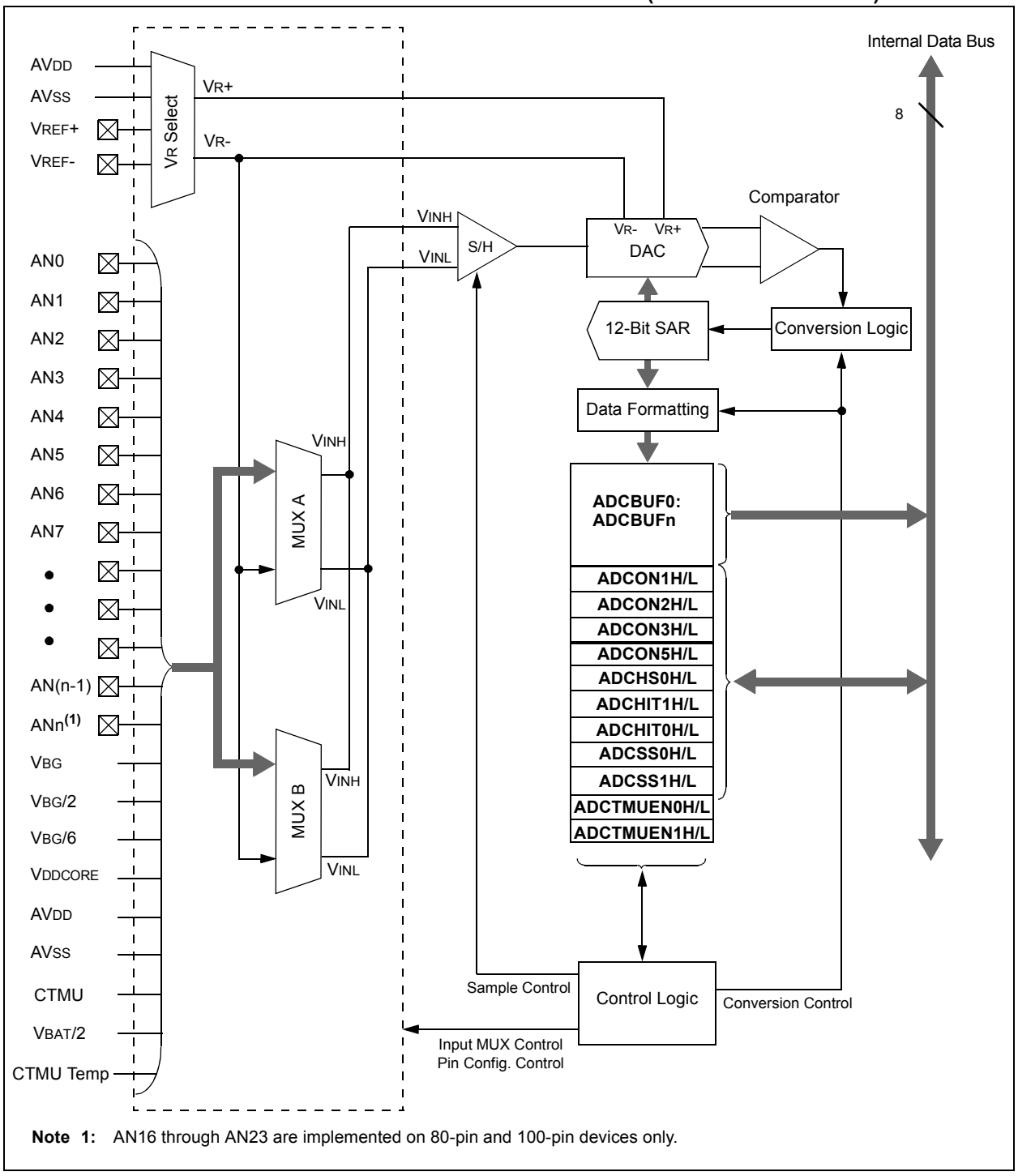
The module sets its interrupt flag after a selectable number of conversions, when the buffer can be read, or after a successful Threshold Detect comparison. After the interrupt, the sequence restarts at the beginning of the buffer. When the interrupt flag is set, according to the earlier selection, scan selections and the Output Buffer Pointer return to their starting positions.

During Sleep or Idle mode, the A/D can wake-up at preconfigured intervals while the device maintains a Low-Power mode. If threshold conditions have not been met on any of the conversions, the module will return to a Low-Power mode.

The A/D module provides configuration to directly interact with the CTMU on specific input channels. This allows the CTMU to automatically turn on only when requested directly by the A/D, even though the rest of the device stays in Sleep mode.

A simplified block diagram for the module is shown in Figure 22-1.

FIGURE 22-1: 12-BIT A/D CONVERTER BLOCK DIAGRAM (PIC18F97J94 FAMILY)



Note 1: AN16 through AN23 are implemented on 80-pin and 100-pin devices only.

22.1 Registers

The 12-bit A/D converter module uses up to 75 registers for its operation. All registers are mapped in the data memory space.

22.1.1 CONTROL REGISTERS

Depending on the specific device, the module has up to twelve control and STATUS registers:

- ADCON1H/L: A/D Control Registers
- ADCON2H/L: A/D Control Registers
- ADCON3H/L: A/D Control Registers
- ADCON5H/L: A/D Control Registers
- ADCHS0H/L: A/D Input Channel Select Registers
- ADCHITH1H/L and ADCHITH0H/L: A/D Scan Compare Hit Registers
- ADCSS1H/L and ADCSS0H/L: A/D Input Scan Select Registers
- ADCTMUEN1H/L and ADCTMUEN0H/L: CTMU Enable Register

The ADCON1H/L, ADCON2H/L and ADCON3H/L registers control the overall operation of the A/D module. This includes enabling the module, configuring the conversion clock and voltage reference sources, selecting the sampling and conversion triggers, and manually controlling the sample/convert sequences. The ADCON5H/L registers specifically controls features of Threshold Detect operation, including its functioning in power-saving modes.

The ADCHS0H/L registers selects the input channels to be connected to the S/H amplifier. It also allows the choice of input multiplexers and the selection of a reference source for differential sampling.

The ADCHITH1H/L and ADCHITH0H/L registers are semaphore registers used with Threshold Detect operations. The status of individual bits, or bit pairs in some cases, indicate if a match condition has occurred. Their use is described in more detail in Section 22.7 “Threshold Detect Operation”. ADCHITH0H/L is always implemented, whereas ADCHITH1H/L may not be implemented in devices with 16 channels or less. The ADCSS0H/L/L registers select the channels to be included for sequential scanning. The ADCTMUEN1H/L/L registers select the channel(s) to be used by the CTMU during conversions. Selecting a particular channel allows the A/D Converter to control the CTMU (particularly, its current source) and read its data through that channel. ADCTMUEN0H/L is always implemented, whereas ADCTMUEN1H/L may not be implemented in devices with 16 channels or less.

22.1.2 A/D RESULT BUFFERS

The module incorporates a multi-word, dual port RAM, called ADCBUF. The buffer is composed of at least the same number of word locations as there are external analog channels for a particular device, with a maximum number of 26. The number of buffer addresses is always even. Each of the locations is mapped into the data memory space and is separately addressable.The buffer locations are referred to as ADCBUF0H/L through ADCBUFnH/L (up to 26).

The A/D result buffers are both readable and writable. When the module is active (ADCON1H<7> = 1), the buffers are read-only, and store the results of A/D conversions. When the module is inactive (ADCON1H<7> = 0), the buffers are both readable and writable. In this state, writing to a buffer location programs a conversion threshold for Threshold Detect operations, as described in Section 22.7.2, Setting Comparison Thresholds.

REGISTER 22-1: ANCON1: ANALOG SELECT CONTROL REGISTER 1 (FOR ANSEL7-ANSEL0)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSEL7	ANSEL6	ANSEL5	ANSEL4	ANSEL3	ANSEL2	ANSEL1	ANSEL0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ANSEL7:** Pin RG0 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 6 **ANSEL6:** Pin RF2 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 5 **ANSEL5:** Pin RA5 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 4 **ANSEL4:** Pin RA4 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 3 **ANSEL3:** Pin RA3 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 2 **ANSEL2:** Pin RA2 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 1 **ANSEL1:** Pin RA1 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 0 **ANSEL0:** Pin RA0 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

REGISTER 22-2: ANCON2: ANALOG SELECT CONTROL REGISTER 2 (FOR ANSEL15-ANSEL8)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSEL15	ANSEL14	ANSEL13	ANSEL12	ANSEL11	ANSEL10	ANSEL9	ANSEL8
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ANSEL15:** Pin RG4 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 6 **ANSEL14:** Pin RG3 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 5 **ANSEL13:** Pin RG2 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 4 **ANSEL12:** Pin RG1 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 3 **ANSEL11:** Pin RF7 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 2 **ANSEL10:** Pin RF6 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 1 **ANSEL9:** Pin RF5 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

bit 0 **ANSEL8:** Pin RC2 Analog Enable bit
1 = Pin configured as an analog channel – digital input is disabled and reads '0'
0 = Pin configured as a digital port

REGISTER 22-3: ANCON3: ANALOG SELECT CONTROL REGISTER 3 (FOR ANSEL23-ANSEL16)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSEL23	ANSEL22	ANSEL21	ANSEL20	ANSEL19	ANSEL18	ANSEL17	ANSEL16
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ANSEL23:** Pin RH7 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

bit 6 **ANSEL22:** Pin RH6 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

bit 5 **ANSEL21:** Pin RH5 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

bit 4 **ANSEL20:** Pin RH4 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

bit 3 **ANSEL19:** Pin RH3 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

bit 2 **ANSEL18:** Pin RH2 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

bit 1 **ANSEL17:** Pin RH1 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

bit 0 **ANSEL16:** Pin RH0 Analog Enable
1 = Pin configured as an analog channel – digital input disabled and reads '0'
0 = Pin configured as a digital port

REGISTER 22-4: ADCON1H: A/D CONTROL REGISTER 1 HIGH

R/W-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
ADON	—	—	—	—	MODE12	FORM1	FORM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **ADON:** A/D Operating Mode bit
1 = A/D Converter module is operating
0 = A/D Converter is off

bit 6-3 **Unimplemented:** Read as ‘0’

bit 2 **MODE12:** 12-Bit Operation Mode bit
1 = 12-bit A/D operation
0 = 10-bit A/D operation

bit 1-0 **FORM<1:0>:** Data Output Format bits (see following formats)
11 = Fractional result, signed, left-justified
10 = Absolute fractional result, unsigned, left-justified
01 = Decimal result, signed, right-justified
00 = Absolute decimal result, unsigned, right-justified

REGISTER 22-5: ADCON1L: A/D CONTROL REGISTER 1 LOW

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0, HSC	R/C-0, HSC
SSRC3	SSRC2	SSRC1	SSRC0	—	ASAM	SAMP	DONE
bit 7							bit 0

Legend:	C = Clearable bit	U = Unimplemented bit, read as '0'	
R = Readable bit	W = Writable bit	HSC = Hardware Settable/Clearable bit	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **SSRC<3:0>:** Sample Clock Source Select bits

1111-1110 = Reserved, do not use
1101 = CMP1
1100 = Reserved, do not use
1011 = CCP4
1010 = ECCP3
1001 = ECCP2
1000 = ECCP1
0111 = The SAMP bit is cleared after SAMC<4:0> number of TAD clocks following the SAMP bit being set (Auto-Convert mode); no extended sample time is present
0110 = Unimplemented
0101 = TMR1
0100 = CTMU
0011 = TMR5
0010 = TMR3
0001 = INT0
0000 = The SAMP bit must be cleared by software to start conversion

bit 3 **Unimplemented:** Read as '0'

bit 2 **ASAM:** A/D Sample Auto-Start bit

1 = Sampling begins immediately after last conversion; SAMP bit is auto-set
0 = Sampling begins when SAMP bit is manually set

bit 1 **SAMP:** A/D Sample Enable bit

1 = A/D Sample-and-Hold amplifiers are sampling
0 = A/D Sample-and-Hold amplifiers are holding

bit 0 **DONE:** A/D Conversion Status bit

1 = A/D conversion cycle has completed
0 = A/D conversion has not started or is in progress

REGISTER 22-6: ADCON2H: A/D CONTROL REGISTER 2 HIGH

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
PVCFG1	PVCFG0	NVCFG0	OFFCAL	BUFREGEN	CSCNA	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **PVCFG<1:0>:** Converter Positive Voltage Reference Configuration bits
1x =Unimplemented, do not use
01 =External V_{REF+}
00 =AV_{DD}

bit 5 **NVCFG0:** Converter Negative Voltage Reference Configuration bit
1 = External V_{REF-}
0 = AV_{SS}

bit 4 **OFFCAL:** Offset Calibration Mode Select bit
1 = Inverting and non-inverting inputs of channel Sample-and-Hold are connected to AV_{SS}
0 = Inverting and non-inverting inputs of channel Sample-and-Hold are connected to normal inputs

bit 3 **BUFREGEN:** A/D Buffer Register Enable bit
1 = Conversion result is loaded into the buffer location determined by the converted channel
0 = A/D result buffer is treated as a FIFO

bit 2 **CSCNA:** Scan Input Selections for CH0+ During Sample A bit
1 = Scans inputs
0 = Does not scan inputs

bit 1-0 **Unimplemented:** Read as '0'

REGISTER 22-7: ADCON2L: A/D CONTROL REGISTER 2 LOW

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUFS(1)	SMPI4	SMPI3	SMPI2	SMPI1	SMPI0	BUFM(1)	ALTS
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **BUFS:** Buffer Fill Status bit(1)
- 1 = A/D is filling the upper half of the buffer; user should access data in the lower half
- 0 = A/D is filling the lower half of the buffer; user should access data in the upper half

bit 6-2 **SMPI<4:0>:** Interrupt Sample Increment Rate Select bits

Selects the number of sample/conversions per each interrupt.
- 11111 = Interrupt/address increment at the completion of conversion for each 32nd sample
- 11110 = Interrupt/address increment at the completion of conversion for each 31st sample
- • • •
- 00001 = Interrupt/address increment at the completion of conversion for every other sample
- 00000 = Interrupt/address increment at the completion of conversion for each sample

bit 1 **BUFM:** Buffer Fill Mode Select bit(1)
- 1 = A/D buffer is two, 13-word buffers, starting at ADC1BUF0 and ADC1BUF12, and sequential conversions fill the buffers alternately (Split mode)
- 0 = A/D buffer is a single, 26-word buffer and fills sequentially from ADC1BUF0 (FIFO mode)

bit 0 **ALTS:** Alternate Input Sample Mode Select bit
- 1 = Uses channel input selects for Sample A on first sample and Sample B on next sample
- 0 = Always uses channel input selects for Sample A

Note 1: These bits are only applicable when the buffer is used in FIFO mode (BUFREGEN = 0). In addition, BUFS is only used when BUFM = 1.

REGISTER 22-8: ADCON3H: A/D CONTROL REGISTER 3 HIGH

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRC	EXTSAM	PUMPEN	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ADRC:** A/D Conversion Clock Source bit
1 = RC Clock
0 = Clock derived from system clock

bit 6 **EXTSAM:** Extended Sampling Time bit
1 = A/D is still sampling after SAMP = 0
0 = A/D is finished sampling

bit 5 **PUMPEN:** Charge Pump Enable bit
1 = Charge pump for switches is enabled
0 = Charge pump for switches is disabled

bit 4-0 **SAMC<4:0>:** Auto-Sample Time Select bits
11111= 31 TAD
• • •
00001= 1 TAD
00000= 0 TAD

REGISTER 22-9: ADCON3L: A/D CONTROL REGISTER 3 LOW

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **ADCS<7:0>:** A/D Conversion Clock Select bits ((ADCS<7:0> + 1) 2/Fosc) = TAD
11111111
• • •= Reserved
01000000
00111111= 64 · 2/Fosc = TAD
• • •
00000001= 2 · 2/Fosc = TAD
00000000= 2/Fosc = TAD

REGISTER 22-10: ADCON5H: A/D CONTROL REGISTER 5 HIGH

R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0	R/W-0
ASENA	LPENA	CTMUREQ	—	—	—	ASINTMD1	ASINTMD0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **ASENA:** Auto-Scan Enable bit
1 = Auto-scan is enabled
0 = Auto-scan is disabled

bit 6 **LPENA:** Low-Power Enable bit
1 = Low power is enabled after scan
0 = Full power is enabled after scan

bit 5 **CTMUREQ:** CTMU Request bit
1 = CTMU is enabled when the A/D is enabled and active
0 = CTMU is not enabled by the A/D

bit 4-2 **Unimplemented:** Read as '0'

bit 1-0 **ASINTMD<1:0>:** Auto-Scan (Threshold Detect) Interrupt Mode bits
11 = Interrupt after Threshold Detect sequence completed and valid compare has occurred
10 = Interrupt after valid compare has occurred
01 = Interrupt after Threshold Detect sequence completed
00 = No interrupt

REGISTER 22-11: ADCON5L: A/D CONTROL REGISTER 5 LOW

U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	—	WM1	WM0	CM1	CM0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Read as '0'

bit 3-2 **WM<1:0>:** Write Mode bits

11 = Reserved
10 = Auto-compare only (conversion results are not saved, but interrupts are generated when a valid match occurs, as defined by the CM<1:0> and ASINTMD<1:0> bits)
01 = Convert and save (conversion results are saved to locations as determined by the register bits when a match occurs, as defined by the CMx bits)
00 = Legacy operation (conversion data is saved to a location determined by the buffer register bits)

bit 1-0 **CM<1:0>:** Compare Mode bits

11 = Outside Window mode (valid match occurs if the conversion result is outside of the window, defined by the corresponding buffer pair)
10 = Inside Window mode (valid match occurs if the conversion result is inside the window, defined by the corresponding buffer pair)
01 = Greater Than mode (valid match occurs if the result is greater than the value in the corresponding buffer register)
00 = Less Than mode (valid match occurs if the result is less than the value in the corresponding buffer register)

REGISTER 22-12: ADCHS0H: A/D SAMPLE SELECT REGISTER 0 HIGH

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB2	CH0NB1	CH0NB0	CH0SB4	CH0SB3	CH0SB2	CH0SB1	CH0SB0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **CH0NB<2:0>:** Sample B Channel 0 Negative Input Select bits
1xx = Unimplemented
011 = Unimplemented
010 = AN1
001 = Unimplemented
000 = V_{REF-}/AV_{SS}

bit 4-0 **CH0SB<4:0>:** Sample B Channel 0 Positive Input Select bits
11111 = $V_{BAT}/2$(1)
11110 = AV_{DD}(1)
11101 = AV_{SS}(1)
11100 = Band gap reference (V_{BG})(1,3)
11011 = $V_{BG}/2$(1)
11010 = $V_{BG}/6$(1)
11001 = CTMU
11000 = CTMU temperature sensor input (does not require ADCTMUEN1H<0> to be set)
10111 = AN23(2)
10110 = AN22(2)
10101 = AN21(2)
10100 = AN20(2)
10011 = AN19
10010 = AN18
10001 = AN17
10000 = AN16
01111 = AN15
01110 = AN14
01101 = AN13
01100 = AN12
01011 = AN11
01010 = AN10
01001 = AN9
01000 = AN8
00111 = AN7
00110 = AN6
00101 = AN5
00100 = AN4
00011 = AN3
00010 = AN2
00001 = AN1
00000 = AN0

Note 1: These input channels do not have corresponding memory mapped result buffers.
2: These channels are implemented in 80-pin and 100-pin devices only.
3: For accurately sampling the band gap set SMPI bits in ADCON2L register to 0, so that the ADC samples the band gap only once on every trigger. When the band gap is sampled multiple times, a large capacitive load is connected to the output of the band gap multiple times, which could cause the output to become unstable for a while and an overshoot or undershoot could be sampled.

REGISTER 22-13: ADCHS0L: A/D SAMPLE SELECT REGISTER 0 LOW

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NA2	CH0NA1	CH0NA0	CH0SA4	CH0SA3	CH0SA2	CH0SA1	CH0SA0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **CH0NA<2:0>:** Sample A Channel 0 Negative Input Select bits
1xx = Unimplemented
011 = Unimplemented
010 = AN1
001 = Unimplemented
000 = V_{REF-}/AV_{SS}

bit 4-0 **CH0SA<4:0>:** Sample A Channel 0 Positive Input Select bits
11111 = $V_{BAT}/2$(1)
11110 = AV_{DD}(1)
11101 = AV_{SS}(1)
11100 = Band gap reference (V_{BG})(1)
11011 = $V_{BG}/2$(1)
11010 = $V_{BG}/6$(1)
11001 = CTMU
11000 = CTMU temperature sensor input (does not require ADCTMUEN1H<0> to be set)
10111 = AN23(2)
10110 = AN22(2)
10101 = AN21(2)
10100 = AN20(2)
10011 = AN19
10010 = AN18
10001 = AN17
10000 = AN16
01111 = AN15
01110 = AN14
01101 = AN13
01100 = AN12
01011 = AN11
01010 = AN10
01001 = AN9
01000 = AN8
00111 = AN7
00110 = AN6
00101 = AN5
00100 = AN4
00011 = AN3
00010 = AN2
00001 = AN1
00000 = AN0

Note 1: These input channels do not have corresponding memory mapped result buffers.
2: These channels are implemented in 80-pin and 100-pin devices only.

REGISTER 22-14: ADCHIT1H: A/D SCAN COMPARE HIT REGISTER 1 HIGH (HIGH WORD)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CHH30	CHH29	CHH28	CHH27	CHH26	CHH25	CHH24
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-0 **CHH<30:24>:** A/D Compare Hit bits

If CM<1:0> = 11:

1 = A/D Result Buffer n has been written with data or a match has occurred

0 = A/D Result Buffer n has not been written with data

For All Other Values of CM<1:0>:

1 = A match has occurred on A/D Result Channel n

0 = No match has occurred on A/D Result Channel n

REGISTER 22-15: ADCHIT1L: A/D SCAN COMPARE HIT REGISTER 1 LOW (LOW WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CHH23	CHH22	CHH21	CHH20	CHH19	CHH18	CHH17	CHH16
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CHH<23:16>:** A/D Compare Hit bits

If CM<1:0> = 11:

1 = A/D Result Buffer n has been written with data or a match has occurred

0 = A/D Result Buffer n has not been written with data

For All Other Values of CM<1:0>:

1 = A match has occurred on A/D Result Channel n

0 = No match has occurred on A/D Result Channel n

REGISTER 22-16: ADCHIT0H: A/D SCAN COMPARE HIT REGISTER 0 HIGH (HIGH WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CHH15	CHH14	CHH13	CHH12	CHH11	CHH10	CHH9	CHH8
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CHH<15:8>:** A/D Compare Hit bits

If CM<1:0> = 11:

1 = A/D Result Buffer n has been written with data or a match has occurred

0 = A/D Result Buffer n has not been written with data

For all other values of CM<1:0>:

1 = A match has occurred on A/D Result Channel n

0 = No match has occurred on A/D Result Channel n

REGISTER 22-17: ADCHIT0L: A/D SCAN COMPARE HIT REGISTER 0 LOW (LOW WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CHH7	CHH6	CHH5	CHH4	CHH3	CHH2	CHH1	CHH0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CHH<7:0>:** A/D Compare Hit bits

If CM<1:0> = 11:

1 = A/D Result Buffer n has been written with data or a match has occurred

0 = A/D Result Buffer n has not been written with data

For all other values of CM<1:0>:

1 = A match has occurred on A/D Result Channel n

0 = No match has occurred on A/D Result Channel n

REGISTER 22-18: ADCSS1H: A/D INPUT SCAN SELECT REGISTER 1 HIGH (HIGH WORD)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CSS30	CSS29	CSS28	CSS27	CSS26	CSS25	CSS24
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-0 **CSS<30:24>:** A/D Input Scan Selection bits

1 = Includes corresponding channel for input scan
0 = Skips channel for input scan

REGISTER 22-19: ADCSS1L: A/D INPUT SCAN SELECT REGISTER 1 LOW (LOW WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS23	CSS22	CSS21	CSS20	CSS19	CSS18	CSS17	CSS16
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-0 **CSS<23:16>:** A/D Input Scan Selection bits

1 = Includes corresponding channel for input scan
0 = Skips channel for input scan

REGISTER 22-20: ADCSS0H: A/D INPUT SCAN SELECT REGISTER 0 HIGH (HIGH WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS15	CSS14	CSS13	CSS12	CSS11	CSS10	CSS9	CSS8
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CSS<15:8>:** A/D Input Scan Selection bits
1 = Includes corresponding channel for input scan
0 = Skips channel for input scan

REGISTER 22-21: ADCSS0L: A/D INPUT SCAN SELECT REGISTER 0 LOW (LOW WORD)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CSS7	CSS6	CSS5	CSS4	CSS3	CSS2	CSS1	CSS0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CSS<7:0>:** A/D Input Scan Selection bits
1 = Includes corresponding channel for input scan
0 = Skips channel for input scan

REGISTER 22-22: ADCTMUEN1H: CTMU ENABLE REGISTER 1 HIGH (HIGH WORD)(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	CTMUEN30	CTMUEN29	CTMUEN28	CTMUEN27	CTMUEN26	CTMUEN25	CTMUEN24
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6-0 **CTMUEN<30:24>:** CTMU Enabled During Conversion bits
1 = CTMU is enabled and connected to the selected channel during conversion
0 = CTMU is not connected to this channel

Note 1: The actual number of channels available depends on which channels are implemented on a specific device; refer to the device data sheet for details. Unimplemented channels are read as '0'.

REGISTER 22-23: ADCTMUEN1L: CTMU ENABLE REGISTER 1 LOW (LOW WORD)(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CTMUEN23	CTMUEN22	CTMUEN21	CTMUEN20	CTMUEN19	CTMUEN18	CTMUEN17	CTMUEN16
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 15-0 **CTMUEN<23:16>:** CTMU Enabled During Conversion bits
1 = CTMU is enabled and connected to the selected channel during conversion
0 = CTMU is not connected to this channel

Note 1: The actual number of channels available depends on which channels are implemented on a specific device; refer to the device data sheet for details. Unimplemented channels are read as '0'.

REGISTER 22-24: ADCTMUEN0H: CTMU ENABLE REGISTER 0 HIGH (HIGH WORD)(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CTMUEN15	CTMUEN14	CTMUEN13	CTMUEN12	CTMUEN11	CTMUEN10	CTMUEN9	CTMUEN8
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CTMUEN<15:8>:** CTMU Enabled During Conversion bits
1 = CTMU is enabled and connected to the selected channel during conversion
0 = CTMU is not connected to this channel

Note 1: The actual number of channels available depends on which channels are implemented on a specific device; refer to the device data sheet for details. Unimplemented channels are read as '0'.

REGISTER 22-25: ADCTMUEN0L: CTMU ENABLE REGISTER 0 LOW (LOW WORD)(1)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CTMUEN7	CTMUEN6	CTMUEN5	CTMUEN4	CTMUEN3	CTMUEN2	CTMUEN1	CTMUEN0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **CTMUEN<7:0>:** CTMU Enabled During Conversion bits
1 = CTMU is enabled and connected to the selected channel during conversion
0 = CTMU is not connected to this channel

Note 1: The actual number of channels available depends on which channels are implemented on a specific device; refer to the device data sheet for details. Unimplemented channels are read as '0'.

REGISTER 22-26: ANCFG – ANALOG INPUT REGISTER

U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0
—	—	—	—	—	VBG6EN	VBG2EN	VBGEN
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
‘1’ = Bit is set	‘0’ = Bit is cleared	

bit 7-3 **Unimplemented:** Read as ‘0’

bit 2 **VBG6EN:** Band Gap Divide-by-6 Control bit
1 = Reference voltage on
0 = Reference voltage off

bit 1 **VBG2EN:** Band Gap Divide-by-2 Control bit
1 = Reference voltage on
0 = Reference voltage off

bit 0 **VBGEN:** Band Gap Control bit
1 = Reference voltage on
0 = Reference voltage off

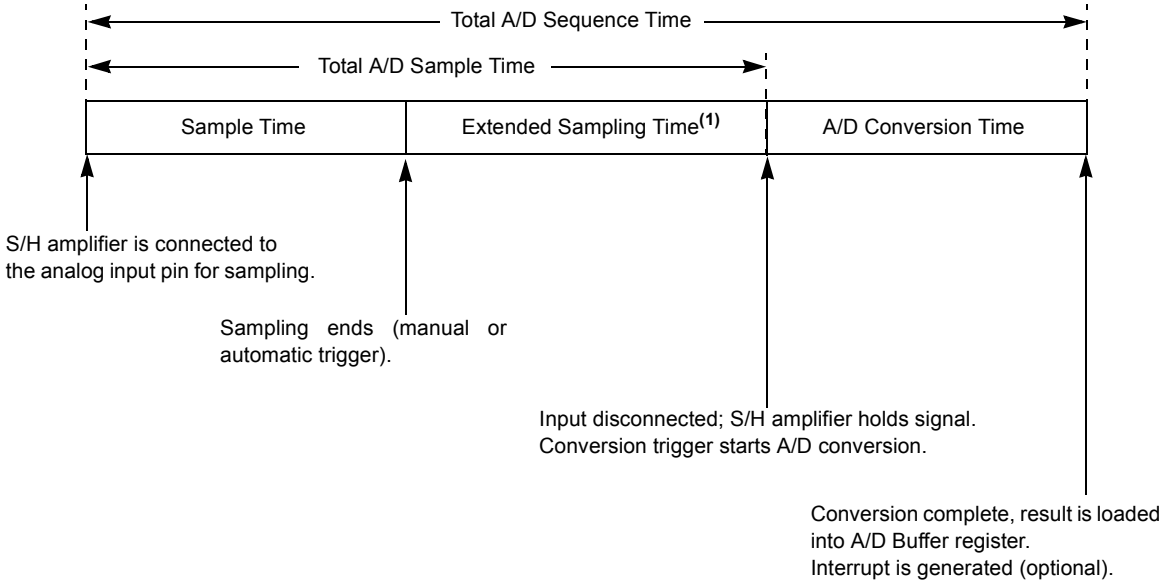
22.2 A/D Terminology and Conversion Sequence

Sample time is the time that the A/D module's S/H amplifier is connected to the analog input pin. The sample time may be started and ended automatically by the A/D Converter's hardware or under direct program control. There is a minimum sample time to ensure that the S/H amplifier will give sufficient accuracy for the A/D conversion.

The conversion trigger ends the sampling time and begins an A/D conversion or a repeating sequence. The conversion trigger sources can be taken from a variety of hardware sources or can be controlled directly in software. One of the conversion trigger options is an auto-conversion, which uses a counter and the A/D clock to set the time between auto-conversions. The Auto-Sample mode and auto-conversion trigger can be used together to provide continuous, automatic conversions without software intervention. When automatic sampling is used, an extended sampling interval is extended between the time the sampling ends and the conversion starts.

Conversion time is the time required for the A/D Converter to convert the voltage held by the S/H amplifier. An A/D conversion requires one A/D clock cycle (TAD) to convert each bit of the result, plus two additional clock cycles, or a total of 14 TAD cycles for a 12-bit conversion. When the conversion is complete, the result is loaded into one of the A/D result buffers. The S/H can be reconnected to the input pin and a CPU interrupt may be generated. The sum of the sample time(s) and the A/D conversion time provides the total A/D sequence time. Figure 22-2 shows the basic conversion sequence and the relationship between intervals.

FIGURE 22-2: A/D SAMPLE/CONVERT SEQUENCE



Note 1: In Automatic Sampling modes, Extended Sampling Time is added to the sequence when the value for the Auto-Sampling Time is greater than 0. Otherwise, sampling ends and conversion starts whenever the SAMP bit is cleared.

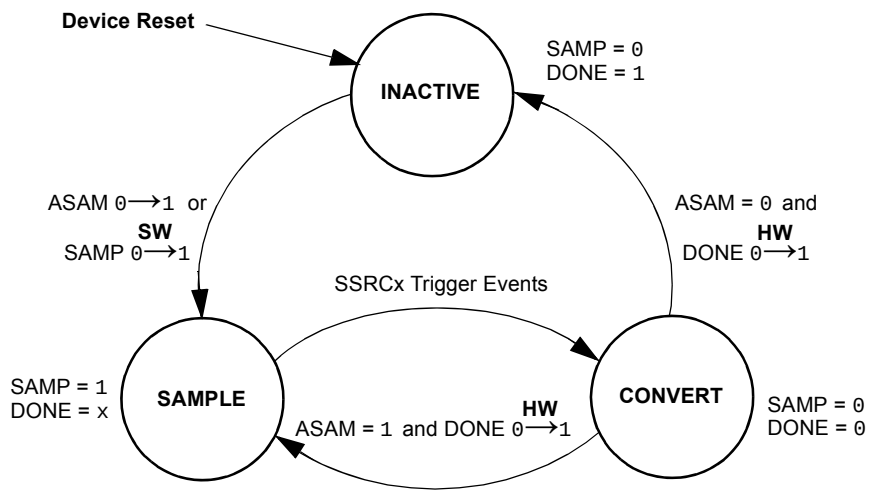
22.2.1 OPERATION AS A STATE MACHINE

The A/D conversion process can be thought of in terms of a finite state machine (Figure 22-3). The sample state represents the time that the input channel is connected to the S/H amplifier and the signal is passed to the converter input. The convert state is transitory. The module enters this state as soon as it exits the sample state and transitions to a different state when that is done. The inactive state is the default state prior to module initialization and following a software-controlled conversion; it can be avoided in operation by using Auto-Sample mode. Machine states are identified by the state of several control and Status bits in ADCON1H/L.

If the module is configured for Auto-Sample mode, the operation “ping-pongs” continuously between the sample and convert states. The module automatically selects the input channels to be sampled (if channel scanning is enabled), while the selected conversion trigger source paces the entire operation. Any time that Auto-Sample mode is not used for conversion, it is available for the sample state. The user needs to make certain that acquisition time is sufficient, in addition to accounting for the normal concerns about system throughput.

Whenever the issue of sampling time is important, the significant event is the transition from sample to convert state. This is the point where the Sample-and-Hold aperture closes, and it is essentially the signal value at this instant, which is applied to the A/D for conversion to digital.

FIGURE 22-3: A/D MODULE STATE MACHINE MODEL



Legend: **HW** = Automatic Hardware event; **SW** = Software Controlled event.

Note: See Register 22-5 for definitions of the ASAM, SAMP, DONE and SSRC<3:0> bits.

22.3 A/D Module Configuration

All of the registers described in the previous section must be configured for module operation to be fully defined. An effective approach is first, to describe the signals and sequences for the particular application. Typically, it is an iterative process to assign signals to port pins, to establish timing methods and to organize a scanning scheme, as well as to integrate the whole process with the software design.

The various configuration and control functions of the module are distributed throughout the module's six control registers. Control functions can be broadly sorted into four groups: input, timing, conversion and output. Table 22-1 shows the register location of control or Status bits by register.

TABLE 22-1: A/D MODULE FUNCTIONS BY REGISTERS AND BITS

A/D Function	Register(s)	Specific Bits
Input	ADCON2H/L	PVCFG<1:0>, NVCFG, OFFCAL, CSCNA, ALTS
	ADCHS0H/L	CH0NB<2:0>, CH0SB<4:0>, CH0NA<2:0>, CH0SA<4:0>
	ADCSS0H/L	CSS<15:0>
	ADCSS1H/L	CSS<30:16>
	ADCTMEN0H/L	CTMEN<15:0>
	ADCTMEN1H/L	CTMEN<31:16>
Conversion	ADCON1H/L	ADON, SSRC<3:0>, ASAM, SAMP, DONE, MODE12
	ADCON2H/L	SMPI<4:0>
	ADCON3H/L	EXTSAM
	ADCON5H/L	ASEN, LPENA, ASINTMD<1:0>
Timing	ADCON3H/L	ADRC, SAMC<4:0>, ADCS<7:0>
Output	ADCON1H/L	FORM<1:0>
	ADCON2H/L	BUFS, BUFM, BUFREGEN
	ADCON5H/L	WM<1:0>, CM<1:0>

Note: Do not write to the SSRCx, BUFS, SMPIx, BUFM and ALTS bits, or the ADCON3H/L, and ADCSS0H/L registers, while ADON = 1; otherwise, indeterminate conversion data may result.

The following steps should be followed for performing an A/D conversion:

1. Configure the A/D module:
   - Select the output resolution (if configurable)
   - Select the voltage reference source to match the expected range on analog inputs
   - Select the analog conversion clock to match the desired data rate with a processor clock
   - Determine how sampling will occur
   - Set the multiplexer input assignments
   - Select the desired sample/conversion sequence
   - Select the output data format
   - Select the output value destination
   - Select the number of readings per interrupt
2. Configure the A/D interrupt (if required):
   - Clear the ADIF bit
   - Select the A/D interrupt priority
3. Turn on the A/D module.

The options for each configuration step are described in the subsequent sections.

22.3.1 SELECTING THE RESOLUTION

The MODE12 bit (ADCON1H<3>) controls output resolution. Setting this bit selects 12-bit resolution.

22.3.2 SELECTING THE VOLTAGE REFERENCE SOURCE

The voltage references for A/D conversions are selected using the PVCFG<1:0> and NVCFG0 control bits (ADCON2H<7:5>). The upper voltage reference (VR+) may be AVDD, the external VREF+ or an internal band gap reference voltage. The lower voltage reference (VR-) may be AVSS or the VREF- input pin. The available options vary between device families.

The external voltage reference pins may be shared with the AN2 and AN3 inputs on low pin count devices. The A/D Converter can still perform conversions on these pins when they are shared with the VREF+ and VREF- input pins.

The voltages applied to the external reference pins must meet certain specifications. Refer to the device data sheet for further details.

22.3.3 SELECTING THE A/D CONVERSION CLOCK

The A/D Converter has a maximum rate at which conversions may be completed. An analog module clock, TAD, controls the conversion timing. The A/D conversion requires 14 clock periods (14 TAD) for a 12-bit conversion and 12 clock periods (12 TAD) for a 10-bit conversion. The A/D clock is derived from the device instruction clock.

The period of the A/D conversion clock is software selected using a 6-bit counter. There are 64 possible options for TAD, specified by the ADCSX bits in the ADCON3L register. Equation 22-1 gives the TAD value as a function of the ADCSx control bits and the device instruction cycle clock period, TCY. For correct A/D conversions, the A/D conversion clock (TAD) must be selected to ensure a minimum TAD time, as specified by the device family data sheet.

EQUATION 22-1: A/D CONVERSION CLOCK PERIOD

$$T_{AD} = 2/F_{OSC}\ (ADCS + 1)$$

$$ADCS = \frac{T_{AD}}{2/F_{OSC}} - 1$$

Note: PLL is disabled.

The A/D Converter also has its own dedicated RC clock source that can be used to perform conversions. The A/D RC clock source should be used when conversions are performed while the device is in Sleep mode. The RC oscillator is selected by setting the ADRC bit (ADCON3H<7>). When the ADRC bit is set, the ADCSx bits have no effect on A/D operation.

22.3.4 CONFIGURING ANALOG PORT PINS

The A/D module does not have an internal provision to configure port pins for analog operation. Instead, input pins are configured as analog inputs through the Analog Select registers (ANSn, where 'n' is the port name). A pin is configured as an analog input when the corresponding ANSn bit is set. By default, pins with multiplexed analog and digital functions are configured as analog pins on device Reset.

For external analog inputs, both the ANSn register and the corresponding TRIS register bits control the operation of the A/D port pins. The port pins that will function as analog inputs must also have their corresponding TRIS bits set, specifying the pins as inputs. After a device Reset, all TRIS bits are set. If the I/O pin associated with an A/D channel is configured as a digital output (TRIS bit is cleared), while the pin is configured for Analog mode, the port digital output level (VOH or VOL) will be converted.

Note 1: When reading a PORT register, any pin configured as an analog input reads as '0'.

2: Analog levels on any pin that is defined as a digital input may cause the input buffer to consume current that is out of the device's specification.

22.3.5 INPUT CHANNEL SELECTION

The A/D Converter incorporates two independent sets of input multiplexers (MUX A and MUX B) that allow users to choose which analog channels are to be sampled. The inputs specified by the CH0SAx and CH0NAx bits are collectively called the MUX A inputs. The inputs specified by the CH0SBx and CH0NBx bits are collectively called the MUX B inputs.

Functionally, MUX A and MUX B are very similar to each other. Both multiplexers allow any of the analog input channels to be selected for individual sampling and allow selection of a negative reference source for differential signals. In addition, MUX A can be configured for sequential analog channel scanning. This is discussed in more detail in **Section 22.3.5.1 "Configuring MUX A And MUX B Inputs"** and **Section 22.3.5.3 "Scanning Through Several Inputs"**.

Note: Different PIC18F devices will have different numbers of analog inputs. Verify the analog input availability against the particular device's data sheet.

22.3.5.1 Configuring MUX A And MUX B Inputs

The user may select any one of up to 32 inputs available to the A/D Converter as the positive input of the S/H amplifier. For MUX A, the CH0SA<4:0> bits (ADCHS0L<4:0>) normally select the analog channel for the positive input. For MUX B, the positive channel is selected by the CH0SB<4:0> bits (ADCHS0H<4:0>).

All of the external analog channels are available as positive inputs. In addition to the external inputs, these may also include device supply voltage (AVDD), the logic core supply voltage (VDDCORE), the internal band gap voltage (VBG) and/or multiples or fractions of VBG. One or more additional input channels are used for the CTMU. These selections leave the A/D disconnected from all other inputs. The options vary by device family; refer to the specific device data sheet for details.

The CTMU input is selected by the ADCTMUEN1H/L, ADCTMUEN0H/L registers. Setting a particular bit in one of these registers effectively assigns the analog output from the CTMU to the corresponding A/D input channel, automatically enabling the CTMU. Many devices will already have a CH0SAx bit combination designated for use of the CTMU. This setting disconnects the converter from any other load. This channel should be the one selected by the appropriate ADCTMUEN bit. If another channel is selected, verify that any other analog sources are disconnected from that channel; otherwise, erroneous readings may result.

For the negative (inverting) input of the amplifier, the user has up to eight options, selected by the CH0NA<2:0> and CH0NB<2:0> bits (ADCHS0L<7:5> and ADCHS0H<7:5>, respectively). Options typically include the device ground (AVSS), the current VR-source designated by the NVCFG0 bit (ADCON2H<5>), and one or more of the external analog input channels. As with the non-inverting inputs, the options vary by device family.

22.3.5.2 Alternating MUX A And MUX B Input Selections

By default, the A/D Converter only samples and converts the inputs selected by MUX A. The ALTS bit (ADCON2L<0>) enables the module to alternate between two sets of inputs selected by MUX A and MUX B during successive samples.

If the ALTS bit is '0', only the inputs specified by the CH0SAx and CH0NAx bits are selected for sampling. When the ALTS bit is '1', the module will alternate between the MUX A inputs on one sample and the MUX B inputs on the subsequent sample.

If the ALTS bit is '1' on the first sample/convert sequence, the inputs specified by the CH0SAx and CH0NAx bits are selected for sampling. On the next sample/convert sequence, the inputs specified by the CH0SBx and CH0NBx bits are selected for sampling. This pattern repeats for subsequent sample conversion sequences.

22.3.5.3 Scanning Through Several Inputs

When using MUX A to select analog inputs, the A/D module has the ability to scan multiple analog channels. When the CSCNA bit (ADCON2H<>) is set, the CH0SA bits are ignored and the channels specified by the ADCSS1H/L, ADCSS0H/L registers are sequentially sampled.

Each bit in the ADCSS1H/L registers and ADCSS0H/L registers (when implemented) corresponds to one of the analog channels. If a bit in the ADCSS0H/L or ADCSS1H/L registers is set, the corresponding analog channel is included in the scan sequence. Inputs are always scanned from lower to higher numbered inputs, starting at the first selected channel after each interrupt occurs.

Note 1: If the number of scanned inputs selected is greater than the number of samples taken per interrupt, the higher numbered inputs will not be sampled.

2: If the CTMU channel is to be included in a scan operation, verify that the proper analog input channel is selected and that the AD1CTMEN register(s) are correctly configured. For more information, see **Section 22.3.5.1 "Configuring MUX A And MUX B Inputs"**.

The ADCSS1H/L, ADCSS0H/L registers' bits specify the positive input of the channel. The CH0NAx bits still select the negative input of the channel during scanning.

Scanning is only available on the MUX A input selection. The MUX B input selection, as specified by the CH0SBx bits, will still select the alternating input. When alternated sampling between MUX A and MUX B is selected (ALTS = 1), the input will alternate between a set of scanning inputs specified by the ADCSS1H/L, ADCSS0H/L registers, and a fixed input specified by the CH0SBx bits.

Automatic scanning can be used in conjunction with the Threshold Detect feature to determine if one or more analog channels meet a predetermined set of conditions while the CPU is inactive. This is described in detail in **Section 22.7 "Threshold Detect Operation"**.

22.3.5.4 Internal Channels In Low-power Modes

While the A/D module can scan and convert analog inputs in low-power modes, some internal analog inputs may be unavailable in Sleep mode. The main examples are the CTMU module, the internal band gap voltage source and the on-chip voltage regulator (for those devices that include one). The A/D module provides a method to make these resources available automatically through the CTMUREQ bit (ADCON5H<5>). Setting one or more of these bits causes the corresponding internal analog source(s) to become active during a channel scan.

22.3.6 ENABLING THE MODULE

When the ADON bit (ADCON1H<7>) is set, the module is fully powered and functional. When ADON is '0', the module is disabled. Although the digital and analog portions of the circuit are turned off for maximum current savings, the contents of all registers are maintained.

Conversion data stored in the ADCBUF registers will also be maintained, including any threshold values stored by the user. It may be necessary to re-initialize these registers to their proper values before re-enabling the module.

When enabling the module by setting the ADON bit, the user must wait for the analog stages to stabilize. For the stabilization time, refer to **Section 30.0 "Electrical Specifications"**.

22.4 Controlling the Sampling Process

22.4.1 MANUAL SAMPLING

Setting the SAMP bit (ADCON1L<1>) while the ASAM bit (ADCON1L<2>) is clear causes the A/D to begin sampling. Clearing the SAMP bit ends sampling and automatically begins the conversion; however, there must be a sufficient delay between setting and clearing SAMP for the sampling process to start. Sampling will not resume until the SAMP bit is once again set. For an example, see Figure 22-4.

22.4.2 AUTOMATIC SAMPLING

Setting the ASAM bit causes the A/D to automatically begin sampling after a conversion has been completed. One of several options can be used as an event to end sampling and complete the conversions. Sampling will continue on the next selected channel after the conversion in progress has completed. For an example, see Figure 22-5.

22.4.3 MONITORING SAMPLE STATUS

The SAMP bit indicates the sampling state of the A/D. Generally, when the SAMP bit clears, indicating the end of sampling, the DONE bit is automatically cleared to indicate the start of conversion. If SAMP is '0' while DONE is '1', the A/D is in an inactive state.

22.4.4 ABORTING A SAMPLE

While in Manual Sampling mode, clearing the SAMP bit will terminate sampling. If SSRC<3:0> = 0000, it may also start a conversion automatically.

Clearing the ASAM bit while in Automatic Sampling mode will not terminate an ongoing sample/convert sequence; however, sampling will not automatically resume after a subsequent conversion.

22.5 Controlling the Conversion Process

The conversion trigger source will terminate sampling and start a selected sequence of conversions. The SSRC<3:0> bits (ADCON1L<7:4>) select the source of the conversion trigger.

Note 1: The available conversion trigger sources may vary depending on the PIC18F device variant. Refer to the specific device data sheet for the available conversion trigger sources.

2: The SSRCx selection bits should not be changed when the A/D module is enabled. If the user wishes to change the conversion trigger source, disable the A/D module first by clearing the ADON bit (AD1CON1<15>).

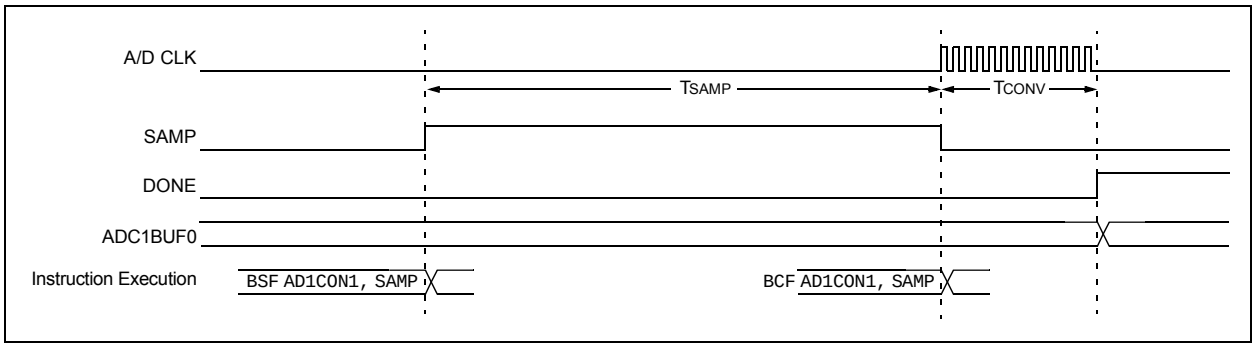
22.5.1 MANUAL CONTROL

When SSRC<3:0> = 0000, the conversion trigger is under software control. Clearing the SAMP bit (ADCON1L<1>) starts the conversion sequence.

Figure 22-4 is an example where setting the SAMP bit initiates sampling, and clearing the SAMP bit terminates sampling and starts conversion. The user software must time the setting and clearing of the SAMP bit to ensure adequate sampling time of the input signal.

Figure 22-5 is an example where setting the ASAM bit initiates automatic sampling, and clearing the SAMP bit terminates sampling and starts conversion. After the conversion completes, the module sets the SAMP bit and returns to the sample state. The user software must time the clearing of the SAMP bit to ensure

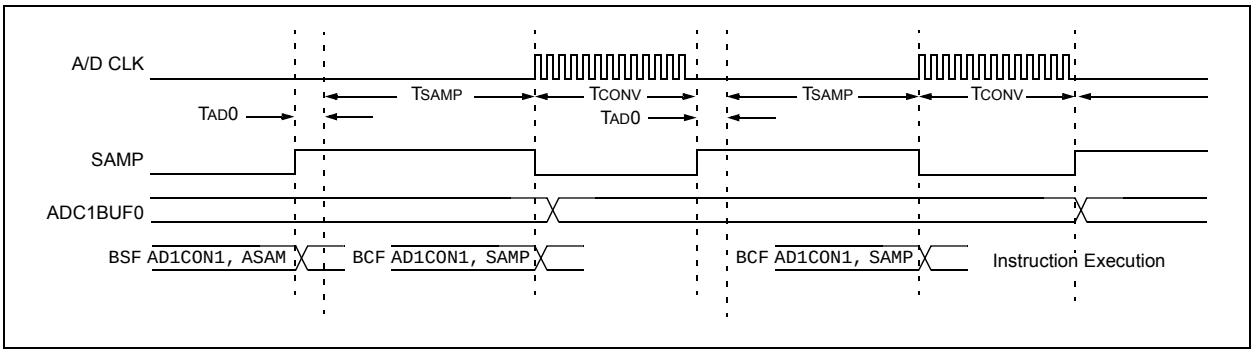
FIGURE 22-4: CONVERTING ONE CHANNEL, MANUAL SAMPLE START, MANUAL CONVERSION START



EXAMPLE 22-1: CONVERTING ONE CHANNEL, MANUAL SAMPLE START, MANUAL CONVERSION START CODE

```
int ADCValue;

ANCON1= 0x02;                       // AN2 as analog, all other pins are digital
ADCON1L = 0x00;                     // SAMP bit = 0 ends sampling and starts converting
ADCHS0L = 0x02;                     // Connect AN2 as S/H+ input
                                    // in this example AN2 is the input
ADCSS0L = 0;
ADCON3L = 0x02;                     // Manual Sample, Tad = 3Tcy
ADCON2L = 0;
ADCON1Hbits.ADON = 1;               // turn ADC ON

while (1)                           // repeat continuously
{
    ADCON1Hbits.SAMP = 1;           // start sampling...
    Delay();                        // Ensure the correct sampling time has elapsed
                                    // before starting conversion.
    ADCON1Hbits.SAMP = 0;           // start converting
    while (!ADCON1Lbits.DONE){};    // conversion done?
    ADCValue = ADCBUF0;             // yes then get ADC value
}
```

FIGURE 22-5: CONVERTING ONE CHANNEL, AUTOMATIC SAMPLE START, MANUAL CONVERSION START



22.5.2 CLOCKED CONVERSION TRIGGER

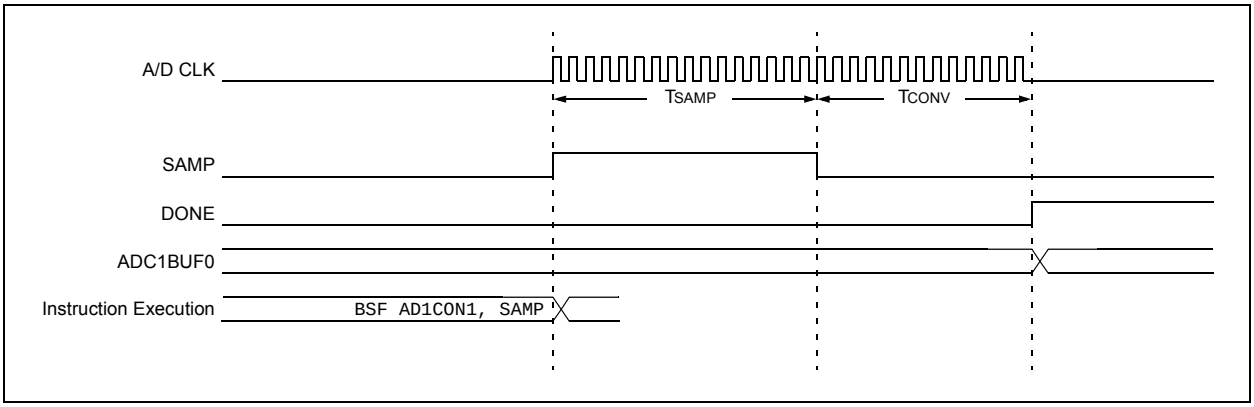
When ADRC = 1, the conversion trigger is under A/D clock control. The SAMCx bits (ADCON3H<4:0>) select the number of TAD clock cycles between the start of sampling and the start of conversion. After the start of sampling, the module will count a number of TAD clocks specified by the SAMCx bits. The SAMCx bits must always be programmed for at least one clock cycle to ensure sampling requirements are met.

EQUATION 22-2: CLOCKED CONVERSION TRIGGER TIME

$$T_{SMP} = SAMC<4:0> * T_{AD}$$

Figure 22-6 shows how to use the clocked conversion trigger with the sampling started by the user software.

FIGURE 22-6: CONVERTING ONE CHANNEL, MANUAL SAMPLE START, TAD-BASED CONVERSION START



EXAMPLE 22-2: CONVERTING ONE CHANNEL, MANUAL SAMPLE START, TAD-BASED CONVERSION START CODE

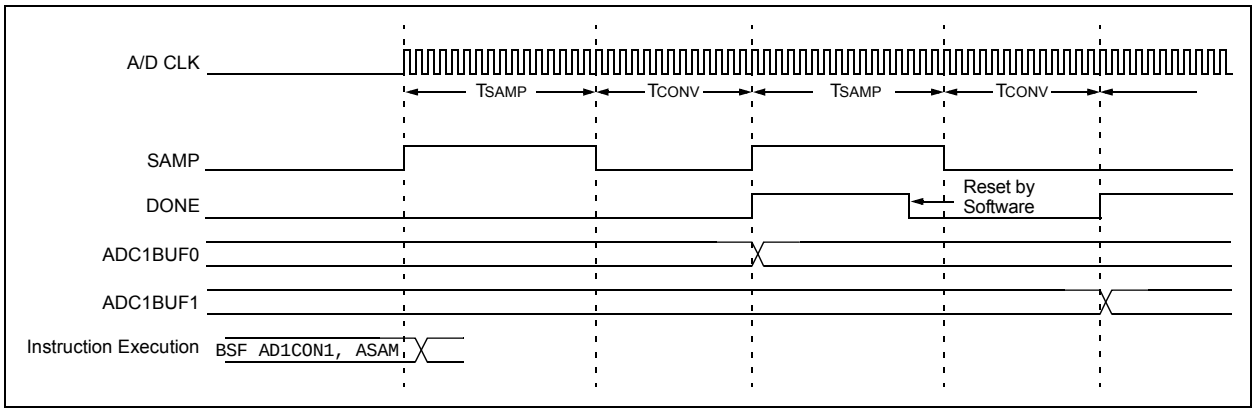
```
int ADCValue;

ANCON2  = 0x10;                     // all PORTB = Digital; RB12 = analog
ADCON1L = 0x70;                     // SSRC<2:0> = 111 implies internal counter ends sampling
                                    // and starts converting.
ADCHS0L = 0x0C;                     // Connect AN12 as S/H input.
                                    // in this example AN12 is the input
ADCSS0H = 0;
ADCON3H = 1F;                       // Sample time = 31Tad, Tad = 3Tcy
ADCON3L=02;
ADCON2L = 0;
ADCON1Hbits.ADON = 1;               // turn ADC ON
while (1)                           // repeat continuously
{
    ADCON1Lbits.SAMP = 1;           // start sampling, then after 31Tad go to conversion
    while (!ADCON1Lbits.DONE){};    // conversion done?
    ADCValue = ADCBUF0;             // yes then get ADC value
}                                   // repeat
```

22.5.2.1 Free-running Sample Conversion Sequence

Using the Auto-Convert Conversion Trigger mode (SSRC<3:0> = 0111), in combination with the Auto-Sample Start mode (ASAM = 1), allows the A/D module to schedule sample/conversion sequences with no intervention by the user or other device resources. This "Clocked" mode, shown in Figure 22-7, allows continuous data collection after module initialization.

Note that all timing in this mode scales with TAD, either from the A/D internal RC clock or from TCY (as prescaled by the ADCS<7:0> bits). In both cases, the SAMC<4:0> bits set the number of TAD clocks in TSAMP. TCONV is fixed at 12 TAD.

FIGURE 22-7: CONVERTING ONE CHANNEL, AUTO-SAMPLE START, TAD-BASED CONVERSION START



22.5.2.2 Sample Time Considerations Using Clocked Conversion Trigger And Automatic Sampling

The user must ensure the sampling time satisfies the sampling requirements, as outlined in **Section 22.9 "A/D Sampling Requirements"**. Assuming that the module is set for automatic sampling and using a clocked conversion trigger, the sampling interval is specified by the SAMCx bits.

22.5.3 EVENT TRIGGER CONVERSION START

It is often desirable to synchronize the end of sampling and the start of conversion with some other time event. Depending on the device family, the A/D module has up to 16 sources available to use as a conversion trigger event. The event trigger is selected by the SSRC<3:0> bits (ADCON1L<7:4>).

As noted, the available event triggers vary between device families. Refer to the specific device data sheet for specific information. The examples that follow represent trigger sources that are implemented in most devices. Note that the SSRCx bit assignments may vary in some devices.

22.5.3.1 External Int0 Pin Trigger

When SSRC<3:0> = 0001, the A/D conversion is triggered by an active transition on the INT0 pin. The pin may be programmed for either a rising edge input or a falling edge input.

22.5.3.2 Special Event Trigger

When SSRC<3:0> = 0010, the A/D is triggered by a Special Event Trigger. Refer to CCP and ECCP section for more information about Special Event Triggers.

22.5.3.3 Synchronizing A/D Operations To Internal Or External Events

The modes where an external event trigger pulse ends sampling and starts conversion may be used in combination with auto-sampling (ASAM = 1) to cause the A/D to synchronize the sample conversion events to the trigger pulse source. For example, in Figure 22-9, where SSRC<3:0> = 0010 and ASAM = 1, the A/D will always end sampling and start conversions synchronously with the timer compare trigger event. The A/D will have a sample conversion rate that corresponds to the timer comparison event rate.

22.5.3.4 Sample Time Considerations For Automatic Sampling/conversion Sequences

Different sample/conversion sequences provide different available sampling times for the S/H channel to acquire the analog signal. The user must ensure the sampling time satisfies the sampling requirements, as outlined in **Section 22.9 "A/D Sampling Requirements"**.

Assuming that the module is set for automatic sampling, and an external trigger pulse is used as the conversion trigger, the sampling interval is a portion of the trigger pulse interval. The sampling time is the trigger pulse period, less the time required to complete the conversion.

EQUATION 22-3: CALCULATING AVAILABLE SAMPLING TIME FOR SEQUENTIAL SAMPLING

$$T_{SMP} = \text{Trigger Pulse Interval } (T_{SEQ}) - \text{Conversion Time } (T_{CONV}) = T_{SEQ} - T_{CONV}$$

FIGURE 22-8: MANUAL SAMPLE START, CONVERSION TRIGGER-BASED CONVERSION START

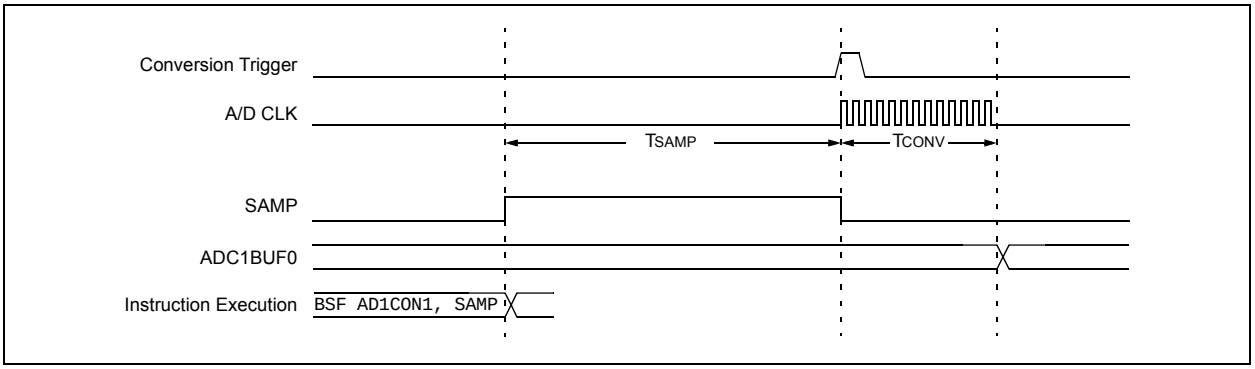
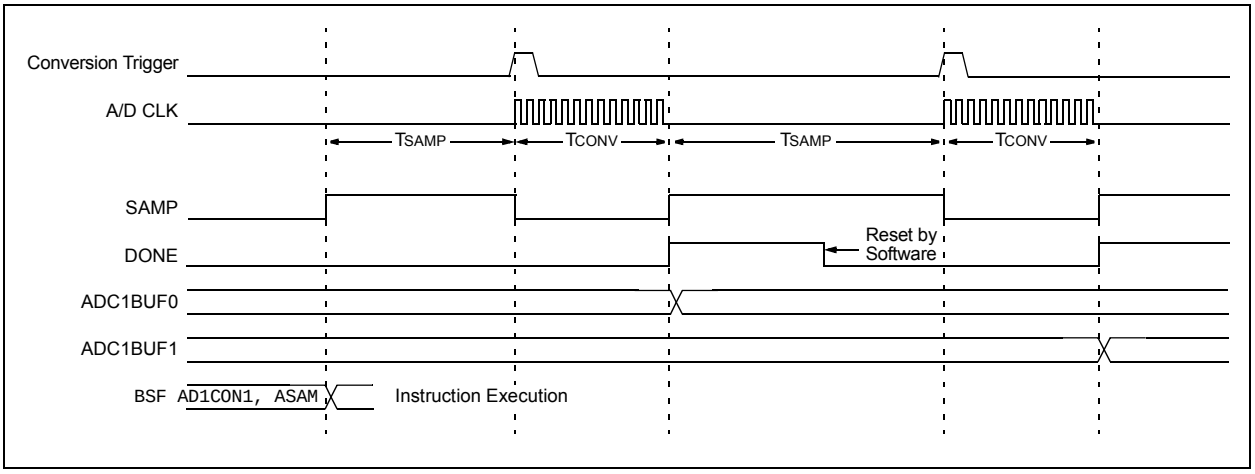


FIGURE 22-9: AUTO-SAMPLE START, CONVERSION TRIGGER-BASED CONVERSION START



22.5.4 MONITORING SAMPLE/ CONVERSION STATUS

The DONE bit (ADCON1L<0>) indicates the conversion state of the A/D. Generally, when the SAMP bit clears, indicating the end of sampling, the DONE bit is automatically cleared to indicate the start of conversion. If SAMP is '0' while DONE is '1', the A/D is in an inactive state.

In some operational modes, the SAMP bit may also invoke and terminate sampling. In these modes, the DONE bit cannot be used to terminate conversions in progress.

22.5.5 GENERATING A/D INTERRUPTS

The SMPI<4:0> bits (ADCON2L<6:2>) control the generation of the A/D Interrupt Flag, ADIF. The A/D Interrupt Flag is set after the number of sample/conversion sequences is specified by the SMPIx bits, after the start of sampling, and continues to recur after that number of samples. The value specified by the SMPIx bits also corresponds to the number of data samples in the buffer, up to the maximum of 16. To enable the interrupt, it is necessary to set the A/D Interrupt Enable bit, ADIE.

If auto-scan is enabled (ADCON5<7> = 1), interrupt generation is controlled by the ASINTMDx bits (ADCON5H<1:0>). For more information, refer to **Section 22.7.4 "Threshold Detect Interrupts"**.

22.5.6 ABORTING A CONVERSION

Clearing the ADON bit during a conversion will abort the current conversion. The A/D results buffer will not be updated with the partially completed A/D conversion sample; that is, the corresponding ADCBUF buffer location will continue to contain the value of the last completed conversion (or the last value written to the buffer).

22.5.7 OFFSET CALIBRATION

The module provides a simple calibration method to offset the effects of internal device noise. While not always necessary, this may be helpful in situations where weak analog signals are being converted. Calibration is performed by using the OFFCAL bit (ADCON2H<4>). This disconnects the S/H amplifier entirely from any inputs. With the OFFCAL bit set, a single reference conversion is performed. The results of this conversion are value added by internal device noise. This result can be stored by the application, then used as an offset value for future conversions.

22.6 A/D Results Buffer

As conversions are completed, the module writes the results of the conversions into the A/D result buffer. This buffer is a RAM array of fixed word size, accessed through the SFR space. The size of the buffer is determined by the number of external analog input channels on the device, allowing one word for each channel. Depending on the device, additional buffer space may be provided for one or more internal analog channels (e.g., band gap sources). The number of buffer addresses is always even and always at least equal to the number of external channels.

User software may attempt to read each A/D conversion result as it is generated; however, this might consume too much CPU time. Generally, to minimize software overhead, the module will fill the buffer with results and then generate an interrupt when the buffer is filled.

Note:	This section describes buffer operation in Legacy mode (ADCON5L<3:2> = 00). Buffer operation is different when the Compare Only or Compare and Save modes are used with the Threshold Detect feature. For more information, see **Section 22.7 "Threshold Detect Operation"**.

22.6.1 NUMBER OF CONVERSIONS PER INTERRUPT

The SMPI<4:0> bits select how many A/D conversions will take place before the CPU is interrupted. This can vary from 1 to 16 samples per interrupt. The A/D Converter module always starts writing its conversion results at the beginning of the buffer, after each interrupt. For example, if SMPI<4:0> = 00000, the conversion results will always be written to the ADCBUF0. In this example, no other buffer locations would be used, since only one sequence per interrupt is specified.

22.6.2 BUFFER FILL MODES

The results buffer can be configured to operate in either of two modes: a standard FIFO mode, compatible with the earlier 10-bit A/D module (default), or a Channel Indexed mode. The Fill mode is selected by the BUFREGEN bit (ADCON2H<3>).

22.6.2.1 FIFO Modes

When BUFREGEN = 0, the results buffer operates in FIFO mode. The first conversion results, after initiating conversions, is written to the first available buffer address. Subsequent conversions are written to the next sequential buffer location, continuing until the process is interrupted. If allowed to continue without interrupts, the module would fill each location and then wrap around to the first address, continuing the process.

The BUFM bit (ADCON2L<1>) controls how the buffer is filled. When BUFM is '1', the buffer is split into two equal halves: a lower half (ADCBUF0 through ADCBUF[(n/2) - 1]) and an upper half (ADCBUF[n/2] through ADCBUFn), where n is the number of available analog channels (both internal and external). The buffers will alternately receive the conversion results after each interrupt event. The initial buffer used after BUFM is set is the lower group.

When BUFM is '0', the entire buffer is used for all conversion sequences.

Note: When the BUFM bit is set, the user should not program the SMPIx bits to a value that specifies more than (n/2) conversions per interrupt.

The decision to use the split buffer feature will depend upon how much time is available to move the buffer contents after the interrupt, as determined by the application. If the application can quickly unload a full buffer within the time it takes to sample and convert one channel, the BUFM bit can be '0', and up to 16 conversions may be done per interrupt. The application will have one sample/convert time before the first buffer location is overwritten.

If the processor cannot unload the buffer within the sample and conversion time, the BUFM bit should be '1'. For example, if SMPI<4:0> = 00111, then eight conversions will be loaded into the lower half of the buffer, following which, an interrupt may occur. The next eight conversions will be loaded into the upper half of the buffer. The processor will, therefore, have the entire time between interrupts to move the eight conversions out of the buffer.

22.6.2.2 Buffer Fill Status

When the conversion result buffer is split (BUFM = 1), the BUFS Status bit (ADCON2L<7>) indicates which half of the buffer that the A/D Converter is currently writing. If BUFS = 0, the A/D Converter is filling the lower group and the user application should read conversion values from the upper group. If BUFS = 1, the situation is reversed, and the user application should read conversion values from the lower group.

22.6.2.3 Channel Indexed Mode

When BUFREGEN = 1, FIFO operation is disabled. In this Fill mode, the conversion result for each channel is written only to the buffer location that corresponds to that channel. For example, any conversions performed on AN0 are stored only in ADCBUF0. The same holds true for AN1 and ADCBUF1, and so on. Subsequent conversions on a particular channel that occur, prior to an interrupt, will result in any previous data in that location being overwritten.

Channel Indexed mode is particularly useful when used with the Threshold Detect feature, as this allows the user to easily test for a particular condition on a specific analog channel without creating an excess of CPU overhead. This is covered in more detail in **Section 22.7 "Threshold Detect Operation"**.

22.6.3 BUFFER DATA FORMATS

The results of each A/D conversion are 12 bits wide (optionally, 10 bits wide in some devices). To maintain data format compatibility, the result of each conversion is automatically converted to one of four selectable, 16-bit formats. The FORM<1:0> bits (ADCON1H<1:0>) select the format. Figure 22-10 and Figure 22-11 show the data output formats that can be selected. Table 22-2 through Table 22-5 show the numerical equivalents for the various conversion result codes.

FIGURE 22-10: A/D OUTPUT DATA FORMATS (12-BIT)

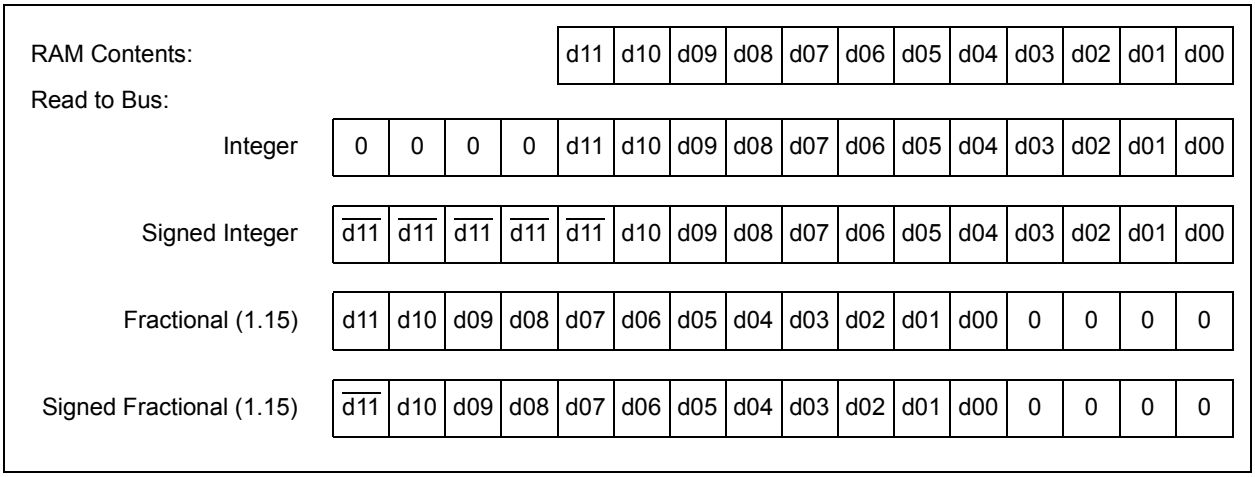


TABLE 22-2: NUMERICAL EQUIVALENTS OF VARIOUS RESULT CODES: 12-BIT INTEGER FORMATS

VIN/VREF	12-Bit Output Code	16-Bit Integer Format/ Equivalent Decimal Value		16-Bit Signed Integer Format/ Equivalent Decimal Value	
4095/4096	1111 1111 1111	0000 1111 1111 1111	4095	0000 0111 1111 1111	2047
4094/4096	1111 1111 1110	0000 1111 1111 1110	4094	0000 0111 1111 1110	2046
•••					
2049/4096	1000 0000 0001	0000 1000 0000 0001	2049	0000 0000 0000 0001	1
2048/4096	1000 0000 0000	0000 1000 0000 0000	2048	0000 0000 0000 0000	0
2047/4096	0111 1111 1111	0000 0111 1111 1111	2047	1111 1111 1111 1111	-1
•••					
1/4096	0000 0000 0001	0000 0000 0000 0001	1	1111 1000 0000 0001	-2047
0/4096	0000 0000 0000	0000 0000 0000 0000	0	1111 1000 0000 0000	-2048

TABLE 22-3: NUMERICAL EQUIVALENTS OF VARIOUS RESULT CODES: 12-BIT FRACTIONAL FORMATS

VIN/VREF	12-Bit Output Code	16-Bit Fractional Format/ Equivalent Decimal Value		16-Bit Signed Fractional Format/ Equivalent Decimal Value	
4095/4096	1111 1111 1111	1111 1111 1111 0000	0.999	0111 1111 1111 0000	0.499
4094/4096	1111 1111 1110	1111 1111 1110 0000	0.998	0111 1111 1110 0000	0.498
•••					
2049/4096	1000 0000 0001	1000 0000 0001 0000	0.501	0000 0000 0001 0000	0.001
2048/4096	1000 0000 0000	1000 0000 0000 0000	0.500	0000 0000 0000 0000	0.000
2047/4096	0111 1111 1111	0111 1111 1111 0000	0.499	1111 1111 1111 0000	-0.001
•••					
1/4096	0000 0000 0001	0000 0000 0001 0000	0.001	1000 0000 0001 0000	-0.499
0/4096	0000 0000 0000	0000 0000 0000 0000	0.000	1000 0000 0000 0000	-0.500

FIGURE 22-11: A/D OUTPUT DATA FORMATS (10-BIT)

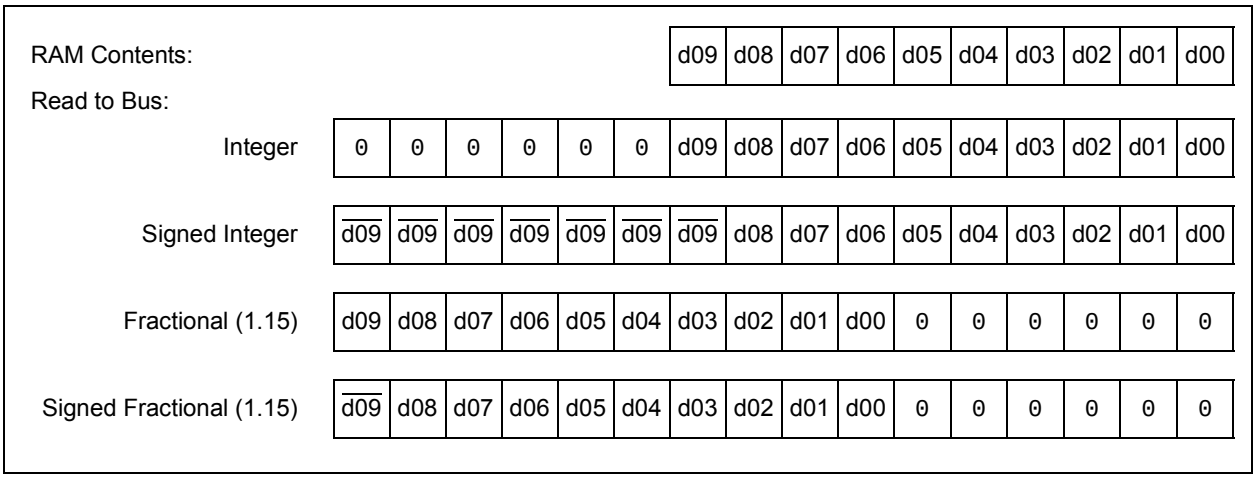


TABLE 22-4: NUMERICAL EQUIVALENTS OF VARIOUS RESULT CODES: 10-BIT INTEGER FORMATS

VIN/VREF	10-Bit Output Code	16-Bit Integer Format/ Equivalent Decimal Value		16-Bit Signed Integer Format/ Equivalent Decimal Value	
1023/1024	11 1111 1111	0000 0011 1111 1111	1023	0000 0001 1111 1111	511
1022/1024	11 1111 1110	0000 0011 1111 1110	1022	0000 0001 1111 1110	510
•••					
513/1024	10 0000 0001	0000 0010 0000 0001	513	0000 0000 0000 0001	1
512/1024	10 0000 0000	0000 0010 0000 0000	512	0000 0000 0000 0000	0
511/1024	01 1111 1111	0000 0001 1111 1111	511	1111 1111 1111 1111	-1
•••					
1/1024	00 0000 0001	0000 0000 0000 0001	1	1111 1110 0000 0001	-511
0/1024	00 0000 0000	0000 0000 0000 0000	0	1111 1110 0000 0000	-512

TABLE 22-5: NUMERICAL EQUIVALENTS OF VARIOUS RESULT CODES: 10-BIT FRACTIONAL FORMATS

VIN/VREF	10-Bit Output Code	16-Bit Fractional Format/ Equivalent Decimal Value		16-Bit Signed Fractional Format/ Equivalent Decimal Value	
1023/1024	11 1111 1111	1111 1111 1100 0000	0.999	0111 1111 1100 0000	0.499
1022/1024	11 1111 1110	1111 1111 1000 0000	0.998	0111 1111 1000 0000	0.498
•••					
513/1024	10 0000 0001	1000 0000 0100 0000	0.501	0000 0000 0100 0000	0.001
512/1024	10 0000 0000	1000 0000 0000 0000	0.500	0000 0000 0000 0000	0.000
511/1024	01 1111 1111	0111 1111 1100 0000	0.499	1111 1111 1100 0000	-0.001
•••					
1/1024	00 0000 0001	0000 0000 0100 0000	0.001	1000 0000 0100 0000	-0.499
0/1024	00 0000 0000	0000 0000 0000 0000	0.000	1000 0000 0000 0000	-0.500

22.7 Threshold Detect Operation

Threshold Detect is a significant extension of the Auto-Scan feature offered in previous 10-bit A/D modules. In addition to being able to repeatedly sample a pre-defined sequence of analog channels, Threshold Detect allows the user to define match conditions based on the conversion results and generate an interrupt based on these conditions. During normal operation, this can potentially reduce the amount of CPU time spent on processing A/D interrupts. For low-power applications, this can allow the CPU to remain inactive for longer periods, waking only when specific analog conditions are met.

When selected by the user, Threshold Detect changes the operation of the A/D results buffer by making it a read/write array for both conversion results and comparison (threshold) values. It also brings into play the ADCHIT registers, which are used to indicate match conditions. Independently selectable comparison and buffer storage settings make a wide range of operating combinations possible.

22.7.1 OPERATING MODES

The operation of Threshold Detect is mostly controlled by the ADCON5H/L registers. The ASENA bit (ADCON5L<7>) controls overall operation of Threshold Detect; setting this bit enables the functionality.

As with Legacy Auto-Scan operation, the channels to be included are selected using the ADCSS1H/L, ADCSS0H/L registers. Setting a particular bit in either register includes the corresponding channel in an automatic sequential scan. One or more channels may be selected. After the channels have been selected, setting both the CSCNA and ASENA bits to enable a single scan of the designated channels. The scan itself is triggered by the trigger source programmed by the SSRC<3:0> bits.

.

Note: Legacy Auto-Scan (i.e., sequential scanning of analog channels on MUX A, without any comparison) is controlled by the CSCNA bit (ADCON2H<2>) and does not depend on the ASEN bit to function.

The LPENA bit (ADCON5H<6>) allows Threshold Detect to function with a low-power feature. By design, Threshold Detect can perform comparison operations when the device is in Sleep or Idle modes, waking the CPU when it generates an interrupt. Setting LPENA configures the device to return to low-power operation after the interrupt has been serviced.

The Compare Mode bits, CM<1:0> (ADCON5L<1:0>), select the type of comparison to be performed. Four types are available:

- The result of the current conversion is greater than a reference threshold
- The result of the current conversion is less than a reference threshold
- The result of the current conversion is between two predefined thresholds (“Inside Window”)
- The result of the current conversion is outside of the predefined thresholds (“Outside Window”)

The Write Mode bits, WM<1:0> (ADCON5L<3:2>), determine the disposition of the conversion. Three options are available:

- Discard the conversion after the comparison has been performed
- Store the conversion after the comparison has been performed
- Store the conversion without comparison (Legacy mode)

22.7.1.1 Buffer Operation And Comparisons

For Buffer Write modes that involve storing conversions (WM<1:0> = 0x), the BUFM and BUFREGEN bits control how the buffer functions (as a channel indexed, single FIFO or split FIFO buffer). However, when the compare and store option is selected (WM<1:0> = 01), using a FIFO mode may overwrite the buffers of other channels and cause unpredictable comparison results. For that reason, always use Channel Indexed Buffer mode (BUFREGEN = 1) when using the compare and store option.

22.7.1.2 Buffer Operation In Windowed Comparisons (Channel Mirroring)

The use of windowed comparisons changes the available options for the results buffer. To accommodate the storage of two threshold values, the buffer is automatically split into halves, similar to Split FIFO mode. Buffer addresses in each half are paired, with the lowest address in one buffer, matched to the buffer address in the upper half. (For example, in a 16-word buffer, ADCBUF0 is paired with ADCBUF9, ADCBUF1 is paired with ADCBUF10, and so on.) This pairing is referred to as “channel mirroring”.

Mirroring can obviously be applied only to the lower A/D channels; for most devices, this corresponds to the lower half of the external analog channels. This does not mean that those buffer locations cannot be used for other purposes. However, storing any other data in a particular buffer location, where channel mirroring is being used, may result in misleading comparison evaluations.

22.7.2 SETTING COMPARISON THRESHOLDS

The comparison thresholds for Threshold Detect are set by writing the desired values to an appropriate location in the A/D results buffer. This can only be done when the module is deactivated (ADCON1H<7> = 0).

The location of the threshold is determined by the comparison type. For simple greater than, and less than, comparisons, the value is written to the buffer location corresponding to the input channel to be monitored. For example, if AN0 is to be monitored for a voltage over a certain level, the ceiling threshold is stored in ADCBUF0.

The location of the thresholds for windowed comparisons are written to two addresses. The lower value is written to the address corresponding to the monitored channel. The upper value is stored in the corresponding mirrored address in the upper half of the buffer. To expand on the previous example, if the conversion on AN0 is to be a windowed comparison, the floor threshold is stored in ADCBUF0, while the ceiling threshold is stored in ADCBUF9.

22.7.3 COMPARE HIT REGISTERS

To determine if a particular event has occurred, the A/D module uses two registers to record match events. These registers are referred to as the Compare Hit registers and are designated, ADCHIT1H/L and ADCHIT0H/L. The registers map their individual bits sequentially to each of the (up to) 32 analog channels. If a particular channel in a device is not implemented, the corresponding Compare Hit bit (CHHn) is not implemented.

Each bit serves as an event semaphore for its corresponding channel. When the programmed event occurs on that channel, the bit becomes set and stays set until it is cleared by the application. It is the user's responsibility to clear the bits after the application has evaluated them.

Depending on the event, more than one Compare Hit bit may be set. The significance of a set bit must be interpreted by the application in the context of the Compare mode selected. Particular examples are covered in **Section 22.7.5 “Comparison Mode Examples”**.

22.7.4 THRESHOLD DETECT INTERRUPTS

The A/D module can generate an interrupt and set the ADIF flag based on Threshold Detect operation. This is based on completion of a Threshold Detect sequence and/or the occurrence of a valid comparison. When Threshold Detect is enabled (ASENA = 1), A/D module interrupt generation is governed by the ASINTMDx bits (ADCON5H<1:0>), superseding any configuration implemented by the SMPIx bits (ADCON2L<6:2>). For information on alternative interrupt settings, refer to **Section 22.6.1 "Number of Conversions Per Interrupt"**.

The Threshold Detect interrupt is configured by the ASINTMD<1:0> bits (ADCON5H<1:0>). Options include interrupt after a scan sequence, interrupt after a scan sequence with a valid match, interrupt after a valid match (without waiting for the sequence to end) or no interrupt.

22.7.5 COMPARISON MODE EXAMPLES

The following examples show the effect of valid comparisons on the results buffer and the

Compare Hit registers. In each figure, changes within the registers are indicated in bold.

For the sake of simplicity, the examples assume a device with only 16 analog inputs. Devices with a greater number of channels, and thus, larger results buffers and two Compare Hit registers, will function in a similar fashion.

Note:	When using any comparison mode, always use channel indexed buffer storage (BUFREGEN = 1). Otherwise, the threshold values for other channels may be overwritten, resulting in unpredictable comparisons.

22.7.5.1 Simple Comparisons (Greater And Less Than Results)

When the Compare Mode bits, CM<1:0> (ADCON5L<1:0>), are programmed as '0x', the converter compares the sampled value to see if it is greater than (CM<1:0> = 01), or less than (CM<1:0> = 00), the threshold value in the buffer location. If the condition is met, both of the following occur:

- The Compare Hit bit (CHHn) for the corresponding channel is set.
- If the Write Mode bits, WM<1:0> (ADCON5L<3:2>), are programmed to '01', the converted value is written to the buffer, replacing the threshold value. If WM<1:0> = 10, the converted value is discarded.

The changes to the result buffer and the Compare Hit register are shown in Figure 22-13. Note that they are the same for both types of simple comparison.

FIGURE 22-12: SIMPLE COMPARISON OPERATIONS (GREATER THAN AND LESS THAN)

Before Conversion and Comparison

ADC1BUF15	—
ADC1BUF14	—
ADC1BUF13	—
ADC1BUF12	—
ADC1BUF11	—
ADC1BUF10	—
ADC1BUF9	—
ADC1BUF8	—
ADC1BUF7	—
ADC1BUF6	—
ADC1BUF5	—
ADC1BUF4	—
ADC1BUF3	—
ADC1BUF2	Threshold Value
ADC1BUF1	—
ADC1BUF0	—

After Conversion and Comparison

	Compare Only (‘10’)	Compare and Store (‘01’)
ADC1BUF15	—	—
ADC1BUF14	—	—
ADC1BUF13	—	—
ADC1BUF12	—	—
ADC1BUF11	—	—
ADC1BUF10	—	—
ADC1BUF9	—	—
ADC1BUF8	—	—
ADC1BUF7	—	—
ADC1BUF6	—	—
ADC1BUF5	—	—
ADC1BUF4	—	—
ADC1BUF3	—	—
ADC1BUF2	Threshold Value	**Conversion Value**
ADC1BUF1	—	—
ADC1BUF0	—	—

AD1CHITL							
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

AD1CHITL							
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	**2**	1	0
0	0	0	0	0	**1**	0	0

22.7.5.2 Inside Window Comparison

When the Compare Mode bits, CM<1:0>, are programmed as '10', the converter compares the sampled value to see if it falls between the threshold values in the buffer and mirrored channel location. Since the value in the mirrored channel location is always the greater value of the two thresholds, the condition is met when:

Threshold 2 > Converted Value > Threshold 1

In this case, both of the following occur:

- The Compare Hit bit (CHHn) for the corresponding channel is set; the Compare Hit bit for the mirrored channel remains cleared.
- If the Write Mode bits, WM<1:0> (ADCON5L<3:2>), are programmed to '01', the converted value is written to the buffer, replacing the lower threshold value. If WM<1:0> = 10, the converted value is discarded.

The changes to the result buffer and the Compare Hit register are shown in Figure 22-14.

FIGURE 22-13: INSIDE WINDOW COMPARISON OPERATION

Before Conversion and Comparison

ADC1BUF15	—
ADC1BUF14	—
ADC1BUF13	—
ADC1BUF12	—
ADC1BUF11	—
ADC1BUF10	Threshold 2
ADC1BUF9	—
ADC1BUF8	—
ADC1BUF7	—
ADC1BUF6	—
ADC1BUF5	—
ADC1BUF4	—
ADC1BUF3	—
ADC1BUF2	Threshold 1
ADC1BUF1	—
ADC1BUF0	—

After Conversion and Comparison

	Compare Only (‘10’)	Compare and Store (‘01’)
ADC1BUF15	—	—
ADC1BUF14	—	—
ADC1BUF13	—	—
ADC1BUF12	—	—
ADC1BUF11	—	—
ADC1BUF10	Threshold 2	Threshold 2
ADC1BUF9	—	—
ADC1BUF8	—	—
ADC1BUF7	—	—
ADC1BUF6	—	—
ADC1BUF5	—	—
ADC1BUF4	—	—
ADC1BUF3	—	—
ADC1BUF2	Threshold 1	**Conversion Value**
ADC1BUF1	—	—
ADC1BUF0	—	—

AD1CHITL							
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

AD1CHITL							
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	**2**	1	0
0	0	0	0	0	**1**	0	0

22.7.5.3 Outside Window Comparison

When the Compare Mode bits CM<1:0> are programmed as '11', the converter compares the sampled value to see if it falls outside of the threshold values in the buffer and mirrored channel location. Again, since the value in the mirrored channel location is always the greater value of the two thresholds, the condition is met when either:

Converted Value >Threshold 2

or

Threshold 1 > Converted Value

In these cases, the following occurs:

- The Compare Hit bit (CHHn) for the corresponding channel is set.
- If the converted value is greater than Threshold 2, the CHHn bit for the mirrored channel is also set. If it is less than Threshold 1, the mirrored channel bit remains '0'.
- If the Write Mode bits, WM<1:0> (ADCON5L<3:2>), are programmed to '01':
  - If the converted value is above Threshold 2, the converted value is written to the mirrored channel address, replacing the upper threshold value.
  - If the converted value is below Threshold 1, the converted value is written to the channel address, replacing the lower threshold value.
- If WM<1:0> = 10, the converted value is discarded.

The changes to the result buffer and the Compare Hit register are shown in Figure 22-15 (over the upper threshold) and Figure 22-16 (under the lower threshold).

Note that when a Windowed Comparison mode is selected and channel mirroring is enabled, nothing prevents a conversion from another operation from being stored in the mirrored channel location. In the previous examples of windowed operation, if AN10 is included in a Threshold Detect operation, a conversion on AN10 might be tested against the upper threshold for AN2, stored in that location. This could result in the threshold value being overwritten and/or the CHH10 bit being set.

For this reason, users must always carefully consider the allocation and use of the upper analog channels (both external and internal) when using Windowed Compare modes. Wherever possible, exclude the upper analog channels for Threshold Detect operations, and convert and test those channels in a separate routine.

FIGURE 22-14: OUTSIDE WINDOW COMPARISON OPERATION (OVER THRESHOLD 2)

Before Conversion and Comparison

ADC1BUF15	—
ADC1BUF14	—
ADC1BUF13	—
ADC1BUF12	—
ADC1BUF11	—
ADC1BUF10	Threshold 2
ADC1BUF9	—
ADC1BUF8	—
ADC1BUF7	—
ADC1BUF6	—
ADC1BUF5	—
ADC1BUF4	—
ADC1BUF3	—
ADC1BUF2	Threshold 1
ADC1BUF1	—
ADC1BUF0	—

After Conversion and Comparison

	Compare Only (‘10’)	Compare and Store (‘01’)
ADC1BUF15	—	—
ADC1BUF14	—	—
ADC1BUF13	—	—
ADC1BUF12	—	—
ADC1BUF11	—	—
ADC1BUF10	Threshold 2	**Conversion Value**
ADC1BUF9	—	—
ADC1BUF8	—	—
ADC1BUF7	—	—
ADC1BUF6	—	—
ADC1BUF5	—	—
ADC1BUF4	—	—
ADC1BUF3	—	—
ADC1BUF2	Threshold 1	Threshold 1
ADC1BUF1	—	—
ADC1BUF0	—	—

AD1CHITL							
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

AD1CHITL							
15	14	13	12	11	**10**	9	8
0	0	0	0	0	**1**	0	0
7	6	5	4	3	**2**	1	0
0	0	0	0	0	**1**	0	0

FIGURE 22-15: OUTSIDE WINDOW COMPARISON OPERATION (UNDER THRESHOLD 1)

Before Conversion and Comparison

ADC1BUF15	—
ADC1BUF14	—
ADC1BUF13	—
ADC1BUF12	—
ADC1BUF11	—
ADC1BUF10	Threshold 2
ADC1BUF9	—
ADC1BUF8	—
ADC1BUF7	—
ADC1BUF6	—
ADC1BUF5	—
ADC1BUF4	—
ADC1BUF3	—
ADC1BUF2	Threshold 1
ADC1BUF1	—
ADC1BUF0	—

After Conversion and Comparison

	Compare Only (‘10’)	Compare and Store (‘01’)
ADC1BUF15	—	—
ADC1BUF14	—	—
ADC1BUF13	—	—
ADC1BUF12	—	—
ADC1BUF11	—	—
ADC1BUF10	Threshold 2	Threshold 2
ADC1BUF9	—	—
ADC1BUF8	—	—
ADC1BUF7	—	—
ADC1BUF6	—	—
ADC1BUF5	—	—
ADC1BUF4	—	—
ADC1BUF3	—	—
ADC1BUF2	Threshold 1	**Conversion Value**
ADC1BUF1	—	—
ADC1BUF0	—	—

AD1CHITL

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

AD1CHITL

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	**2**	1	0
0	0	0	0	0	**1**	0	0

22.8 Examples

22.8.1 INITIALIZATION

Example 22-1 shows a simple initialization code example for the A/D module. Operation in Idle mode is disabled, output data is in unsigned fractional format, and AVDD and AVSS are used for VR+ and VR-. The start of sampling, as well as the start of conversion (conversion trigger), are performed directly in software. Scanning of inputs is disabled and an interrupt occurs after every sample/convert sequence (one conversion result) with only one channel (AN0) being converted. The A/D conversion clock is TCY/2.

In this particular configuration, all 16 analog input pins are set up as analog inputs. It is important to note that with this A/D module, I/O pins are configured for analog or digital operation at the I/O port with the ANSn Analog Select registers. The use of these registers is described in detail in the I/O Port chapter of the specific device data sheet.

This example shows one method of controlling a sample/convert sequence by manually setting and clearing the SAMP bit (ADCON1L<1>). This method, among others, is more fully discussed in **Section 22.4 “Controlling the Sampling Process”** and **Section 22.5 “Controlling the Conversion Process”**.

EXAMPLE 22-3: A/D INITIALIZATION CODE EXAMPLE

```
ADCON1H = 0x22;         // Configure sample clock source
ADCON1L = 0x00;         // and conversion trigger mode.
                        // Unsigned Fraction format (FORM<1:0>=10),
                        // Manual conversion trigger (SSRC<3:0>=0000),
                        // Manual start of sampling (ASAM=0),
                        // S/H in Sample (SAMP = 1)
ADCON2H = 0;            // Configure A/D voltage reference
ADCON2L = 0;            // and buffer fill modes.
                        // Vr+ and Vr- from AVdd and AVss(PVCFG<1:0>=00, NVCFG=0),
                        // Inputs are not scanned,
                        // Interrupt after every sample
ADCON3H = 0;            // Configure sample time = 1Tad,
ADCON3L = 0;            // A/D conversion clock as Tcy

ADCHS0H = 0;            // Configure input channels,
ADCHS0L = 0;            // S/H+ input is AN0,
                        // S/H- input is Vr- (AVss).
ADCSS0L = 0;            // No inputs are scanned.
ADCSS0H = 0;            // No inputs are scanned.
PIR1bits.ADIF = 0;      // Clear A/D conversion interrupt.

// Configure A/D interrupt priority bits (ADIP) here, if
// required. Default priority level is high.

PIE1bits.ADIE = 1;      // Enable A/D conversion interrupt
ADCON1Hbits.ADON = 1;   // Turn on A/D
ADCON1Lbits.SAMP = 1;   // Start sampling the input
Delay();                // Ensure the correct sampling time has elapsed
                        // before starting conversion.
ADCON1Lbits.SAMP = 0;   // End A/D sampling and start conversion

// Example code for A/D ISR:
#pragma interrupt _ADC1Interrupt
void _ADC1Interrupt(void)
{
    PIR1bits.ADIF = 0;
}
```

22.8.2 CONVERSION SEQUENCE EXAMPLES

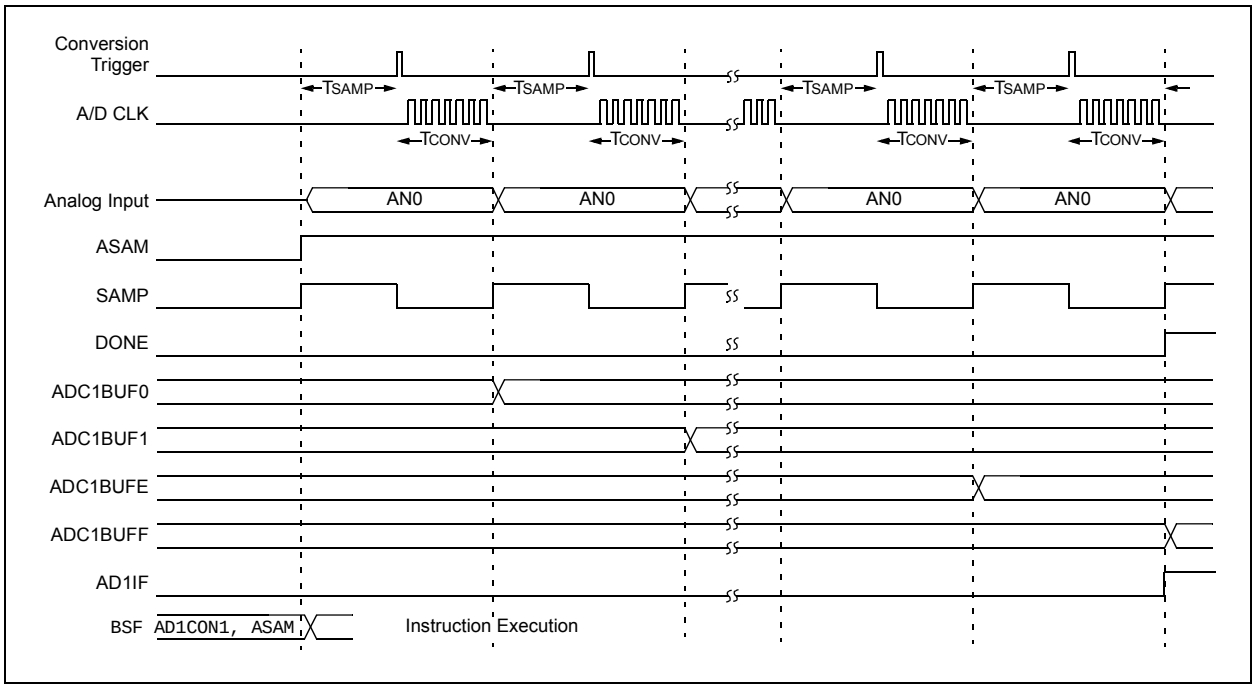
The following configuration examples show the A/D operation in different sampling and buffering configurations. In each example, setting the ASAM bit starts automatic sampling. A conversion trigger ends sampling and starts conversion.

22.8.2.1 Sampling and Converting a Single Channel Multiple Times

In this case Figure 22-16, one A/D input, AN0, will be sampled and converted. The results are stored in the ADCBUFn buffer. This process repeats 16 times until the buffer is full and then the module generates an interrupt. The entire process will then repeat.

With the ALTS bit clear, only the MUX A inputs are active. The CH0SAx and CH0NAx bits are specified (AN0 - VR-) as the inputs to the Sample-and-Hold channel. All other input selection bits are unused.

FIGURE 22-16: CONVERTING ONE CHANNEL 16 TIMES PER INTERRUPT



EXAMPLE 22-4: CONVERTING A SINGLE CHANNEL 16 TIMES PER INTERRUPT

A/D Configuration:

- Select AN0 for S/H+ Input (CH0SA<4:0> = 00000)
- Select V_{R-} for S/H- Input (CH0NA<2:0> = 000)
- Configure for No Input Scan (CSCNA = 0)
- Use Only MUX A for Sampling (ALTS = 0)
- Set AD1IF on Every 16th Sample (SMPI<4:0> = 01111)
- Configure Buffers for Single, 16-Word Results (BUFM = 0)

Operational Sequence:

1. Sample MUX A Input AN0; Convert and Write to Buffer 0h.
2. Sample MUX A Input AN0; Convert and Write to Buffer 1h.
3. Sample MUX A Input AN0; Convert and Write to Buffer 2h.
4. Sample MUX A Input AN0; Convert and Write to Buffer 3h.
5. Sample MUX A Input AN0; Convert and Write to Buffer 4h.
6. Sample MUX A Input AN0; Convert and Write to Buffer 5h.
7. Sample MUX A Input AN0; Convert and Write to Buffer 6h.
8. Sample MUX A Input AN0; Convert and Write to Buffer 7h.
9. Sample MUX A Input AN0; Convert and Write to Buffer 8h.
10. Sample MUX A Input AN0; Convert and Write to Buffer 9h.
11. Sample MUX A Input AN0; Convert and Write to Buffer Ah.
12. Sample MUX A Input AN0; Convert and Write to Buffer Bh.
13. Sample MUX A Input AN0; Convert and Write to Buffer Ch.
14. Sample MUX A Input AN0; Convert and Write to Buffer Dh.
15. Sample MUX A Input AN0; Convert and Write to Buffer Eh.
16. Sample MUX A Input AN0; Convert and Write to Buffer Fh.
17. Set AD1IF Flag (and generate interrupt, if enabled).
18. Repeat (1-16) After Return from Interrupt.

Results Stored in Buffer (after 2 cycles):

Buffer Address	Buffer Contents at 1st AD1IF Event	Buffer Contents at 2nd AD1IF Event
ADC1BUF0	AN0, Sample 1	AN0, Sample 17
ADC1BUF1	AN0, Sample 2	AN0, Sample 18
ADC1BUF2	AN0, Sample 3	AN0, Sample 19
ADC1BUF3	AN0, Sample 4	AN0, Sample 20
ADC1BUF4	AN0, Sample 5	AN0, Sample 21
ADC1BUF5	AN0, Sample 6	AN0, Sample 22
ADC1BUF6	AN0, Sample 7	AN0, Sample 23
ADC1BUF7	AN0, Sample 8	AN0, Sample 24
ADC1BUF8	AN0, Sample 9	AN0, Sample 25
ADC1BUF9	AN0, Sample 10	AN0, Sample 26
ADC1BUFA	AN0, Sample 11	AN0, Sample 27
ADC1BUFB	AN0, Sample 12	AN0, Sample 28
ADC1BUFC	AN0, Sample 13	AN0, Sample 29
ADC1BUFD	AN0, Sample 14	AN0, Sample 30
ADC1BUFE	AN0, Sample 15	AN0, Sample 31
ADC1BUFF	AN0, Sample 16	AN0, Sample 32

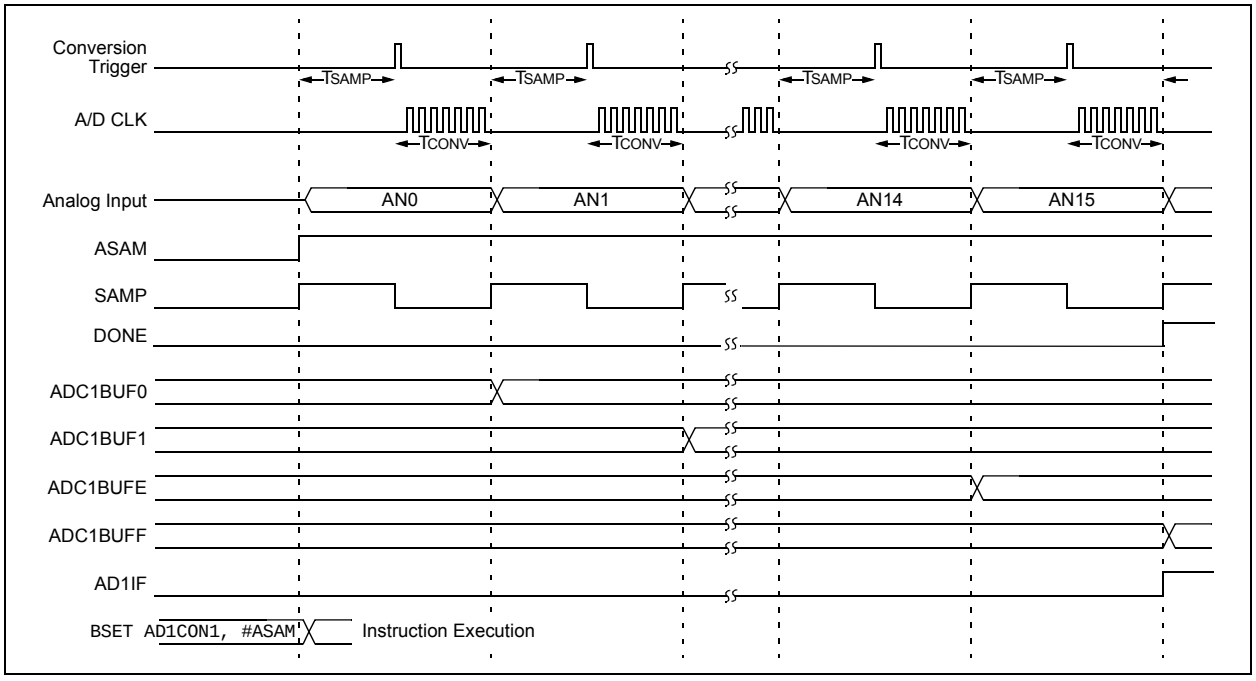
22.8.2.2 A/D Conversions While Scanning Through All Analog Inputs

Figure 22-17 and Example 22-5 illustrate a typical setup, where all available analog input channels are sampled and converted. In this instance, 16 analog inputs are assumed. The set CSCNA bit specifies scanning of the A/D inputs to the S/H positive input.

Other conditions are similar to those located in Section **Section 22.8.2.1 "Sampling and Converting a Single Channel Multiple Times"**.

Initially, the AN0 input is sampled and converted. The result is stored in the ADCBUFn buffer. Then, the AN1 input is sampled and converted. This process of scanning the inputs repeats 16 times, until the buffer is full, and then the module generates an interrupt. The entire process will then repeat.

FIGURE 22-17: SCANNING ALL 16 INPUTS PER SINGLE INTERRUPT



EXAMPLE 22-5: SCANNING AND CONVERTING ALL 16 CHANNELS PER SINGLE INTERRUPT

A/D Configuration:

- Select Any Channel for S/H+ Input (CH0SA<4:0> = xxxxx)
- Select V_{R-} for S/H- Input (CH0NA<2:0> = 000)
- Use Only MUX A for Sampling (ALTS = 0)
- Configure MUX A for Input Scan (CSCNA = 1)
- Include All Analog Channels in Scanning (AD1CSSL = 1111 1111 1111 1111)
- Set AD1IF on Every 16th Sample (SMPI<4:0> = 01111)
- Configure Buffers for Single, 16-Word Results (BUFM = 0)

Operational Sequence:

1. Sample MUX A Input AN0; Convert and Write to Buffer 0h.
2. Sample MUX A Input AN1; Convert and Write to Buffer 1h.
3. Sample MUX A Input AN2; Convert and Write to Buffer 2h.
4. Sample MUX A Input AN3; Convert and Write to Buffer 3h.
5. Sample MUX A Input AN4; Convert and Write to Buffer 4h.
6. Sample MUX A Input AN5; Convert and Write to Buffer 5h.
7. Sample MUX A Input AN6; Convert and Write to Buffer 6h.
8. Sample MUX A Input AN7; Convert and Write to Buffer 7h.
9. Sample MUX A Input AN8; Convert and Write to Buffer 8h.
10. Sample MUX A Input AN9; Convert and Write to Buffer 9h.
11. Sample MUX A Input AN10; Convert and Write to Buffer Ah.
12. Sample MUX A Input AN11; Convert and Write to Buffer Bh.
13. Sample MUX A Input AN12; Convert and Write to Buffer Ch.
14. Sample MUX A Input AN13; Convert and Write to Buffer Dh.
15. Sample MUX A Input AN14; Convert and Write to Buffer Eh.
16. Sample MUX A Input AN15; Convert and Write to Buffer Fh.
17. Set AD1IF Flag (and generate interrupt, if enabled).
18. Repeat (1-16) after Return from Interrupt.

Results Stored in Buffer (after 2 cycles):

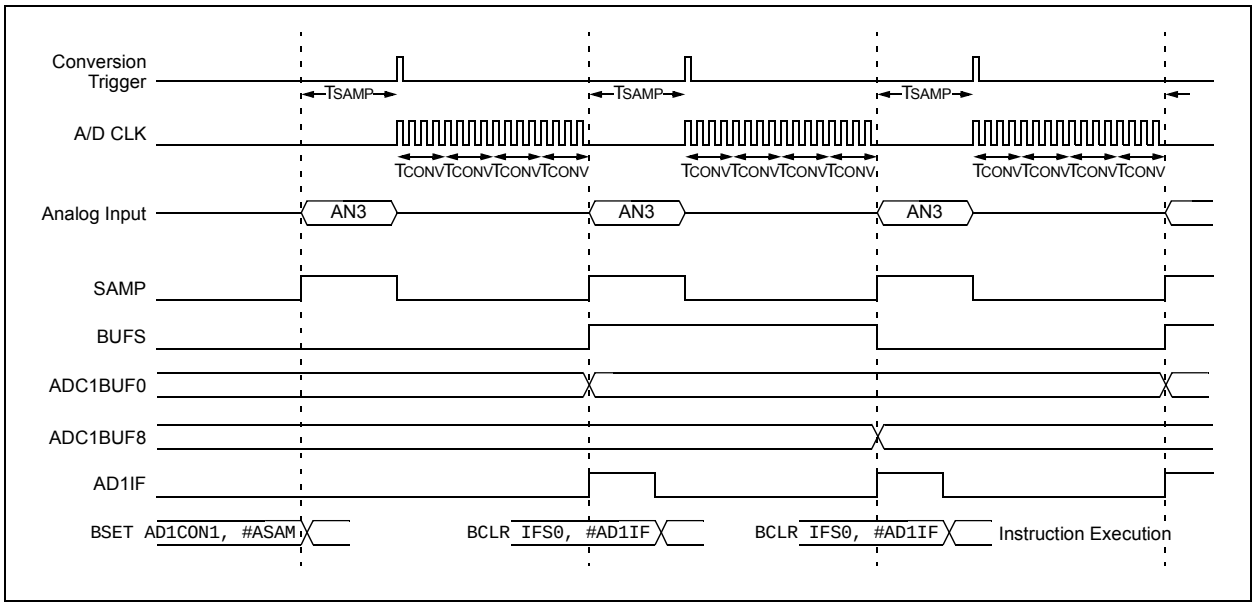
Buffer Address	Buffer Contents at 1st AD1IF Event	Buffer Contents at 2nd AD1IF Event
ADC1BUF0	Sample 1 (AN0, Sample 1)	Sample 17 (AN0, Sample 2)
ADC1BUF1	Sample 2 (AN1, Sample 1)	Sample 18 (AN1, Sample 2)
ADC1BUF2	Sample 3 (AN2, Sample 1)	Sample 19 (AN2, Sample 2)
ADC1BUF3	Sample 4 (AN3, Sample 1)	Sample 20 (AN3, Sample 2)
ADC1BUF4	Sample 5 (AN4, Sample 1)	Sample 21 (AN4, Sample 2)
ADC1BUF5	Sample 6 (AN5, Sample 1)	Sample 22 (AN5, Sample 2)
ADC1BUF6	Sample 7 (AN6, Sample 1)	Sample 23 (AN6, Sample 2)
ADC1BUF7	Sample 8 (AN7, Sample 1)	Sample 24 (AN7, Sample 2)
ADC1BUF8	Sample 9 (AN8, Sample 1)	Sample 25 (AN8, Sample 2)
ADC1BUF9	Sample 10 (AN9, Sample 1)	Sample 26 (AN9, Sample 2)
ADC1BUF10	Sample 11 (AN10, Sample 1)	Sample 27 (AN10, Sample 2)
ADC1BUF11	Sample 12 (AN11, Sample 1)	Sample 28 (AN11, Sample 2)
ADC1BUF12	Sample 13 (AN12, Sample 1)	Sample 29 (AN12, Sample 2)
ADC1BUF13	Sample 14 (AN13, Sample 1)	Sample 30 (AN13, Sample 2)
ADC1BUF14	Sample 15 (AN14, Sample 1)	Sample 31 (AN14, Sample 2)
ADC1BUF15	Sample 16 (AN15, Sample 1)	Sample 32 (AN15, Sample 2)

22.8.3 USING DUAL BUFFERS

Figure 22-18 and Example 22-6 demonstrate using dual buffers and alternating the buffer fill.

Setting the BUFM bit enables dual buffers. In this example, an interrupt is generated after each sample. The BUFM setting does not affect other operational parameters. First, the conversion sequence starts filling the buffer at ADCBUF0. After the first interrupt occurs, the buffer begins to fill at ADCBUF8. The BUFS Status bit is toggled after each interrupt.

FIGURE 22-18: CONVERTING A SINGLE CHANNEL, ONCE PER INTERRUPT, USING DUAL, 8-WORD BUFFERS



EXAMPLE 22-6: CONVERTING A SINGLE CHANNEL, ONCE PER INTERRUPT, DUAL BUFFER MODE

A/D Configuration:

- Select AN3 for S/H+ Input (CH0SA<4:0> = 00011)
- Select VR- for S/H- Input (CH0NA<2:0> = 000)
- Configure for No Input Scan (CSCNA = 0)
- Use Only MUX A for Sampling (ALTS = 0)
- Set AD1IF on Every Sample (SMPI<4:0> = 00000)
- Configure Buffer as Dual, 8-Word Segments (BUFM = 1)

Operational Sequence:

1. Sample MUX A Input, AN3; Convert and Write to Buffer 0h.
2. Set AD1IF Flag (and generate interrupt, if enabled); Write Access Automatically Switches to Alternate Buffer.
3. Sample MUX A Input, AN3; Convert and Write to Buffer 8h.
4. Set AD1IF Flag (and generate interrupt, if enabled); Write Access Automatically Switches to Alternate Buffer.
5. Repeat (1-4).

Results Stored in Buffer (after 2 cycles):

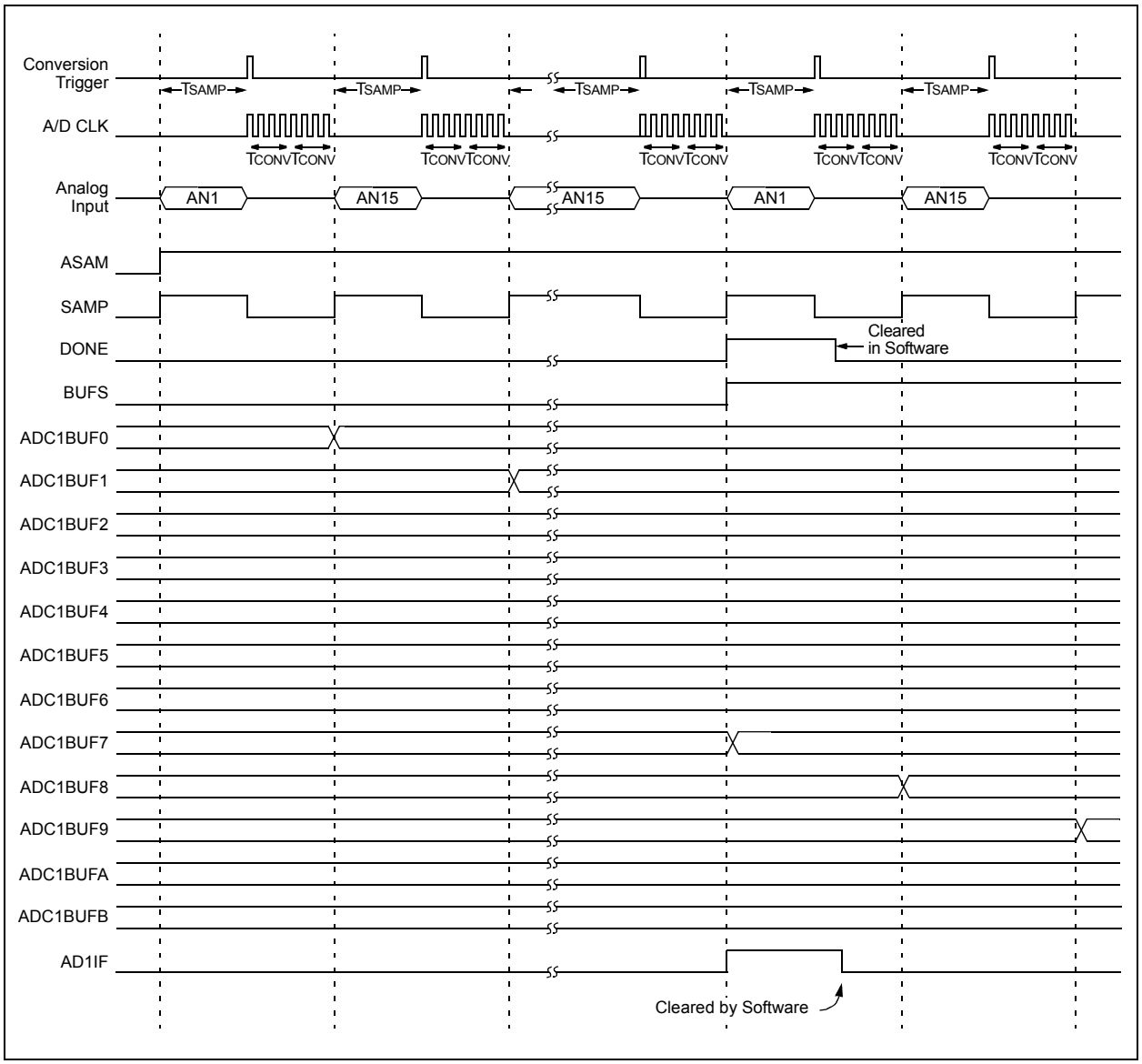
Buffer Address	Buffer Contents at 1st AD1IF Event	Buffer Contents at 2nd AD1IF Event
ADC1BUF0	Sample 1 (AN3, Sample 1)	(undefined)
ADC1BUF1	(undefined)	(undefined)
ADC1BUF2	(undefined)	(undefined)
ADC1BUF3	(undefined)	(undefined)
ADC1BUF4	(undefined)	(undefined)
ADC1BUF5	(undefined)	(undefined)
ADC1BUF6	(undefined)	(undefined)
ADC1BUF7	(undefined)	(undefined)
ADC1BUF8	(undefined)	Sample 2 (AN3, Sample 2)
ADC1BUF9	(undefined)	(undefined)
ADC1BUFA	(undefined)	(undefined)
ADC1BUFB	(undefined)	(undefined)
ADC1BUFC	(undefined)	(undefined)
ADC1BUFD	(undefined)	(undefined)
ADC1BUFE	(undefined)	(undefined)
ADC1BUFF	(undefined)	(undefined)

22.8.3.1 Using Alternating MUX A and MUX B Input Selections

Figure 22-19 and Example 22-7 demonstrate alternate sampling of the inputs assigned to MUX A and MUX B. Setting the ALTS bit enables alternating input selections. The first sample uses the MUX A inputs specified by the CH0SAx and CH0NAx bits. The next sample uses the MUX B inputs, specified by the CH0SBx and CH0NBx bits.

This example also demonstrates use of the dual, 8-word buffers. An interrupt occurs after every 8th sample, resulting in filling eight words into the buffer on each interrupt.

FIGURE 22-19: CONVERTING TWO INPUTS USING ALTERNATING INPUT SELECTIONS



EXAMPLE 22-7: CONVERTING TWO INPUTS BY ALTERNATING MUX A AND MUX B

A/D Configuration:

- Select AN1 for MUX A S/H+ Input (CH0SA<4:0> = 00001)
- Select V_{R-} for MUX A S/H- Input (CH0NA<2:0> = 000)
- Configure for No Input Scan (CSCNA = 0)
- Select AN15 for MUX B S/H+ Input (CH0SB<4:0> = 11111)
- Select V_{R-} for MUX B S/H- Input (CH0NB<2:0> = 000)
- Alternate MUX A and MUX B for Sampling (ALTS = 1)
- Set AD1IF on Every 8th Sample (SMPI<4:0> = 00111)
- Configure Buffer as Two, 8-Word Segments (BUFM = 1)

Operational Sequence:

1. Sample MUX A Input AN1; Convert and Write to Buffer 0h.
2. Sample MUX B Input AN15; Convert and Write to Buffer 1h.
3. Sample MUX A Input AN1; Convert and Write to Buffer 2h.
4. Sample MUX B Input AN15; Convert and Write to Buffer 3h.
5. Sample MUX A Input AN1; Convert and Write to Buffer 4h.
6. Sample MUX B Input AN15; Convert and Write to Buffer 5h.
7. Sample MUX A Input AN1; Convert and Write to Buffer 6h.
8. Sample MUX B Input AN15; Convert and Write to Buffer 7h.
9. Set AD1IF Flag (and generate interrupt, if enabled); Write Access Automatically Switches to Alternate Buffer.
10. Repeat (1-9); Resume Writing to Buffer with Buffer 8h (first address of alternate buffer).

Results Stored in Buffer (after 2 cycles):

Buffer Address	Buffer Contents at 1st AD1IF Event	Buffer Contents at 2nd AD1IF Event
ADC1BUF0	Sample 1 (AN1, Sample 1)	(undefined)
ADC1BUF1	Sample 2 (AN15, Sample 1)	(undefined)
ADC1BUF2	Sample 3 (AN1, Sample 2)	(undefined)
ADC1BUF3	Sample 4 (AN15, Sample 2)	(undefined)
ADC1BUF4	Sample 5 (AN1, Sample 3)	(undefined)
ADC1BUF5	Sample 6 (AN15, Sample 3)	(undefined)
ADC1BUF6	Sample 7 (AN1, Sample 4)	(undefined)
ADC1BUF7	Sample 8 (AN15, Sample 4)	(undefined)
ADC1BUF8	(undefined)	Sample 9 (AN1, Sample 5)
ADC1BUF9	(undefined)	Sample 10 (AN15, Sample 5)
ADC1BUFA	(undefined)	Sample 11 (AN1, Sample 6)
ADC1BUFB	(undefined)	Sample 12 (AN15, Sample 6)
ADC1BUFC	(undefined)	Sample 13 (AN1, Sample 7)
ADC1BUFD	(undefined)	Sample 14 (AN15, Sample 7)
ADC1BUFE	(undefined)	Sample 15 (AN1, Sample 8)
ADC1BUFF	(undefined)	Sample 16 (AN15, Sample 8)

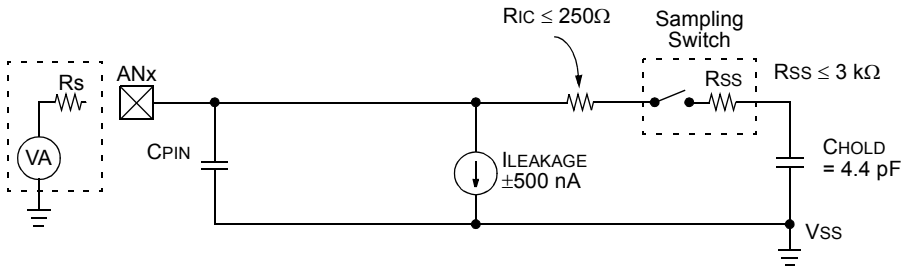
22.9 A/D Sampling Requirements

The Analog Input model of the 12-bit A/D Converter is shown in Figure 22-20. The total sampling time for the A/D is a function of the holding capacitor charge time.

For the A/D Converter to meet its specified accuracy, the charge holding capacitor (CHOLD) must be allowed to fully charge to the voltage level on the analog input pin. The source impedance (RS), the interconnect impedance (RIC) and the internal sampling switch (RSS) impedance combine to directly affect the time required to charge CHOLD. The combined impedance of the analog sources must, therefore, be small enough to fully charge the holding capacitor within the chosen sample time. To minimize the effects of pin leakage currents on the accuracy of the A/D Converter, the maximum recommended source impedance, RS, is 2.5 k. After the analog input channel is selected (changed), this sampling function must be completed prior to starting the conversion. The internal holding capacitor will be in a discharged state prior to each sample operation.

At least 1 TAD time period should be allowed between conversions for the sample time. For more details, see **Section 30.0 "Electrical Specifications"**.

FIGURE 22-20: 12-BIT A/D CONVERTER ANALOG INPUT MODEL



Legend:		
	CPIN	= Input Capacitance
	VT	= Threshold Voltage
	ILEAKAGE	= Leakage Current at the Pin due to Various Junctions
	RIC	= Interconnect Resistance
	RSS	= Sampling Switch Resistance
	CHOLD	= Sample/Hold Capacitance (from DAC)

Note: CPIN value depends on device package and is not tested. The effect of the CPIN is negligible if Rs ≤ 5 kΩ.

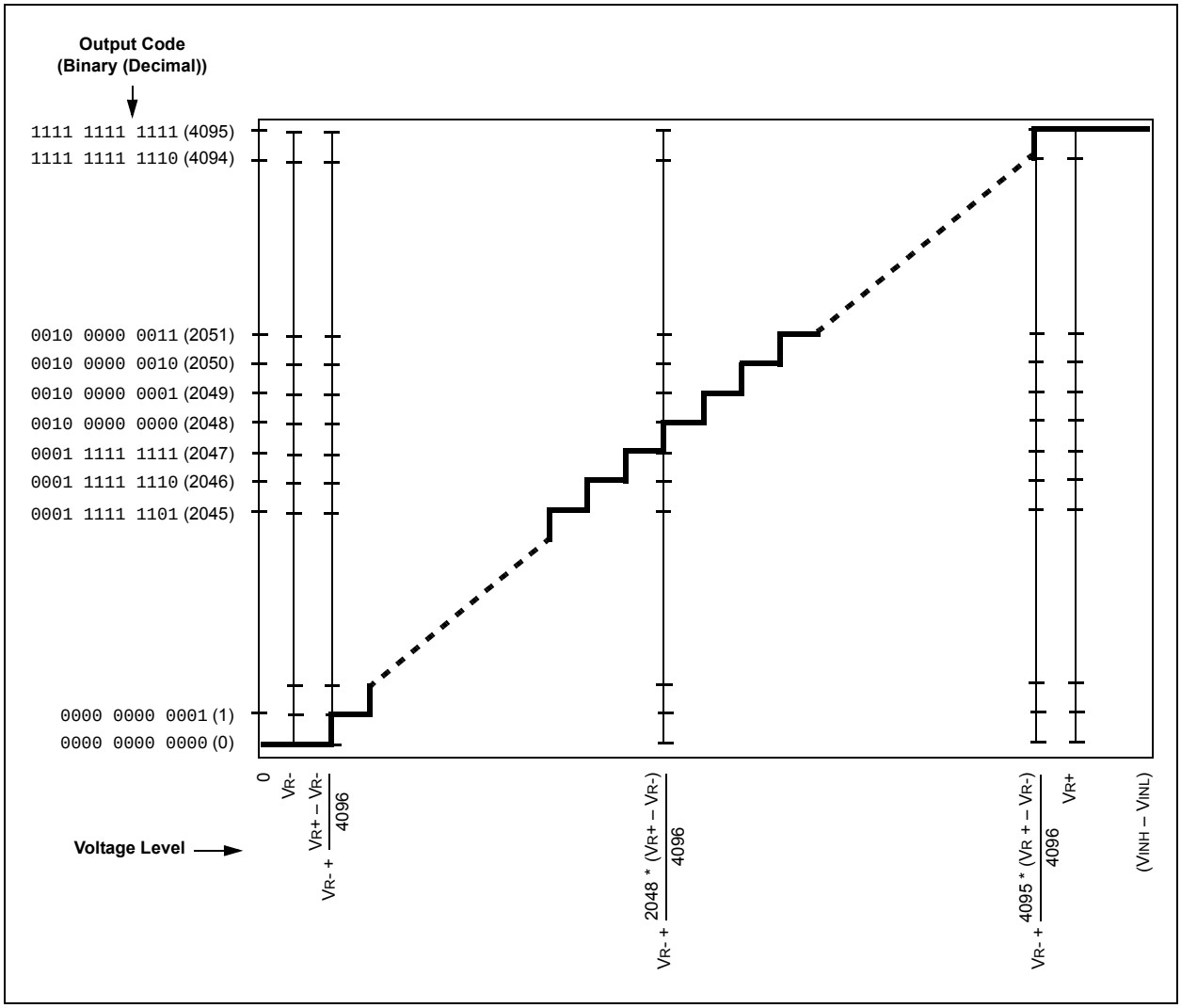
22.10 Transfer Functions

The transfer functions of the A/D Converter, in 12-bit and 10-bit resolution, are shown in Figure 22-21 and Figure 22-22, respectively. In both cases, the difference of the input voltages, (VINH - VINL), is compared to the reference, ((VR+) - (VR-)).

For the 12-bit transfer function:

- The first code transition occurs when the input voltage is ((VR+) - (VR-))/4096 or 1.0 LSb.
- The '0000 0000 0001' code is centered at VR- + (1.5 * ((VR+) - (VR-)) / 4096).
- The '0010 0000 0000' code is centered at VREFL + (2048.5 * ((VR+) - (VR-)) /4096).
- An input voltage less than VR- + (((VR-) - (VR-)) / 4096) converts as '0000 0000 0000'.
- An input voltage greater than (VR-) + (4096 ((VR+) - (VR-))/4096) converts as '1111 1111 1111'.

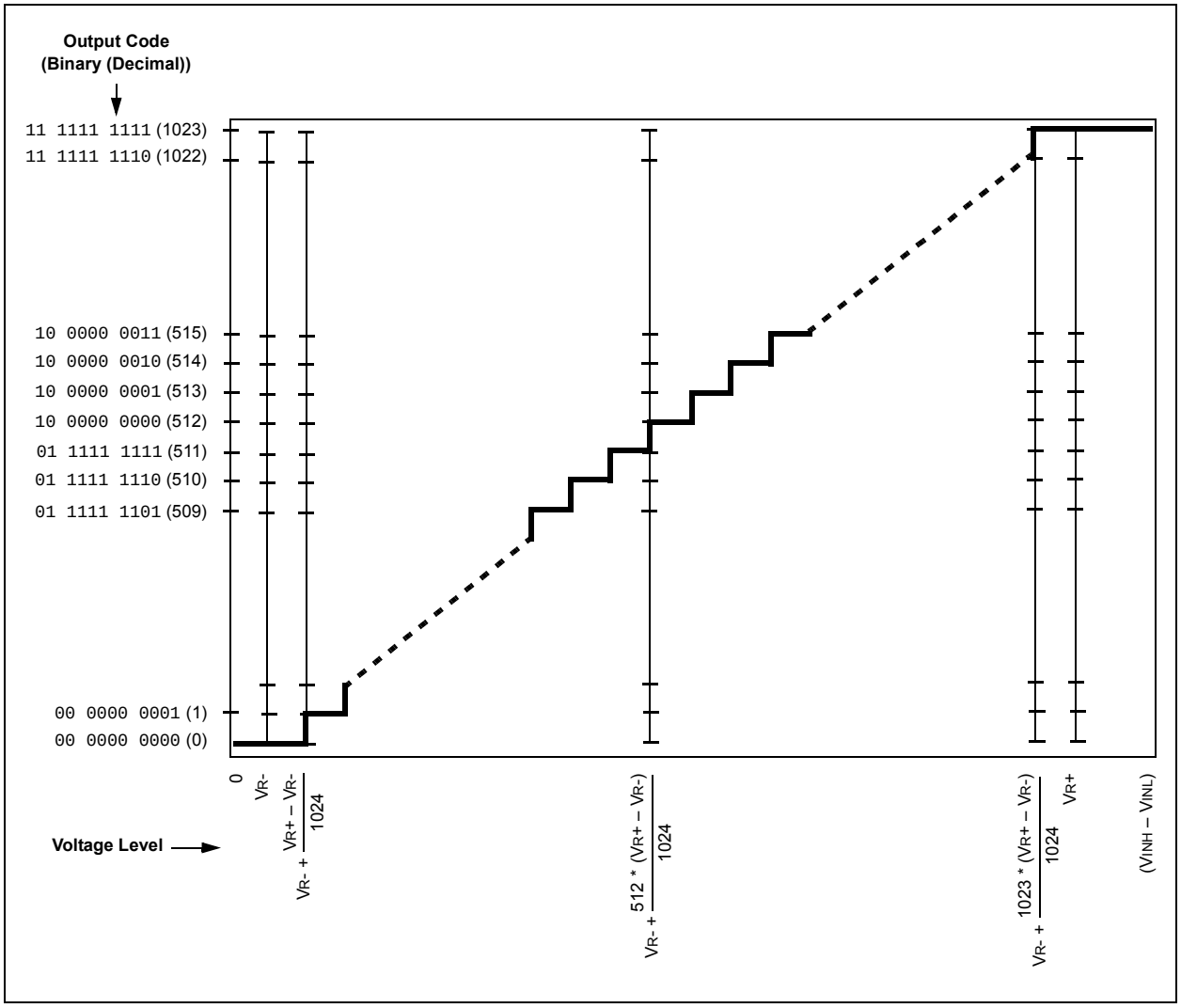
FIGURE 22-21: 12-BIT A/D TRANSFER FUNCTION



For the 10-bit transfer function (when 10-bit resolution is available):

- The first code transition occurs when the input voltage is ((VR+) - (VR-))/1024 or 1.0 LSb.
- The '00 0000 0001' code is centered at VR- + (1.5 * (((VR+) - (VR-)) / 1024).
- The '10 0000 0000' code is centered at VREFL + (512.5 * (((VR+) - (VR-)) /1024).
- An input voltage less than VR- + (((VR-) - (VR-)) / 1024) converts as '00 0000 0000'.
- An input voltage greater than (VR-) + ((1023 (VR+)) - (VR-))/1024) converts as '11 1111 1111'.

FIGURE 22-22: 10-BIT A/D TRANSFER FUNCTION



22.11 Operation During Sleep and Idle Modes

Sleep and Idle modes are useful for minimizing conversion noise because the digital activity of the CPU, buses and other peripherals is minimized.

22.11.1 CPU SLEEP MODE WITHOUT RC A/D CLOCK

When the device enters Sleep mode, all clock sources to the module are shut down and stay at logic '0'.

If Sleep occurs in the middle of a conversion, the conversion is aborted unless the A/D is clocked from its internal RC clock generator. The converter will not resume a partially completed conversion on exiting from Sleep mode.

Register contents are not affected by the device entering or leaving Sleep mode.

22.11.2 CPU SLEEP MODE WITH RC A/D CLOCK

The A/D module can operate during Sleep mode if the A/D clock source is set to the internal A/D RC oscillator (ADRC = 1). This eliminates digital switching noise from the conversion. When the conversion is completed, the DONE bit will be set and the result is loaded into the A/D Result Buffer n, ADCBUFn.

If the A/D interrupt is enabled (ADIE = 1), the device will wake-up from Sleep when the A/D interrupt occurs. Program execution will resume at the A/D Interrupt Service Routine (ISR). After the ISR completes execution will continue from the instruction after the SLEEP instruction that placed the device in Sleep mode.

If the A/D interrupt is not enabled, the A/D module will then be turned off, although the ADON bit will remain set.

To minimize the effects of digital noise on the A/D module operation, the user should select a conversion trigger source that ensures the A/D conversion will take place in Sleep mode. The automatic conversion trigger option can be used for sampling and conversion in Sleep (SSRC<3:0> = 0111). To use the automatic conversion option, the ADON bit should be set in the instruction prior to the SLEEP instruction.

Note:	For the A/D module to operate in Sleep, the A/D clock source must be set to RC (ADRC = 1).

22.11.3 A/D OPERATION DURING CPU IDLE MODE

The module will continue normal operation when the device enters Idle mode. If the A/D interrupt is enabled (ADIE = 1), the device will wake-up from Idle mode when the A/D interrupt occurs. If the respective global interrupt enable bit(s) are also set, program execution will resume at the A/D Interrupt Service Routine (ISR). After the ISR completes, execution will continue from the instruction after the SLEEP instruction that placed the device in Idle mode.

22.11.4 PERIPHERAL MODULE DISABLE (PMD) REGISTER

The Peripheral Module Disable (PMD) registers provide a method to disable the A/D module by stopping all clock sources supplied to that module. When a peripheral is disabled via the appropriate PMDx control bit, the peripheral is in a minimum power consumption state. The control and STATUS registers associated with the peripheral will also be disabled, so writes to those registers will have no effect and read values will be invalid. The A/D module is enabled only when the ADCMD bit in the PMD3 register is cleared.

22.12 Design Tips

Question 1: How can I optimize the system performance of the A/D Converter?

Answer: There are three main things to consider in optimizing A/D performance:

1. Make sure you are meeting all of the timing specifications. If you are turning the module off and on, there is a minimum delay you must wait before taking a sample. If you are changing input channels, there is a minimum delay you must wait for this as well, and finally, there is TAD, which is the time selected for each bit conversion. This is selected in AD1CON3 and should be within a certain range, as specified in **Section 30.0 "Electrical Specifications"**. If TAD is too short, the result may not be fully con- verted before the con- version is terminated, and if TAD is made too long, the voltage on the sampling capacitor can decay before the conversion is complete. These timing specifications are provided in the "Electrical Characteristics" section of the device data sheets.
2. Often, the source impedance of the analog signal is high (greater than 2.5 kΩ), so the current drawn from the source by leakage, and to charge the sample capacitor, can affect accuracy. If the input signal does not change too quickly, try putting a 0.1 uF capacitor on the analog input. This capacitor will charge to the analog voltage being sampled and supply the instantaneous current needed to charge the internal holding capacitor.
3. Put the device into Sleep mode before the start of the A/D conversion. The RC clock source selection is required for conversions in Sleep mode. This technique increases accuracy, because digital noise from the CPU and other peripherals is minimized.

Question 2: Do you know of a good reference on A/D Converters?

Answer: A good reference for understanding A/D conversions is the "Analog-Digital Conversion

Handbook third edition, published by Prentice Hall (ISBN 0-13-03-2848-0).

Question 3: My combination of channels/samples and samples/interrupt is greater than the size of the buffer. What will happen to the buffer?

Answer: This configuration is not recommended. The buffer will contain unknown results.

22.13 Related Application Notes

This section lists application notes that are related to this section of the data sheet. These application notes may not be written specifically for the PIC18F device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the 12-Bit A/D Converter with Threshold Detect module are:

AN546, Using the Analog-to-Digital (A/D) Converter (DS00546)

AN557, Four-Channel Digital Voltmeter with Display and Keyboard (DS00557)

AN693, Understanding A/D Converter Performance Specifications (DS00693)

Note:	Visit the Microchip web site (www.microchip.com) for additional application notes and code examples for the PIC18F family of devices.

23.0 COMPARATOR MODULE

The analog comparator module contains three comparators that can be independently configured in a variety of ways. The inputs can be selected from the analog inputs and two internal voltage references. The digital outputs are available at the pin level, via PPS-Lite, and can also be read through the control register. Multiple output and interrupt event generations are also available. A generic single comparator from the module is shown in Figure 23-1.

Key features of the module includes:

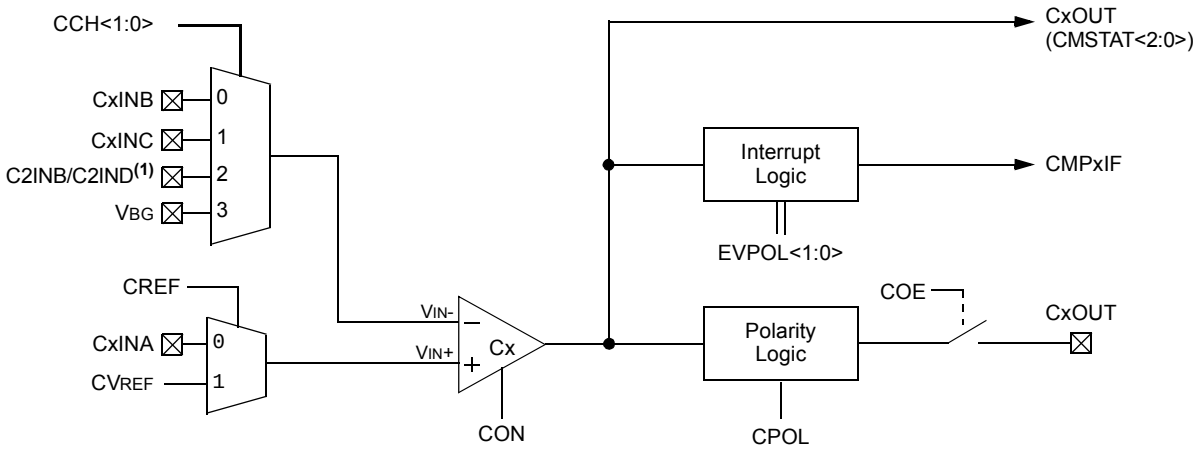
- Independent comparator control
- Programmable input configuration
- Output to both pin and register levels
- Programmable output polarity
- Independent interrupt generation for each comparator with configurable interrupt-on-change

23.1 Registers

The CMxCON registers (CM1CON, CM2CON and CM3CON) select the input and output configuration for each comparator, as well as the settings for interrupt generation (see Register 23-1).

The CMSTAT register (Register 23-2) provides the output results of the comparators. The bits in this register are read-only.

FIGURE 23-1: COMPARATOR SIMPLIFIED BLOCK DIAGRAM



Note 1: Comparator 1 and Comparator 3 use C2INB as an input to the inverted terminal. Comparator 2 uses C2IND as an input to the inverted terminal.

REGISTER 23-1: CMxCON: COMPARATOR CONTROL x REGISTER

R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
CON	COE	CPOL	EVPOL1	EVPOL0	CREF	CCH1	CCH0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CON:** Comparator Enable bit
1 = Comparator is enabled
0 = Comparator is disabled

bit 6 **COE:** Comparator Output Enable bit
1 = Comparator output is present on the CxOUT pin
0 = Comparator output is internal only

bit 5 **CPOL:** Comparator Output Polarity Select bit
1 = Comparator output is inverted
0 = Comparator output is not inverted

bit 4-3 **EVPOL<1:0>:** Interrupt Polarity Select bits
11 = Interrupt generation on any change of the output(1)
10 = Interrupt generation only on high-to-low transition of the output
01 = Interrupt generation only on low-to-high transition of the output
00 = Interrupt generation is disabled

bit 2 **CREF:** Comparator Reference Select bit (non-inverting input)
1 = Non-inverting input connects to internal CV_{REF} voltage
0 = Non-inverting input connects to CxINA pin

bit 1-0 **CCH<1:0>:** Comparator Channel Select bits
11 = Inverting input of comparator connects to V_{BG}
10 = Inverting input of comparator connects to C2INB pin
01 = Inverting input of comparator connects to CxINC pin
00 = Inverting input of comparator connects to CxINx pin(2)

Note 1: The CMPxIF is automatically set any time this mode is selected and must be cleared by the application after the initial configuration.

2: Comparator 1 and Comparator 3 use C2INB as an input to the inverting terminal. Comparator 2 uses C2IND as an input to the inverting terminal.

REGISTER 23-2: CMSTAT: COMPARATOR STATUS REGISTER

U-0	U-0	U-0	U-0	U-0	R-x	R-x	R-x
—	—	—	—	—	C3OUT	C2OUT	C1OUT
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-3 **Unimplemented:** Read as '0'

bit 2-0 **C3OUT:C1OUT:** Comparator x Status bits

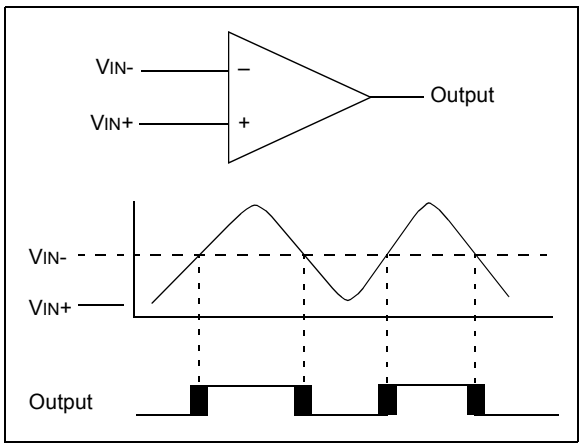
If CPOL (CMxCON<5>)= 0 (non-inverted polarity):
1 = Comparator x's VIN+ > VIN-
0 = Comparator x's VIN+ < VIN-

CPOL = 1 (inverted polarity):
1 = Comparator x's VIN+ < VIN-
0 = Comparator x's VIN+ > VIN-

23.2 Comparator Operation

A single comparator is shown in Figure 23-2, along with the relationship between the analog input levels and the digital output. When the analog input at VIN+ is less than the analog input, VIN-, the output of the comparator is a digital low level. When the analog input at VIN+ is greater than the analog input, VIN-, the output of the comparator is a digital high level. The shaded areas of the output of the comparator in Figure 23-2 represent the uncertainty due to input offsets and response time.

FIGURE 23-2: SINGLE COMPARATOR



23.3 Comparator Response Time

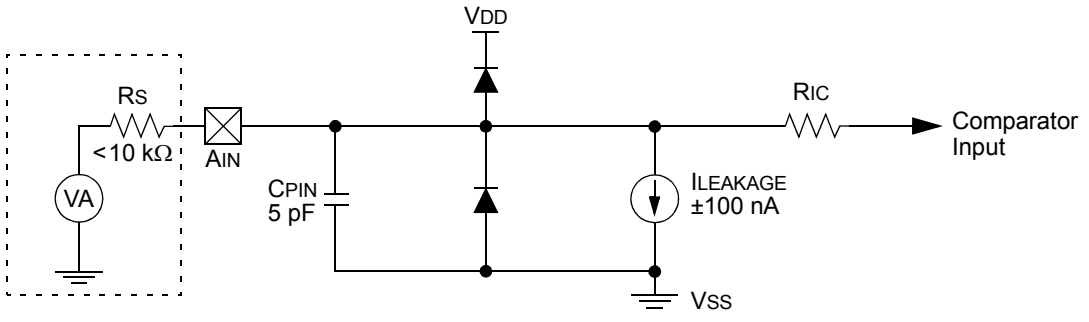
Response time is the minimum time, after selecting a new reference voltage or input source, before the comparator output has a valid level. The response time of the comparator differs from the settling time of the voltage reference. Therefore, both of these times must be considered when determining the total response to a comparator input change. Otherwise, the maximum delay of the comparators should be used (see **Section 30.0 "Electrical Specifications"**).

23.4 Analog Input Connection Considerations

A simplified circuit for an analog input is shown in Figure 23-3. Since the analog pins are connected to a digital output, they have reverse biased diodes to VDD and VSS. The analog input, therefore, must be between VSS and VDD. If the input voltage deviates from this range by more than 0.6V in either direction, one of the diodes is forward biased and a latch-up condition may occur.

A maximum source impedance of 10 kΩ is recommended for the analog sources. Any external component connected to an analog input pin, such as a capacitor or a Zener diode, should have very little leakage current.

FIGURE 23-3: COMPARATOR ANALOG INPUT MODEL



Legend:

CPIN	=	Input Capacitance
VT	=	Threshold Voltage
ILEAKAGE	=	Leakage Current at the pin due to various junctions
RIC	=	Interconnect Resistance
RS	=	Source Impedance
VA	=	Analog Voltage

23.5 Comparator Control and Configuration

Each comparator has up to eight possible combinations of inputs: up to four external analog inputs and one of two internal voltage references.

All of the comparators allow a selection of the signal from pin, CxINA, or the voltage from the Comparator Reference (CVREF) on the non-inverting channel. This is compared to either CxINB, CxINC, C2IND or the microcontroller's fixed internal reference voltage (VBG, 1.2V nominal) on the inverting channel. The comparator inputs and outputs are tied to fixed I/O pins, defined in Table 23-1. The available comparator configurations and their corresponding bit settings are shown in Figure 23-4.

TABLE 23-1: COMPARATOR INPUTS AND OUTPUTS

Comparator	Input or Output	I/O Pin(†)
1	C1INA (VIN+)	RA5/RF6
	C1INB (VIN-)	RF5
	C1INC (VIN-)	RH6(2)
	C2INB(VIN-)	RF2
	CVREF (VIN+)	RF5
	C1OUT	RPn(1)
2	C2INA (VIN+)	RA5
	C2INB (VIN-)	RF2
	C2INC (VIN-)	RH4(2)
	C2IND (VIN-)	RH5(2)
	CVREF (VIN+)	RF5
	C2OUT	RPn(1)
3	C3INA (VIN+)	RA5/RG2
	C3INB (VIN-)	RG3
	C2INB (VIN-)	RF2
	C3INC (VIN-)	RG4
	CVREF (VIN+)	RF5
	C3OUT	RPn(1)

† The I/O pin is dependent on package type.

Note 1: These pins are remappable I/Os.

2: These pins are not available on 64-pin devices.

23.5.1 COMPARATOR ENABLE AND INPUT SELECTION

Setting the CON bit of the CMxCON register (CMxCON<7>) enables the comparator for operation. Clearing the CON bit disables the comparator, resulting in minimum current consumption.

The CCH<1:0> bits in the CMxCON register (CMxCON<1:0>) direct either one of three analog input pins, or the Internal Reference Voltage (V_{BG}), to the comparator, V_{IN}-. Depending on the comparator operating mode, either an external or internal voltage reference may be used.

The analog signal present at V_{IN}- is compared to the signal at V_{IN}+ and the digital output of the comparator is adjusted accordingly.

The external reference is used when CREF = 0 (CMxCON<2>) and V_{IN}+ is connected to the CxINA pin. When external voltage references are used, the comparator module can be configured to have the reference sources externally. The reference signal must be between V_{SS} and V_{DD} and can be applied to either pin of the comparator.

The comparator module also allows the selection of an internally generated voltage reference from the Comparator Voltage Reference (CV_{REF}) module. This module is described in more detail in **Section 24.0 "Comparator Voltage Reference Module"**. The reference from the comparator voltage reference module is only available when CREF = 1. In this mode, the internal voltage reference is applied to the comparator's V_{IN}+ pin.

Note:	The comparator input pin selected by CCH<1:0> must be configured as an input by setting both the corresponding TRISx bit and the corresponding ANSELx bit in the ANCONx register.

23.5.2 COMPARATOR ENABLE AND OUTPUT SELECTION

The comparator outputs are read through the CMSTAT register. The CMSTAT<0> bit reads the Comparator 1 output, CMSTAT<2> reads the Comparator 2 output and the CMSTAT<3> bit reads the Comparator 3 output These bits are read-only.

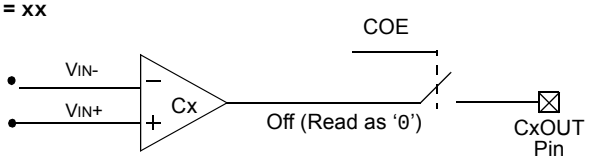
The comparator outputs may also be directly output to the RPn I/O pins by setting the COE bit (CMxCON<6>). When enabled, multiplexers in the output path of the pins switch to the output of the comparator. While in this mode, the respective TRISx bits still function as the digital output enable bits for the RPn I/O pins.

By default, the comparator's output is at logic high whenever the voltage on V_{IN}+ is greater than on V_{IN}-. The polarity of the comparator outputs can be inverted using the CPOL bit (CMxCON<5>).

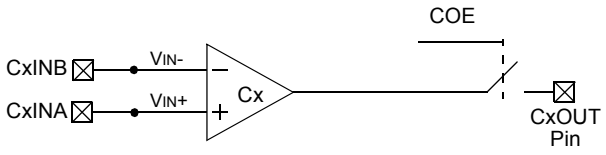
The uncertainty of each of the comparators is related to the input offset voltage and the response time given in the specifications, as discussed in **Section 23.2 "Comparator Operation"**.

FIGURE 23-4: COMPARATOR CONFIGURATIONS

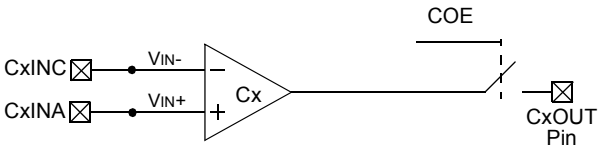
Comparator Off
CON = 0, CREF = x, CCH<1:0> = xx



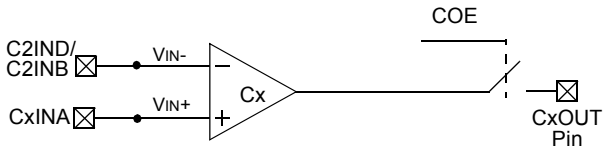
Comparator CxINB > CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 00



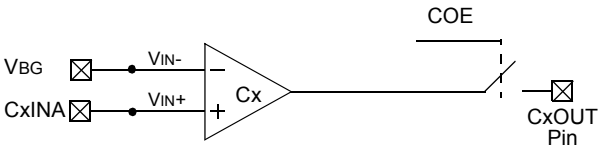
Comparator CxINC > CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 01



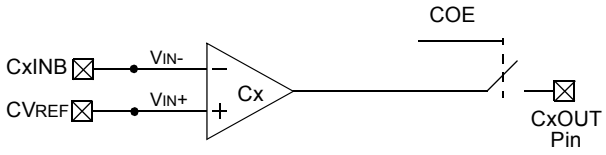
Comparator C2IND/C2INB > CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 10



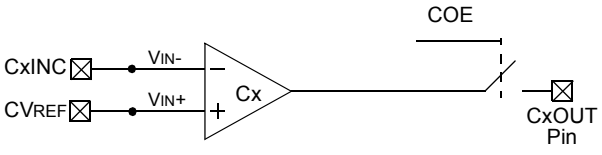
Comparator VBG > CxINA Compare
CON = 1, CREF = 0, CCH<1:0> = 11



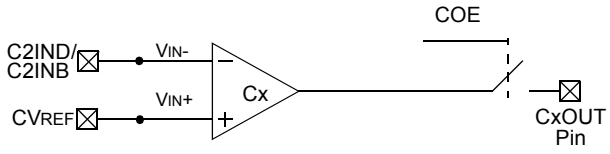
Comparator CxINB > CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 00



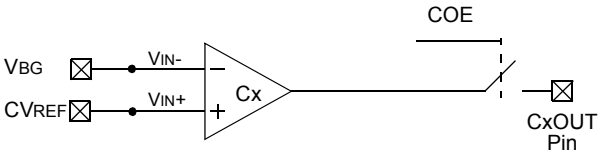
Comparator CxINC > CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 01



Comparator C2IND/C2INB > CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 10



Comparator VBG > CVREF Compare
CON = 1, CREF = 1, CCH<1:0> = 11



Note 1: VBG is the Internal Reference Voltage (see Table 30-14).

23.6 Comparator Interrupts

The comparator interrupt flag is set whenever any of the following occurs:

- Low-to-high transition of the comparator output
- High-to-low transition of the comparator output
- Any change in the comparator output

The comparator interrupt selection is done by the EVPOL<1:0> bits in the CMxCON register (CMxCON<4:3>).

In order to provide maximum flexibility, the output of the comparator may be inverted using the CPOL bit in the CMxCON register (CMxCON<5>). This is functionally identical to reversing the inverting and non-inverting inputs of the comparator for a particular mode.

An interrupt is generated on the low-to-high or high-to-low transition of the comparator output. This mode of interrupt generation is dependent on EVPOL<1:0> in the CMxCON register. When EVPOL<1:0> = 01 or 10, the interrupt is generated on a low-to-high or high-to-low transition of the comparator output. Once the interrupt is generated, it is required to clear the interrupt flag by software.

When EVPOL<1:0> = 11, the comparator interrupt flag is set whenever there is a change in the output value of either comparator. Software will need to maintain information about the status of the output bits, as read from CMSTAT<2:0>, to determine the actual change that occurred.

The CMPxIF<2:0> (PIR6<2:0>) bits are the Comparator Interrupt Flags. The CMPxIF bits must be reset by clearing them. Since it is also possible to write a '1' to this register, a simulated interrupt may be initiated. Table 23-2 shows the interrupt generation with respect to comparator input voltages and EVPOL bit settings.

Both the CMPxIE bits (PIE6<2:0>) and the PEIE bit (INTCON<6>) must be set to enable the interrupt. In addition, the GIE bit (INTCON<7>) must also be set. If any of these bits are clear, the interrupt is not enabled, though the CMPxIF bits will still be set if an interrupt condition occurs.

A simplified diagram of the interrupt section is shown in Figure 23-3.

Note:	CMPxIF will not be set when EVPOL<1:0> = 00.

TABLE 23-2: COMPARATOR INTERRUPT GENERATION

CPOL	EVPOL<1:0>	Comparator Input Change	CxOUT Transition	Interrupt Generated
0	00	VIN+ > VIN-	Low-to-High	No
		VIN+ < VIN-	High-to-Low	No
	01	VIN+ > VIN-	Low-to-High	Yes
		VIN+ < VIN-	High-to-Low	No
	10	VIN+ > VIN-	Low-to-High	No
		VIN+ < VIN-	High-to-Low	Yes
	11	VIN+ > VIN-	Low-to-High	Yes
		VIN+ < VIN-	High-to-Low	Yes
1	00	VIN+ > VIN-	High-to-Low	No
		VIN+ < VIN-	Low-to-High	No
	01	VIN+ > VIN-	High-to-Low	No
		VIN+ < VIN-	Low-to-High	Yes
	10	VIN+ > VIN-	High-to-Low	Yes
		VIN+ < VIN-	Low-to-High	No
	11	VIN+ > VIN-	High-to-Low	Yes
		VIN+ < VIN-	Low-to-High	Yes

23.7 Comparator Operation During Sleep

When a comparator is active and the device is placed in Sleep mode, the comparator remains active and the interrupt is functional, if enabled. This interrupt will wake-up the device from Sleep mode when enabled. Each operational comparator will consume additional current.

To minimize power consumption while in Sleep mode, turn off the comparators (CON = 0) before entering Sleep. If the device wakes up from Sleep, the contents of the CMxCON register are not affected.

23.8 Effects of a Reset

A device Reset forces the CMxCON registers to their Reset state. This forces both comparators and the voltage reference to the OFF state.

24.0 COMPARATOR VOLTAGE REFERENCE MODULE

The comparator voltage reference is a 32-tap resistor ladder network that provides a selectable reference voltage. Although its primary purpose is to provide a reference for the analog comparators, it may also be used independently of them.

A block diagram of the module is shown in Figure 24-1. The resistor ladder is segmented to provide a range of CV_{REF} values and has a power-down function to conserve power when the reference is not being used. The module's supply reference can be provided from either device V_{DD}/V_{SS} or an external voltage reference.

24.1 Configuring the Comparator Voltage Reference

The comparator voltage reference module is controlled through the CVRCONH register (Register 24-1). The comparator voltage reference provides a range of output voltage with 32 levels.

The CVR<4:0> selection bits (CVRCONH<4:0>) offer a range of output voltages. Equation 24-1 shows how the comparator voltage reference is computed.

EQUATION 24-1:

If CVRSS = 1:

$$CV_{REF} = (V_{REF-} + \frac{CVR<4:0>}{32}) \bullet (V_{REF+} - V_{REF-})$$

If CVRSS = 0:

$$CV_{REF} = (AV_{SS} + \frac{CVR<4:0>}{32}) \bullet (AV_{DD} - AV_{SS})$$

The comparator voltage reference supply can come from either V_{DD} and V_{SS}, or the external V_{REF+} and V_{REF-} that are multiplexed with RA3 and RA2. The voltage source is selected by the CVRPSS<1:0> bits (CVRCONL<5:4>).

The settling time of the comparator voltage reference must be considered when changing the CV_{REF} output (see Table 30-13 in **Section 30.0 "Electrical Specifications"**).

REGISTER 24-1: CVRCONH: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER HIGH

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	CVR4	CVR3	CVR2	CVR1	CVR0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4-0 **CVR<4:0>:** Comparator V_{REF} Value Selection $0 \le CVR<4:0> \le 31$ bits

$CV_{REF} = V_{NEGSRC} + (CVR<4:0>/32) \bullet (V_{POSSRC} - V_{NEGSRC})$

REGISTER 24-2: CVRCONL: COMPARATOR VOLTAGE REFERENCE CONTROL REGISTER LOW

R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	R/W-0
CVREN	CVROE	CVRPSS1	CVRPSS0	—	—	—	CVRNSS
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **CVREN:** Comparator Voltage Reference Enable bit
1 = CVREF circuit is powered on
0 = CVREF circuit is powered down

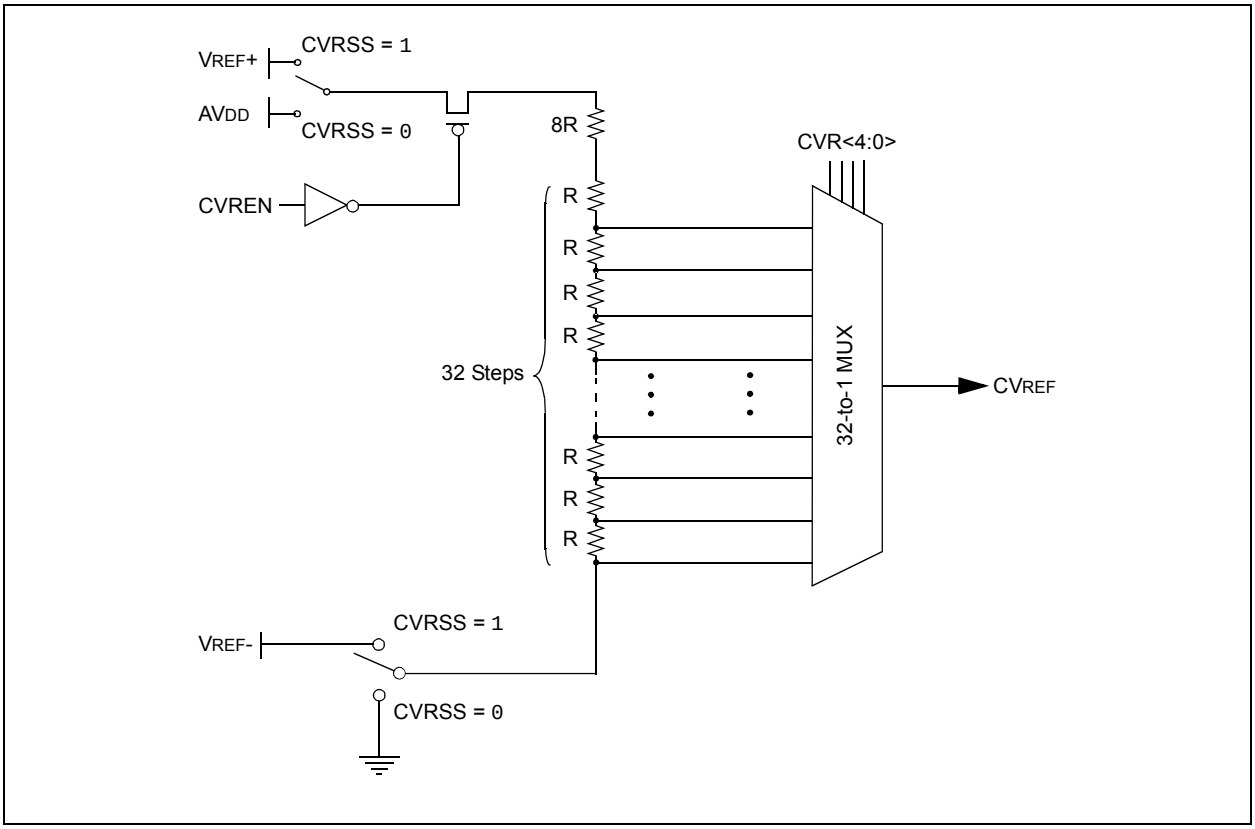
bit 6 **CVROE:** Comparator VREF Output Enable bit
1 = CVREF voltage level is output on CVREF pin
0 = CVREF voltage level is disconnected from CVREF pin

bit 5-4 **CVRPSS<1:0>:** Comparator VREF Positive Source (VPOSSRC) Selection bits
11 = Reserved, do not use. Positive source is floating
10 = VBG (Band gap)
01 = VREF+
00 = AVDD

bit 3-1 **Unimplemented:** Read as '0'

bit 0 **CVRNSS:** Comparator VREF Negative Source (VNEGSRC) Selection bit
01 = VREF-
00 = AVSS

FIGURE 24-1: COMPARATOR VOLTAGE REFERENCE BLOCK DIAGRAM



24.2 Voltage Reference Accuracy/Error

The full range of voltage reference cannot be realized due to the construction of the module. The transistors on the top and bottom of the resistor ladder network (Figure 24-1) keep CVREF from approaching the reference source rails. The voltage reference is derived from the reference source; therefore, the CVREF output changes with fluctuations in that source. The tested absolute accuracy of the voltage reference can be found in **Section 30.0 "Electrical Specifications"**.

24.3 Operation During Sleep

When the device wakes up from Sleep through an interrupt or a Watchdog Timer time-out, the contents of the CVRCON register are not affected. To minimize current consumption in Sleep mode, the voltage reference should be disabled.

24.4 Effects of a Reset

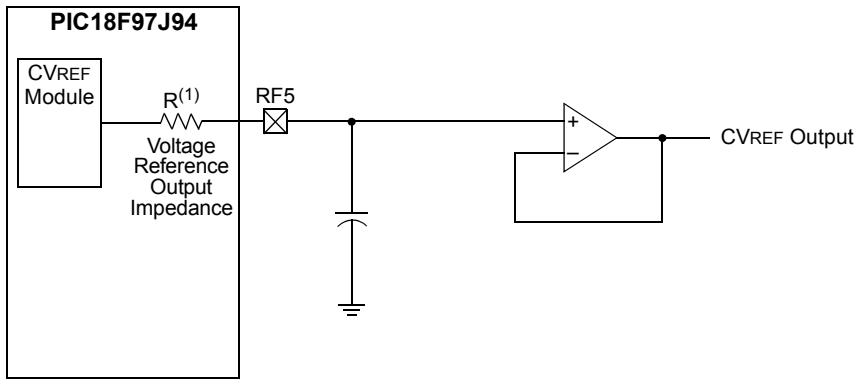
A device Reset disables the voltage reference by clearing bit, CVREN (CVRCONL<7>). This Reset also disconnects the reference from the RF5 pin by clearing bit, CVROE (CVRCONL<6>).

24.5 Connection Considerations

The voltage reference module operates independently of the comparator module. The output of the reference generator may be connected to the RA0 pin if the CVROE bit is set. Enabling the voltage reference output onto RA0, when it is configured as a digital input, will increase current consumption. Connecting RA0 as a digital output with CVRSS enabled will also increase current consumption.

The RA0 pin can be used as a simple D/A output with limited drive capability. Due to the limited current drive capability, a buffer must be used on the voltage reference output for external connections to VREF. Figure 24-2 shows an example buffering technique.

FIGURE 24-2: COMPARATOR VOLTAGE REFERENCE OUTPUT BUFFER EXAMPLE



Note 1: R is dependent upon the Comparator Voltage Reference bits, CVRCONH<4:0>, CVRCONL<5:4> and CVRCONL<0>.

25.0 HIGH/LOW-VOLTAGE DETECT (HLVD)

The PIC18FXXJ94 of devices has a High/Low-Voltage Detect module (HLVD). This is a programmable circuit that sets both a device voltage trip point and the direction of change from that point. If the device experiences an excursion past the trip point in that direction, an interrupt flag is set. If the interrupt is enabled, the program execution branches to the interrupt vector address and the software responds to the interrupt.

The High/Low-Voltage Detect Control register (Register 25-1) completely controls the operation of the HLVD module. This allows the circuitry to be “turned off” by the user under software control, which minimizes the current consumption for the device.

The module’s block diagram is shown in Figure 25-1.

REGISTER 25-1: HLVDCON: HIGH/LOW-VOLTAGE DETECT CONTROL REGISTER

R/W-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
VDIRMAG	BGVST	IRVST	HLVDEN	HLVDL3(1)	HLVDL2(1)	HLVDL1(1)	HLVDL0(1)
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **VDIRMAG:** Voltage Direction Magnitude Select bit
1 = Event occurs when voltage equals or exceeds trip point (HLVDL<3:0>)
0 = Event occurs when voltage equals or falls below trip point (HLVDL<3:0>)

bit 6 **BGVST:** Band Gap Reference Voltages Stable Status Flag bit
1 = Internal band gap voltage references are stable
0 = Internal band gap voltage references are not stable

bit 5 **IRVST:** Internal Reference Voltage Stable Flag bit
1 = Indicates that the voltage detect logic will generate the interrupt flag at the specified voltage range
0 = Indicates that the voltage detect logic will not generate the interrupt flag at the specified voltage range and the HLVD interrupt should not be enabled

bit 4 **HLVDEN:** High/Low-Voltage Detect Power Enable bit
1 = HLVD is enabled
0 = HLVD is disabled

bit 3-0 **HLVDL<3:0>:** Voltage Detection Limit bits(1)
1111 = External analog input is used (input comes from the HLVDIN pin)
1110 = Maximum setting
.
.
.
0100 = Minimum setting

Note 1: For the electrical specifications, see Parameter D420.

The module is enabled by setting the HLVDEN bit (HLVDCON<4>). Each time the HLVD module is enabled, the circuitry requires some time to stabilize. The IRVST bit (HLVDCON<5>) is a read-only bit used to indicate when the circuit is stable. The module can only generate an interrupt after the circuit is stable and IRVST is set.

The VDIRMAG bit (HLVDCON<7>) determines the overall operation of the module. When VDIRMAG is cleared, the module monitors for drops in VDD below a predetermined set point. When the bit is set, the module monitors for rises in VDD above the set point.

25.1 Operation

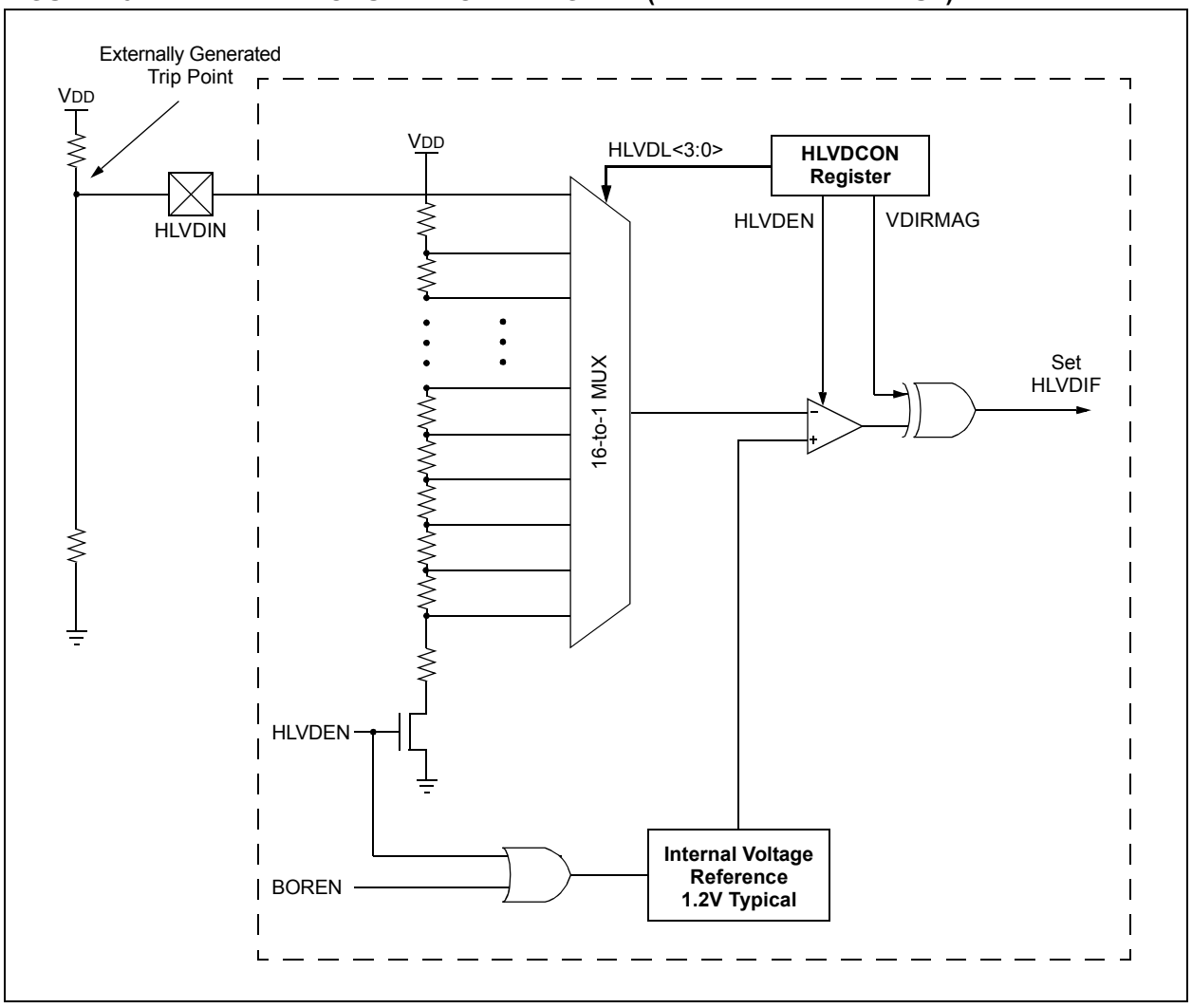
When the HLVD module is enabled, a comparator uses an internally generated reference voltage as the set point. The set point is compared with the trip point, where each node in the resistor divider represents a trip point voltage. The "trip point" voltage is the voltage level at which the device detects a high or low-voltage event, depending on the configuration of the module.

When the supply voltage is equal to the trip point, the voltage tapped off of the resistor array is equal to the internal reference voltage generated by the voltage reference module. The comparator then generates an interrupt signal by setting the HLVDIF bit.

The trip point voltage is software programmable to any of 16 values. The trip point is selected by programming the HLVDL<3:0> bits (HLVDCON<3:0>).

The HLVD module has an additional feature that allows the user to supply the trip voltage to the module from an external source. This mode is enabled when bits, HLVDL<3:0>, are set to '1111'. In this state, the comparator input is multiplexed from the external input pin, HLVDIN. This gives users the flexibility of configuring the High/Low-Voltage Detect interrupt to occur at any voltage in the valid operating range.

FIGURE 25-1: HLVD MODULE BLOCK DIAGRAM (WITH EXTERNAL INPUT)



25.2 HLVD Setup

To set up the HLVD module:

1. Select the desired HLVD trip point by writing the value to the HLVDL<3:0> bits.
2. Set the VDIRMAG bit to detect high voltage (VDIRMAG = 1) or low voltage (VDIRMAG = 0).
3. Enable the HLVD module by setting the HLVDEN bit.
4. Clear the HLVD interrupt flag (PIR2<2>), which may have been set from a previous interrupt.
5. If interrupts are desired, enable the HLVD interrupt by setting the HLVDIE and GIE bits (PIE2<2> and INTCON<7>, respectively).

An interrupt will not be generated until the IRVST bit is set.

Note:	Before changing any module settings (VDIRMAG, HLVDL<3:0>), first disable the module (HLVDEN = 0), make the changes and re-enable the module. This prevents the generation of false HLVD events.

25.3 Current Consumption

When the module is enabled, the HLVD comparator and voltage divider are enabled and consume static current.

Depending on the application, the HLVD module does not need to operate constantly. To reduce current requirements, the HLVD circuitry may only need to be enabled for short periods where the voltage is checked. After such a check, the module could be disabled.

25.4 HLVD Start-up Time

The internal reference voltage of the HLVD module, specified in electrical specification, Parameter 37 (**Section 30.0 "Electrical Specifications"**), may be used by other internal circuitry, such as the programmable Brown-out Reset. If the HLVD or other circuits using the voltage reference are disabled to lower the device's current consumption, the reference voltage circuit will require time to become stable before a low or high-voltage condition can be reliably detected. This start-up time, TIRVST, is an interval that is independent of device clock speed. It is specified in electrical specification, Parameter 37 (Table 30-26).

The HLVD interrupt flag is not enabled until TIRVST has expired and a stable reference voltage is reached. For this reason, brief excursions beyond the set point may not be detected during this interval (see Figure 25-2 or Figure 25-3).

FIGURE 25-2: LOW-VOLTAGE DETECT OPERATION (VDIRMAG = 0)

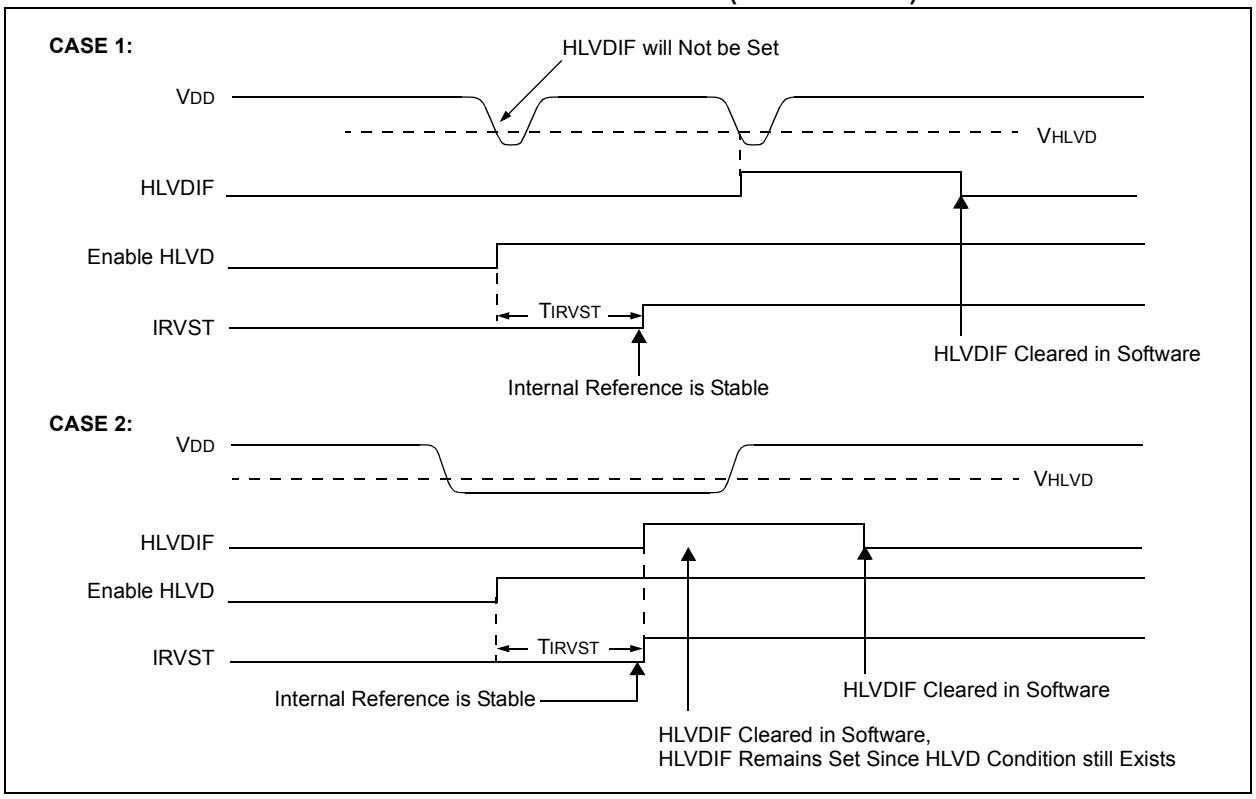
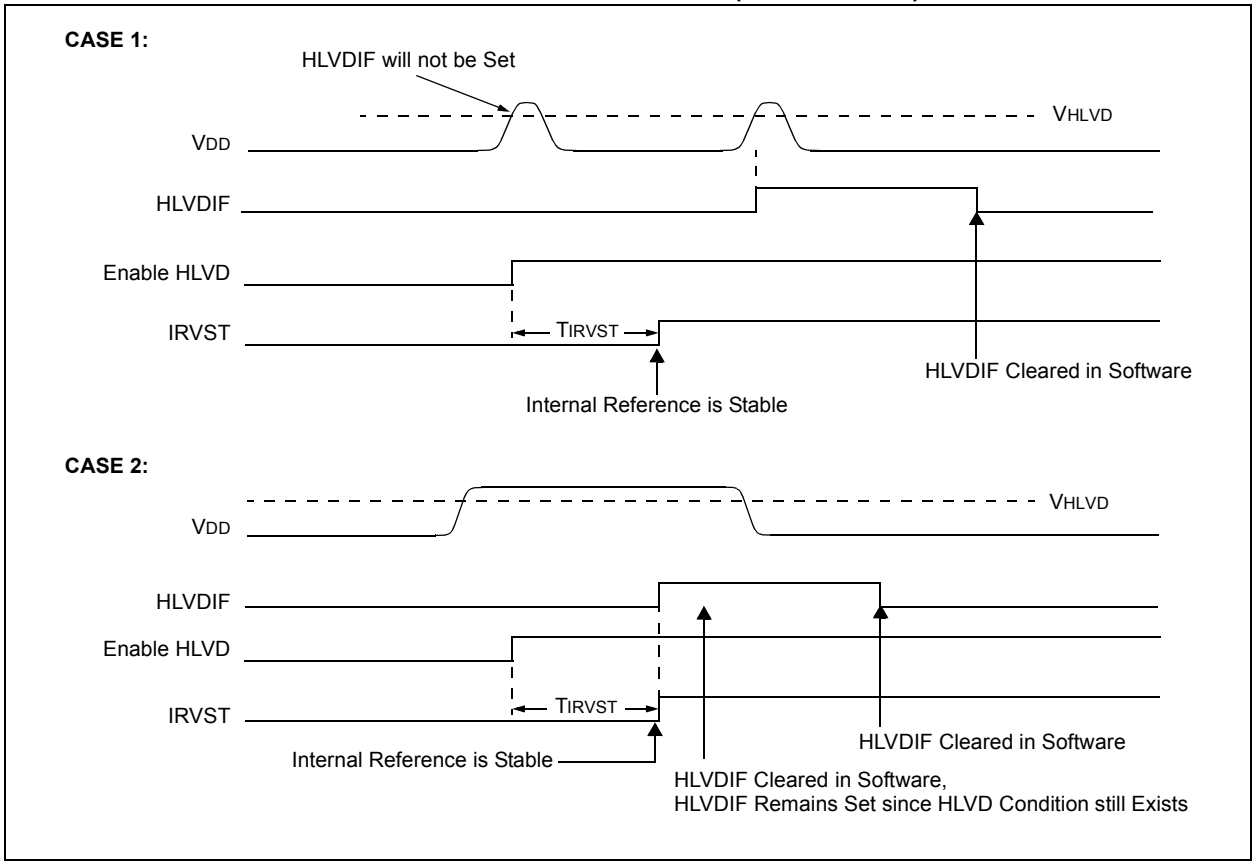


FIGURE 25-3: HIGH-VOLTAGE DETECT OPERATION (VDIRMAG = 1)

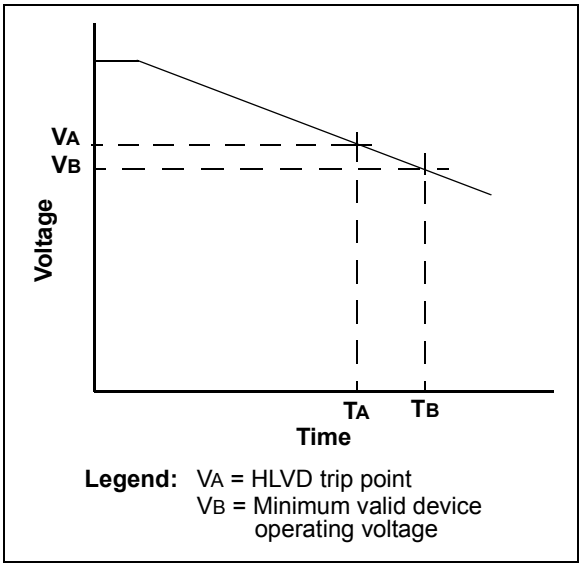


25.5 Applications

In many applications, it is desirable to detect a drop below, or rise above, a particular voltage threshold. For example, the HLVD module could be periodically enabled to detect Universal Serial Bus (USB) attach or detach. This assumes the device is powered by a lower voltage source than the USB when detached. An attach would indicate a high-voltage detect from, for example, 3.3V to 5V (the voltage on USB) and vice versa for a detach. This feature could save a design a few extra components and an attach signal (input pin).

For general battery applications, Figure 25-4 shows a possible voltage curve. Over time, the device voltage decreases. When the device voltage reaches voltage, VA, the HLVD logic generates an interrupt at time, TA. The interrupt could cause the execution of an ISR, which would allow the application to perform "housekeeping tasks" and a controlled shutdown, before the device voltage exits the valid operating range at TB. This would give the application a time window, represented by the difference between TA and TB, to safely exit.

FIGURE 25-4: TYPICAL LOW-VOLTAGE DETECT APPLICATION



25.6 Operation During Sleep

When enabled, the HLVD circuitry continues to operate during Sleep. If the device voltage crosses the trip point, the HLVDIF bit will be set and the device will wake-up from Sleep. Device execution will continue from the interrupt vector address if interrupts have been globally enabled.

25.7 Effects of a Reset

A device Reset forces all registers to their Reset state. This forces the HLVD module to be turned off.

26.0 CHARGE TIME MEASUREMENT UNIT (CTMU)

The Charge Time Measurement Unit (CTMU) is a flexible analog module that provides accurate differential time measurement between pulse sources, as well as asynchronous pulse generation. By working with other on-chip analog modules, the CTMU can precisely measure time, capacitance and relative changes in capacitance or generate output pulses with a specific time delay. The CTMU is ideal for interfacing with capacitive-based sensors.

The module includes these key features:

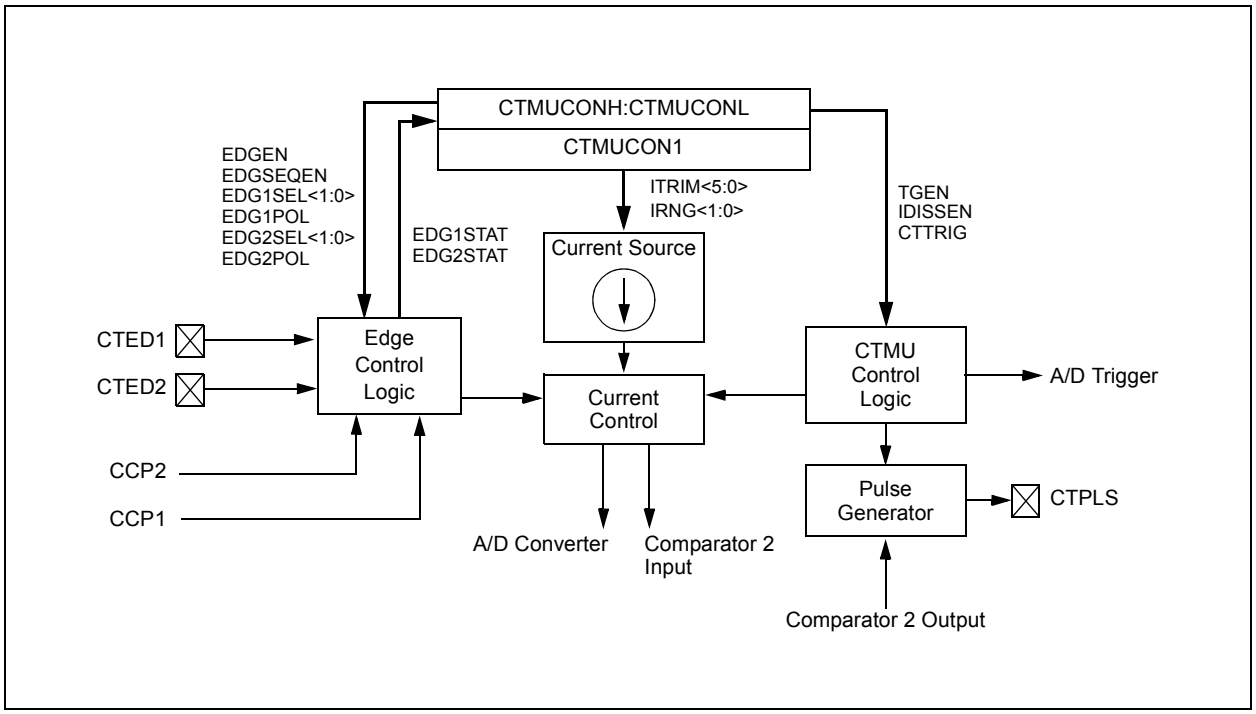
- Up to 24 channels available for capacitive or time measurement input
- Low-cost temperature measurement using on-chip diode channel
- On-chip precision current source
- Sixteen-edge input trigger sources
- Polarity control for each edge source
- Provides a trigger for the A/D Converter
- Control of edge sequence
- Control of response to edges
- Time measurement resolution of 1 nanosecond
- High-precision time measurement
- Time delay of external or internal signal asynchronous to system clock
- Accurate current source suitable for capacitive measurement

The CTMU works in conjunction with the A/D Converter to provide up to 24 channels for time or charge measurement, depending on the specific device and the number of A/D channels available. When configured for time delay, the CTMU is connected to one of the analog comparators. The level-sensitive input edge sources can be selected from four sources: two external inputs or the CCP1/CCP2 Special Event Triggers.

The CTMU special event can trigger the Analog-to-Digital Converter module.

Figure 26-1 provides a block diagram of the CTMU.

FIGURE 26-1: CTMU BLOCK DIAGRAM



26.1 CTMU Registers

The control registers for the CTMU are:

- CTMUCON1
- CTMUCON2
- CTMUCON3
- CTMUCON4

The CTMUCON1 and CTMUCON3 registers (Register 26-1 and Register 26-3) contain control bits for configuring the CTMU module edge source selection, edge source polarity selection, edge sequencing, A/D trigger, analog circuit capacitor discharge and enables. The CTMUCON2 register (Register 26-2) has bits for selecting the current source range and current source trim.

REGISTER 26-1: CTMUCON1: CTMU CONTROL REGISTER 1

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CTMUEN	—	CTMUSIDL	TGEN	EDGEN	EDGSEQEN	IDISSEN	CTTRIG
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **CTMUEN:** CTMU Enable bit
1 = Module is enabled
0 = Module is disabled

bit 6 **Unimplemented:** Read as ‘0’

bit 5 **CTMUSIDL:** Stop in Idle Mode bit
1 = Discontinues module operation when device enters Idle mode
0 = Continues module operation in Idle mode

bit 4 **TGEN:** Time Generation Enable bit
1 = Enables edge delay generation
0 = Disables edge delay generation

bit 3 **EDGEN:** Edge Enable bit
1 = Edges are not blocked
0 = Edges are blocked

bit 2 **ESGSEQEN:** Edge Sequence Enable bit
1 = Edge 1 event must occur before Edge 2 event can occur
0 = No edge sequence is needed

bit 1 **IDISSEN:** Analog Current Source Control bit
1 = Analog current source output is grounded
0 = Analog current source output is not grounded

bit 0 **CTTRIG:** CTMU Special Event Trigger bit
1 = CTMU Special Event Trigger is enabled
0 = CTMU Special Event Trigger is disabled

REGISTER 26-2: CTMUCON2: CTMU CURRENT CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ITRIM5	ITRIM4	ITRIM3	ITRIM2	ITRIM1	ITRIM0	IRNG1	IRNG0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-2 **ITRIM<5:0>:** Current Source Trim bits

011111 = Maximum positive change (+62% typ.) from nominal current
011110
.
.
.
000001 = Minimum positive change (+2% typ.) from nominal current
000000 = Nominal current output specified by IRNG<1:0>
111111 = Minimum negative change (-2% typ.) from nominal current
.
.
.
100010
100001 = Maximum negative change (-62% typ.) from nominal current

bit 1-0 **IRNG<1:0>:** Current Source Range Select bits

11 = 100 x Base Current
10 = 10 x Base Current
01 = Base Current Level (0.55 μA nominal)
00 = 1000 x Base Current

REGISTER 26-3: CTMUCON3: CTMU CURRENT CONTROL REGISTER 3

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
EDG2EN	EDG2POL	EDG2SEL3	EDG2SEL2	EDG2SEL1	EDG2SEL0	—	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **EDG2EN:** Edge 2 Edge-Sensitive Select bit
1 = Input is edge-sensitive
0 = Input is level-sensitive

bit 6 **EDG2POL:** Edge 2 Polarity Select bit
1 = Edge 2 is programmed for a positive edge response
0 = Edge 2 is programmed for a negative edge response

bit 5-2 **EDG2SEL<3:0>:** Edge 2 Source Select bits
1111 = CMP3 selected
1110 = CMP2 selected
1101 = CMP1 selected
1100 = Reserved
1011 = CCP3 trigger selected
1010 = CCP2 trigger selected
1001 = CCP1 trigger selected
1000 = CTED13 selected
0111 = CTED12 selected
0110 = CTED11 selected
0101 = CTED10 selected
0100 = CTED9 selected
0011 = CTED1 selected
0010 = CTED2 selected
0001 = CCP1 interrupt selected
0000 = TMR1 interrupt selected

bit 1-0 **Unimplemented:** Read as '0'

REGISTER 26-4: CTMUCON4: CTMU CURRENT CONTROL REGISTER 4

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	U-0
EDG1EN	EDG1POL	EDG1SEL3	EDG1SEL2	EDG1SEL1	EDG1SEL0	EDG2STAT	EDG1STAT
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **EDG1EN:** Edge 1 Edge-Sensitive Select bit
1 = Input is edge-sensitive
0 = Input is level-sensitive

bit 6 **EDG1POL:** Edge 1 Polarity Select bit
1 = Edge 1 is programmed for a positive edge response
0 = Edge 1 is programmed for a negative edge response

bit 5-2 **EDG1SEL<3:0>:** Edge 1 Source Select bits
1111 = CMP3 selected
1110 = CMP2 selected
1101 = CMP1 selected
1100 = CCP3 trigger selected
1011 = CCP2 trigger selected
1010 = CCP1 trigger selected
1001 = CTED8 selected
1000 = CTED7 selected
0111 = CTED6 selected
0110 = CTED5 selected
0101 = CTED4 selected
0100 = CTED3 selected
0011 = CTED1 selected
0010 = CTED2 selected
0001 = CCP1 interrupt selected
0000 = TMR1 interrupt selected

bit 1-0 **EDG2STAT:** Edge 2 Status bit
Indicates the status of Edge 2 and can be written to control edge source.
1 = Edge2 has occurred
0 = Edge2 has not occurred

bit 1-0 **EDG1STAT:** Edge 1 Status bit
1 = Edge 1 has occurred
0 = Edge 1 has not occurred

26.2 CTMU Operation

The CTMU works by using a fixed current source to charge a circuit. The type of circuit depends on the type of measurement being made.

In the case of charge measurement, the current is fixed and the amount of time the current is applied to the circuit is fixed. The amount of voltage read by the A/D becomes a measurement of the circuit's capacitance.

In the case of time measurement, the current, as well as the capacitance of the circuit, is fixed. In this case, the voltage read by the A/D is representative of the amount of time elapsed from the time the current source starts and stops charging the circuit.

If the CTMU is being used as a time delay, both capacitance and current source are fixed, as well as the voltage supplied to the comparator circuit. The delay of a signal is determined by the amount of time it takes the voltage to charge to the comparator threshold voltage.

26.2.1 THEORY OF OPERATION

The operation of the CTMU is based on the equation for charge:

$$I = C \bullet \frac{dV}{dT}$$

More simply, the amount of charge measured in coulombs in a circuit is defined as current in amperes (I), multiplied by the amount of time in seconds that the current flows (t). Charge is also defined as the capacitance in farads (C), multiplied by the voltage of the circuit (V). It follows that:

$$I \bullet t = C \bullet V$$

The CTMU module provides a constant, known current source. The A/D Converter is used to measure (V) in the equation, leaving two unknowns: capacitance (C) and time (t). The above equation can be used to calculate capacitance or time, by either the relationship using the known fixed capacitance of the circuit:

$$t = (C \bullet V)/I$$

or by:

$$C = (I \bullet t)/V$$

using a fixed time that the current source is applied to the circuit.

26.2.2 CURRENT SOURCE

At the heart of the CTMU is a precision current source, designed to provide a constant reference for measurements. The level of current is user-selectable across three ranges, or a total of two orders of magnitude, with the ability to trim the output in ±2% increments (nominal). The current range is selected by the IRNG<1:0> bits (CTMUCON1<1:0>), with a value of '01' representing the lowest range.

Current trim is provided by the ITRIM<5:0> bits (CTMUCON1<7:2>). These six bits allow trimming of the current source, in steps of approximately 2% per step. Half of the range adjusts the current source positively and the other half reduces the current source. A value of '000000' is the neutral position (no change). A value of '100001' is the maximum negative adjustment (approximately -62%) and '011111' is the maximum positive adjustment (approximately +62%).

26.2.3 EDGE SELECTION AND CONTROL

CTMU measurements are controlled by edge events occurring on the module's two input channels. Each channel, referred to as Edge 1 and Edge 2, can be configured to receive input pulses from one of the edge input pins (CTED1 and CTED2) or CCPx Special Event Triggers (CCP1 and CCP2). The input channels are level-sensitive, responding to the instantaneous level on the channel rather than a transition between levels. The inputs are selected using the EDG1SELx (CTMUCON2<5:2>) and EDG2SELx (CTMUCON3<5:2>) bit pairs.

In addition to source, each channel can be configured for event polarity using the EDGE2POL (CTMUCON2<6>) and EDGE1POL (CTMUCON3<6> bits. The input channels can also be filtered for an edge event sequence (Edge 1 occurring before Edge 2) by setting the EDGSEQEN bit (CTMUCON<2>).

26.2.4 EDGE STATUS

The CTMUCON3 register also contains two Edge Status bits: EDG2STAT and EDG1STAT (CTMUCON3<1:0>). Their primary function is to show if an edge response has occurred on the corresponding channel. The CTMU automatically sets a particular bit when an edge response is detected on its channel. The level-sensitive nature of the input channels also means that the Status bits become set immediately if the channel's configuration is changed and matches the channel's current state.

The module uses the Edge Status bits to control the current source output to external analog modules (such as the A/D Converter). Current is only supplied to external modules when only one (not both) of the Status bits is set. Current is shut off when both bits are either set or cleared. This allows the CTMU to measure current only during the interval between edges. After both Status bits are set, it is necessary to clear them before another measurement is taken. Both bits should be cleared simultaneously, if possible, to avoid re-enabling the CTMU current source.

In addition to being set by the CTMU hardware, the Edge Status bits can also be set by software. This permits a user application to manually enable or disable the current source. Setting either (but not both) of the bits enables the current source. Setting or clearing both bits at once disables the source.

26.2.5 INTERRUPTS

The CTMU sets its interrupt flag (PIR3<3>) whenever the current source is enabled, then disabled. An interrupt is generated only if the corresponding interrupt enable bit (PIE3<3>) is also set. If edge sequencing is not enabled (i.e., Edge 1 must occur before Edge 2), it is necessary to monitor the Edge Status bits, and determine which edge occurred last and caused the interrupt.

26.3 CTMU Module Initialization

The following sequence is a general guideline used to initialize the CTMU module:

1. Select the current source range using the IRNGx bits (CTMUCON1<1:0>).
2. Adjust the current source trim using the ITRIMx bits (CTMUCON1<7:2>).
3. Configure the edge input sources for Edge 1 and Edge 2 by setting the EDG1SELx and EDG2SELx bits (CTMUCON3<5:2> and CTMUCON2<5:2>, respectively).
4. Configure the input polarities for the edge inputs using the EDG1POL and EDG2POL bits (CTMUCON3<6> and CTMUCON2<6>).

   The default configuration is for negative edge polarity (high-to-low transitions).
5. Enable edge sequencing using the EDGSEQEN bit (CTMUCON<2>).

   By default, edge sequencing is disabled.
6. Select the operating mode (Measurement or Time Delay) with the TGEN bit (CTMUCON<4>).

   The default mode is Time/Capacitance Measurement mode.
7. Configure the module to automatically trigger an A/D conversion when the second edge event has occurred using the CTTRIG bit (CTMUCON<0>).

   The conversion trigger is disabled by default.
8. Discharge the connected circuit by setting the IDISSEN bit (CTMUCON<1>).
9. After waiting a sufficient time for the circuit to discharge, clear the IDISSEN bit.
10. Disable the module by clearing the CTMUEN bit (CTMUCON<7>).
11. Clear the Edge Status bits, EDG2STAT and EDG1STAT (CTMUCON3<1:0>).

    Both bits should be cleared simultaneously, if possible, to avoid re-enabling the CTMU current source.
12. Enable both edge inputs by setting the EDGEN bit (CTMUCON<3>).
13. Enable the module by setting the CTMUEN bit.

Depending on the type of measurement or pulse generation being performed, one or more additional modules may also need to be initialized and configured with the CTMU module:

- Edge Source Generation: In addition to the external edge input pins, CCP1/CCP2 Special Event Triggers can be used as edge sources for the CTMU.
- Capacitance or Time Measurement: The CTMU module uses the A/D Converter to measure the voltage across a capacitor that is connected to one of the analog input channels.
- Pulse Generation: When generating system clock independent, output pulses, the CTMU module uses Comparator 2 and the associated comparator voltage reference.

26.4 Calibrating the CTMU Module

The CTMU requires calibration for precise measurements of capacitance and time, as well as for accurate time delay. If the application only requires measurement of a relative change in capacitance or time, calibration is usually not necessary. An example of a less precise application is a capacitive touch switch, in which the touch circuit has a baseline capacitance and the added capacitance of the human body changes the overall capacitance of a circuit.

If actual capacitance or time measurement is required, two hardware calibrations must take place:

- The current source needs calibration to set it to a precise current.
- The circuit being measured needs calibration to measure or nullify any capacitance other than that to be measured.

26.4.1 CURRENT SOURCE CALIBRATION

The current source on board the CTMU module has a range of ±62% nominal for each of three current ranges. For precise measurements, it is possible to measure and adjust this current source by placing a high-precision resistor, RCAL, onto an unused analog channel. An example circuit is shown in Figure 26-2.

To measure the current source:

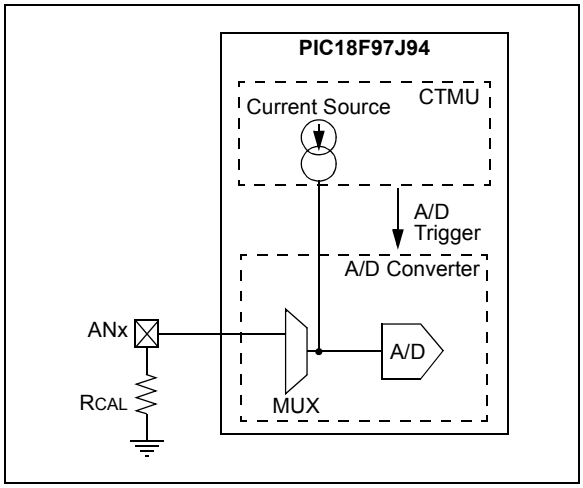
1. Initialize the A/D Converter.
2. Initialize the CTMU.
3. Enable the current source by setting EDG1STAT (CTMUCON3<0>).
4. Issue time delay for voltage across RCAL to stabilize and the A/D Sample-and-Hold (S/H) capacitor to charge.
5. Perform the A/D conversion.
6. Calculate the current source current using $I = V/R_{CAL}$, where RCAL is a high-precision resistance and V is measured by performing an A/D conversion.

The CTMU current source may be trimmed with the ITRIMx bits in CTMUCON1, using an iterative process to get the exact current desired. Alternatively, the nominal value without adjustment may be used. That value may be stored by software for use in all subsequent capacitive or time measurements.

To calculate the optimal value for RCAL, the nominal current must be chosen.

For example, if the A/D Converter reference voltage is 3.3V, use 70% of full scale (or 2.31V) as the desired approximate voltage to be read by the A/D Converter. If the range of the CTMU current source is selected to be 0.55 μA, the resistor value needed is calculated as RCAL = 2.31V/0.55 μA, for a value of 4.2 MΩ. Similarly, if the current source is chosen to be 5.5 μA, RCAL would be 420,000Ω, and 42,000Ω if the current source is set to 55 μA.

FIGURE 26-2: CTMU CURRENT SOURCE CALIBRATION CIRCUIT



A value of 70% of full-scale voltage is chosen to make sure that the A/D Converter is in a range that is well above the noise floor. If an exact current is chosen to incorporate the trimming bits from CTMUCON1, the resistor value of RCAL may need to be adjusted accordingly. RCAL also may be adjusted to allow for available resistor values. RCAL should be of the highest precision available in light of the precision needed for the circuit that the CTMU will be measuring. A recommended minimum would be 0.1% tolerance.

The following examples show a typical method for performing a CTMU current calibration.

- Example 26-1 demonstrates how to initialize the A/D Converter and the CTMU.

  This routine is typical for applications using both modules.

- Example 26-2 demonstrates one method for the actual calibration routine.

  This method manually triggers the A/D Converter to demonstrate the entire step-wise process. It is also possible to automatically trigger the conversion by setting the CTMU's CTTRIG bit (CTMUCON<0>).

EXAMPLE 26-1: SETUP FOR CTMU CALIBRATION ROUTINES

```
#include "p18cxxx.h"
/*****************************************************************************/
/*Setup CTMU *****************************************************************/
/*****************************************************************************/
void setup(void)

{ //CTMUCON - CTMU Control register

    CTMUCON = 0x00;              //make sure CTMU is disabled
    CTMUCON3 = 0x90;
    //CTMU continues to run when emulator is stopped,CTMU continues
    //to run in idle mode,Time Generation mode disabled, Edges are blocked
    //No edge sequence order, Analog current source not grounded, trigger
    //output disabled, Edge2 polarity = positive level, Edge2 source =
    //source 0, Edge1 polarity = positive level, Edge1 source = source 0,
    // Set Edge status bits to zero


    //CTMUCON1 - CTMU Current Control Register
    CTMUCON1 = 0x01;             //0.55uA, Nominal - No Adjustment

/*****************************************************************************/
//Setup AD converter;
/*****************************************************************************/

 TRISBbits.TRISB0=0;
    TRISAbits.TRISA2=1;                     //set channel 2 as an input
    ANCON1bits.ANSEL2=1;                    // Configured AN2 as an analog channel
    ADCON1Hbits.FORM=0b00;                  // Result format 1= Right justified
    ADCON1Lbits.SSRC=0b0111;
    ADCON3Hbits.SAMC=0b00111;               // Acquisition time 7 = 20TAD 2 = 4TAD 1=2TAD
    ADCON3Lbits.ADCS=0x3F;                  // Clock conversion bits 6= FOSC/64 2=FOSC/32
                                            // ADCON1
    ADCON2Hbits.PVCFG=0b00;                 // Vref+ = AVdd
    ADCON2Hbits.NVCFG0=0;                   // Vref- = AVss
    ADCHS0Lbits.CHONA=0b000;
    ADCHS0Lbits.CHOSA=0b00010;              // Select ADC channel
    ADCON1Hbits.ADON=1; // Turn on ADC

}
```

EXAMPLE 26-2: CTMU CURRENT CALIBRATION ROUTINE

```
#include "p18cxxx.h"

#define COUNT 500                       //@ 8MHz = 125uS.
#define DELAY for(i=0;i<COUNT;i++)
#define RCAL .027                       //R value is 4200000 (4.2M)
                                        //scaled so that result is in
                                        //1/100th of uA
#define ADSCALE 1023                    //for unsigned conversion 10 sig bits
#define ADREF 3.3                       //Vdd connected to A/D Vr+

int main(void)
{
    int i;
    int j = 0; //index for loop
    unsigned int Vread = 0;
    double VTot = 0;
    float Vavg=0, Vcal=0, CTMUISrc = 0;  //float values stored for calcs

//assume CTMU and A/D have been setup correctly
//see Example 25-1 for CTMU & A/D setup
setup();

CTMUCONbits.CTMUEN = 1;                 //Enable the CTMU
    for(j=0;j<10;j++)
    {
        CTMUCONbits.IDISSEN = 1;        //drain charge on the circuit
        DELAY;                          //wait 125us
        CTMUCONbits.IDISSEN = 0;        //end drain of circuit

        CTMUCON3bits.EDG1STAT = 1;      //Begin charging the circuit
                                        //using CTMU current source
        DELAY;                          //wait for 125us
        CTMUCON3bits.EDG1STAT = 0;      //Stop charging circuit

        PIR1bits.ADIF = 0;              //make sure A/D Int not set
        ADCON1Lbits.SAMP=1;             //and begin A/D conv.
        while(!PIR1bits.ADIF);          //Wait for A/D convert complete

        Vread = ADRES;                  //Get the value from the A/D
        PIR1bits.ADIF = 0;              //Clear A/D Interrupt Flag
        VTot += Vread;                  //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000);        //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL;               //CTMUISrc is in 1/100ths of uA

}
```

26.4.2 CAPACITANCE CALIBRATION

There is a small amount of capacitance from the internal A/D Converter sample capacitor, as well as stray capacitance from the circuit board traces and pads that affect the precision of capacitance measurements. A measurement of the stray capacitance can be taken by making sure the desired capacitance to be measured has been removed.

After removing the capacitance to be measured:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT (= 1).
3. Wait for a fixed delay of time, *t*.
4. Clear EDG1STAT.
5. Perform an A/D conversion.
6. Calculate the stray and A/D sample capacitances:

$$C_{OFFSET} = C_{STRAY} + C_{AD} = (I \bullet t)/V$$

Where:

- I is known from the current source measurement step
- t is a fixed delay
- V is measured by performing an A/D conversion

This measured value is then stored and used for calculations of time measurement or subtracted for capacitance measurement. For calibration, it is expected that the capacitance of C_{STRAY} + C_{AD} is approximately known; C_{AD} is approximately 4 pF.

An iterative process may be required to adjust the time, t, that the circuit is charged to obtain a reasonable voltage reading from the A/D Converter. The value of *t* may be determined by setting C_{OFFSET} to a theoretical value and solving for *t*. For example, if C_{STRAY} is theoretically calculated to be 11 pF, and *V* is expected to be 70% of V_{DD} or 2.31V, t would be:

$$(4\ pF + 11\ pF) \bullet 2.31V/0.55\ mA$$

or 63 μs.

See Example 26-3 for a typical routine for CTMU capacitance calibration.

EXAMPLE 26-3: CTMU CAPACITANCE CALIBRATION ROUTINE

```
#include "p18cxxx.h"

#define COUNT 25                        //@ 8MHz INTFRC = 62.5 us.
#define ETIME COUNT*2.5                 //time in uS
#define DELAY for(i=0;i<COUNT;i++)
#define ADSCALE 1023                    //for unsigned conversion 10 sig bits
#define ADREF 3.3                       //Vdd connected to A/D Vr+
#define RCAL .027                       //R value is 4200000 (4.2M)
                                        //scaled so that result is in
                                        //1/100th of uA

int main(void)
{
    int i;
    int j = 0;                          //index for loop
    unsigned int Vread = 0;
    float CTMUISrc, CTMUCap, Vavg, VTot, Vcal;

//assume CTMU and A/D have been setup correctly
//see Example 25-1 for CTMU & A/D setup
setup();

CTMUCONbits.CTMUEN = 1;                 //Enable the CTMU
    for(j=0;j<10;j++)
    {
        CTMUCONbits.IDISSEN = 1;        //drain charge on the circuit
        DELAY;                          //wait 125us
        CTMUCONbits.IDISSEN = 0;        //end drain of circuit

        CTMUCON3bits.EDG1STAT = 1;      //Begin charging the circuit
                                        //using CTMU current source
        DELAY;                          //wait for 125us
        CTMUCON3bits.EDG1STAT = 0;      //Stop charging circuit

        PIR1bits.ADIF = 0;              //make sure A/D Int not set
        ADCON1Lbits.SAMP=1;             //and begin A/D conv.
        while(!PIR1bits.ADIF);          //Wait for A/D convert complete


        Vread = ADRES;                  //Get the value from the A/D
        PIR1bits.ADIF = 0;              //Clear A/D Interrupt Flag
        VTot += Vread;                  //Add the reading to the total
    }

    Vavg = (float)(VTot/10.000);        //Average of 10 readings
    Vcal = (float)(Vavg/ADSCALE*ADREF);
    CTMUISrc = Vcal/RCAL;               //CTMUISrc is in 1/100ths of uA
    CTMUCap = (CTMUISrc*ETIME/Vcal)/100;
}
```

26.5 Measuring Capacitance with the CTMU

There are two ways to measure capacitance with the CTMU. The absolute method measures the actual capacitance value. The relative method only measures for any change in the capacitance.

26.5.1 ABSOLUTE CAPACITANCE MEASUREMENT

For absolute capacitance measurements, both the current and capacitance calibration steps found in **Section 26.4 "Calibrating the CTMU Module"** should be followed.

To perform these measurements:

1. Initialize the A/D Converter.
2. Initialize the CTMU.
3. Set EDG1STAT.
4. Wait for a fixed delay, T.
5. Clear EDG1STAT.
6. Perform an A/D conversion.
7. Calculate the total capacitance, $C_{TOTAL} = (I * T)/V$, where:
   - I is known from the current source measurement step (**Section 26.4.1 "Current Source Calibration"**)
   - T is a fixed delay
   - V is measured by performing an A/D conversion
8. Subtract the stray and A/D capacitance (C_{OFFSET} from **Section 26.4.2 "Capacitance Calibration"**) from C_{TOTAL} to determine the measured capacitance.

26.5.2 CAPACITIVE TOUCH SENSE USING RELATIVE CHARGE MEASUREMENT

Not all applications require precise capacitance measurements. When detecting a valid press of a capacitance-based switch, only a relative change of capacitance needs to be detected.

In such an application when the switch is open (or not touched), the total capacitance is the capacitance of the combination of the board traces, the A/D Converter and other elements. A larger voltage will be measured by the A/D Converter. When the switch is closed (or touched), the total capacitance is larger due to the addition of the capacitance of the human body to the above listed capacitances and a smaller voltage will be measured by the A/D Converter.

To detect capacitance changes simply:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT.
3. Wait for a fixed delay.
4. Clear EDG1STAT.
5. Perform an A/D conversion.

The voltage measured by performing the A/D conversion is an indication of the relative capacitance. In this case, no calibration of the current source or circuit capacitance measurement is needed. (For a sample software routine for a capacitive touch switch, see Example 26-4.)

EXAMPLE 26-4: CTMU ROUTINE FOR CAPACITIVE TOUCH SWITCH

```
#include "p18cxxx.h"

#define COUNT 500                       //@ 8MHz = 125uS.
#define DELAY for(i=0;i<COUNT;i++)
#define OPENSW 1000                     //Un-pressed switch value
#define TRIP 300                        //Difference between pressed
                                        //and un-pressed switch
#define HYST 65                         //amount to change
                                        //from pressed to un-pressed
#define PRESSED 1
#define UNPRESSED 0

int main(void)
{
    unsigned int Vread;                 //storage for reading
    unsigned int switchState;
    int i;

    //assume CTMU and A/D have been setup correctly
    //see Example 25-1 for CTMU & A/D setup
    setup();

    CTMUCONbits.CTMUEN = 1;             //Enable the CTMU

    CTMUCONbits.IDISSEN = 1;            //drain charge on the circuit
    DELAY;                              //wait 125us
    CTMUCONbits.IDISSEN = 0;            //end drain of circuit

    CTMUCON3bits.EDG1STAT = 1;          //Begin charging the circuit
                                        //using CTMU current source
    DELAY;                              //wait for 125us
    CTMUCON3bits.EDG1STAT = 0;          //Stop charging circuit

    PIR1bits.ADIF = 0;                  //make sure A/D Int not set
    ADCON1Lbits.SAMP=1;;                //and begin A/D conv.
    while(!PIR1bits.ADIF);              //Wait for A/D convert complete

    Vread = ADRES;                      //Get the value from the A/D

    if(Vread < OPENSW - TRIP)
    {
        switchState = PRESSED;
    }
    else if(Vread > OPENSW - TRIP + HYST)
    {
        switchState = UNPRESSED;
    }
}
```

26.6 Measuring Time with the CTMU Module

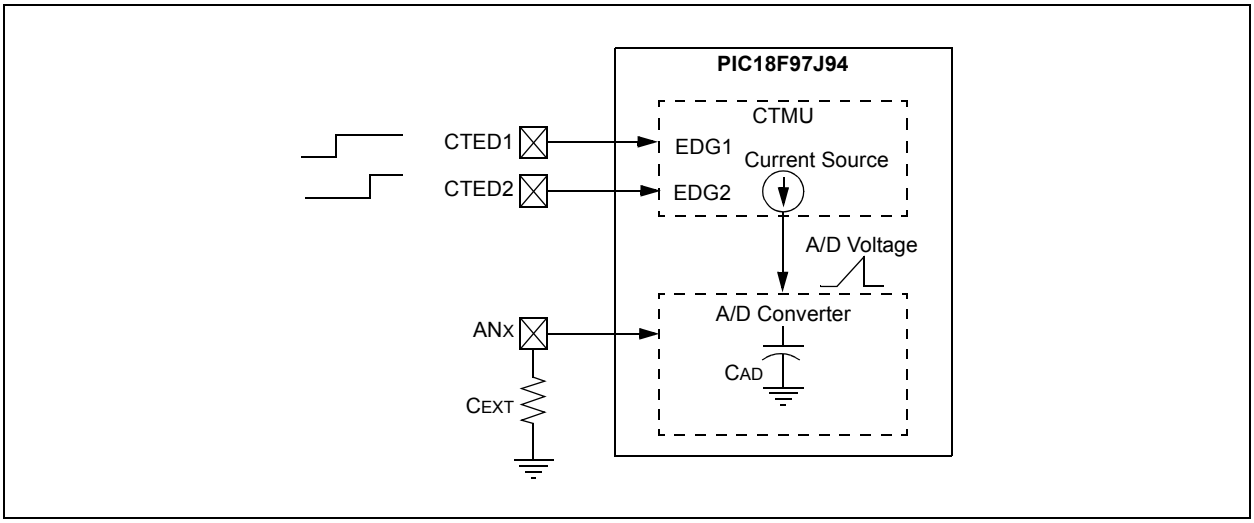
Time can be precisely measured after the ratio (C/I) is measured from the current and capacitance calibration step. To do that:

1. Initialize the A/D Converter and the CTMU.
2. Set EDG1STAT.
3. Set EDG2STAT.
4. Perform an A/D conversion.
5. Calculate the time between edges as T = (C/I) * V, where:
   - I is calculated in the current calibration step (**Section 26.4.1 "Current Source Calibration"**)
   - C is calculated in the capacitance calibration step (**Section 26.4.2 "Capacitance Calibration"**)
   - V is measured by performing the A/D conversion

It is assumed that the time measured is small enough that the capacitance, CAD + CEXT, provides a valid voltage to the A/D Converter. For the smallest time measurement, always set the A/D Channel Select bits via ADCON1L/H to an unused A/D channel; the corresponding pin for which is not connected to any circuit board trace. This minimizes added stray capacitance, keeping the total circuit capacitance close to that of the A/D Converter itself (25 pF).

To measure longer time intervals, an external capacitor may be connected to an A/D channel and that channel selected whenever making a time measurement.

FIGURE 26-3: CTMU TYPICAL CONNECTIONS AND INTERNAL CONFIGURATION FOR TIME MEASUREMENT



26.7 Measuring Temperature with the CTMU

The constant-current source provided by the CTMU module can be used for low-cost temperature measurement by exploiting a basic property of common and inexpensive diodes. An on-chip temperature sense diode is provided on A/D Channel 29 to further simplify design and cost.

26.7.1 BASIC PRINCIPAL

We can show that the forward voltage (V_F) of a P-N junction, such as a diode, is an extension of the equation for the junction's thermal voltage:

$$V_F = \frac{kT}{q} \ln\left(1 - \frac{I_F}{I_S}\right)$$

where k is the Boltzmann constant (1.38×10^{-23} J K^{-1}), T is the absolute junction temperature in kelvin, q is the electron charge (1.6×10^{-19} C), I_F is the forward current applied to the diode and I_S is the diode's characteristic saturation current, which varies between devices.

Since k and q are physical constants, and I_S is a constant for the device, this only leaves T and I_F as independent variables. If I_F is held constant, it follows from the equation that V_F will vary as a function of T. As the natural log term of the equation will always be negative, the temperature will be negatively proportional to V_F. In other words, as temperature increases, V_F decreases.

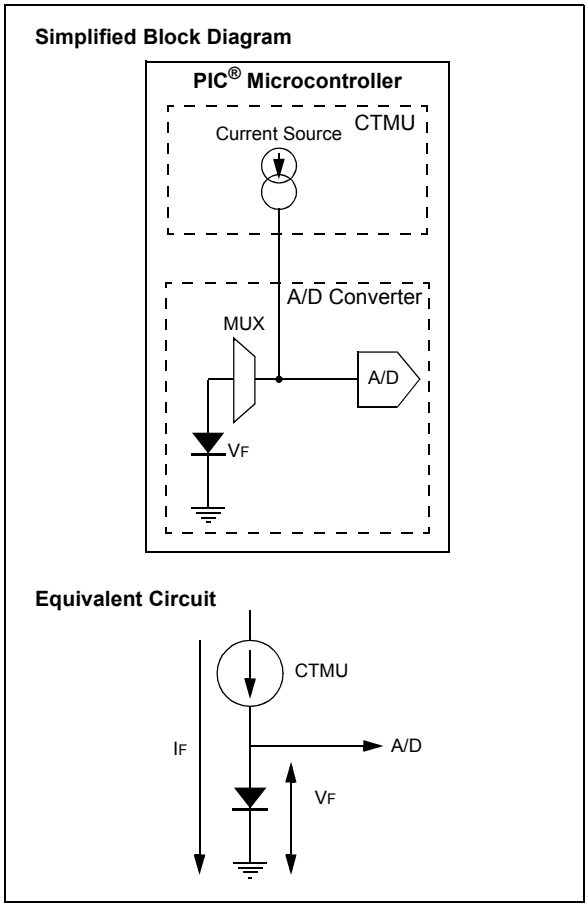
By using the CTMU's current source to provide a constant I_F, it becomes possible to calculate the temperature by measuring the V_F across the diode.

26.7.2 IMPLEMENTATION

To implement this theory, all that is needed is to connect a regular junction diode to one of the microcontroller's A/D pins (Figure 26-2). The A/D channel multiplexer is shared by the CTMU and the A/D.

To perform a measurement, the multiplexer is configured to select the pin connected to the diode. The CTMU current source is then turned on and an A/D conversion is performed on the channel. As shown in the equivalent circuit diagram in Figure 26-4, the diode is driven by the CTMU at I_F. The resulting V_F across the diode is measured by the A/D. A code snippet is shown in Example 26-5.

FIGURE 26-4: CTMU TEMPERATURE MEASUREMENT CIRCUIT



EXAMPLE 26-5: CTMU ROUTINE FOR TEMPERATURE MEASUREMENT USING INTERNAL DIODE

```
// Initialize CTMU
CTMUICON = 0x03;
CTMUCONbits.CTMUEN = 1;
CTMUCON3bits.EDG1STAT = 1;


ADCON1Hbits.FORM    = 0;              // Right Justified
ADCON1Hbits.MODE12  = 0;              // 12-Bit A/D Operation
ADCHS0Lbits.CHOSA   = 0x18;           // Enable ADC and  connect to Internal diode

ADCON1Hbits.ADON    = 1;              // Enable ADC
```

Note: The temperature diode is not calibrated or standardized; the user must calibrate the diode to their application.

26.8 Operation During Sleep/Idle Modes

26.8.1 SLEEP MODE

When the device enters any Sleep mode, the CTMU module current source is always disabled. If the CTMU is performing an operation that depends on the current source when Sleep mode is invoked, the operation may not terminate correctly. Capacitance and time measurements may return erroneous values.

26.8.2 IDLE MODE

The behavior of the CTMU in Idle mode is determined by the CTMUSIDL bit (CTMUCON<5>). If CTMUSIDL is cleared, the module will continue to operate in Idle mode. If CTMUSIDL is set, the module's current source is disabled when the device enters Idle mode. In this case, if the module is performing an operation when Idle mode is invoked, the results will be similar to those with Sleep mode.

26.9 Effects of a Reset on CTMU

Upon Reset, all registers of the CTMU are cleared. This disables the CTMU module, turns off its current source and returns all configuration options to their default settings. The module needs to be re-initialized following any Reset.

If the CTMU is in the process of taking a measurement at the time of Reset, the measurement will be lost. A partial charge may exist on the circuit that was being measured, which should be properly discharged before the CTMU makes subsequent attempts to take a measurement. The circuit is discharged by setting and clearing the IDISSEN bit (CTMUCON<1>) while the A/D Converter is connected to the appropriate channel.

27.0 UNIVERSAL SERIAL BUS (USB)

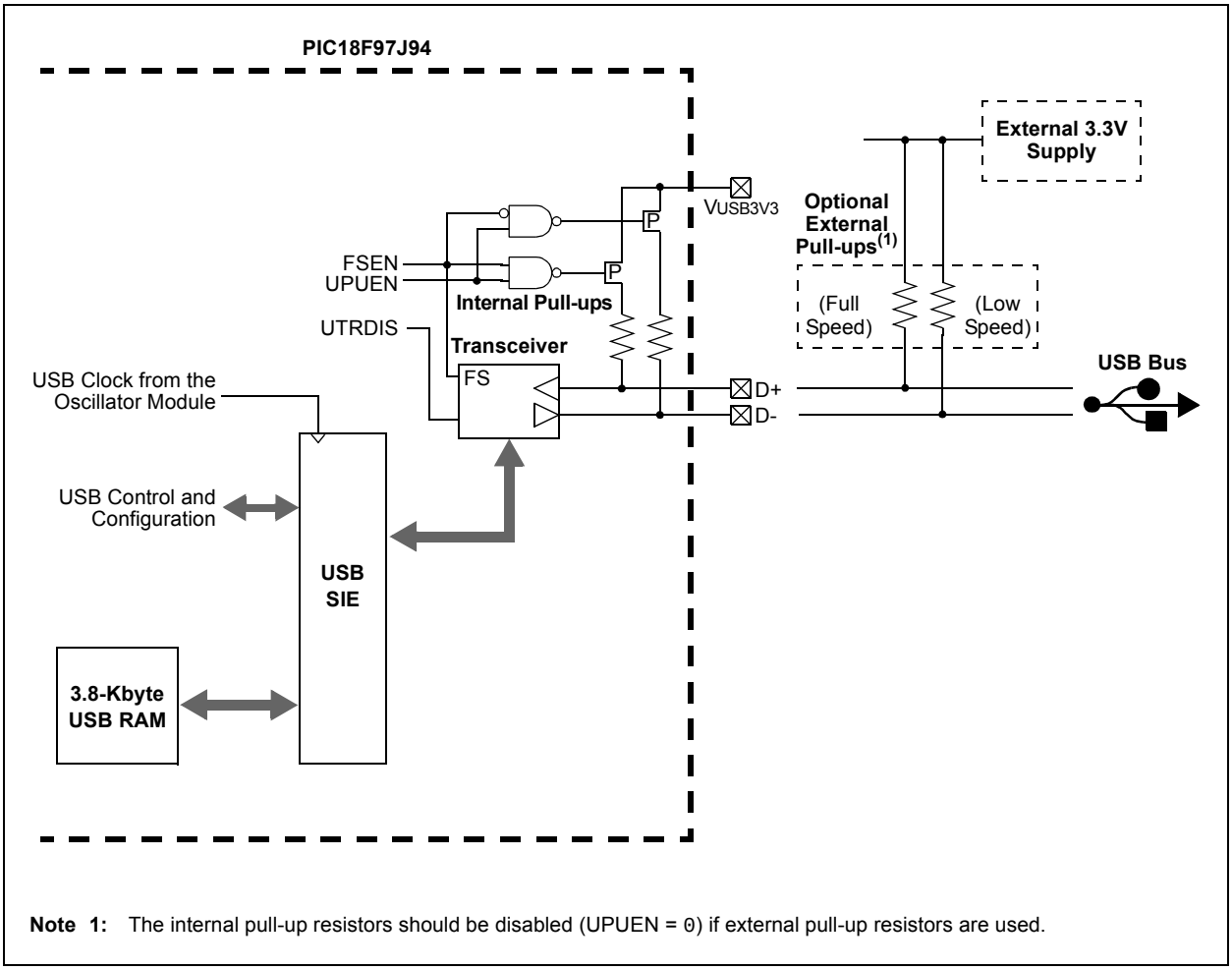
This section describes the details of the USB peripheral. Because of the very specific nature of the module, some knowledge of USB is expected. Some high-level USB information is provided in **Section 27.9 "Overview of USB"** only for application design reference. Designers are encouraged to refer to the official specification published by the USB Implementers Forum (USB-IF) for the latest information. USB Specification Revision 2.0 is the most current specification at the time of publication of this document.

27.1 Overview of the USB Peripheral

PIC18FXXJ94 devices contain a full-speed and low-speed, compatible USB Serial Interface Engine (SIE) that allows fast communication between any USB host and the PIC® MCU. The SIE can be interfaced directly to the USB, utilizing the internal transceiver.

Some special hardware features have been included to improve performance. Dual access port memory in the device's data memory space (USB RAM) has been supplied to share Direct Memory Access (DMA) between the microcontroller core and the SIE. Buffer descriptors are also provided, allowing users to freely program endpoint memory usage within the USB RAM space. Figure 27-1 provides a general overview of the USB peripheral and its features.

FIGURE 27-1: USB PERIPHERAL AND OPTIONS



Note 1: The internal pull-up resistors should be disabled (UPUEN = 0) if external pull-up resistors are used.

27.2 USB Status and Control

The operation of the USB module is configured and managed through three control registers. In addition, a total of 22 registers are used to manage the actual USB transactions. The registers are:

- USB Control Register (UCON)
- USB Configuration Register (UCFG)
- USB Transfer STATUS Register (USTAT)
- USB Device Address Register (UADDR)
- Frame Number Registers (UFRMH:UFRML)
- Endpoint Enable Registers 0 through 15 (UEPn)

27.2.1 USB CONTROL REGISTER (UCON)

The USB Control register (Register 27-1) contains bits needed to control the module behavior during transfers. The register contains bits that control the following:

- Main USB Peripheral Enable
- Ping-Pong Buffer Pointer Reset
- Control of the Suspend mode
- Packet Transfer Disable

In addition, the USB Control register contains a Status bit, SE0 (UCON<5>), which is used to indicate the occurrence of a single-ended zero on the bus. When the USB module is enabled, this bit should be monitored to determine whether the differential data lines have come out of a single-ended zero condition. This helps to differentiate the initial power-up state from the USB Reset signal.

The overall operation of the USB module is controlled by the USBEN bit (UCON<3>). Setting this bit activates the module and resets all of the PPBI bits in the Buffer Descriptor Table (BDT) to '0'. This bit also activates the internal pull-up resistors if they are enabled. Thus, this bit can be used as a soft attach/detach to the USB. Although all status and control bits are ignored when this bit is clear, the module needs to be fully preconfigured prior to setting this bit. The USB clock source should have been already configured for the correct frequency and running. If the PLL is being used, it should be enabled for at least 2 ms (enough time for the PLL to lock) before attempting to set the USBEN bit.

Note:	When disabling the USB module, make sure the SUSPND bit (UCON<1>) is clear prior to clearing the USBEN bit. Clearing the USBEN bit when the module is in the suspended state may prevent the module from fully powering down

REGISTER 27-1: UCON: USB CONTROL REGISTER

U-0	R/W-0	R-x	R/C-0	R/W-0	R/W-0	R/W-0	U-0
—	PPBRST(2)	SE0	PKTDIS	USBEN(1)	RESUME	SUSPND	—
bit 7							bit 0

Legend:	C = Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6 **PPBRST:** Ping-Pong Buffers Reset bit(2)
- 1 = Reset all Ping-Pong Buffer Pointers to the Even Buffer Descriptor (BD) banks
- 0 = Ping-Pong Buffer Pointers are not being reset

bit 5 **SE0:** Live Single-Ended Zero Flag bit
- 1 = Single-ended zero is active on the USB bus
- 0 = No single-ended zero is detected

bit 4 **PKTDIS:** Packet Transfer Disable bit
- 1 = SIE token and packet processing are disabled, automatically set when a SETUP token is received
- 0 = SIE token and packet processing are enabled

bit 3 **USBEN:** USB Module Enable bit(1)
- 1 = USB module and supporting circuitry are enabled (device attached)
- 0 = USB module and supporting circuitry are disabled (device detached)

bit 2 **RESUME:** Resume Signaling Enable bit
- 1 = Resume signaling is activated
- 0 = Resume signaling is disabled

bit 1 **SUSPND:** Suspend USB bit
- 1 = USB module and supporting circuitry are in Power Conserve mode, SIE clock is inactive
- 0 = USB module and supporting circuitry are in normal operation, SIE is clocked at the configured rate

bit 0 **Unimplemented**: Read as ‘0’

Note 1: Make sure the USB clock source is correctly configured before setting this bit.

2: There should be at least four cycles of delay between the setting and PPBRST.

The PPBRST bit (UCON<6>) controls the Reset status when Double-Buffering mode (ping-pong buffering) is used. When the PPBRST bit is set, all Ping-Pong Buffer Pointers are set to the Even buffers. PPBRST has to be cleared by firmware. This bit is ignored in buffering modes not using ping-pong buffering.

The PKTDIS bit (UCON<4>) is a flag indicating that the SIE has disabled packet transmission and reception. This bit is set by the SIE when a SETUP token is received to allow setup processing. This bit cannot be set by the microcontroller, only cleared; clearing it allows the SIE to continue transmission and/or reception. Any pending events within the Buffer Descriptor Table (BDT) will still be available, indicated within the USTAT register's FIFO buffer.

The RESUME bit (UCON<2>) allows the peripheral to perform a remote wake-up by executing resume signaling. To generate a valid remote wake-up, firmware must set RESUME for 10 ms and then clear the bit. For more information on resume signaling, see Sections 7.1.7.5, 11.4.4 and 11.9 in the "*USB 2.0 Specification*".

The SUSPND bit (UCON<1>) places the module and supporting circuitry in a Low-Power mode. The input clock to the SIE is also disabled. This bit should be set by the software in response to an IDLEIF interrupt. It should be reset by the microcontroller firmware after an ACTVIF interrupt is observed. When this bit is active, the device remains attached to the bus but the transceiver outputs remain Idle. The voltage on the V_{USB3V3} pin may vary depending on the value of this bit. Setting this bit before a IDLEIF request will result in unpredictable bus behavior.

Note:	While in Suspend mode, a typical bus-powered USB device is limited to 2.5 mA of current. This is the complete current which may be drawn by the PIC MCU device and its supporting circuitry. Care should be taken to assure minimum current draw when the device enters Suspend mode.

27.2.2 USB CONFIGURATION REGISTER (UCFG)

Prior to communicating over USB, the module's associated internal and/or external hardware must be configured. Most of the configuration is performed with the UCFG register (Register 27-2).The UFCG register contains most of the bits that control the system-level behavior of the USB module. These include:

- Bus Speed (full speed versus low speed)
- On-Chip Pull-up Resistor Enable
- On-Chip Transceiver Enable
- Ping-Pong Buffer Usage

The UCFG register also contains two bits, which aid in module testing, debugging and USB certifications. These bits control output enable state monitoring and eye pattern generation.

Note:	The USB speed, transceiver and pull-up should only be configured during the module setup phase. It is not recommended to switch these settings while the module is enabled.

27.2.2.1 Internal Transceiver

The USB peripheral has a built-in, "*USB 2.0 Specification*", full-speed and low-speed capable transceiver, internally connected to the SIE. This feature is useful for low-cost, single chip applications. The UTRDIS bit (UCFG<3>) controls the transceiver; it is enabled by default (UTRDIS = 0). The FSEN bit (UCFG<2>) controls the transceiver speed; setting this bit enables full-speed operation.

The on-chip USB pull-up resistors are controlled by the UPUEN bit (UCFG<4>). They can only be selected when the on-chip transceiver is enabled.

The internal USB transceiver obtains power from the V_{USB3V3} pin. In order to meet USB signalling level specifications, V_{USB3V3} must be supplied with a voltage source between 3.0V and 3.6V. The best electrical signal quality is obtained when a 3.3V supply is used and locally bypassed with a high quality ceramic capacitor (ex: 0.1 μF). The capacitor should be placed as close as possible to the V_{USB3V3} and V_{SS} pins.

V_{USB3V3} should always be maintained $\geq V_{DD}$. If the USB module is not used, but RC4 or RC5 are used as general purpose inputs, V_{USB3V3} should still be connected to a power source (such as V_{DD}). The input thresholds for the RC4 and RC5 pins are dependent upon the V_{USB3V3} supply level.

The D+ and D- signal lines can be routed directly to their respective pins on the USB connector or cable (for hard-wired applications). No additional resistors, capacitors or magnetic components are required, as the D+ and D- drivers have controlled slew rate and output impedance, intended to match with the characteristic impedance of the USB cable.

In order to achieve optimum USB signal quality, the D+ and D- traces between the microcontroller and USB connector (or cable) should be less than 19 cm long. Both traces should be equal in length and they should be routed parallel to each other. Ideally, these traces should be designed to have a characteristic impedance matching that of the USB cable.

REGISTER 27-2: UCFG: USB CONFIGURATION REGISTER

R/W-0	R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
UTEYE	UOEMON	—	UPUEN(1,2)	UTRDIS(1,3)	FSEN(1)	PPB1	PPB0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **UTEYE:** USB Eye Pattern Test Enable bit
1 = Eye pattern test is enabled
0 = Eye pattern test is disabled

bit 6 **UOEMON:** USB OE Monitor Enable bit
1 = $\overline{\text{UOE}}$ signal is active, indicating intervals during which the D+/D- lines are driving
0 = $\overline{\text{UOE}}$ signal is inactive

bit 5 **Unimplemented:** Read as '0'

bit 4 **UPUEN:** USB On-Chip Pull-up Enable bit(1,2)
1 = On-chip pull-up is enabled (pull-up on D+ with FSEN = 1 or D- with FSEN = 0)
0 = On-chip pull-up is disabled

bit 3 **UTRDIS:** On-Chip Transceiver Disable bit(1,3)
1 = On-chip transceiver is disabled
0 = On-chip transceiver is active

bit 2 **FSEN:** Full-Speed Enable bit(1)
1 = Full-speed device: Controls transceiver edge rates; requires input clock at 48 MHz
0 = Low-speed device: Controls transceiver edge rates; requires input clock at 6 MHz

bit 1-0 **PPB<1:0>:** Ping-Pong Buffers Configuration bits
11 = Even/Odd ping-pong buffers are enabled for Endpoints 1 to 15
10 = Even/Odd ping-pong buffers are enabled for all endpoints
01 = Even/Odd ping-pong buffer are enabled for OUT Endpoint 0
00 = Even/Odd ping-pong buffers are disabled

Note 1: The UPUEN, UTRDIS and FSEN bits should never be changed while the USB module is enabled. These values must be preconfigured prior to enabling the module.
2: This bit is only valid when the on-chip transceiver is active (UTRDIS = 0); otherwise, it is ignored.
3: If UTRDIS is set, the $\overline{\text{UOE}}$ signal will be active, independent of the UOEMON bit setting.

27.2.2.2 Internal Pull-up Resistors

The PIC18FXXJ94 devices have built-in pull-up resistors, designed to meet the requirements for low-speed and full-speed USB. The UPUEN bit (UCFG<4>) enables the internal pull-ups. Figure 27-1 shows the pull-ups and their control.

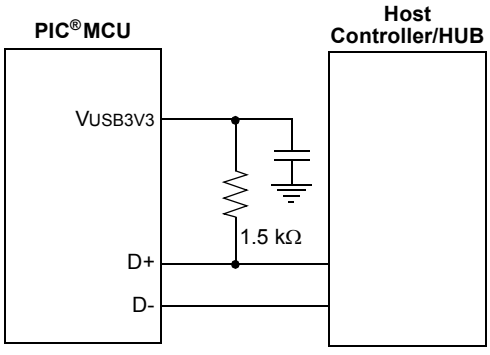
Note: A compliant USB device should never source any current onto the +5V VBUS line of the USB cable. Additionally, USB devices should not source any current on the D+ and D- data lines whenever the +5V VBUS line is less than 1.17V. In order to be USB compliant, applications which are not purely bus-powered should monitor the VBUS line, and avoid turning on the USB module and the D+ or D- pull-up resistor until VBUS is greater than 1.17V. VBUS can be connected to and monitored by a 5V tolerant I/O pin, or if a resistive divider is used, by an analog capable pin.

27.2.2.3 External Pull-up Resistors

External pull-ups may also be used. The VUSB3V3 pin may be used to pull up D+ or D-. The pull-up resistor must be 1.5 kΩ (±5%) as required by the USB specifications.

Figure 27-2 provides an example of external circuitry.

FIGURE 27-2: EXTERNAL CIRCUITRY



Note: The above setting shows a typical connection for a full-speed configuration using an on-chip regulator and an external pull-up resistor.

27.2.2.4 Ping-Pong Buffer Configuration

The usage of ping-pong buffers is configured using the PPB<1:0> bits. Refer to **Section 27.4.4 "Ping-Pong Buffering"** for a complete explanation of the ping-pong buffers.

27.2.2.5 Eye Pattern Test Enable

An automatic eye pattern test can be generated by the module when the UCFG<7> bit is set. The eye pattern output will be observable based on module settings, meaning that the user is first responsible for configuring the SIE clock settings, pull-up resistor and Transceiver mode. In addition, the module has to be enabled.

Once UTEYE is set, the module emulates a switch from a receive to transmit state and will start transmitting a J-K-J-K bit sequence (K-J-K-J for full speed). The sequence will be repeated indefinitely while the Eye Pattern Test mode is enabled.

Note that this bit should never be set while the module is connected to an actual USB system. This Test mode is intended for board verification to aid with USB certification tests. It is intended to show a system developer the noise integrity of the USB signals which can be affected by board traces, impedance mismatches and proximity to other system components. It does not properly test the transition from a receive to a transmit state. Although the eye pattern is not meant to replace the more complex USB certification test, it should aid during first order system debugging.

27.2.3 USB STATUS REGISTER (USTAT)

The USB STATUS register reports the transaction status within the SIE. When the SIE issues a USB transfer complete interrupt, USTAT should be read to determine the status of the transfer. USTAT contains the transfer endpoint number, direction and Ping-Pong Buffer Pointer value (if used).

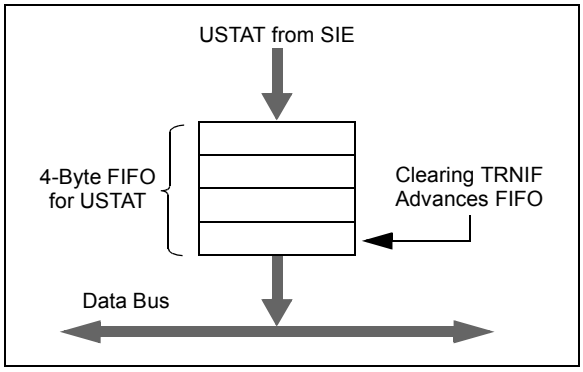
Note:	The data in the USB STATUS register is valid only when the TRNIF interrupt flag is asserted.

The USTAT register is actually a read window into a 4-byte status FIFO, maintained by the SIE. It allows the microcontroller to process one transfer while the SIE processes additional endpoints (Figure 27-3). When the SIE completes using a buffer for reading or writing data, it updates the USTAT register. If another USB transfer is performed before a transaction complete interrupt is serviced, the SIE will store the status of the next transfer into the status FIFO.

Clearing the Transfer Complete Flag bit, TRNIF, causes the SIE to advance the FIFO. If the next data in the FIFO holding register is valid, the SIE will reassert the interrupt within 5 TCY of clearing TRNIF. If no additional data is present, TRNIF will remain clear; USTAT data will no longer be reliable.

Note:	If an endpoint request is received while the USTAT FIFO is full, the SIE will automatically issue a NAK back to the host.

FIGURE 27-3: USTAT FIFO



REGISTER 27-3: USTAT: USB STATUS REGISTER (ACCESS F64H)

U-0	R-x	R-x	R-x	R-x	R-x	R-x	U-0
—	ENDP3	ENDP2	ENDP1	ENDP0	DIR	PPBI(1)	—
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as ‘0’	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as ‘0’

bit 6-3 **ENDP<3:0>:** Encoded Number of Last Endpoint Activity bits
(represents the number of the BDT updated by the last USB transfer)
1111 = Endpoint 15
1110 = Endpoint 14
.
.
.
0001 = Endpoint 1
0000 = Endpoint 0

bit 2 **DIR:** Last BD Direction Indicator bit
1 = The last transaction was an IN token
0 = The last transaction was an OUT or SETUP token

bit 1 **PPBI:** Ping-Pong BD Pointer Indicator bit(1)
1 = The last transaction was to the Odd BD bank
0 = The last transaction was to the Even BD bank

bit 0 **Unimplemented:** Read as ‘0’

Note 1: This bit is only valid for endpoints with available Even and Odd BD registers.

27.2.4 USB ENDPOINT CONTROL

Each of the 16 possible bidirectional endpoints has its own independent control register, UEPn (where 'n' represents the endpoint number). Each register has an identical complement of control bits.

Register 27-4 provides the prototype.

The EPHSHK bit (UEPn<4>) controls handshaking for the endpoint; setting this bit enables USB handshaking. Typically, this bit is always set except when using isochronous endpoints.

The EPCONDIS bit (UEPn<3>) is used to enable or disable USB control operations (SETUP) through the endpoint. Clearing this bit enables SETUP transactions. Note that the corresponding EPINEN and EPOUTEN bits must be set to enable IN and OUT transactions. For Endpoint 0, this bit should always be cleared since the USB specifications identify Endpoint 0 as the default control endpoint.

The EPOUTEN bit (UEPn<2>) is used to enable or disable USB OUT transactions from the host. Setting this bit enables OUT transactions. Similarly, the EPINEN bit (UEPn<1>) enables or disables USB IN transactions from the host.

The EPSTALL bit (UEPn<0>) is used to indicate a STALL condition for the endpoint. If a STALL is issued on a particular endpoint, the EPSTALL bit for that endpoint pair will be set by the SIE. This bit remains set until it is cleared through firmware or until the SIE is reset.

REGISTER 27-4: UEPn: USB ENDPOINT n CONTROL REGISTER (UEP0 THROUGH UEP15)

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as '0'

bit 4 **EPHSHK:** Endpoint Handshake Enable bit
1 = Endpoint handshake is enabled
0 = Endpoint handshake is disabled (typically used for isochronous endpoints)

bit 3 **EPCONDIS:** Bidirectional Endpoint Control bit
If EPOUTEN = 1 and EPINEN = 1:
1 = Disables Endpoint n from control transfers; only IN and OUT transfers are allowed
0 = Enables Endpoint n for control (SETUP) transfers; IN and OUT transfers are also allowed

bit 2 **EPOUTEN:** Endpoint Output Enable bit
1 = Endpoint n output is enabled
0 = Endpoint n output is disabled

bit 1 **EPINEN:** Endpoint Input Enable bit
1 = Endpoint n input is enabled
0 = Endpoint n input is disabled

bit 0 **EPSTALL:** Endpoint Stall Indicator bit
1 = Endpoint n has issued one or more STALL packets
0 = Endpoint n has not issued any STALL packets

27.2.5 USB ADDRESS REGISTER (UADDR)

The USB Address register contains the unique USB address that the peripheral will decode when active. UADDR is reset to 00h when a USB Reset is received, indicated by URSTIF, or when a Reset is received from the microcontroller. The USB address must be written by the microcontroller during the USB setup phase (enumeration) as part of the Microchip USB firmware support.

27.2.6 USB FRAME NUMBER REGISTERS (UFRMH:UFRML)

The Frame Number registers contain the 11-bit frame number. The low-order byte is contained in UFRML, while the three high-order bits are contained in UFRMH. The register pair is updated with the current frame number whenever a SOF token is received. For the microcontroller, these registers are read-only. The Frame Number registers are primarily used for isochronous transfers. The contents of the UFRMH and UFRML registers are only valid when the 48 MHz SIE clock is active (i.e., contents are inaccurate when SUSPND (UCON<1>) bit = 1).

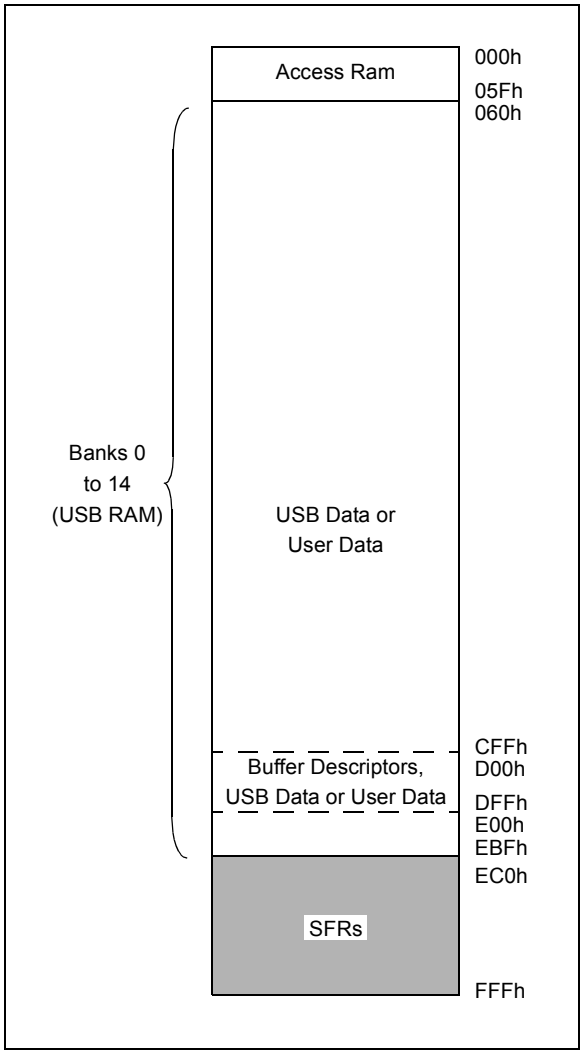
27.3 USB RAM

USB data moves between the microcontroller core and the SIE through a memory space, known as the USB RAM. This is a special dual access memory that is mapped into the normal data memory space in Banks 0 through 14 (00h to EBFh), for a total of 3.8 Kbytes (Figure 27-4).

Bank 13 (D00h through DFFh) is used specifically for endpoint buffer control, while Banks 0 through 12 and Bank 14 are available for USB data. Depending on the type of buffering being used, all but 8 bytes of Bank 13 may also be available for use as USB buffer space.

Although USB RAM is available to the microcontroller as data memory, the sections that are being accessed by the SIE should not be accessed by the microcontroller. A semaphore mechanism is used to determine the access to a particular buffer at any given time. This is discussed in **Section 27.4.1.1 "Buffer Ownership"**.

FIGURE 27-4: IMPLEMENTATION OF USB RAM IN DATA MEMORY SPACE



27.4 Buffer Descriptors and the Buffer Descriptor Table

The registers in Bank 13 are used specifically for endpoint buffer control in a structure known as the Buffer Descriptor Table (BDT). This provides a flexible method for users to construct and control endpoint buffers of various lengths and configuration.

The BDT is composed of Buffer Descriptors (BD) which are used to define and control the actual buffers in the USB RAM space. Each BD, in turn, consists of four registers, where n represents one of the 64 possible BDs (range of 0 to 63):

- BDnSTAT: BD STATUS Register
- BDnCNT: BD Byte Count Register
- BDnADRL: BD Address Low Register
- BDnADRH: BD Address High Register

BDs always occur as a four-byte block in the sequence, BDnSTAT:BDnCNT:BDnADRL:BDnADRH. The address of BDnSTAT is always an offset of (4n – 1, in hexadecimal) from D00h, with n being the buffer descriptor number.

Depending on the buffering configuration used (**Section 27.4.4 "Ping-Pong Buffering"**), there are up to 32, 33 or 64 sets of buffer descriptors. At a minimum, the BDT must be at least 8 bytes long. This is because the USB Specification mandates that every device must have Endpoint 0 with both input and output for initial setup. Depending on the endpoint and buffering configuration, the BDT can be as long as 256 bytes.

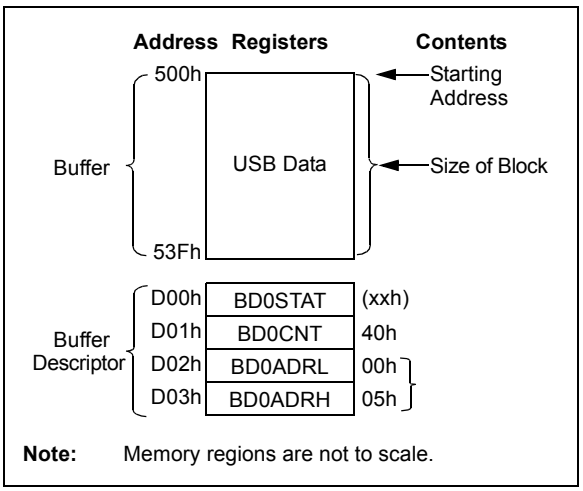
Although they can be thought of as Special Function Registers, the Buffer Descriptor Status and Address registers are not hardware mapped, as conventional microcontroller SFRs in Bank 15 are. If the endpoint corresponding to a particular BD is not enabled, its registers are not used. Instead of appearing as unimplemented addresses, however, they appear as available RAM. Only when an endpoint is enabled by setting the UEPn<1> bit does the memory at those addresses become functional as BD registers. As with any address in the data memory space, the BD registers have an indeterminate value on any device Reset.

Figure 27-5 provides an example of a BD for a 64-byte buffer, starting at 500h. A particular set of BD registers is only valid if the corresponding endpoint has been enabled using the UEPn register. All BD registers are available in USB RAM. The BD for each endpoint should be set up prior to enabling the endpoint.

27.4.1 BD STATUS AND CONFIGURATION

Buffer descriptors not only define the size of an endpoint buffer, but also determine its configuration and control. Most of the configuration is done with the BD STATUS register, BDnSTAT. Each BD has its own unique and correspondingly numbered BDnSTAT register.

FIGURE 27-5: EXAMPLE OF A BUFFER DESCRIPTOR



Note: Memory regions are not to scale.

Unlike other control registers, the bit configuration for the BDnSTAT register is context-sensitive. There are two distinct configurations, depending on whether the microcontroller or the USB module is modifying the BD and buffer at a particular time. Only three bit definitions are shared between the two.

27.4.1.1 Buffer Ownership

Because the buffers and their BDs are shared between the CPU and the USB module, a simple semaphore mechanism is used to distinguish which is allowed to update the BD and associated buffers in memory.

This is done by using the UOWN bit (BDnSTAT<7>) as a semaphore to distinguish which is allowed to update the BD and associated buffers in memory. UOWN is the only bit that is shared between the two configurations of BDnSTAT.

When UOWN is clear, the BD entry is "owned" by the microcontroller core. When the UOWN bit is set, the BD entry and the buffer memory are "owned" by the USB peripheral. The core should not modify the BD or its corresponding data buffer during this time. Note that the microcontroller core can still read BDnSTAT while the SIE owns the buffer and vice versa.

The buffer descriptors have a different meaning based on the source of the register update. Prior to placing ownership with the USB peripheral, the user can configure the basic operation of the peripheral through the BDnSTAT bits. During this time, the byte count and buffer location registers can also be set.

When UOWN is set, the user can no longer depend on the values that were written to the BDs. From this point, the SIE updates the BDs as necessary, overwriting the original BD values. The BDnSTAT register is updated by the SIE with the token PID and the transfer count, BDnCNT, is updated.

The BDnSTAT byte of the BDT should always be the last byte updated when preparing to arm an endpoint. The SIE will clear the UOWN bit when a transaction has completed.

No hardware mechanism exists to block access when the UOWN bit is set. Thus, unexpected behavior can occur if the microcontroller attempts to modify memory when the SIE owns it. Similarly, reading such memory may produce inaccurate data until the USB peripheral returns ownership to the microcontroller.

27.4.1.2 BDnSTAT Register (CPU Mode)

When UOWN = 0, the microcontroller core owns the BD. At this point, the other seven bits of the register take on control functions.

The Data Toggle Sync Enable bit, DTSEN (BDnSTAT<3>), controls data toggle parity checking. Setting DTSEN enables data toggle synchronization by the SIE. When enabled, it checks the data packet's parity against the value of DTS (BDnSTAT<6>). If a packet arrives with an incorrect synchronization, the data will essentially be ignored. It will not be written to the USB RAM and the USB transfer complete interrupt flag will not be set. The SIE will send an ACK token back to the host to Acknowledge receipt, however. The effects of the DTSEN bit on the SIE are summarized in Table 27-1.

The Buffer Stall bit, BSTALL (BDnSTAT<2>), provides support for control transfers, usually one-time stalls on Endpoint 0. It also provides support for the SET_FEATURE/CLEAR_FEATURE commands specified in Chapter 9 of the USB Specification; typically, continuous STALLs to any endpoint other than the default control endpoint.

The BSTALL bit enables buffer stalls. Setting BSTALL causes the SIE to return a STALL token to the host if a received token would use the BD in that location. The EPSTALL bit in the corresponding UEPn Control register is set and a STALL interrupt is generated when a STALL is issued to the host. The UOWN bit remains set and the BDs are not changed unless a SETUP token is received. In this case, the STALL condition is cleared and the ownership of the BD is returned to the microcontroller core.

The BC<9:8> bits (BDnSTAT<1:0>) store the two most significant digits of the SIE byte count. The lower 8 digits are stored in the corresponding BDnCNT register. See **Section 27.4.2 "BD Byte Count"** for more information.

TABLE 27-1: EFFECT OF DTSEN BIT ON ODD/EVEN (DATA0/DATA1) PACKET RECEPTION

OUT Packet from Host	BDnSTAT Settings		Device Response after Receiving Packet			
	DTSEN	DTS	Handshake	UOWN	TRNIF	BDnSTAT and USTAT Status
DATA0	1	0	$\overline{\text{ACK}}$	0	1	Updated
DATA1	1	0	$\overline{\text{ACK}}$	1	0	Not Updated
DATA0	1	1	$\overline{\text{ACK}}$	1	0	Not Updated
DATA1	1	1	$\overline{\text{ACK}}$	0	1	Updated
Either	0	x	$\overline{\text{ACK}}$	0	1	Updated
Either, with error	x	x	NAK	1	0	Not Updated

Legend: x = don't care

REGISTER 27-5: BDnSTAT: BUFFER DESCRIPTOR n STATUS REGISTER (BD0STAT THROUGH BD63STAT), CPU MODE

R/W-x	R/W-x	U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x
UOWN(1)	DTS(2)	—(3)	—(3)	DTSEN	BSTALL	BC9	BC8
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **UOWN:** USB Own bit(1)
0 = The microcontroller core owns the BD and its corresponding buffer

bit 6 **DTS:** Data Toggle Synchronization bit(2)
1 = Data 1 packet
0 = Data 0 packet

bit 5-4 **Unimplemented:** These bits should always be programmed to '0'(3)

bit 3 **DTSEN:** Data Toggle Synchronization Enable bit
1 = Data toggle synchronization is enabled; data packets with incorrect Sync value will be ignored, except for a SETUP transaction, which is accepted even if the data toggle bits do not match
0 = No data toggle synchronization is performed

bit 2 **BSTALL:** Buffer Stall Enable bit
1 = Buffer stall is enabled; STALL handshake issued if a token is received that would use the BD in the given location (UOWN bit remains set, BD value is unchanged)
0 = Buffer stall is disabled

bit 1-0 **BC<9:8>:** Byte Count 9 and 8 bits
The byte count bits represent the number of bytes that will be transmitted for an IN token or received during an OUT token. Together with BC<7:0>, the valid byte counts are 0-1023.

Note 1: This bit must be initialized by the user to the desired value prior to enabling the USB module.
2: This bit is ignored unless DTSEN = 1.
3: If these bits are set, USB communication may not work. Hence, these bits should always be maintained as '0'.

27.4.1.3 BDnSTAT Register (SIE Mode)

When the BD and its buffer are owned by the SIE, most of the bits in BDnSTAT take on a different meaning. The configuration is shown in Register 27-6. Once UOWN is set, any data or control settings previously written there by the user will be overwritten with data from the SIE.

The BDnSTAT register is updated by the SIE with the token Packet Identifier (PID) which is stored in BDnSTAT<5:2>. The transfer count in the corresponding BDnCNT register is updated. Values that overflow the 8-bit register carry over to the two most significant digits of the count, stored in BDnSTAT<1:0>.

27.4.2 BD BYTE COUNT

The byte count represents the total number of bytes that will be transmitted during an IN transfer. After an IN transfer, the SIE will return the number of bytes sent to the host.

For an OUT transfer, the byte count represents the maximum number of bytes that can be received and stored in USB RAM. After an OUT transfer, the SIE will return the actual number of bytes received. If the number of bytes received exceeds the corresponding byte count, the data packet will be rejected and a NAK handshake will be generated. When this happens, the byte count will not be updated.

The 10-bit byte count is distributed over two registers. The lower 8 bits of the count reside in the BDnCNT register; the upper two bits reside in BDnSTAT<1:0>. This represents a valid byte range of 0 to 1023.

27.4.3 BD ADDRESS VALIDATION

The BD Address register pair contains the starting RAM address location for the corresponding endpoint buffer. No mechanism is available in hardware to validate the BD address.

If the value of the BD address does not point to an address in the USB RAM, or if it points to an address within another endpoint's buffer, data is likely to be lost or overwritten. Similarly, overlapping a receive buffer (OUT endpoint) with a BD location in use can yield unexpected results. When developing USB applications, the user may want to consider the inclusion of software-based address validation in their code.

REGISTER 27-6: BDnSTAT: BUFFER DESCRIPTOR n STATUS REGISTER (BD0STAT THROUGH BD63STAT), SIE MODE (DATA RETURNED BY THE SIE TO THE MCU)

R/W-x	r-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
UOWN	r	PID3	PID2	PID1	PID0	BC9	BC8
bit 7							bit 0

Legend:	r = Reserved bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **UOWN:** USB Own bit
1 = The SIE owns the BD and its corresponding buffer

bit 6 **Reserved:** Not written by the SIE

bit 5-2 **PID<3:0>:** Packet Identifier bits
The received token PID value of the last transfer (IN, OUT or SETUP transactions only).

bit 1-0 **BC<9:8>:** Byte Count 9 and 8 bits
These bits are updated by the SIE to reflect the actual number of bytes received on an OUT transfer and the actual number of bytes transmitted on an IN transfer.

27.4.4 PING-PONG BUFFERING

An endpoint is defined to have a ping-pong buffer when it has two sets of BD entries: one set for an Even transfer and one set for an Odd transfer. This allows the CPU to process one BD while the SIE is processing the other BD. Double-buffering BDs in this way allows for maximum throughput to/from the USB.

The USB module supports four modes of operation:

- No ping-pong support
- Ping-pong buffer support for OUT Endpoint 0 only
- Ping-pong buffer support for all endpoints
- Ping-pong buffer support for all other endpoints except Endpoint 0

The ping-pong buffer settings are configured using the PPB<1:0> bits in the UCFG register.

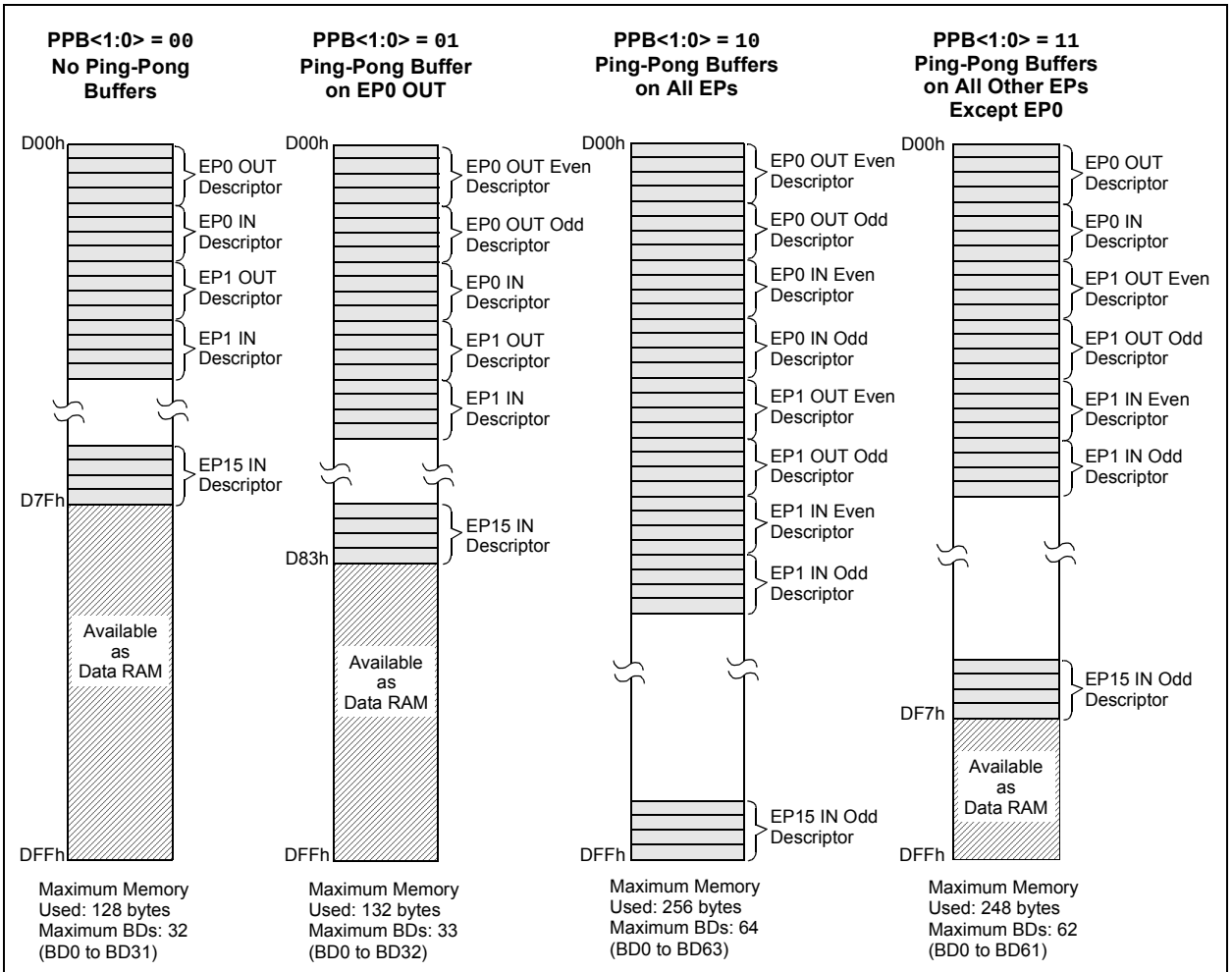
The USB module keeps track of the Ping-Pong Pointer individually for each endpoint. All pointers are initially reset to the Even BD when the module is enabled. After the completion of a transaction (UOWN cleared by the SIE), the pointer is toggled to the Odd BD. After the completion of the next transaction, the pointer is toggled back to the Even BD and so on.

The Even/Odd status of the last transaction is stored in the PPBI bit of the USTAT register. The user can reset all Ping-Pong Pointers to Even using the PPBRST bit.

Figure 27-6 shows the four different modes of operation and how USB RAM is filled with the BDs.

BDs have a fixed relationship to a particular endpoint, depending on the buffering configuration. Table 27-2 provides the mapping of BDs to endpoints. This relationship also means that gaps may occur in the BDT if endpoints are not enabled contiguously. This, theoretically, means that the BDs for disabled endpoints could be used as buffer space. In practice, users should avoid using such spaces in the BDT unless a method of validating BD addresses is implemented.

FIGURE 27-6: BUFFER DESCRIPTOR TABLE MAPPING FOR BUFFERING MODES



Note: Memory area is not shown to scale.

TABLE 27-2: ASSIGNMENT OF BUFFER DESCRIPTORS FOR THE DIFFERENT BUFFERING MODES

Endpoint	BDs Assigned to Endpoint							
	Mode 0 (No Ping-Pong)		Mode 1 (Ping-Pong on EP0 OUT)		Mode 2 (Ping-Pong on All EPs)		Mode 3 (Ping-Pong on All Other EPs, except EP0)	
	Out	In	Out	In	Out	In	Out	In
0	0	1	0 (E), 1 (O)	2	0 (E), 1 (O)	2 (E), 3 (O)	0	1
1	2	3	3	4	4 (E), 5 (O)	6 (E), 7 (O)	2 (E), 3 (O)	4 (E), 5 (O)
2	4	5	5	6	8 (E), 9 (O)	10 (E), 11 (O)	6 (E), 7 (O)	8 (E), 9 (O)
3	6	7	7	8	12 (E), 13 (O)	14 (E), 15 (O)	10 (E), 11 (O)	12 (E), 13 (O)
4	8	9	9	10	16 (E), 17 (O)	18 (E), 19 (O)	14 (E), 15 (O)	16 (E), 17 (O)
5	10	11	11	12	20 (E), 21 (O)	22 (E), 23 (O)	18 (E), 19 (O)	20 (E), 21 (O)
6	12	13	13	14	24 (E), 25 (O)	26 (E), 27 (O)	22 (E), 23 (O)	24 (E), 25 (O)
7	14	15	15	16	28 (E), 29 (O)	30 (E), 31 (O)	26 (E), 27 (O)	28 (E), 29 (O)
8	16	17	17	18	32 (E), 33 (O)	34 (E), 35 (O)	30 (E), 31 (O)	32 (E), 33 (O)
9	18	19	19	20	36 (E), 37 (O)	38 (E), 39 (O)	34 (E), 35 (O)	36 (E), 37 (O)
10	20	21	21	22	40 (E), 41 (O)	42 (E), 43 (O)	38 (E), 39 (O)	40 (E), 41 (O)
11	22	23	23	24	44 (E), 45 (O)	46 (E), 47 (O)	42 (E), 43 (O)	44 (E), 45 (O)
12	24	25	25	26	48 (E), 49 (O)	50 (E), 51 (O)	46 (E), 47 (O)	48 (E), 49 (O)
13	26	27	27	28	52 (E), 53 (O)	54 (E), 55 (O)	50 (E), 51 (O)	52 (E), 53 (O)
14	28	29	29	30	56 (E), 57 (O)	58 (E), 59 (O)	54 (E), 55 (O)	56 (E), 57 (O)
15	30	31	31	32	60 (E), 61 (O)	62 (E), 63 (O)	58 (E), 59 (O)	60 (E), 61 (O)

Legend: (E) = Even transaction buffer, (O) = Odd transaction buffer

TABLE 27-3: SUMMARY OF USB BUFFER DESCRIPTOR TABLE REGISTERS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BDnSTAT(1)	UOWN	DTS(4)	PID3(2)	PID2(2)	PID1(2) DTSEN(3)	PID0(2) BSTALL(3)	BC9	BC8
BDnCNT(1)	Byte Count							
BDnADRL(1)	Buffer Address Low							
BDnADRH(1)	Buffer Address High							

Note 1: For buffer descriptor registers, n may have a value of 0 to 63. For the sake of brevity, all 64 registers are shown as one generic prototype. All registers have indeterminate Reset values (xxxx xxxx).

2: Bits 5 through 2 of the BDnSTAT register are used by the SIE to return PID<3:0> values once the register is turned over to the SIE (UOWN bit is set). Once the registers have been under SIE control, the values written for DTSEN and BSTALL are no longer valid.

3: Prior to turning the buffer descriptor over to the SIE (UOWN bit is cleared), bits 5 through 2 of the BDnSTAT register are used to configure the DTSEN and BSTALL settings.

4: This bit is ignored unless DTSEN = 1.

27.5 USB Interrupts

The USB module can generate multiple interrupt conditions. To accommodate all of these interrupt sources, the module is provided with its own interrupt logic structure, similar to that of the microcontroller. USB interrupts are enabled with one set of control registers and trapped with a separate set of flag registers. All sources are funneled into a single USB Oscillator Fail Interrupt Flag bit, USBIF (PIR2<4>), in the microcontroller's interrupt logic.

Figure 27-7 provides the interrupt logic for the USB module. There are two layers of interrupt registers in the USB module. The top level consists of overall USB status interrupts; these are enabled and flagged in the UIE and UIR registers, respectively. The second level consists of USB error conditions, which are enabled and flagged in the UEIR and UEIE registers. An interrupt condition in any of these triggers a USB Error Interrupt Flag (UERRIF) in the top level.

Interrupts may be used to trap routine events in a USB transaction. Figure 27-8 provides some common events within a USB frame and its corresponding interrupts.

FIGURE 27-7: USB INTERRUPT LOGIC FUNNEL

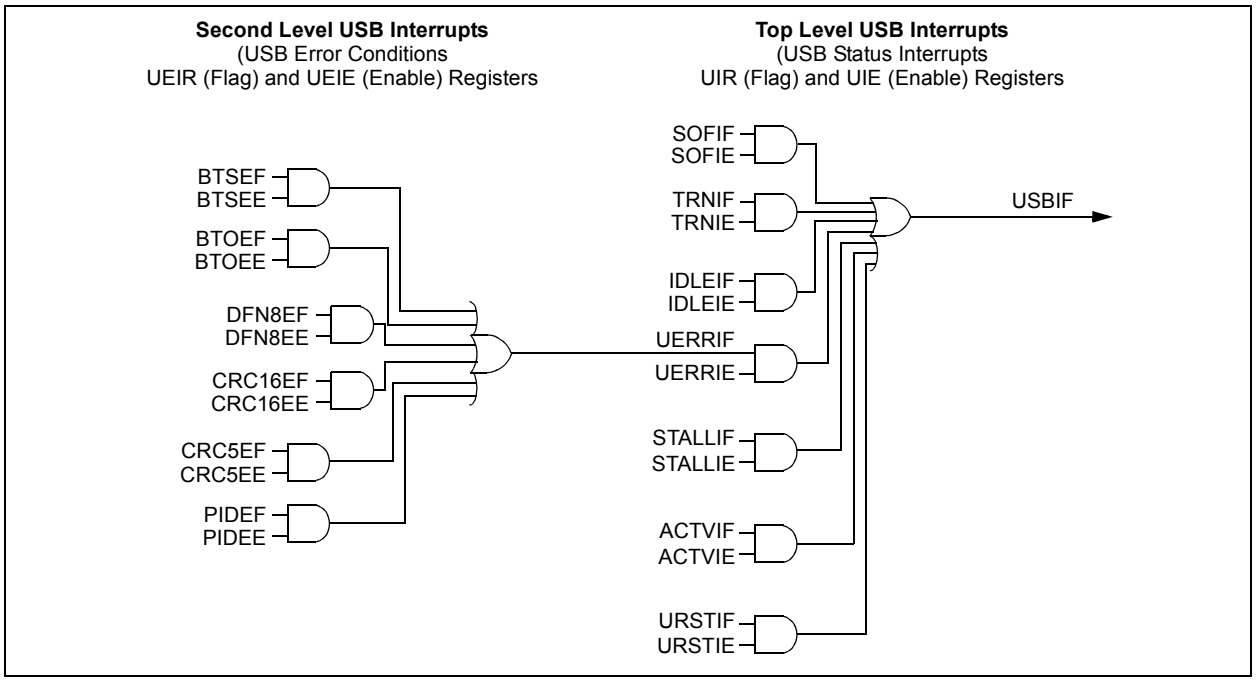
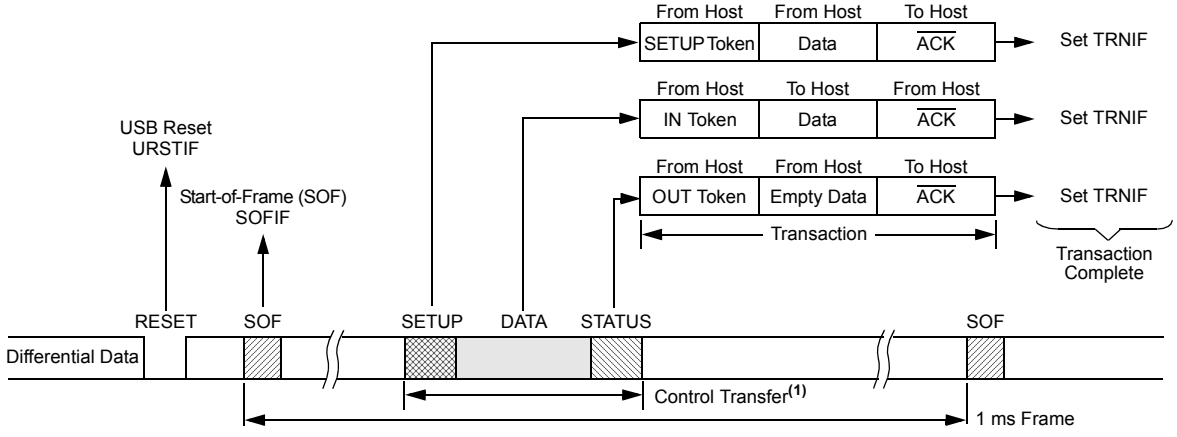


FIGURE 27-8: EXAMPLE OF A USB TRANSACTION AND INTERRUPT EVENTS



Note 1: The control transfer shown here is only an example showing events that can occur for every transaction. Typical control transfers will spread across multiple frames.

27.5.1 USB INTERRUPT STATUS REGISTER (UIR)

The USB Interrupt STATUS register (Register 27-7) contains the flag bits for each of the USB status interrupt sources. Each of these sources has a corresponding interrupt enable bit in the UIE register. All of the USB status flags are ORed together to generate the USBIF interrupt flag for the microcontroller's interrupt funnel.

Once an interrupt bit has been set by the SIE, it must be cleared in software by writing a '0'. The flag bits can also be set in software, which can aid in firmware debugging.

When the USB module is in the Low-Power Suspend mode (UCON<1> = 1), the SIE does not get clocked. When in this state, the SIE cannot process packets, and therefore, cannot detect new interrupt conditions other than the Activity Detect Interrupt, ACTVIF. The ACTVIF bit is typically used by USB firmware to detect when the microcontroller should bring the USB module out of the Low-Power Suspend mode (UCON<1> = 0).

Register 27-7: UIR: USB INTERRUPT STATUS REGISTER (ACCESS F62h)

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R/W-0
—	SOFIF	STALLIF	IDLEIF(1)	TRNIF(2)	ACTVIF(3)	UERRIF(4)	URSTIF
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6 **SOFIF:** Start-of-Frame Token Interrupt bit
1 = A Start-of-Frame token is received by the SIE
0 = No Start-of-Frame token is received by the SIE

bit 5 **STALLIF:** A STALL Handshake Interrupt bit
1 = A STALL handshake was sent by the SIE
0 = A STALL handshake has not been sent

bit 4 **IDLEIF:** Idle Detect Interrupt bit(1)
1 = Idle condition is detected (constant Idle state of 3 ms or more)
0 = No Idle condition is detected

bit 3 **TRNIF:** Transaction Complete Interrupt bit(2)
1 = Processing of pending transaction is complete; read the USTAT register for endpoint information
0 = Processing of pending transaction is not complete or no transaction is pending

bit 2 **ACTVIF:** Bus Activity Detect Interrupt bit(3)
1 = Activity on the D+/D- lines was detected
0 = No activity detected on the D+/D- lines

bit 1 **UERRIF:** USB Error Condition Interrupt bit(4)
1 = An unmasked error condition has occurred
0 = No unmasked error condition has occurred.

bit 0 **URSTIF:** USB Reset Interrupt bit
1 = Valid USB Reset occurred; 00h is loaded into the UADDR register
0 = No USB Reset has occurred

Note 1: Once an Idle state is detected, the user may want to place the USB module in Suspend mode.
2: Clearing this bit will cause the USTAT FIFO to advance (valid only for IN, OUT and SETUP tokens).
3: This bit is typically unmasked only following the detection of a UIDLE interrupt event.
4: Only error conditions enabled through the UEIE register will set this bit. This bit is a Status bit only and cannot be set or cleared by the user.

27.5.1.1 Bus Activity Detect Interrupt Bit (ACTVIF)

The ACTVIF bit cannot be cleared immediately after the USB module wakes up from Suspend mode or while the USB module is suspended. A few clock cycles are required to synchronize the internal hardware state machine before the ACTVIF bit can be cleared by firmware. Clearing the ACTVIF bit before the internal hardware is synchronized may not have an effect on the value of ACTVIF. Additionally, if the USB module uses the clock from the 96 MHz PLL source, then after clearing the SUSPND bit, the USB module may not be immediately operational while waiting for the 96 MHz PLL to lock. The application code should clear the ACTVIF flag as provided in Example 27-1.

Note:	Only one ACTVIF interrupt is generated when resuming from the USB bus Idle condition. If user firmware clears the ACTVIF bit, the bit will not immediately become set again, even when there is continuous bus traffic. Bus traffic must cease long enough to generate another IDLEIF condition before another ACTVIF interrupt can be generated.

EXAMPLE 27-1: CLEARING ACTVIF BIT (UIR<2>)

Assembly:

```
    BCF     UCON, SUSPND
LOOP:
    BTFSS   UIR, ACTVIF
    BRA     DONE
    BCF     UIR, ACTVIF
    BRA     LOOP
DONE:
```

C:

```
UCONbits.SUSPND = 0;
while (UIRbits.ACTVIF) { UIRbits.ACTVIF = 0; }
```

27.5.2 USB INTERRUPT ENABLE REGISTER (UIE)

The USB Interrupt Enable (UIE) register (Register 27-8) contains the enable bits for the USB status interrupt sources. Setting any of these bits will enable the respective interrupt source in the UIR register.

The values in this register only affect the propagation of an interrupt condition to the microcontroller's interrupt logic. The flag bits are still set by their interrupt conditions, allowing them to be polled and serviced without actually generating an interrupt.

Register 27-8: UIE: USB INTERRUPT ENABLE REGISTER

U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	SOFIE	STALLIE	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **Unimplemented:** Read as '0'

bit 6 **SOFIE:** Start-of-Frame Token Interrupt Enable bit
1 = Start-of-Frame token interrupt is enabled
0 = Start-of-Frame token interrupt is disabled

bit 5 **STALLIE:** STALL Handshake Interrupt Enable bit
1 = STALL interrupt is enabled
0 = STALL interrupt is disabled

bit 4 **IDLEIE:** Idle Detect Interrupt Enable bit
1 = Idle detect interrupt is enabled
0 = Idle detect interrupt is disabled

bit 3 **TRNIE:** Transaction Complete Interrupt Enable bit
1 = Transaction interrupt is enabled
0 = Transaction interrupt is disabled

bit 2 **ACTVIE:** Bus Activity Detect Interrupt Enable bit
1 = Bus activity detect interrupt is enabled
0 = Bus activity detect interrupt is disabled

bit 1 **UERRIE:** USB Error Interrupt Enable bit
1 = USB error interrupt is enabled
0 = USB error interrupt is disabled

bit 0 **URSTIE:** USB Reset Interrupt Enable bit
1 = USB Reset interrupt is enabled
0 = USB Reset interrupt is disabled

27.5.3 USB ERROR INTERRUPT STATUS REGISTER (UEIR)

The USB Error Interrupt STATUS register (Register 27-9) contains the flag bits for each of the error sources within the USB peripheral. Each of these sources is controlled by a corresponding interrupt enable bit in the UEIE register. All of the USB error flags are ORed together to generate the USB Error Interrupt Flag (UERRIF) at the top level of the interrupt logic.

Each error bit is set as soon as the error condition is detected. Thus, the interrupt will typically not correspond with the end of a token being processed.

Once an interrupt bit has been set by the SIE, it must be cleared in software by writing a '0'.

Register 27-9: UEIR: USB ERROR INTERRUPT STATUS REGISTER (ACCESS F63h)

R/C-0	U-0	U-0	R/C-0	R/C-0	R/C-0	R/C-0	R/C-0
BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF
bit 7							bit 0

Legend:	C = Clearable bit		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **BTSEF:** Bit Stuff Error Flag bit
1 = A bit stuff error has been detected
0 = No bit stuff error has been detected

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **BTOEF:** Bus Turnaround Time-out Error Flag bit
1 = Bus turnaround time-out has occurred (more than 16 bit times of Idle from previous EOP elapsed)
0 = No bus turnaround time-out has occurred

bit 3 **DFN8EF:** Data Field Size Error Flag bit
1 = The data field was not an integral number of bytes
0 = The data field was an integral number of bytes

bit 2 **CRC16EF:** CRC16 Failure Flag bit
1 = The CRC16 failed
0 = The CRC16 passed

bit 1 **CRC5EF:** CRC5 Host Error Flag bit
1 = The token packet was rejected due to a CRC5 error
0 = The token packet was accepted

bit 0 **PIDEF:** PID Check Failure Flag bit
1 = PID check failed
0 = PID check passed

27.5.4 USB ERROR INTERRUPT ENABLE REGISTER (UEIE)

The USB Error Interrupt Enable register (Register 27-10) contains the enable bits for each of the USB error interrupt sources. Setting any of these bits will enable the respective error interrupt source in the UEIR register to propagate into the UERR bit at the top level of the interrupt logic.

As with the UIE register, the enable bits only affect the propagation of an interrupt condition to the microcontroller's interrupt logic. The flag bits are still set by their interrupt conditions, allowing them to be polled and serviced without actually generating an interrupt.

Register 27-10: UEIE: USB ERROR INTERRUPT ENABLE REGISTER (BANKED F37h)

R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **BTSEE:** Bit Stuff Error Interrupt Enable bit
1 = Bit stuff error interrupt is enabled
0 = Bit stuff error interrupt is disabled

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **BTOEE:** Bus Turnaround Time-out Error Interrupt Enable bit
1 = Bus turnaround time-out error interrupt is enabled
0 = Bus turnaround time-out error interrupt is disabled

bit 3 **DFN8EE:** Data Field Size Error Interrupt Enable bit
1 = Data field size error interrupt is enabled
0 = Data field size error interrupt is disabled

bit 2 **CRC16EE:** CRC16 Failure Interrupt Enable bit
1 = CRC16 failure interrupt is enabled
0 = CRC16 failure interrupt is disabled

bit 1 **CRC5EE:** CRC5 Host Error Interrupt Enable bit
1 = CRC5 host error interrupt is enabled
0 = CRC5 host error interrupt is disabled

bit 0 **PIDEE:** PID Check Failure Interrupt Enable bit
1 = PID check failure interrupt is enabled
0 = PID check failure interrupt is disabled

27.6 USB Power Modes

Many USB applications will likely have several different sets of power requirements and configuration. The most common power modes encountered are Bus Power Only, Self-Power Only and Dual Power with Self-Power Dominance. The most common cases are presented here. Also provided is a means of estimating the current consumption of the USB transceiver.

27.6.1 BUS POWER ONLY

In Bus Power Only mode, all power for the application is drawn from the USB (Figure 27-9). This is effectively the simplest power method for the device.

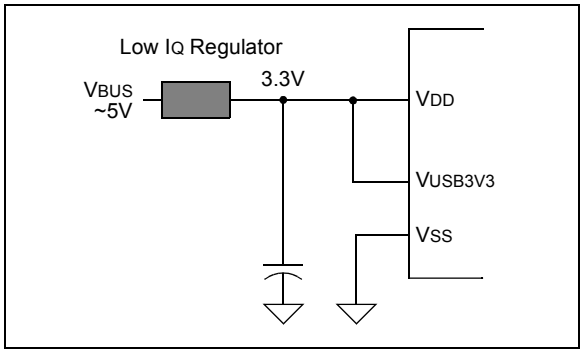
In order to meet the inrush current requirements of the "*USB 2.0 Specification*", the total effective capacitance appearing across VBUS and ground must be no more than 10 µF. If not, some kind of inrush timing is required. For more details, see **Section 7.2.4 of the "*USB 2.0 Specification*"**.

According to the "*USB 2.0 Specification*", all USB devices must also support a Low-Power Suspend mode. In the USB Suspend mode, devices must consume no more than 2.5 mA from the 5V VBUS line of the USB cable.

The host signals the USB device to enter the Suspend mode by stopping all USB traffic to that device for more than 3 ms. This condition will cause the IDLEIF bit in the UIR register to become set.

During the USB Suspend mode, the D+ or D- pull-up resistor must remain active, which will consume some of the allowed suspend current: 2.5 mA budget.

FIGURE 27-9: BUS POWER ONLY



27.6.2 SELF-POWER ONLY

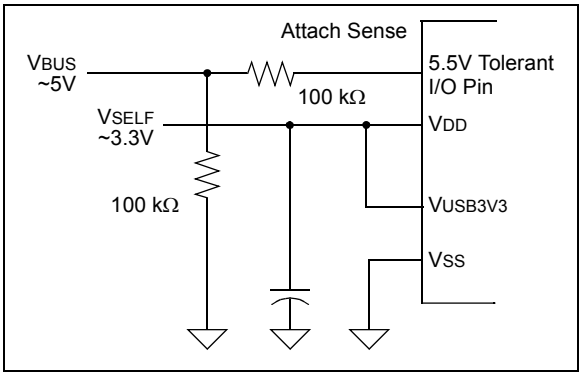
In Self-Power Only mode, the USB application provides its own power, with very little power being pulled from the USB. See Figure 27-10 for an example.

Note that an attach indication is added to indicate when the USB has been connected and the host is actively powering VBUS.

In order to meet compliance specifications, the USB module (and the D+ or D- pull-up resistor) should not be enabled until the host actively drives VBUS high. One of the 5.5V tolerant I/O pins may be used for this purpose.

The application should never source any current onto the 5V VBUS pin of the USB cable.

FIGURE 27-10: SELF-POWER ONLY

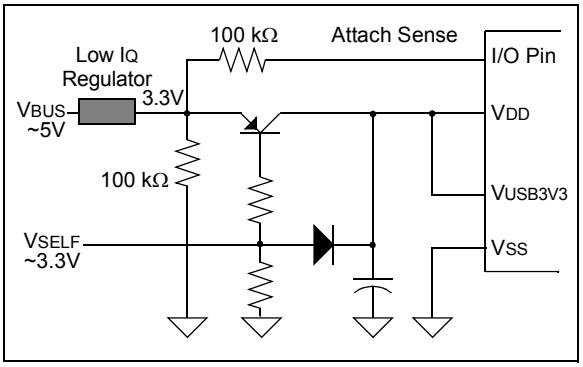


27.6.3 DUAL POWER WITH SELF-POWER DOMINANCE

Some applications may require a dual power option. This allows the application to use internal power primarily, but switch to power from the USB when no internal power is available. See Figure 27-11 for a simple Dual Power with Self-Power Dominance mode example, which automatically switches between Self-Power Only and USB Bus Power Only modes.

Dual power devices must also meet all of the special requirements for inrush current and Suspend mode current, and must not enable the USB module until VBUS is driven high. See **Section 27.6.1 "Bus Power Only"** and **Section 27.6.2 "Self-Power Only"** for descriptions of those requirements. Additionally, dual power devices must never source current onto the 5V VBUS pin of the USB cable.

FIGURE 27-11: DUAL POWER WITH SELF-POWER DOMINANCE



Note:	Users should keep in mind the limits for devices drawing power from the USB. According to USB Specification 2.0, this cannot exceed 100 mA per low-power device or 500 mA per high-power device.

27.6.4 USB TRANSCEIVER CURRENT CONSUMPTION

The USB transceiver consumes a variable amount of current depending on the characteristic impedance of the USB cable, the length of the cable, the V_{USB3V3} supply voltage and the actual data patterns moving across the USB cable. Longer cables have larger capacitances and consume more total energy when switching output states.

Data patterns that consist of “IN” traffic consume far more current than “OUT” traffic. IN traffic requires the PIC® MCU to drive the USB cable, whereas OUT traffic requires that the host drive the USB cable.

The data that is sent across the USB cable is NRZI encoded. In the NRZI encoding scheme, ‘0’ bits cause a toggling of the output state of the transceiver (either from a “J” state to a “K” state or vise versa). With the exception of the effects of bit stuffing, NRZI encoded ‘1’ bits do not cause the output state of the transceiver to change. Therefore, IN traffic consisting of data bits of value, ‘0’, cause the most current consumption, as the transceiver must charge/discharge the USB cable in order to change states.

More details about NRZI encoding and bit stuffing can be found in the “*USB 2.0 Specification*”, Section 7.1, although knowledge of such details is not required to make USB applications using the PIC18FXXJ94 of microcontrollers. Among other things, the SIE handles bit stuffing/unstuffing, NRZI encoding/decoding and CRC generation/checking in hardware.

The total transceiver current consumption will be application-specific. However, to help estimate how much current actually may be required in full-speed applications, Equation 27-1 can be used.

See Equation 27-2 to know how this equation can be used for a theoretical application.

EQUATION 27-1: ESTIMATING USB TRANSCEIVER CURRENT CONSUMPTION

$$I_{XCVR} = \frac{(40\ mA \bullet V_{USB3V3} \bullet P_{ZERO} \bullet P_{IN} \bullet L_{CABLE})}{(3.3V \bullet 5m)} + I_{PULLUP}$$

Legend: V_{USB3V3} – Voltage applied to the V_{USB3V3} pin in volts (should be 3.0V to 3.6V).

P_{ZERO} – Percentage (in decimal) of the IN traffic bits sent by the PIC® MCU that are a value of ‘0’.

P_{IN} – Percentage (in decimal) of total bus bandwidth that is used for IN traffic.

L_{CABLE} – Length (in meters) of the USB cable. The “*USB 2.0 Specification*” requires that full-speed applications use cables no longer than 5m.

I_{PULLUP} – Current which the nominal, 1.5 kΩ pull-up resistor (when enabled) must supply to the USB cable. On the host or hub end of the USB cable, 15 kΩ nominal resistors (14.25 kΩ to 24.8 kΩ) are present which pull both the D+ and D- lines to ground. During bus Idle conditions (such as between packets or during USB Suspend mode), this results in up to 218 μA of quiescent current drawn at 3.3V.

I_{PULLUP} is also dependant on bus traffic conditions and can be as high as 2.2 mA when the USB bandwidth is fully utilized (either IN or OUT traffic) for data that drives the lines to the “K” state most of the time.

EQUATION 27-2: CALCULATING USB TRANSCEIVER CURRENT†

For this example, the following assumptions are made about the application:

- 3.3V will be applied to V_{USB3V3} and V_{DD}, with the core voltage regulator enabled.
- This is a full-speed application that uses one interrupt IN endpoint that can send one packet of 64 bytes every 1 ms, with no restrictions on the values of the bytes being sent. The application may or may not have additional traffic on OUT endpoints.
- A regular USB “B” or “mini-B” connector will be used on the application circuit board.

In this case, P_{ZERO} = 100% = 1, because there should be no restriction on the value of the data moving through the IN endpoint. All 64 kbps of data could potentially be bytes of value, 00h. Since ‘0’ bits cause toggling of the output state of the transceiver, they cause the USB transceiver to consume extra current charging/discharging the cable. In this case, 100% of the data bits sent can be of value ‘0’. This should be considered the “max” value, as normal data will consist of a fair mix of ones and zeros.

This application uses 64 kbps for IN traffic out of the total bus bandwidth of 1.5 Mbps (12 Mbps), therefore:

$$Pin = \frac{64\ kbps}{1.5\ Mbps} = 4.3\% = 0.043$$

Since a regular “B” or “mini-B” connector is used in this application, the end user may plug in any type of cable up to the maximum allowed 5m length. Therefore, we use the worst-case length:

L_{CABLE} = 5 meters

Assume I_{PULLUP} = 2.2 mA. The actual value of I_{PULLUP} will likely be closer to 218 μA, but allowance for the worst-case. USB bandwidth is shared between all the devices which are plugged into the root port (via hubs). If the application is plugged into a USB 1.1 hub that has other devices plugged into it, your device may see host to device traffic on the bus, even if it is not addressed to your device. Since any traffic, regardless of source, can increase the I_{PULLUP} current above the base 218 μA, it is safest to allow for the worst-case of 2.2 mA.

Therefore:

$$I_{XCVR} = \frac{(40\ mA \bullet 3.3V \bullet 1 \bullet 0.043 \bullet 5m)}{(3.3V \bullet 5m)} + 2.2\ mA = 3.9\ mA$$

† The calculated value should be considered an approximation and additional guardband or application-specific product testing is recommended. The transceiver current is “in addition to” the rest of the current consumed by the PIC18FXXJ94 device that is needed to run the core, drive the other I/O lines, power the various modules, etc.

27.7 Oscillator

The USB module has specific clock requirements. For full-speed operation, the clock source must be 48 MHz. Even so, the microcontroller core and other peripherals are not required to run at that clock speed.

27.8 USB Firmware and Drivers

Microchip provides a number of application-specific resources, such as USB firmware and driver support. Refer to www.microchip.com for the latest firmware and driver support.

TABLE 27-4: REGISTERS ASSOCIATED WITH USB MODULE OPERATION(1)

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
INTCON	GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	IOCIE	TMR0IF	INT0IF	IOCIF
IPR2	OSCFIP	SSP2IP	BCL2IP	USBIP	BCL1IP	HLVDIP	TMR3IP	TMR3GIP
PIR2	OSCFIF	SSP2IF	BCL2IF	USBIF	BCL1IF	HLVDIF	TMR3IF	TMR3GIF
PIE2	OSCFIE	SSP2IE	BCL2IE	USBIE	BCL1IE	HLVDIE	TMR3IE	TMR3GIE
UCON	—	PPBRST	SE0	PKTDIS	USBEN	RESUME	SUSPND	—
UCFG	UTEYE	UOEMON	—	UPUEN	UTRDIS	FSEN	PPB1	PPB0
USTAT	—	ENDP3	ENDP2	ENDP1	ENDP0	DIR	PPBI	—
UADDR	—	ADDR6	ADDR5	ADDR4	ADDR3	ADDR2	ADDR1	ADDR0
UFRML	FRM7	FRM6	FRM5	FRM4	FRM3	FRM2	FRM1	FRM0
UFRMH	—	—	—	—	—	FRM10	FRM9	FRM8
UIR	—	SOFIF	STALLIF	IDLEIF	TRNIF	ACTVIF	UERRIF	URSTIF
UIE	—	SOFIE	STALLIE	IDLEIE	TRNIE	ACTVIE	UERRIE	URSTIE
UEIR	BTSEF	—	—	BTOEF	DFN8EF	CRC16EF	CRC5EF	PIDEF
UEIE	BTSEE	—	—	BTOEE	DFN8EE	CRC16EE	CRC5EE	PIDEE
UEP0	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP1	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP2	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP3	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP4	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP5	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP6	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP7	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP8	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP9	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP10	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP11	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP12	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP13	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP14	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL
UEP15	—	—	—	EPHSHK	EPCONDIS	EPOUTEN	EPINEN	EPSTALL

Legend: — = unimplemented, read as '0'. Shaded cells are not used by the USB module.

Note 1: This table includes only those hardware mapped SFRs located in Bank 15 of the data memory space. The Buffer Descriptor registers, which are mapped into Bank 4 and are not true SFRs, are listed separately in Table 27-3.

27.9 Overview of USB

This section presents some of the basic USB concepts and useful information necessary to design a USB device. Although much information is provided in this section, there is a plethora of information provided within the USB specifications and class specifications. Thus, the reader is encouraged to refer to the USB specifications for more information (www.usb.org). If you are very familiar with the details of USB, then this section serves as a basic, high-level refresher of USB.

27.9.1 LAYERED FRAMEWORK

USB device functionality is structured into a layered framework, graphically illustrated in Figure 27-12. Each level is associated with a functional level within the device. The highest layer, other than the device, is the configuration. A device may have multiple configurations. For example, a particular device may have multiple power requirements based on Self-Power Only or Bus Power Only modes.

For each configuration, there may be multiple interfaces. Each interface could support a particular mode of that configuration.

Below the interface is the endpoint(s). Data is directly moved at this level. There can be as many as 16 bidirectional endpoints. Endpoint 0 is always a control endpoint, and by default, when the device is on the bus, Endpoint 0 must be available to configure the device.

27.9.2 FRAMES

Information communicated on the bus is grouped into 1 ms time slots, referred to as frames. Each frame can contain many transactions to various devices and endpoints. See Figure 27-8 for an example of a transaction within a frame.

27.9.3 TRANSFERS

There are four transfer types defined in the USB specification.

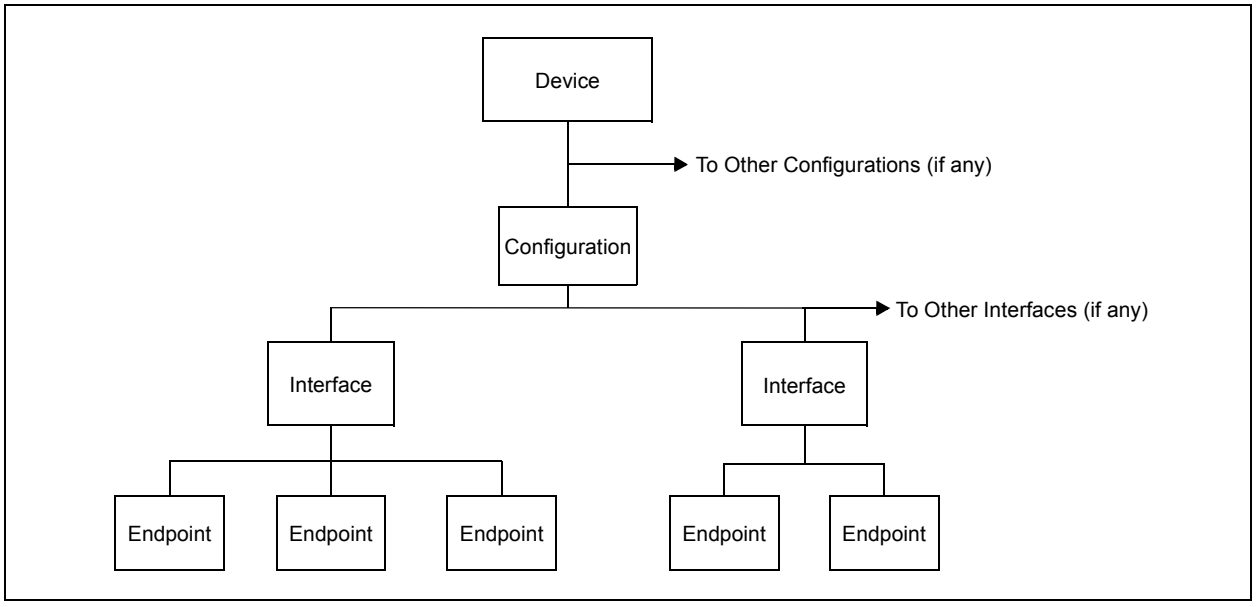
- **Isochronous:** This type provides a transfer method for large amounts of data (up to 1023 bytes) with timely delivery ensured; however, the data integrity is not ensured. This is good for streaming applications where small data loss is not critical, such as audio.
- **Bulk:** This type of transfer method allows for large amounts of data to be transferred with ensured data integrity; however, the delivery timeliness is not ensured.
- **Interrupt:** This type of transfer provides for ensured timely delivery for small blocks of data, plus data integrity is ensured.
- **Control:** This type provides for device setup control.

While full-speed devices support all transfer types, low-speed devices are limited to interrupt and control transfers only.

27.9.4 POWER

Power is available from the USB. The USB specification defines the bus power requirements. Devices may either be self-powered or bus-powered. Self-powered devices draw power from an external source, while bus-powered devices use power supplied from the bus.

FIGURE 27-12: USB LAYERS



The USB Specification limits the power taken from the bus. Each device is ensured 100 mA at approximately 5V (one unit load). Additional power may be requested, up to a maximum of 500 mA.

Note that power above one unit load is a request and the host or hub is not obligated to provide the extra current. Thus, a device capable of consuming more than one unit load must be able to maintain a low-power configuration of a one unit load or less, if necessary.

The USB Specification also defines a Suspend mode. In this situation, current must be limited to 500 μA, averaged over one second. A device must enter a suspend state after 3 ms of inactivity (i.e., no SOF tokens for 3 ms). A device entering Suspend mode must drop current consumption within 10 ms after suspend. Likewise, when signaling a wake-up, the device must signal a wake-up within 10 ms of drawing current above the suspend limit.

27.9.5 ENUMERATION

When the device is initially attached to the bus, the host enters an enumeration process in an attempt to identify the device. Essentially, the host interrogates the device, gathering information, such as power consumption, data rates and sizes, protocol and other descriptive information; descriptors contain this information. A typical enumeration process would be as follows:

1. USB Reset – Reset the device. Thus, the device is not configured and does not have an address (Address 0).
2. Get Device Descriptor – The host requests a small portion of the device descriptor.
3. USB Reset – Reset the device again.
4. Set Address – The host assigns an address to the device.
5. Get Device Descriptor – The host retrieves the device descriptor, gathering info, such as manufacturer, type of device, maximum control packet size.
6. Get configuration descriptors.
7. Get any other descriptors.
8. Set a configuration.

The exact enumeration process depends on the host.

27.9.6 DESCRIPTORS

There are eight different standard descriptor types, of which, five are most important for this device.

27.9.6.1 Device Descriptor

The device descriptor provides general information, such as manufacturer, product number, serial number, the class of the device and the number of configurations. There is only one device descriptor.

27.9.6.2 Configuration Descriptor

The configuration descriptor provides information on the power requirements of the device and how many different interfaces are supported when in this configuration. There may be more than one configuration for a device (i.e., low-power and high-power configurations).

27.9.6.3 Interface Descriptor

The interface descriptor details the number of endpoints used in this interface, as well as the class of the interface. There may be more than one interface for a configuration.

27.9.6.4 Endpoint Descriptor

The endpoint descriptor identifies the transfer type (**Section 27.9.3 "Transfers"**) and direction, and some other specifics for the endpoint. There may be many endpoints in a device and endpoints may be shared in different configurations.

27.9.6.5 String Descriptor

Many of the previous descriptors reference one or more string descriptors. String descriptors provide human readable information about the layer (**Section 27.9.1 "Layered Framework"**) they describe. Often these strings show up in the host to help the user identify the device. String descriptors are generally optional to save memory and are encoded in a unicode format.

27.9.7 BUS SPEED

Each USB device must indicate its bus presence and speed to the host. This is accomplished through a 1.5 kΩ resistor, which is connected to the bus at the time of the attachment event.

Depending on the speed of the device, the resistor either pulls up the D+ or D- line to 3.3V. For a low-speed device, the pull-up resistor is connected to the D- line. For a full-speed device, the pull-up resistor is connected to the D+ line.

27.9.8 CLASS SPECIFICATIONS AND DRIVERS

USB specifications include class specifications, which operating system vendors optionally support. Examples of classes include: Audio, Mass Storage, Communications and Human Interface (HID). In most cases, a driver is required at the host side to 'talk' to the USB device. In custom applications, a driver may need to be developed. Fortunately, drivers are available for most common host systems for the most common classes of devices. Thus, these drivers can be reused.

28.0 SPECIAL FEATURES OF THE CPU

The PIC18FXXJ94 of devices includes several features intended to maximize reliability and minimize cost through elimination of external components. These include:

- Oscillator Selection
- Resets:
  - Power-on Reset (POR)
  - Power-up Timer (PWRT)
  - Oscillator Start-up Timer (OST)
  - Brown-out Reset (BOR)
- Interrupts
- Watchdog Timer (WDT) and On-chip Regulator
- Fail-Safe Clock Monitor
- Two-Speed Start-up
- Code Protection
- ID Locations
- In-Circuit Serial Programming™

The oscillator can be configured for the application depending on frequency, power, accuracy and cost. All of the options are discussed in detail in **Section 3.0 "Oscillator Configurations"**.

A complete discussion of device Resets and interrupts is available in previous sections of this data sheet.

In addition to their Power-up and Oscillator Start-up Timers provided for Resets, the PIC18FXXJ94 of devices has a Watchdog Timer, which is either permanently enabled via the Configuration bits or software controlled (if configured as disabled).

The inclusion of an internal RC Oscillator (LF-INTOSC) also provides the additional benefits of a Fail-Safe Clock Monitor (FSCM) and Two-Speed Start-up. FSCM provides for background monitoring of the peripheral clock and automatic switchover in the event of its failure. Two-Speed Start-up enables code to be executed almost immediately on start-up, while the primary clock source completes its start-up delays.

All of these features are enabled and configured by setting the appropriate Configuration register bits.

28.1 Configuration Bits

Devices of the PIC18FXXJ94 do not use persistent memory registers to store configuration information. The Configuration registers, CONFIG1L through CONFIG8H, are implemented as volatile memory. Immediately after power-up, or after a device Reset, the microcontroller hardware automatically loads the CONFIG1L through CONFIG8H registers with configuration data stored in nonvolatile Flash program memory. The last eight words of Flash program memory, known as the Flash Configuration Words (FCW), are used to store the configuration data.

Table 28-2 provides the Flash program memory, which will be loaded into the corresponding Configuration register.

When creating applications for these devices, users should always specifically allocate the location of the FCW for configuration data. This is to make certain that program code is not stored in this address when the code is compiled.

The four Most Significant bits (MSb) of the FCW, corresponding to CONFIG1H, CONFIG2H, CONFIG3H, CONFIG4H, CONFIG5H, CONFIG6H, CONFIG7H and CONFIG8H, should always be programmed to '1111'.

This makes these FCWs appear to be NOP instructions in the remote event that their locations are ever executed by accident.

The four MSbs of the CONFIG1H, CONFIG2H, CONFIG3H, CONFIG4H CONFIG5H, CONFIG6H, CONFIG7H and CONFIG8H, registers are not implemented, so writing '1's to their corresponding FCW has no effect on device operation.

To prevent inadvertent configuration changes during code execution, the Configuration registers, CONFIG1L through CONFIG8H, are loaded only once per power-up or Reset cycle. User's firmware can still change the configuration by using self-reprogramming to modify the contents of the FCW.

Modifying the FCW will not change the active contents being used in the CONFIG1L through CONFIG8H registers until after the device is reset.

TABLE 28-1: MAPPING OF THE FLASH CONFIGURATION WORDS TO THE CONFIGURATION REGISTERS

Configuration Register (Volatile)	Configuration Register Address	Flash Configuration Byte Address
CONFIG1L	300000h	XXXF0h
CONFIG1H	300001h	XXXF1h
CONFIG2L	300002h	XXXF2h
CONFIG2H	300003h	XXXF3h
CONFIG3L	300004h	XXXF4h
CONFIG3H	300005h	XXXF5h
CONFIG4L	300006h	XXXF6h
CONFIG4H	300007h	XXXF7h
CONFIG5L	300008h	XXXF8h
CONFIG5H	300009h	XXXF9h
CONFIG6L	30000Ah	XXXFAh
CONFIG6H	30000Bh	XXXFBh
CONFIG7L	30000Ch	XXXFCh
CONFIG7H	30000Dh	XXXFDh
CONFIG8L	30000Eh	XXXFEh
CONFIG8H	30000Fh	XXXFFh

TABLE 28-2: CONFIGURATION BITS AND DEVICE IDs

File Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default/ Unprogrammed Value
300000h	CONFIG1L	$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	—	111- ----
300001h	CONFIG1H	—(2)	—(2)	—(2)	—(2)	—(1)	CP0	BORV	BOREN	---- -111
300002h	CONFIG2L	IESO	—	CLKOEN	—	SOSCSEL	FOSC2	FOSC1	FOSC0	1-1- 1111
300003h	CONFIG2H	—(2)	—(2)	—(2)	—(2)	PLLDIV3	PLLDIV2	PLLDIV1	PLLDIV0	---- 1111
300004h	CONFIG3L	—	—	FSCM1	FSCM0	—	—	POSCMD1	POSCMD0	--11 --11
300005h	CONFIG3H	—(2)	—(2)	—(2)	—(2)	—	—	—	—	1111 ----
300006h	CONFIG4L	WPFP7	WPFP6	WPFP5	WPFP4	WPFP3	WPFP2	WPFP1	WPFP0	1111 1111
300007h	CONFIG4H	—(2)	—(2)	—(2)	—(2)	—	WPCFG	WPEND	WPDIS	---- -111
300008h	CONFIG5L	WAIT	BW	ABW1	ABW0	EASHFT	—	CINASEL	T5GSEL	1111 1-11
300009h	CONFIG5H	—(2)	—(2)	—(2)	—(2)	MSSPMSK1	MSSPMSK2	LS48MHZ	IOL1WAY	1111 1111
30000Ah	CONFIG6L	WDPS3	WDPS2	WDPS1	WDPS0	WDTCLK1	WDTCLK0	WDTWIN1	WDTWIN0	1111 1111
30000Bh	CONFIG6H	—(2)	—(2)	—(2)	—(2)	WPSA	WINDIS	WDTEN1	WDTEN0	1111 1111
30000Ch	CONFIG7L	—	—	—	DSBITEN	DSBOREN	VBTBOR	—	$\overline{\text{RETEN}}$	---1 11-1
30000Dh	CONFIG7H	—(2)	—(2)	—(2)	—(2)	—	—	—	—	1111 ----
30000Eh	CONFIG8L	DSWDTPS4	DSWDTPS3	DSWDTPS2	DSWDTPS1	DSWDTPS0	—	—	—	1111 1---
30000Fh	CONFIG8H	—(2)	—(2)	—(2)	—(2)	—	—	DSWDTOSC	DSWDTEN	1111 --11
3FFFFEh	DEVID1	DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0	See Register 28-16
3FFFFFh	DEVID2	DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3	See Register 28-15

Legend: x = unknown, u = unchanged, - = unimplemented, q = value depends on condition. Shaded cells are unimplemented, read as '0'.

Note 1: This bit should always be maintained as '0'.

2: The value of these bits in program memory should always be programmed to '1'. This ensures that the location is executed as a NOP if it is accidentally executed.

REGISTER 28-1: CONFIG1L: CONFIGURATION REGISTER 1 LOW (BYTE ADDRESS 300000h)

R/WO-1	R/WO-1	R/WO-1	U-1	U-1	U-1	U-1	U-1
$\overline{\text{DEBUG}}$	XINST	STVREN	—	—	—	—	—
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **$\overline{\text{DEBUG}}$:** Background Debugger Enable bit
1 = Background debugger is disabled, and RB6 and RB7 are configured as general purpose I/O pins
0 = Background debugger is enabled, and RB6 and RB7 are dedicated to In-Circuit Debug

bit 6 **XINST:** Extended Instruction Set Enable bit
1 = Instruction set extension and Indexed Addressing mode are enabled
0 = Instruction set extension and Indexed Addressing mode are disabled (Legacy mode)

bit 5 **STVREN:** Stack Overflow Reset Enable bit
1 = Reset on stack overflow/underflow is enabled
0 = Reset on stack overflow/underflow is disabled

bit 4-0 **Unimplemented:** Read as '1'

REGISTER 28-2: CONFIG1H: CONFIGURATION REGISTER 1 HIGH (BYTE ADDRESS 300001h)

U-1	U-1	U-1	U-1	U-0	R/WO-1	R/WO-1	R/WO-1
—	—	—	—	—	CP0	BORV	BOREN
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Read as '–'

bit 3 **Unimplemented:** Maintain as '0'

bit 2 **CP0:** Code Protection bit 0
1 = Program memory is not code-protected
0 = Program memory is code-protected (and write-protected in test modes)

bit 1 **BORV:** BOR Trip Point Select bit
1 = BOR trip point is 1.8V
0 = BOR trip point is 2.0V

bit 0 **BOREN:** Brown-out Reset Enable bit
1 = Brown-out Reset is disabled
0 = Brown-out Reset is enabled outside of Deep Sleep (BORV is always disabled in Deep Sleep)

REGISTER 28-3: CONFIG2L: CONFIGURATION REGISTER 2 LOW (BYTE ADDRESS 300002h)(1,2,3,4)

R/WO-1	U-1	R/WO-0	U-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
IESO	—	CLKOEN	—	SOSCSEL	FOSC2	FOSC1	FOSC0
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **IESO:** Internal External Switch Over bit
1 = Internal/External Switchover mode is enabled (Two-Speed Start-up is enabled)
0 = Internal/External Switchover mode is disabled (Two-Speed Start-up is disabled)

bit 6 **Unimplemented:** Read as '1'

bit 5 **CLKOEN:** CLKO Enable Configuration bit
1 = CLKO output signal is active on the OSC2 pin; Primary Oscillator must be disabled or configured for the External Clock mode (EC) for the CLKO to be active (POSCMD<1:0> = 11 or 00)
0 = CLKO output disabled

bit 4 **Unimplemented:** Read as '0'

bit 3 **SOSCSEL:** SOSC Selection Configuration bit
1 = Low-power SOSC circuit is selected (typical IDD of 1 µA)
0 = Digital (SCLKI) mode

bit 2-0 **FOSC<2:0>:** Oscillator Selection bits
000 =Fast RC Oscillator (FRC)
001 =Fast RC Oscillator with divide-by-N with PLL module (FRCDIV+PLL)
010 =Primary Oscillator (MS, HS, EC)
011 =Primary Oscillator with PLL module (MS+PLL, HS+PLL, EC+PLL)
100 =Secondary Oscillator (SOSC)
101 =Low-Power RC Oscillator (LPRC)
110 =Fast RC Oscillator (FRC) divided by 16 (500 kHz)
111 =Fast RC Oscillator with divide-by-N (FRCDIV)

Note 1: The CONFIG2L bits can only be programmed indirectly by programming the Flash Configuration Word.
2: The CONFIG2L is reset to '1' only on VDD Reset; it is reloaded with the programmed value at any device Reset.
3: Although CONFIG2L is reset to '1' only on VDD Reset, these values are not used until after the actual configuration values are read out and stored in the register bits. Therefore, for these bits, the Reset value has no effect on the operation of the system.
4: Unlike other Configuration registers, the CLKOEN holding register is reset to a '0' on any VDD Reset. This prevents the CLKO pin from driving until the actual configuration values are read out and stored in the register.

REGISTER 28-4: CONFIG2H: CONFIGURATION REGISTER 2 HIGH (BYTE ADDRESS 300003h)(1,2)

U-1	U-1	U-1	U-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
—	—	—	—	PLLDIV3(3)	PLLDIV2(3)	PLLDIV1(3)	PLLDIV0(3)
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Program the corresponding Flash Configuration bit to '1'

bit 3-0 **PLLDIV<3:0>:** Frequency Multiplier Select bits(3)
Divider must be selected so as to not exceed 64 MHz output.
1111 = No PLL used; PLLEN bit is not available to user
1110 = 8x PLL is selected
1101 = 6x PLL is selected
1100 = 4x PLL is selected
1011 = Reserved; do not use
1010 = Reserved; do not use
1001 = Reserved; do not use
1000 = Reserved; do not use
0111 = 96 MHz PLL is selected; oscillator divided by 12 (48 MHz input)
0110 = 96 MHz PLL is selected; oscillator divided by 8 (32 MHz input)
0101 = 96 MHz PLL is selected; oscillator divided by 6 (24 MHz input)
0100 = 96 MHz PLL is selected; oscillator divided by 5 (20 MHz input)
0011 = 96 MHz PLL is selected; oscillator divided by 4 (16 MHz input)
0010 = 96 MHz PLL is selected; oscillator divided by 3 (12 MHz input)
0001 = 96 MHz PLL is selected; oscillator divided by 2 (8 MHz input)
0000 = 96 MHz PLL is selected; no divide – oscillator is used directly (4 MHz input)

Note 1: The CONFIG2H bits can only be programmed indirectly by programming the Flash Configuration Word.

2: The CONFIG2H is reset to '1' only on VDD Reset; it is reloaded with the programmed value at any device Reset.

3: If USB functionality is used, then this field must be set to '0xxx' (i.e., 96 MHz PLL is selected).

REGISTER 28-5: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)(1,2)

U-1	U-1	R/WO-1	R/WO-1	U-1	U-0	R/WO-1	R/WO-1
—	—	FSCM1	FSCM0	—	—	POSCMD1	POSCMD0
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-6 **Unimplemented:** Read as '1'

bit 5-4 **FSCM<1:0>:** Clock Switching and Monitor Selection Configuration bits
1x =Clock switching is disabled, Fail-Safe Clock Monitor is disabled
01 =Clock switching is enabled, Fail-Safe Clock Monitor is disabled
00 =Clock switching is enabled, Fail-Safe Clock Monitor is enabled

bit 3-2 **Unimplemented:** Read as '0'

bit 1-0 **POSCMD<1:0>:** Primary Oscillator Configuration bits
11 =Primary Oscillator is disabled
10 =HS Oscillator mode is selected (10 MHz-40 MHz)
01 =MS Oscillator mode is selected (3.5 MHz-10 MHz)
00 =External Clock mode is selected

Note 1: The CONFIG3L bits can only be programmed indirectly by programming the Flash Configuration Word.
2: The CONFIG3L is reset to '1' only on VDD Reset; it is reloaded with the programmed value at any device Reset.

REGISTER 28-6: CONFIG4L: CONFIGURATION REGISTER 4 LOW (BYTE ADDRESS 300006h)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
WPFP7	WPFP6	WPFP5	WPFP4	WPFP3	WPFP2	WPFP1	WPFP0
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **WPFP<7:0>:** Write-Protect Program Flash Pages bits (valid when WPDIS = 0)
When WPEND = 0:
Write/erase protect Flash memory pages, starting at Page 0 and ending with Page WPFP<7:0>.
When WPEND = 1:
Write/erase protect Flash memory pages, starting at Page WPFP<7:0> and ending with the last page in user Flash.

REGISTER 28-7: CONFIG4H: CONFIGURATION REGISTER 4 HIGH (BYTE ADDRESS 300007h)

U-1	U-1	U-1	U-1	U-1	R/WO-1	R/WO-1	R/WO-1
—	—	—	—	—	WPCFG	WPEND	WPDIS
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-3 **Unimplemented:** Program the corresponding Flash Configuration bit to '1'

bit 2 **WPCFG:** Write/Erase Protect Last Page in User Flash bit
1 = Write/erase protection of last page is disabled, regardless of the WPFP<7:0> setting
0 = Write/erase protection of last page is enabled, regardless of the WPFP<7:0> setting

bit 1 **WPEND:** Write Protection End Page bit
This bit is valid when WPDIS = 0.
When WPEND = 0:
Write/erase protect Flash Memory pages, starting at Page 0 and ending with Page WPFP<7:0>.
When WPEND = 1:
Write/erase protect Flash memory pages, starting at Page WPFP<7:0> and ending with the last page in user Flash.

bit 0 **WPDIS:** Write-Protect Disable bit
1 = WPFP<7:0>, WPEND and WPCFG bits are ignored
0 = WPFP<7:0>, WPEND and WPCFG bits are enabled; write-protect is active

REGISTER 28-8: CONFIG5L: CONFIGURATION REGISTER 5 LOW (BYTE ADDRESS 300008h)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-1	R/WO-1	R/WO-1
WAIT(1)	BW	ABW1	ABW0	EASHFT	—	CINASEL	T5GSEL
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **WAIT:** External Bus Wait Enable bit(1)
- 1 = Wait selections from WAIT<1:0> (MEMCON<5:4>) are unavailable and the device will not wait
- 0 = Wait is programmed by WAIT<1:0> (MEMCON<5:4>)

bit 6 **BW:** Data Bus Width Select bit
- 1 = 16-Bit External Bus mode
- 0 = 8-Bit External Bus mode

bit 5-4 **ABW<1:0>:** External Memory Bus Configuration bits
- 00 =Extended Microcontroller Mode – 20-Bit Address mode
- 01 =Extended Microcontroller Mode – 16-Bit Address mode
- 10 =Extended Microcontroller Mode – 12-Bit Address mode
- 11 =Microcontroller Mode – External bus is disabled

bit 3 **EASHFT:** External Address Bus Shift Enable bit
- 1 = Address shifting is enabled – External address bus is shifted to start at 000000h
- 0 = Address shifting is disabled – External address bus reflects the PC value

bit 2 **Unimplemented:** Read as '0'

bit 1 **CINASEL:** CxINA Gate Select bit
- 1 = C1INA and C3INA are on their default pin locations
- 0 = C1INA and C3INA are all remapped to pin, RA5

bit 0 **T5GSEL:** TMR5 Gate Select bit
- 1 = TMR5 gate is driven by the T5G input
- 0 = TMR5 gate is driven by the T3G input

Note 1: This bit was previously referred to as 'WAIT', but a set condition actually indicates the case where the EMB does not wait and the name was therefore changed to reflect this. No change in functionality or polarity occurred, only a change in the name of the register bit.

REGISTER 28-9: CONFIG5H: CONFIGURATION REGISTER 5 HIGH (BYTE ADDRESS 300009h)

U-1	U-1	U-1	U-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
—	—	—	—	MSSPMSK1	MSSPMSK2	LS48MHZ	IOL1WAY
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Program the corresponding Flash Configuration bit to '1'

bit 3 **MSSPMSK1:** MSSP1 7-Bit Address Masking Mode Enable bit
1 = 7-Bit Address Masking mode enable
0 = 5-Bit Address Masking mode enable

bit 2 **MSSPMSK2:** MSSP2 7-Bit Address Masking Mode Enable bit
1 = 7-Bit Address Masking mode enable
0 = 5-Bit Address Masking mode enable

bit 1 **LS48MHZ:** Low-Speed USB Clock Selection bit
1 = 48 MHz system clock is expected; divide-by-8 generates low-speed USB clock
0 = 24 MHz system clock is expected; divide-by-4 generates low-speed USB clock

bit 0 **IOL1WAY:** IOLOCK Bit One-Way Set Enable bit
1 = The IOLOCK bit can only be set once (provided an unlocking sequence is executed); this prevents any possible future RP register changes
0 = The IOLOCK bit can be set and cleared as needed (provided an unlocking sequence is executed)

REGISTER 28-10: CONFIG6L: CONFIGURATION REGISTER 6 LOW (BYTE ADDRESS 30000Ah)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
WDPS3	WDPS2	WDPS1	WDPS0	WDTCLK1	WDTCLK0	WDTWIN1	WDTWIN0
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-4 **WDPS<3:0>:** Watchdog Timer Postscale Select bits

1111 = 1:32,768
1110 = 1:16,384
1101 = 1:8,192
1100 = 1:4,096
1011 = 1:2,048
1010 = 1:1,024
1001 = 1:512
1000 = 1:256
0111 = 1:128
0110 = 1:64
0101 = 1:32
0100 = 1:16
0011 = 1:8
0010 = 1:4
0001 = 1:2
0000 = 1:1

bit 3-2 **WDTCLK<1:0>:** Watchdog Timer Clock Source bits

00 =Use the peripheral clock when the system clock is not INTOSC/LPRC and device is not in Sleep; otherwise, use INTOSC/LPRC
01 =Always use SOSC
10 =Always use INTOSC/LPRC
11 =Use FRC when WINDIS = 0, system clock is not INTOSC/LPRC and device is not in Sleep; otherwise, use INTOSC/LPRC

bit 1-0 **WDTWIN<1:0>:** Watchdog Timer Window Width bits

11 = 25%
10 = 37.5%
01 = 50%
00 = 75%

REGISTER 28-11: CONFIG6H: CONFIGURATION REGISTER 6 HIGH (BYTE ADDRESS 30000Bh)

U-1	U-1	U-1	U-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1
—	—	—	—	WPSA	WINDIS	WDTEN1	WDTEN0
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Program the corresponding Flash Configuration bit to ‘1’

bit 3 **WPSA:** WDT Prescaler bit
1 = WDT prescaler ratio of 1:128
0 = WDT prescaler ratio of 1:32

bit 2 **WINDIS:** Windowed Watchdog Timer Disable bit
1 = Standard WDT is selected; windowed WDT is disabled
0 = Windowed WDT is enabled when executing a CLRWDT instruction while the WDT is disabled in hardware

bit 1-0 **WDTEN<1:0>:** Watchdog Timer Enable bits
11 = WDT is enabled in hardware
10 = WDT is controlled with the SWDTEN bit setting
01 = WDT is enabled only while device is active and disabled in Sleep; SWDTEN bit is disabled
00 = WDT is disabled in hardware; SWDTEN bit is disabled

REGISTER 28-12: CONFIG7L: CONFIGURATION REGISTER 7 LOW (BYTE ADDRESS 30000Ch)

U-1	U-1	U-1	R/WO-1	R/WO-1	R/WO-1	U-1	R/WO-1
—	—	—	DSBITEN	DSBOREN	VBTBOR	—	$\overline{\text{RETEN}}$
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-5 **Unimplemented:** Read as ‘1’

bit 4 **DSBITEN:** DSEN Bit Enable bit
1 = Deep Sleep is controlled by the register bit, DSEN
0 = Deep Sleep operation is always disabled

bit 3 **DSBOREN:** Deep Sleep BOR Enable bit
1 = DSBOR is enabled in Deep Sleep
0 = DSBOR is disabled in Deep Sleep (does not affect operation in non-Deep Sleep modes)

bit 2 **VBTBOR:** V_{BAT} BOR Enable bit
1 = V_{BAT} BOR is enabled
0 = V_{BAT} BOR is disabled

bit 1 **Unimplemented:** Read as ‘1’

bit 0 **$\overline{\text{RETEN}}$:** Retention Voltage Regulator Control Enable bit
1 = Retention voltage regulator is disabled
0 = Retention voltage regulator is enabled; regulator power in Sleep mode is controlled by SRETEN (RCON4<4>)

REGISTER 28-13: CONFIG8L: CONFIGURATION REGISTER 8 LOW (BYTE ADDRESS 30000Eh)

R/WO-1	R/WO-1	R/WO-1	R/WO-1	R/WO-1	U-1	U-1	U-1
DSWDTPS4	DSWDTPS3	DSWDTPS2	DSWDTPS1	DSWDTPS0	—	—	—
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-3 **DSWDTPS<4:0>:** Deep Sleep Watchdog Timer Postscale Select bits
The DS WDT prescaler is 32; this creates an approximate base time unit of 1 ms.
11111 = 1:2^36 (25.7 days)
11110 = 1:2^35 (12.8 days)
11101 = 1:2^34 (6.4 days)
11100 = 1:2^33 (77.0 hours)
11011 = 1:2^32 (38.5 hours)
11010 = 1:2^31 (19.2 hours)
11001 = 1:2^30 (9.6 hours)
11000 = 1:2^29 (4.8 hours)
10111 = 1:2^28 (2.4 hours)
10110 = 1:2^27 (72.2 minutes)
10101 = 1:2^26 (36.1 minutes)
10100 = 1:2^25 (18.0 minutes)
10011 = 1:2^24 (9.0 minutes)
10010 = 1:2^23 (4.5 minutes)
10001 = 1:2^22 (135.3s)
10000 = 1:2^21 (67.7s)
01111 = 1:2^20 (33.825s)
01110 = 1:2^19 (16.912s)
01101 = 1:2^18 (8.456s)
01100 = 1:2^17 (4.228s)
01011 = 1:65536 (2.114s)
01010 = 1:32768 (1.057s)
01001 = 1:16384 (528.5 ms)
01000 = 1:8192 (264.3 ms)
00111 = 1:4096 (132.1 ms)
00110 = 1:2048 (66.1 ms)
00101 = 1:1024 (33 ms)
00100 = 1:512 (16.5 ms)
00011 = 1:256 (8.3 ms)
00010 = 1:128 (4.1 ms)
00001 = 1:64 (2.1 ms)
00000 = 1:32 (1 ms)

bit 2-0 **Unimplemented:** Read as '1'

REGISTER 28-14: CONFIG8H: CONFIGURATION REGISTER 8 HIGH (BYTE ADDRESS 30000Fh)

U-1	U-1	U-1	U-1	U-1	U-1	R/WO-1	R/WO-1
—	—	—	—	—	—	DSWDTOSC	DSWDTEN
bit 7							bit 0

Legend:	P = Programmable bit	WO = Write-Once bit	
R = Readable bit	W = Writable bit	U = Unimplemented bit	
-n = Value at POR	‘1’ = Bit is set	‘0’ = Bit is cleared	x = Bit is unknown

bit 7-4 **Unimplemented:** Program the corresponding Flash Configuration bit to ‘1’

bit 3-2 **Unimplemented:** Read as ‘1’

bit 1 **DSWDTOSC:** DSWDT Reference Clock Select bit
1 = DSWDT uses INTOSC/LPRC as the reference clock
0 = DSWDT uses T1OSC/SOSC as the reference clock

bit 0 **DSWDTEN:** Deep Sleep Watchdog Timer Enable bit
1 = DSWDT is enabled
0 = DSWDT is disabled

REGISTER 28-15: DEVID2: DEVICE ID REGISTER 2 FOR THE PIC18FXXJ94

R	R	R	R	R	R	R	R
DEV10	DEV9	DEV8	DEV7	DEV6	DEV5	DEV4	DEV3
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-0 **DEV<10:3>:** Device ID bits
These bits are used with the DEV<2:0> bits in the Device ID Register 1 to identify the part number.

REGISTER 28-16: DEVID1: DEVICE ID REGISTER 1 FOR THE PIC18FXXJ94

R	R	R	R	R	R	R	R
DEV2	DEV1	DEV0	REV4	REV3	REV2	REV1	REV0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7-5 **DEV<2:0>:** Device ID bits
These bits are used with the DEV<10:3> bits in the Device ID Register 2 to identify the part number:

```
0110 0010 101 = PIC18F97J94
0110 0010 111 = PIC18F96J94
0110 0011 000 = PIC18F95J94
0110 0011 001 = PIC18F87J94
0110 0011 011 = PIC18F86J94
0110 0011 100 = PIC18F85J94
0110 0011 101 = PIC18F67J94
0110 0011 111 = PIC18F66J94
0110 0100 000 = PIC18F65J94
```

bit 4-0 **REV<4:0>:** Revision ID bits
These bits are used to indicate the device revision.

28.2 Watchdog Timer (WDT)

For the PIC18FXXJ94 of devices, the WDT is driven by the LF-INTOSC source. When the WDT is enabled, the clock source is also enabled. The nominal WDT period is 4 ms and has the same stability as the LF-INTOSC Oscillator.

The 4 ms period of the WDT is multiplied by a 16-bit postscaler. Any output of the WDT postscaler is selected by a multiplexer, controlled by bits in Configuration Register 2H. Available periods range from 4 ms to 4,194 seconds (about one hour). The WDT and postscaler are cleared when any of the following events occur: a SLEEP or CLRWDT instruction is executed, the IRCFx bits (OSCCON3<2:0>) are changed or a clock failure has occurred.

28.2.1 WINDOWED OPERATION

The Watchdog Timer has an optional Fixed Window mode of operation. In this Windowed mode, CLRWDT instructions can only reset the WDT during the last 1/4 of the programmed WDT period. A CLRWDT instruction executed before that window causes a WDT Reset, similar to a WDT time-out.

Windowed WDT mode is enabled by programming the WINDIS Configuration bit (CONFIG6H<2>) to '0'.

The WDT can be operated in one of four modes, as determined by WDTEN<1:0> (CONFIG6H<1:0>. The four modes are:

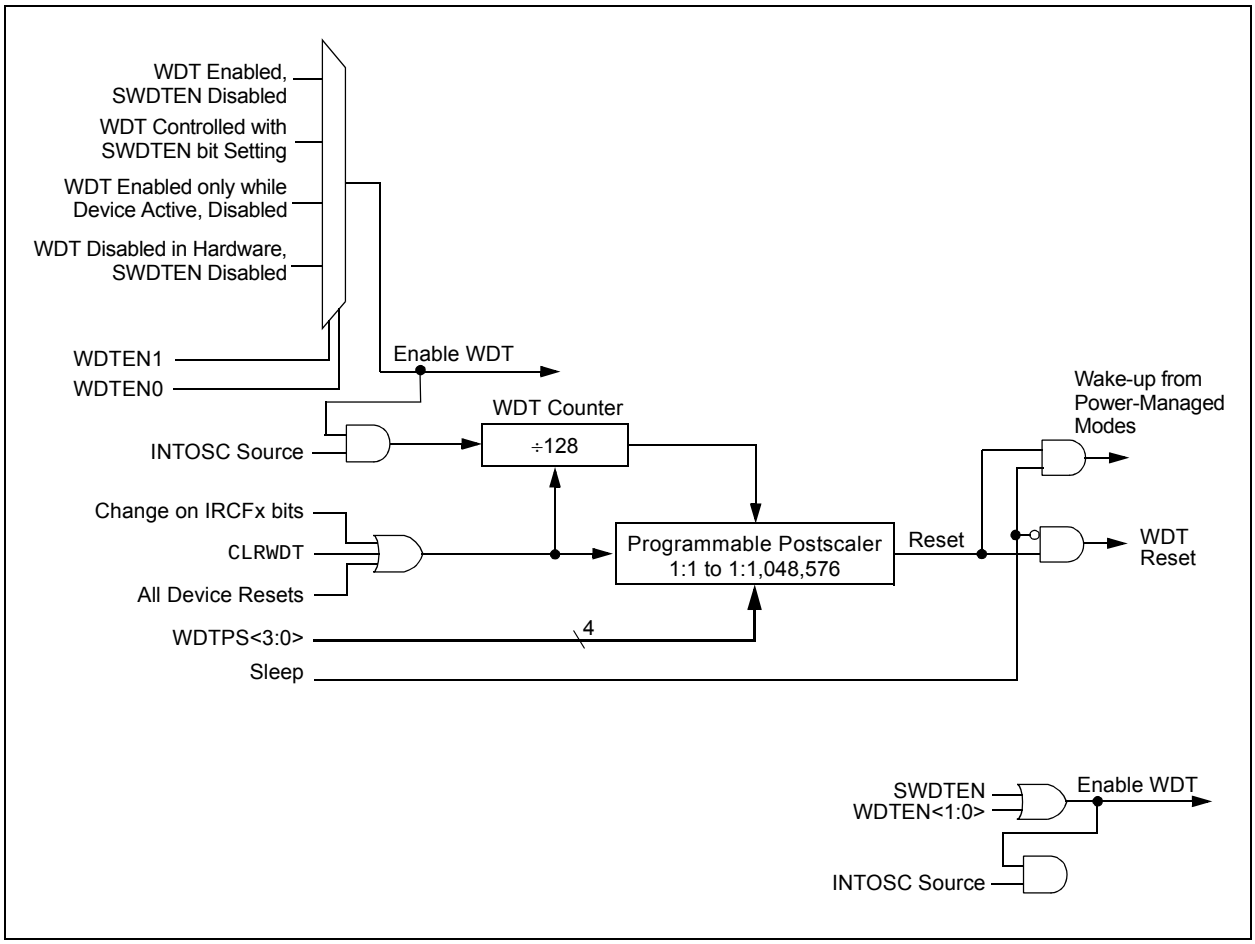
- WDT Enabled
- WDT Disabled
- WDT under Software Control, SWDTEN (RCON2<5>)
- WDT:
  - Enabled during normal operation
  - Disabled during Sleep

Note 1: The CLRWDT and SLEEP instructions clear the WDT and postscaler counts when executed.

2: Changing the setting of the IRCFx bits (OSCCON3<2:0>) clear the WDT and postscaler counts.

3: When a CLRWDT instruction is executed, the postscaler count will be cleared.

FIGURE 28-1: WDT BLOCK DIAGRAM



28.2.2 CONTROL REGISTER

Register 28-17 shows the RCON2 register. This is a readable and writable register which contains a control bit that allows software to override the WDT Enable Configuration bit, but only if the Configuration bit has disabled the WDT.

REGISTER 28-17: RCON2: RESET CONTROL REGISTER 2

R/W, HS-0	U-0	R/W-0	U-0	U-0	U-0	U-0	U-0
EXTR(1)	—	SWDTEN(2)	—	—	—	—	—
bit 7							bit 0

Legend:			
HS = Hardware Settable bit			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

bit 7 **EXTR:** External Reset ($\overline{MCLR}$) Pin bit(1)
- 1 = A Master Clear (pin) Reset has occurred
- 0 = A Master Clear (pin) Reset has not occurred

bit 6 **Unimplemented**: Read as '0'

bit 5 **SWDTEN:** Software Controlled Watchdog Timer Enable bit(2)
- 1 = Watchdog Timer is on
- 0 = Watchdog Timer is off

bit 4-0 **Unimplemented**: Read as '0'

Note 1: This bit is set in hardware; it can be cleared in software.
2: This bit has no effect unless the Configuration bits, WDTEN<1:0> = 10.

28.3 Two-Speed Start-up

The Two-Speed Start-up feature helps to minimize the latency period from oscillator start-up to code execution by allowing the microcontroller to use the INTOSC (LF-INTOSC, MF-INTOSC, HF-INTOSC) Oscillator as a clock source until the primary clock source is available. It is enabled by setting the IESO Configuration bit.

Two-Speed Start-up should be enabled only if the Primary Oscillator mode is LP, MS or HS (Crystal-Based modes). Other sources do not require an OST start-up delay; for these, Two-Speed Start-up should be disabled.

When enabled, Resets and wake-ups from Sleep mode cause the device to configure itself to run from the internal oscillator block as the clock source, following the time-out of the Power-up Timer (PWRT) after a Power-on Reset is enabled. This allows almost immediate code execution while the Primary Oscillator starts and the OST is running. Once the OST times out, the device automatically switches to PRI_RUN mode.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF<2:0>, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF2:0> bits prior to entering Sleep mode.

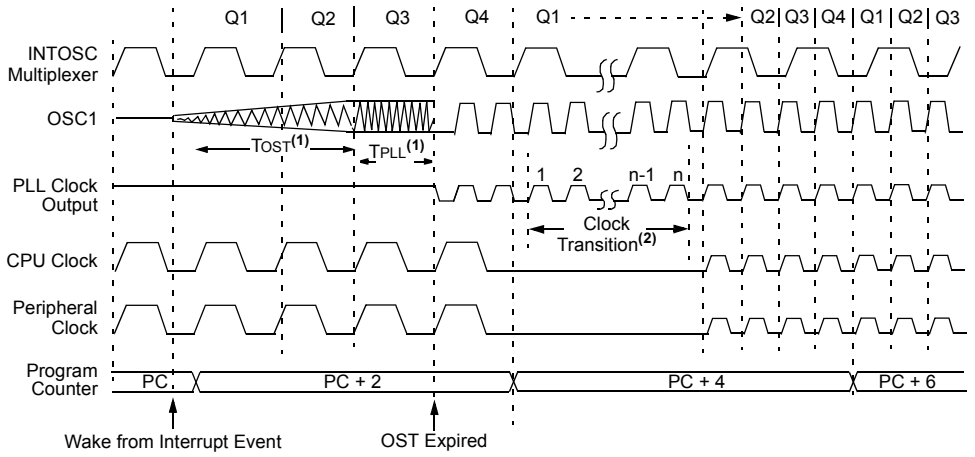
In all other power-managed modes, Two-Speed Start-up is not used. The device will be clocked by the currently selected clock source until the primary clock source becomes available. The setting of the IESO Configuration bit is ignored.

28.3.1 SPECIAL CONSIDERATIONS FOR USING TWO-SPEED START-UP

While using the INTOSC Oscillator in Two-Speed Start-up, the device still obeys the normal command sequences for entering power-managed modes, including multiple SLEEP instructions. In practice, this means that user code can change the NOSC<2:0> bit settings or issue SLEEP instructions before the OST times out. This would allow an application to briefly wake-up, perform routine “housekeeping” tasks and return to Sleep before the device starts to operate from the Primary Oscillator.

User code can also check if the primary clock source is currently providing the device clocking by checking the status of the COSC<2:0> bits (OSCCON<2:0>). If the bit is set, the Primary Oscillator is providing the clock. Otherwise, the internal oscillator block is providing the clock during wake-up from Reset or Sleep mode.

FIGURE 28-2: TIMING TRANSITION FOR TWO-SPEED START-UP (INTOSC TO HSPLL)



Note 1: TOST = 1024 TOSC; TPLL = 2 ms (approx). These intervals are not shown to scale.

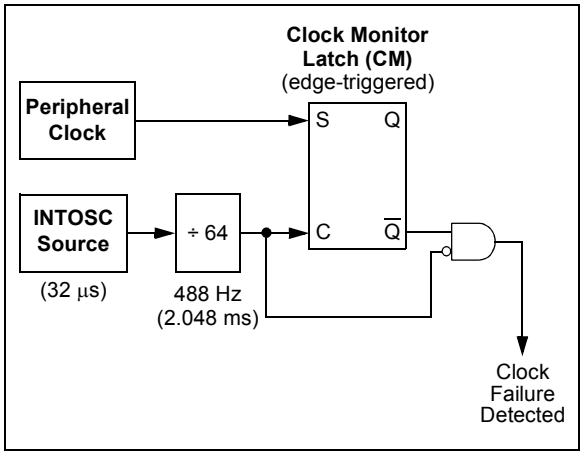
2: Clock transition typically occurs within 2-4 TOSC.

28.4 Fail-Safe Clock Monitor

The Fail-Safe Clock Monitor (FSCM) allows the microcontroller to continue operation in the event of an external oscillator failure by automatically switching the device clock to the internal oscillator block. The FSCM function is enabled by clearing the FSCMx Configuration bits.

When FSCM is enabled, the LF-INTOSC Oscillator runs at all times to monitor clocks to peripherals and provides a backup clock in the event of a clock failure. Clock monitoring (shown in Figure 28-3) is accomplished by creating a sample clock signal, which is the output from the LF-INTOSC, divided by 64. This allows ample time between FSCM sample clocks for a peripheral clock edge to occur. The peripheral device clock and the sample clock are presented as inputs to the Clock Monitor (CM) latch. The CM is set on the falling edge of the device clock source, but cleared on the rising edge of the sample clock.

FIGURE 28-3: FSCM BLOCK DIAGRAM



Clock failure is tested for on the falling edge of the sample clock. If a sample clock falling edge occurs while CM is still set, a clock failure has been detected (Figure 28-4). This causes the following:

- The FSCM generates an oscillator fail interrupt by setting bit, OSCFIF (PIR2<7>)
- The device clock source switches to the internal oscillator block (OSCCON is not updated to show the current clock source – this is the fail-safe condition)
- The WDT is reset

During switchover, the postscaler frequency from the internal oscillator block may not be sufficiently stable for timing-sensitive applications. In these cases, it may be desirable to select another clock configuration and enter an alternate power-managed mode. This can be done to attempt a partial recovery or execute a controlled shutdown. See **Section 28.3.1 "Special Considerations for Using Two-Speed Start-up"** for more details.

To use a higher clock speed on wake-up, the INTOSC or postscaler clock sources can be selected to provide a higher clock speed by setting bits, IRCF<2:0>, immediately after Reset. For wake-ups from Sleep, the INTOSC or postscaler clock sources can be selected by setting the IRCF<2:0> bits prior to entering Sleep mode.

The FSCM will detect only failures of the primary or secondary clock sources. If the internal oscillator block fails, no failure would be detected nor would any action be possible.

28.4.1 FSCM AND THE WATCHDOG TIMER

Both the FSCM and the WDT are clocked by the INTOSC Oscillator. Since the WDT operates with a separate divider and counter, disabling the WDT has no effect on the operation of the INTOSC Oscillator when the FSCM is enabled.

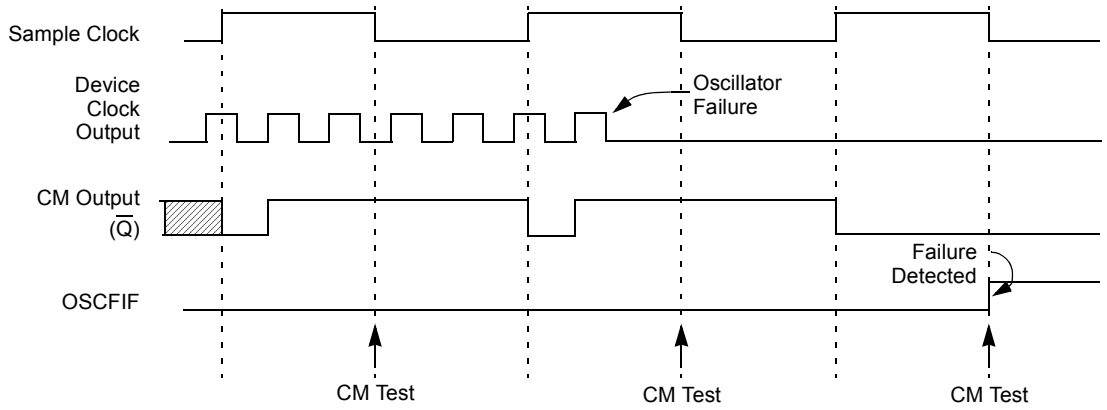
As already noted, the clock source is switched to the INTOSC clock when a clock failure is detected. Depending on the frequency selected by the IRCF<2:0> bits, this may mean a substantial change in the speed of code execution. If the WDT is enabled with a small prescale value, a decrease in clock speed allows a WDT time-out to occur and a subsequent device Reset. For this reason, Fail-Safe Clock events also reset the WDT and postscaler, allowing it to start timing from when execution speed was changed and decreasing the likelihood of an erroneous time-out.

28.4.2 EXITING FAIL-SAFE OPERATION

The Fail-Safe condition is terminated by either a device Reset or by entering a power-managed mode. On Reset, the controller starts the primary clock source, specified in Configuration Register 1H (with any required start-up delays that are required for the oscillator mode, such as the OST or PLL timer). The INTOSC multiplexer provides the device clock until the primary clock source becomes ready (similar to a Two-Speed Start-up). The clock source is then switched to the primary clock automatically after an OST. The Fail-Safe Clock Monitor then resumes monitoring the peripheral clock.

The primary clock source may never become ready during start-up. In this case, operation is clocked by the INTOSC multiplexer. The OSCCON register will remain in its Reset state until a power-managed mode is entered.

FIGURE 28-4: FSCM TIMING DIAGRAM



Note: The device clock is normally at a much higher frequency than the sample clock. The relative frequencies in this example have been chosen for clarity.

28.4.3 FSCM INTERRUPTS IN POWER-MANAGED MODES

By entering a power-managed mode, the clock multiplexer selects the clock source selected by the OSCCON register. Fail-Safe Clock Monitoring of the power-managed clock source resumes in the power-managed mode.

If an oscillator failure occurs during power-managed operation, the subsequent events depend on whether or not the oscillator failure interrupt is enabled. If enabled (OSCFIF = 1), code execution will be clocked by the INTOSC multiplexer. An automatic transition back to the failed clock source will not occur.

If the interrupt is disabled, subsequent interrupts while in Idle mode will cause the CPU to begin executing instructions while being clocked by the INTOSC source.

28.4.4 POR OR WAKE FROM SLEEP

The FSCM is designed to detect oscillator failure at any point after the device has exited Power-on Reset (POR) or low-power Sleep mode. When the primary device clock is EC, RC or INTOSC modes, monitoring can begin immediately following these events.

For oscillator modes involving a crystal or resonator (HS, HSPLL, LP or MS), the situation is somewhat different. Since the oscillator may require a start-up time considerably longer than the FCSM sample clock time, a false clock failure may be detected. To prevent this, the internal oscillator block is automatically configured as the device clock and functions until the primary clock is stable (when the OST and PLL timers have timed out).

This is identical to Two-Speed Start-up mode. Once the primary clock is stable, the INTOSC returns to its role as the FSCM source.

Note: The same logic that prevents false oscillator failure interrupts on POR, or wake from Sleep, also prevents the detection of the oscillator's failure to start at all following these events. This is avoided by an OST time-out condition and by using a timing routine to determine if the oscillator is taking too long to start. Even so, no oscillator failure interrupt will be flagged.

As noted in **Section 28.3.1 "Special Considerations for Using Two-Speed Start-up"**, it is also possible to select another clock configuration and enter an alternate power-managed mode while waiting for the primary clock to become stable. When the new power-managed mode is selected, the primary clock is disabled.

28.4.5 PROGRAM VERIFICATION AND CODE PROTECTION

For all devices in the PIC18FXXJ94 of devices, the on-chip program memory space is treated as a single block. Code protection for this block is controlled by one Configuration bit, CP0. This bit inhibits external reads and writes to the program memory space. It has no direct effect in normal execution mode.

28.4.6 CONFIGURATION REGISTER PROTECTION

The Configuration registers are protected against untoward changes or reads in two ways. The primary protection is the write-once feature of the Configuration bits, which prevents reconfiguration once the bit has been programmed during a power cycle. To safeguard against unpredictable events, Configuration bit changes, resulting from individual cell level disruptions (such as ESD events), will cause a parity error and trigger a device Reset. This is seen by the user as a Configuration Mismatch (CM) Reset.

The data for the Configuration registers is derived from the FCW in program memory. When the CP0 bit is set, the source data for device configuration is also protected as a consequence.

28.5 In-Circuit Serial Programming

The PIC18FXXJ94 of devices can be serially programmed while in the end application circuit. This is simply done with two lines for clock and data and three other lines for power, ground and the programming voltage. This allows customers to manufacture boards with unprogrammed devices and then program the microcontroller just before shipping the product. This also allows the most recent firmware or a custom firmware to be programmed.

For the various programming modes, see the programming specification

28.6 In-Circuit Debugger

When the $\overline{\text{DEBUG}}$ Configuration bit is programmed to a '0', the In-Circuit Debugger functionality is enabled. This function allows simple debugging functions when used with MPLAB® IDE. When the microcontroller has this feature enabled, some resources are not available for general use. Table 28-3 shows which resources are required by the background debugger.

TABLE 28-3: DEBUGGER RESOURCES

I/O Pins:	RB6, RB7
Stack:	Two levels
Program Memory:	512 bytes
Data Memory:	10 bytes

To use the In-Circuit Debugger function of the microcontroller, the design must implement In-Circuit Serial Programming connections to $\overline{\text{MCLR}}$, V_{DD}, V_{SS}, RB7 and RB6. This will interface to the In-Circuit Debugger module available from Microchip or one of the third-party development tool companies.

29.0 INSTRUCTION SET SUMMARY

The PIC18FXXJ94 of devices incorporates the standard set of 75 PIC18 core instructions, as well as an extended set of eight new instructions for the optimization of code that is recursive or that utilizes a software stack. The extended set is discussed later in this section.

29.1 Standard Instruction Set

The standard PIC18 MCU instruction set adds many enhancements to the previous PIC® MCU instruction sets, while maintaining an easy migration from these PIC MCU instruction sets. Most instructions are a single program memory word (16 bits), but there are four instructions that require two program memory locations.

Each single-word instruction is a 16-bit word divided into an opcode, which specifies the instruction type and one or more operands, which further specify the operation of the instruction.

The instruction set is highly orthogonal and is grouped into four basic categories:

- **Byte-oriented** operations
- **Bit-oriented** operations
- **Literal** operations
- **Control** operations

The PIC18 instruction set summary in Table 29-2 lists **byte-oriented**, **bit-oriented**, **literal** and **control** operations. Table 29-1 shows the opcode field descriptions.

Most **byte-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The destination of the result (specified by ‘d’)
3. The accessed memory (specified by ‘a’)

The file register designator, ‘f’, specifies which file register is to be used by the instruction. The destination designator, ‘d’, specifies where the result of the operation is to be placed. If ‘d’ is zero, the result is placed in the WREG register. If ‘d’ is one, the result is placed in the file register specified in the instruction.

All **bit-oriented** instructions have three operands:

1. The file register (specified by ‘f’)
2. The bit in the file register (specified by ‘b’)
3. The accessed memory (specified by ‘a’)

The bit field designator, ‘b’, selects the number of the bit affected by the operation, while the file register designator, ‘f’, represents the number of the file in which the bit is located.

The **literal** instructions may use some of the following operands:

- A literal value to be loaded into a file register (specified by ‘k’)
- The desired FSR register to load the literal value into (specified by ‘f’)
- No operand required (specified by ‘—’)

The **control** instructions may use some of the following operands:

- A program memory address (specified by ‘n’)
- The mode of the CALL or RETURN instructions (specified by ‘s’)
- The mode of the table read and table write instructions (specified by ‘m’)
- No operand required (specified by ‘—’)

All instructions are a single word, except for four double-word instructions. These instructions were made double-word to contain the required information in 32 bits. In the second word, the 4 MSbs are ‘1’s. If this second word is executed as an instruction (by itself), it will execute as a NOP.

All single-word instructions are executed in a single instruction cycle, unless a conditional test is true or the Program Counter is changed as a result of the instruction. In these cases, the execution takes two instruction cycles with the additional instruction cycle(s) executed as a NOP.

The double-word instructions execute in two instruction cycles.

One instruction cycle consists of four oscillator periods. Thus, for an oscillator frequency of 4 MHz, the normal instruction execution time is 1 μs. If a conditional test is true, or the Program Counter is changed as a result of an instruction, the instruction execution time is 2 μs. Two-word branch instructions (if true) would take 3 μs.

Figure 29-1 shows the general formats that the instructions can have. All examples use the convention ‘nnh’ to represent a hexadecimal number.

The Instruction Set Summary, shown in Table 29-2, lists the standard instructions recognized by the Microchip MPASM™ Assembler.

Section 29.1.1 “Standard Instruction Set” provides a description of each instruction.

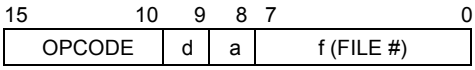
TABLE 29-1: OPCODE FIELD DESCRIPTIONS

Field	Description
a	RAM access bit: a = 0: RAM location in Access RAM (BSR register is ignored) a = 1: RAM bank is specified by BSR register
bbb	Bit address within an 8-bit file register (0 to 7).
BSR	Bank Select Register. Used to select the current RAM bank.
C, DC, Z, OV, N	ALU Status bits: **C**arry, **D**igit **C**arry, **Z**ero, **Ov**erflow, **N**egative.
d	Destination select bit: d = 0: store result in WREG d = 1: store result in file register f
dest	Destination: either the WREG register or the specified register file location.
f	8-bit register file address (00h to FFh), or 2-bit FSR designator (0h to 3h).
f_s	12-bit register file address (000h to FFFh). This is the source address.
f_d	12-bit register file address (000h to FFFh). This is the destination address.
GIE	Global Interrupt Enable bit.
k	Literal field, constant data or label (may be either an 8-bit, 12-bit or a 20-bit value).
label	Label name.
mm * *+ *- +*	The mode of the TBLPTR register for the table read and table write instructions. Only used with table read and table write instructions: No Change to register (such as TBLPTR with table reads and writes) Post-Increment register (such as TBLPTR with table reads and writes) Post-Decrement register (such as TBLPTR with table reads and writes) Pre-Increment register (such as TBLPTR with table reads and writes)
n	The relative address (2's complement number) for relative branch instructions or the direct address for Call/Branch and Return instructions.
PC	Program Counter.
PCL	Program Counter Low Byte.
PCH	Program Counter High Byte.
PCLATH	Program Counter High Byte Latch.
PCLATU	Program Counter Upper Byte Latch.
$\overline{PD}$	Power-Down bit.
PRODH	Product of Multiply High Byte.
PRODL	Product of Multiply Low Byte.
s	Fast Call/Return mode select bit: s = 0: do not update into/from shadow registers s = 1: certain registers loaded into/from shadow registers (Fast mode)
TBLPTR	21-bit Table Pointer (points to a Program Memory location).
TABLAT	8-bit Table Latch.
$\overline{TO}$	Time-out bit.
TOS	Top-of-Stack.
u	Unused or Unchanged.
WDT	Watchdog Timer.
WREG	Working register (accumulator).
x	Don't care ('0' or '1'). The assembler will generate code with x = 0. It is the recommended form of use for compatibility with all Microchip software tools.
z_s	7-bit offset value for Indirect Addressing of register files (source).
z_d	7-bit offset value for Indirect Addressing of register files (destination).
{ }	Optional argument.
[text]	Indicates an Indexed Address.
(text)	The contents of text.
[expr]<n>	Specifies bit n of the register indicated by the pointer expr.
→	Assigned to.
< >	Register bit field.
∈	In the set of.
italics	User-defined term (font is Courier New).

FIGURE 29-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations

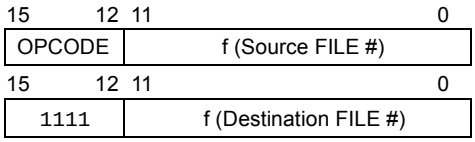
Example Instruction




```
ADDWF MYREG, W, B
```

d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

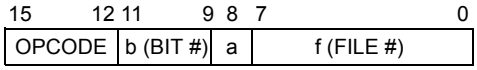
Byte to Byte move operations (2-word)




```
MOVFF MYREG1, MYREG2
```

f = 12-bit file register address

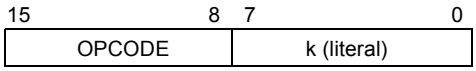
Bit-oriented file register operations




```
BSF MYREG, bit, B
```

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Literal operations

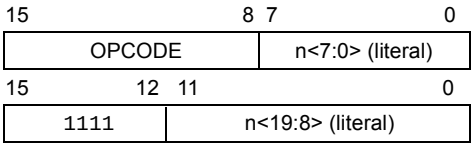



```
MOVLW 7Fh
```

k = 8-bit immediate value

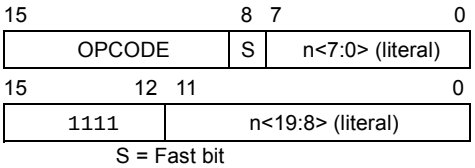
Control operations

CALL, GOTO and Branch operations



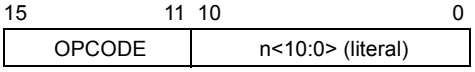

```
GOTO Label
```

n = 20-bit immediate value

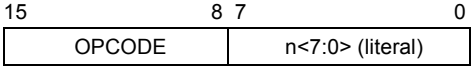



```
CALL MYFUNC
```

S = Fast bit




```
BRA MYFUNC
```




```
BC MYFUNC
```

TABLE 29-2: PIC18F97J94 FAMILY INSTRUCTION SET

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
BYTE-ORIENTED OPERATIONS					
ADDWF f, d, a	Add WREG and f	1	0010 01da ffff ffff	C, DC, Z, OV, N	1, 2
ADDWFC f, d, a	Add WREG and Carry bit to f	1	0010 00da ffff ffff	C, DC, Z, OV, N	1, 2
ANDWF f, d, a	AND WREG with f	1	0001 01da ffff ffff	Z, N	1, 2
CLRF f, a	Clear f	1	0110 101a ffff ffff	Z	2
COMF f, d, a	Complement f	1	0001 11da ffff ffff	Z, N	1, 2
CPFSEQ f, a	Compare f with WREG, Skip =	1 (2 or 3)	0110 001a ffff ffff	None	4
CPFSGT f, a	Compare f with WREG, Skip >	1 (2 or 3)	0110 010a ffff ffff	None	4
CPFSLT f, a	Compare f with WREG, Skip <	1 (2 or 3)	0110 000a ffff ffff	None	1, 2
DECF f, d, a	Decrement f	1	0000 01da ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010 11da ffff ffff	None	1, 2, 3, 4
DCFSNZ f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100 11da ffff ffff	None	1, 2
INCF f, d, a	Increment f	1	0010 10da ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011 11da ffff ffff	None	4
INFSNZ f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100 10da ffff ffff	None	1, 2
IORWF f, d, a	Inclusive OR WREG with f	1	0001 00da ffff ffff	Z, N	1, 2
MOVF f, d, a	Move f	1	0101 00da ffff ffff	Z, N	1
MOVFF f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 ffff ffff ffff 1111 ffff ffff ffff	None	
MOVWF f, a	Move WREG to f	1	0110 111a ffff ffff	None	
MULWF f, a	Multiply WREG with f	1	0000 001a ffff ffff	None	1, 2
NEGF f, a	Negate f	1	0110 110a ffff ffff	C, DC, Z, OV, N	
RLCF f, d, a	Rotate Left f through Carry	1	0011 01da ffff ffff	C, Z, N	1, 2
RLNCF f, d, a	Rotate Left f (No Carry)	1	0100 01da ffff ffff	Z, N	
RRCF f, d, a	Rotate Right f through Carry	1	0011 00da ffff ffff	C, Z, N	
RRNCF f, d, a	Rotate Right f (No Carry)	1	0100 00da ffff ffff	Z, N	
SETF f, a	Set f	1	0110 100a ffff ffff	None	1, 2
SUBFWB f, d, a	Subtract f from WREG with Borrow	1	0101 01da ffff ffff	C, DC, Z, OV, N	
SUBWF f, d, a	Subtract WREG from f	1	0101 11da ffff ffff	C, DC, Z, OV, N	1, 2
SUBWFB f, d, a	Subtract WREG from f with Borrow	1	0101 10da ffff ffff	C, DC, Z, OV, N	
SWAPF f, d, a	Swap Nibbles in f	1	0011 10da ffff ffff	None	4
TSTFSZ f, a	Test f, Skip if 0	1 (2 or 3)	0110 011a ffff ffff	None	1, 2
XORWF f, d, a	Exclusive OR WREG with f	1	0001 10da ffff ffff	Z, N	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

TABLE 29-2: PIC18F97J94 FAMILY INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
BIT-ORIENTED OPERATIONS						
BCF	f, b, a	Bit Clear f	1	1001 bbba ffff ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000 bbba ffff ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011 bbba ffff ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010 bbba ffff ffff	None	3, 4
BTG	f, b, a	Bit Toggle f	1	0111 bbba ffff ffff	None	1, 2
CONTROL OPERATIONS						
BC	n	Branch if Carry	1 (2)	1110 0010 nnnn nnnn	None	
BN	n	Branch if Negative	1 (2)	1110 0110 nnnn nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110 0011 nnnn nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110 0111 nnnn nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110 0101 nnnn nnnn	None	
BNZ	n	Branch if Not Zero	1 (2)	1110 0001 nnnn nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110 0100 nnnn nnnn	None	
BRA	n	Branch Unconditionally	2	1101 0nnn nnnn nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110 0000 nnnn nnnn	None	
CALL	n, s	Call Subroutine 1st word	2	1110 110s kkkk kkkk	None	
		2nd word		1111 kkkk kkkk kkkk		
CLRWDT	—	Clear Watchdog Timer	1	0000 0000 0000 0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000 0000 0000 0111	C	
GOTO	n	Go to Address 1st word	2	1110 1111 kkkk kkkk	None	
		2nd word		1111 kkkk kkkk kkkk		
NOP	—	No Operation	1	0000 0000 0000 0000	None	
NOP	—	No Operation	1	1111 xxxx xxxx xxxx	None	4
POP	—	Pop Top of Return Stack (TOS)	1	0000 0000 0000 0110	None	
PUSH	—	Push Top of Return Stack (TOS)	1	0000 0000 0000 0101	None	
RCALL	n	Relative Call	2	1101 1nnn nnnn nnnn	None	
RESET		Software Device Reset	1	0000 0000 1111 1111	All	
RETFIE	s	Return from Interrupt Enable	2	0000 0000 0001 000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with Literal in WREG	2	0000 1100 kkkk kkkk	None	
RETURN	s	Return from Subroutine	2	0000 0000 0001 001s	None	
SLEEP	—	Go into Standby mode	1	0000 0000 0000 0011	$\overline{TO}$, $\overline{PD}$	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a NOP unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

TABLE 29-2: PIC18F97J94 FAMILY INSTRUCTION SET (CONTINUED)

Mnemonic, Operands	Description	Cycles	16-Bit Instruction Word				Status Affected	Notes
			MSb			LSb		
LITERAL OPERATIONS								
ADDLW k	Add Literal and WREG	1	0000	1111	kkkk	kkkk	C, DC, Z, OV, N	
ANDLW k	AND Literal with WREG	1	0000	1011	kkkk	kkkk	Z, N	
IORLW k	Inclusive OR Literal with WREG	1	0000	1001	kkkk	kkkk	Z, N	
LFSR f, k	Move literal (12-bit) 2nd word to FSR(f) 1st word	2	1110	1110	00ff	kkkk	None	
			1111	0000	kkkk	kkkk		
MOVLB k	Move Literal to BSR<3:0>	1	0000	0001	0000	kkkk	None	
MOVLW k	Move Literal to WREG	1	0000	1110	kkkk	kkkk	None	
MULLW k	Multiply Literal with WREG	1	0000	1101	kkkk	kkkk	None	
RETLW k	Return with Literal in WREG	2	0000	1100	kkkk	kkkk	None	
SUBLW k	Subtract WREG from Literal	1	0000	1000	kkkk	kkkk	C, DC, Z, OV, N	
XORLW k	Exclusive OR Literal with WREG	1	0000	1010	kkkk	kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS								
TBLRD*	Table Read	2	0000	0000	0000	1000	None	
TBLRD*+	Table Read with Post-Increment		0000	0000	0000	1001	None	
TBLRD*-	Table Read with Post-Decrement		0000	0000	0000	1010	None	
TBLRD+*	Table Read with Pre-Increment		0000	0000	0000	1011	None	
TBLWT*	Table Write	2	0000	0000	0000	1100	None	
TBLWT*+	Table Write with Post-Increment		0000	0000	0000	1101	None	
TBLWT*-	Table Write with Post-Decrement		0000	0000	0000	1110	None	
TBLWT+*	Table Write with Pre-Increment		0000	0000	0000	1111	None	

Note 1: When a PORT register is modified as a function of itself (e.g., `MOVF PORTB, 1, 0`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and where applicable, d = 1), the prescaler will be cleared if assigned.

3: If the Program Counter (PC) is modified or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a `NOP`.

4: Some instructions are two-word instructions. The second word of these instructions will be executed as a `NOP` unless the first word of the instruction retrieves the information embedded in these 16 bits. This ensures that all program memory locations have a valid instruction.

29.1.1 STANDARD INSTRUCTION SET

ADDLW **ADD Literal to W**

Syntax: ADDLW k

Operands: $0 \le k \le 255$

Operation: $(W) + k \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal 'k' and the result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: ADDLW 15h

Before Instruction
W = 10h
After Instruction
W = 25h

ADDWF **ADD W to f**

Syntax: ADDWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(W) + (f) \rightarrow$ dest

Status Affected: N, OV, C, DC, Z

Encoding:

0010	01da	ffff	ffff

Description: Add W to register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: ADDWF REG, 0, 0

Before Instruction
W = 17h
REG = 0C2h
After Instruction
W = 0D9h
REG = 0C2h

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

ADDWFC **ADD W and Carry bit to f**

Syntax: ADDWFC f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) + (f) + (C) → dest

Status Affected: N,OV, C, DC, Z

Encoding:

0010	00da	ffff	ffff

Description: Add W, the Carry flag and data memory location 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in data memory location 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: ADDWFC REG, 0, 1

Before Instruction
Carry bit = 1
REG = 02h
W = 4Dh
After Instruction
Carry bit = 0
REG = 02h
W = 50h

ANDLW **AND Literal with W**

Syntax: ANDLW k

Operands: $0 \le k \le 255$

Operation: (W) .AND. k → W

Status Affected: N, Z

Encoding:

0000	1011	kkkk	kkkk

Description: The contents of W are ANDed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: ANDLW 05Fh

Before Instruction
W = A3h
After Instruction
W = 03h

ANDWF **AND W with f**

Syntax: ANDWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) .AND. (f) → dest

Status Affected: N, Z

Encoding:

0001	01da	ffff	ffff

Description: The contents of W are ANDed with register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `ANDWF REG, 0, 0`

Before Instruction
W = 17h
REG = C2h
After Instruction
W = 02h
REG = C2h

BC **Branch if Carry**

Syntax: BC n

Operands: $-128 \le n \le 127$

Operation: if Carry bit is '1',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0010	nnnn	nnnn

Description: If the Carry bit is '1', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: `HERE BC 5`

Before Instruction
PC = address (HERE)
After Instruction
If Carry = 1;
PC = address (HERE + 12)
If Carry = 0;
PC = address (HERE + 2)

BCF **Bit Clear f**

Syntax: BCF f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: $0 \rightarrow f<b>$

Status Affected: None

Encoding:

1001	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is cleared.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `BCF FLAG_REG, 7, 0`

Before Instruction
FLAG_REG = C7h
After Instruction
FLAG_REG = 47h

BN **Branch if Negative**

Syntax: BN n

Operands: $-128 \le n \le 127$

Operation: if Negative bit is '1',
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0110	nnnn	nnnn

Description: If the Negative bit is '1', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example:
```
HERE    BN   Jump
```
Before Instruction
PC = address (HERE)
After Instruction
If Negative = 1;
PC = address (Jump)
If Negative = 0;
PC = address (HERE + 2)

BNC **Branch if Not Carry**

Syntax: BNC n

Operands: -128 ≤ n ≤ 127

Operation: if Carry bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0011	nnnn	nnnn

Description: If the Carry bit is '0', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNC Jump

Before Instruction
PC = address (HERE)
After Instruction
If Carry = 0;
PC = address (Jump)
If Carry = 1;
PC = address (HERE + 2)

BNN **Branch if Not Negative**

Syntax: BNN n

Operands: -128 ≤ n ≤ 127

Operation: if Negative bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0111	nnnn	nnnn

Description: If the Negative bit is '0', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNN Jump

Before Instruction
PC = address (HERE)
After Instruction
If Negative = 0;
PC = address (Jump)
If Negative = 1;
PC = address (HERE + 2)

BNOV **Branch if Not Overflow**

Syntax: BNOV n

Operands: -128 ≤ n ≤ 127

Operation: if Overflow bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0101	nnnn	nnnn

Description: If the Overflow bit is '0', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNOV Jump

Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 0;
PC = address (Jump)
If Overflow = 1;
PC = address (HERE + 2)

BNZ **Branch if Not Zero**

Syntax: BNZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '0',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0001	nnnn	nnnn

Description: If the Zero bit is '0', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BNZ Jump

Before Instruction
PC = address (HERE)
After Instruction
If Zero = 0;
PC = address (Jump)
If Zero = 1;
PC = address (HERE + 2)

BRA **Unconditional Branch**

Syntax: BRA n

Operands: -1024 ≤ n ≤ 1023

Operation: (PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1101	0nnn	nnnn	nnnn

Description: Add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a 2-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: HERE BRA Jump

Before Instruction
PC = address (HERE)
After Instruction
PC = address (Jump)

BSF **Bit Set f**

Syntax: BSF f, b {,a}

Operands: 0 ≤ f ≤ 255
0 ≤ b ≤ 7
a ∈ [0,1]

Operation: 1 → f<b>

Status Affected: None

Encoding:

1000	bbba	ffff	ffff

Description: Bit 'b' in register 'f' is set.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: BSF FLAG_REG, 7, 1

Before Instruction
FLAG_REG = 0Ah
After Instruction
FLAG_REG = 8Ah

BTFSC **Bit Test File, Skip if Clear**

Syntax: BTFSC f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b \le 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 0

Status Affected: None

Encoding:

1011	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '0', then the next instruction is skipped. If bit 'b' is '0', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    BTFSC   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (TRUE)
If FLAG<1> = 1;
PC = address (FALSE)

BTFSS **Bit Test File, Skip if Set**

Syntax: BTFSS f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: skip if (f<b>) = 1

Status Affected: None

Encoding:

1010	bbba	ffff	ffff

Description: If bit 'b' in register 'f' is '1', then the next instruction is skipped. If bit 'b' is '1', then the next instruction fetched during the current instruction execution is discarded and a NOP is executed instead, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    BTFSS   FLAG, 1, 0
FALSE   :
TRUE    :
```

Before Instruction
PC = address (HERE)
After Instruction
If FLAG<1> = 0;
PC = address (FALSE)
If FLAG<1> = 1;
PC = address (TRUE)

BTG **Bit Toggle f**

Syntax: BTG f, b {,a}

Operands: $0 \le f \le 255$
$0 \le b < 7$
$a \in [0,1]$

Operation: $(\overline{f<b>}) \rightarrow f<b>$

Status Affected: None

Encoding:

0111	bbba	ffff	ffff

Description: Bit ‘b’ in data memory location, ‘f’, is inverted.

If ‘a’ is ‘0’, the Access Bank is selected. If ‘a’ is ‘1’, the BSR is used to select the GPR bank.

If ‘a’ is ‘0’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write register ‘f’

Example: `BTG PORTC, 4, 0`

Before Instruction:
PORTC = 0111 0101 [75h]
After Instruction:
PORTC = 0110 0101 [65h]

BOV **Branch if Overflow**

Syntax: BOV n

Operands: $-128 \le n \le 127$

Operation: if Overflow bit is ‘1’,
$(PC) + 2 + 2n \rightarrow PC$

Status Affected: None

Encoding:

1110	0100	nnnn	nnnn

Description: If the Overflow bit is ‘1’, then the program will branch.

The 2’s complement number ‘2n’ is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:
If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal ‘n’	Process Data	No operation

Example:
```
HERE    BOV   Jump
```
Before Instruction
PC = address (HERE)
After Instruction
If Overflow = 1;
PC = address (Jump)
If Overflow = 0;
PC = address (HERE + 2)

BZ **Branch if Zero**

Syntax: BZ n

Operands: -128 ≤ n ≤ 127

Operation: if Zero bit is '1',
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1110	0000	nnnn	nnnn

Description: If the Zero bit is '1', then the program will branch.

The 2's complement number '2n' is added to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is then a 2-cycle instruction.

Words: 1

Cycles: 1(2)

Q Cycle Activity:

If Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	Write to PC
No operation	No operation	No operation	No operation

If No Jump:

Q1	Q2	Q3	Q4
Decode	Read literal 'n'	Process Data	No operation

Example: HERE BZ Jump

Before Instruction
PC = address (HERE)
After Instruction
If Zero = 1;
PC = address (Jump)
If Zero = 0;
PC = address (HERE + 2)

CALL **Subroutine Call**

Syntax: CALL k {,s}

Operands: 0 ≤ k ≤ 1048575
s ∈ [0,1]

Operation: (PC) + 4 → TOS,
k → PC<20:1>;
if s = 1
(W) → WS,
(STATUS) → STATUSS,
(BSR) → BSRS

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	110s	$k_7$kkk	kkkk$_0$
2nd word(k<19:8>)	1111	k_{19}kkk	kkkk	kkkk$_8$

Description: Subroutine call of entire 2-Mbyte memory range. First, return address (PC+ 4) is pushed onto the return stack. If 's' = 1, the W, STATUS and BSR registers are also pushed into their respective shadow registers, WS, STATUSS and BSRS. If 's' = 0, no update occurs. Then, the 20-bit value 'k' is loaded into PC<20:1>. CALL is a 2-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	Push PC to stack	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: HERE CALL THERE,1

Before Instruction
PC = address (HERE)
After Instruction
PC = address (THERE)
TOS = address (HERE + 4)
WS = W
BSRS = BSR
STATUSS = STATUS

CLRF **Clear f**

Syntax: CLRF f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: 000h → f,
1 → Z

Status Affected: Z

Encoding:

0110	101a	ffff	ffff

Description: Clears the contents of the specified register.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: CLRF FLAG_REG,1

Before Instruction
FLAG_REG = 5Ah
After Instruction
FLAG_REG = 00h

CLRWDT **Clear Watchdog Timer**

Syntax: CLRWDT

Operands: None

Operation: 000h → WDT,
000h → WDT postscaler,
1 → $\overline{TO}$,
1 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0100

Description: CLRWDT instruction resets the Watchdog Timer. It also resets the post-scaler of the WDT. Status bits, $\overline{TO}$ and $\overline{PD}$, are set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	No operation

Example: CLRWDT

Before Instruction
WDT Counter = ?
After Instruction
WDT Counter = 00h
WDT Postscaler = 0
$\overline{TO}$ = 1
$\overline{PD}$ = 1

COMF **Complement f**

Syntax: COMF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $\bar{f} \rightarrow$ dest

Status Affected: N, Z

Encoding:

0001	11da	ffff	ffff

Description: The contents of register 'f' are complemented. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: `COMF REG, 0, 0`

Before Instruction
REG = 13h
After Instruction
REG = 13h
W = ECh

CPFSEQ **Compare f with W, Skip if f = W**

Syntax: CPFSEQ f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) = (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	001a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.

If 'f' = W, then the fetched instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSEQ REG, 0
NEQUAL  :
EQUAL   :
```

Before Instruction
PC Address = HERE
W = ?
REG = ?
After Instruction
If REG = W;
PC = Address (EQUAL)
If REG ≠ W;
PC = Address (NEQUAL)

CPFSGT **Compare f with W, Skip if f > W**

Syntax: CPFSGT f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) > (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	010a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of the W by performing an unsigned subtraction.

If the contents of 'f' are greater than the contents of WREG, then the fetched instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE      CPFSGT REG, 0
NGREATER  :
GREATER   :
```
Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG > W;
PC = Address (GREATER)
If REG ≤ W;
PC = Address (NGREATER)

CPFSLT **Compare f with W, Skip if f < W**

Syntax: CPFSLT f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (f) – (W),
skip if (f) < (W)
(unsigned comparison)

Status Affected: None

Encoding:

0110	000a	ffff	ffff

Description: Compares the contents of data memory location 'f' to the contents of W by performing an unsigned subtraction.

If the contents of 'f' are less than the contents of W, then the fetched instruction is discarded and a NOP is executed instead, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    CPFSLT REG, 1
NLESS   :
LESS    :
```
Before Instruction
PC = Address (HERE)
W = ?
After Instruction
If REG < W;
PC = Address (LESS)
If REG ≥ W;
PC = Address (NLESS)

DAW **Decimal Adjust W Register**

Syntax: DAW

Operands: None

Operation: If [W<3:0> > 9] or [DC = 1], then
(W<3:0>) + 6 → W<3:0>;
else
(W<3:0>) → W<3:0>

If [W<7:4> > 9] or [C = 1], then
(W<7:4>) + 6 → W<7:4>;
C = 1;
else
(W<7:4>) → W<7:4>

Status Affected: C

Encoding:

0000	0000	0000	0111

Description: DAW adjusts the 8-bit value in W, resulting from the earlier addition of two variables (each in packed BCD format) and produces a correct packed BCD result.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register W	Process Data	Write W

Example 1: DAW

Before Instruction
W = A5h
C = 0
DC = 0
After Instruction
W = 05h
C = 1
DC = 0

Example 2:

Before Instruction
W = CEh
C = 0
DC = 0
After Instruction
W = 34h
C = 1
DC = 0

DECF **Decrement f**

Syntax: DECF f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f) – 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0000	01da	ffff	ffff

Description: Decrement register, 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: DECF CNT, 1, 0

Before Instruction
CNT = 01h
Z = 0
After Instruction
CNT = 00h
Z = 1

DECFSZ **Decrement f, Skip if 0**

Syntax: DECFSZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0010	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If the result is '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    DECFSZ  CNT, 1, 1
        GOTO    LOOP
CONTINUE
```
Before Instruction
PC = Address (HERE)
After Instruction
CNT = CNT – 1
If CNT = 0;
PC = Address (CONTINUE)
If CNT ≠ 0;
PC = Address (HERE + 2)

DCFSNZ **Decrement f, Skip if Not 0**

Syntax: DCFSNZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	11da	ffff	ffff

Description: The contents of register 'f' are decremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    DCFSNZ  TEMP, 1, 0
ZERO    :
NZERO   :
```
Before Instruction
TEMP = ?
After Instruction
TEMP = TEMP – 1,
If TEMP = 0;
PC = Address (ZERO)
If TEMP ≠ 0;
PC = Address (NZERO)

GOTO **Unconditional Branch**

Syntax: GOTO k

Operands: $0 \le k \le 1048575$

Operation: $k \rightarrow PC<20:1>$

Status Affected: None

Encoding:

1st word (k<7:0>)	1110	k_7kkk	$kkkk_0$
2nd word(k<19:8>)	1111	$k_{19}kkk$	$kkkk_8$

Description: GOTO allows an unconditional branch anywhere within entire 2-Mbyte memory range. The 20-bit value 'k' is loaded into PC<20:1>. GOTO is always a 2-cycle instruction.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'<7:0>,	No operation	Read literal 'k'<19:8>, Write to PC
No operation	No operation	No operation	No operation

Example: GOTO THERE

After Instruction
PC = Address (THERE)

INCF **Increment f**

Syntax: INCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) + 1 → dest

Status Affected: C, DC, N, OV, Z

Encoding:

0010	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: INCF CNT, 1, 0

Before Instruction
CNT = FFh
Z = 0
C = ?
DC = ?
After Instruction
CNT = 00h
Z = 1
C = 1
DC = 1

INCFSZ **Increment f, Skip if 0**

Syntax: INCFSZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) + 1 → dest,
skip if result = 0

Status Affected: None

Encoding:

0011	11da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If the result is '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    INCFSZ   CNT, 1, 0
NZERO   :
ZERO    :
```
Before Instruction
PC = Address (HERE)
After Instruction
CNT = CNT + 1
If CNT = 0;
PC = Address (ZERO)
If CNT ≠ 0;
PC = Address (NZERO)

INFSNZ **Increment f, Skip if Not 0**

Syntax: INFSNZ f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) + 1 → dest,
skip if result ≠ 0

Status Affected: None

Encoding:

0100	10da	ffff	ffff

Description: The contents of register 'f' are incremented. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If the result is not '0', the next instruction which is already fetched is discarded and a NOP is executed instead, making it a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    INFSNZ  REG, 1, 0
ZERO
NZERO
```
Before Instruction
PC = Address (HERE)
After Instruction
REG = REG + 1
If REG ≠ 0;
PC = Address (NZERO)
If REG = 0;
PC = Address (ZERO)

IORLW **Inclusive OR Literal with W**

Syntax: IORLW k

Operands: $0 \le k \le 255$

Operation: (W) .OR. k → W

Status Affected: N, Z

Encoding:

0000	1001	kkkk	kkkk

Description: The contents of W are ORed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: IORLW 35h

Before Instruction
W = 9Ah
After Instruction
W = BFh

IORWF **Inclusive OR W with f**

Syntax: IORWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (W) .OR. (f) → dest

Status Affected: N, Z

Encoding:

0001	00da	ffff	ffff

Description: Inclusive OR W with register 'f'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: IORWF RESULT, 0, 1

Before Instruction
RESULT = 13h
W = 91h
After Instruction
RESULT = 13h
W = 93h

LFSR **Load FSR**

Syntax: LFSR f, k

Operands: $0 \le f \le 2$
$0 \le k \le 4095$

Operation: $k \rightarrow FSRf$

Status Affected: None

Encoding:

1110 1111	1110 0000	00ff $k_7$kkk	k_{11}kkk kkkk

Description: The 12-bit literal 'k' is loaded into the file select register pointed to by 'f'.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k' MSB	Process Data	Write literal 'k' MSB to FSRfH
Decode	Read literal 'k' LSB	Process Data	Write literal 'k' to FSRfL

Example: `LFSR 2, 3ABh`

After Instruction
FSR2H = 03h
FSR2L = ABh

MOVF **Move f**

Syntax: MOVF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $f \rightarrow dest$

Status Affected: N, Z

Encoding:

0101	00da	ffff	ffff

Description: The contents of register 'f' are moved to a destination dependent upon the status of 'd'. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'. Location 'f' can be anywhere in the 256-byte bank.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write W

Example: `MOVF REG, 0, 0`

Before Instruction
REG = 22h
W = FFh
After Instruction
REG = 22h
W = 22h

MOVFF **Move f to f**

Syntax: MOVFF f_s,f_d

Operands: $0 \le f_s \le 4095$
$0 \le f_d \le 4095$

Operation: $(f_s) \rightarrow f_d$

Status Affected: None

Encoding:

1st word (source)	1100	ffff	ffff	$ffff_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of source register, 'f_s', are moved to destination register 'f_d'. Location of source 'f_s' can be anywhere in the 4096-byte data space (000h to FFFh) and location of destination 'f_d' can also be anywhere from 000h to FFFh.

Either source or destination can be W (a useful special situation).

MOVFF is particularly useful for transferring a data memory location to a peripheral register (such as the transmit buffer or an I/O port).

The MOVFF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f' (src)	Process Data	No operation
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example:
```
MOVFF   REG1, REG2
```
Before Instruction
REG1 = 33h
REG2 = 11h
After Instruction
REG1 = 33h
REG2 = 33h

MOVLB **Move Literal to Low Nibble in BSR**

Syntax: MOVLB k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow BSR$

Status Affected: None

Encoding:

0000	0001	kkkk	kkkk

Description: The 8-bit literal 'k' is loaded into the Bank Select Register (BSR). The value of BSR<7:4> always remains '0' regardless of the value of k_7:k_4.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write literal 'k' to BSR

Example:
```
MOVLB   5
```
Before Instruction
BSR Register = 02h
After Instruction
BSR Register = 05h

MOVLW **Move Literal to W**

Syntax: MOVLW k

Operands: $0 \le k \le 255$

Operation: $k \rightarrow W$

Status Affected: None

Encoding:

0000	1110	kkkk	kkkk

Description: The 8-bit literal 'k' is loaded into W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example: MOVLW 5Ah

After Instruction
W = 5Ah

MOVWF **Move W to f**

Syntax: MOVWF f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: $(W) \rightarrow f$

Status Affected: None

Encoding:

0110	111a	ffff	ffff

Description: Move data from W to register 'f'. Location 'f' can be anywhere in the 256-byte bank.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: MOVWF REG, 0

Before Instruction
W = 4Fh
REG = FFh
After Instruction
W = 4Fh
REG = 4Fh

MULLW **Multiply Literal with W**

Syntax: MULLW k

Operands: $0 \le k \le 255$

Operation: (W) x k → PRODH:PRODL

Status Affected: None

Encoding:

0000	1101	kkkk	kkkk

Description: An unsigned multiplication is carried out between the contents of W and the 8-bit literal 'k'. The 16-bit result is placed in the PRODH:PRODL register pair. PRODH contains the high byte.

W is unchanged.

None of the Status flags are affected.

Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write registers PRODH: PRODL

Example: MULLW 0C4h

Before Instruction
W = E2h
PRODH = ?
PRODL = ?
After Instruction
W = E2h
PRODH = ADh
PRODL = 08h

MULWF **Multiply W with f**

Syntax: MULWF f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: (W) x (f) → PRODH:PRODL

Status Affected: None

Encoding:

0000	001a	ffff	ffff

Description: An unsigned multiplication is carried out between the contents of W and the register file location 'f'. The 16-bit result is stored in the PRODH:PRODL register pair. PRODH contains the high byte. Both W and 'f' are unchanged.

None of the Status flags are affected.

Note that neither Overflow nor Carry is possible in this operation. A Zero result is possible but not detected.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write registers PRODH: PRODL

Example: MULWF REG, 1

Before Instruction
W = C4h
REG = B5h
PRODH = ?
PRODL = ?
After Instruction
W = C4h
REG = B5h
PRODH = 8Ah
PRODL = 94h

NEGF **Negate f**

Syntax: NEGF f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: $(\bar{f}) + 1 \rightarrow f$

Status Affected: N, OV, C, DC, Z

Encoding:

0110	110a	ffff	ffff

Description: Location ‘f’ is negated using two’s complement. The result is placed in the data memory location ‘f’.

If ‘a’ is ‘0’, the Access Bank is selected. If ‘a’ is ‘1’, the BSR is used to select the GPR bank.

If ‘a’ is ‘0’ and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 “Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode”** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register ‘f’	Process Data	Write register ‘f’

Example:
```
NEGF    REG, 1
```
Before Instruction
REG = 0011 1010 [3Ah]
After Instruction
REG = 1100 0110 [C6h]

NOP **No Operation**

Syntax: NOP

Operands: None

Operation: No operation

Status Affected: None

Encoding:

0000 1111	0000 xxxx	0000 xxxx	0000 xxxx

Description: No operation.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation

Example:

None.

POP **Pop Top of Return Stack**

Syntax: POP

Operands: None

Operation: (TOS) → bit bucket

Status Affected: None

Encoding:

0000	0000	0000	0110

Description: The TOS value is pulled off the return stack and is discarded. The TOS value then becomes the previous value that was pushed onto the return stack. This instruction is provided to enable the user to properly manage the return stack to incorporate a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	POP TOS value	No operation

Example:
```
POP
GOTO NEW
```
Before Instruction
TOS = 0031A2h
Stack (1 level down) = 014332h

After Instruction
TOS = 014332h
PC = NEW

PUSH **Push Top of Return Stack**

Syntax: PUSH

Operands: None

Operation: (PC + 2) → TOS

Status Affected: None

Encoding:

0000	0000	0000	0101

Description: The PC + 2 is pushed onto the top of the return stack. The previous TOS value is pushed down on the stack. This instruction allows implementing a software stack by modifying TOS and then pushing it onto the return stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	PUSH PC + 2 onto return stack	No operation	No operation

Example:
```
PUSH
```
Before Instruction
TOS = 345Ah
PC = 0124h

After Instruction
PC = 0126h
TOS = 0126h
Stack (1 level down) = 345Ah

RCALL **Relative Call**

Syntax: RCALL n

Operands: -1024 ≤ n ≤ 1023

Operation: (PC) + 2 → TOS,
(PC) + 2 + 2n → PC

Status Affected: None

Encoding:

1101	1nnn	nnnn	nnnn

Description: Subroutine call with a jump up to 1K from the current location. First, return address (PC + 2) is pushed onto the stack. Then, add the 2's complement number '2n' to the PC. Since the PC will have incremented to fetch the next instruction, the new address will be PC + 2 + 2n. This instruction is a 2-cycle instruction.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'n' PUSH PC to stack	Process Data	Write to PC
No operation	No operation	No operation	No operation

Example: HERE RCALL Jump

Before Instruction
PC = Address (HERE)
After Instruction
PC = Address (Jump)
TOS = Address (HERE + 2)

RESET **Reset**

Syntax: RESET

Operands: None

Operation: Reset all registers and flags that are affected by a $\overline{\text{MCLR}}$ Reset.

Status Affected: All

Encoding:

0000	0000	1111	1111

Description: This instruction provides a way to execute a $\overline{\text{MCLR}}$ Reset in software.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Start reset	No operation	No operation

Example: RESET

After Instruction
Registers = Reset Value
Flags* = Reset Value

RETFIE **Return from Interrupt**

Syntax: RETFIE {s}

Operands: s ∈ [0,1]

Operation: (TOS) → PC,
1 → GIE/GIEH or PEIE/GIEL;
if s = 1,
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: GIE/GIEH, PEIE/GIEL.

Encoding:

0000	0000	0001	000s

Description: Return from interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low-priority Global Interrupt Enable bit. If 's' = 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers W, STATUS and BSR. If 's' = 0, no update of these registers occurs.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	POP PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example: RETFIE 1

After Interrupt
PC = TOS
W = WS
BSR = BSRS
STATUS = STATUSS
GIE/GIEH, PEIE/GIEL = 1

RETLW **Return Literal to W**

Syntax: RETLW k

Operands: 0 ≤ k ≤ 255

Operation: k → W,
(TOS) → PC,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	1100	kkkk	kkkk

Description: W is loaded with the 8-bit literal 'k'. The Program Counter is loaded from the top of the stack (the return address). The high address latch (PCLATH) remains unchanged.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	POP PC from stack, write to W
No operation	No operation	No operation	No operation

Example:

```
    CALL TABLE ; W contains table
               ; offset value
               ; W now has
               ; table value
    :
TABLE
    ADDWF PCL  ; W = offset
    RETLW k0   ; Begin table
    RETLW k1   ;
    :
    :
    RETLW kn   ; End of table
```

Before Instruction
W = 07h
After Instruction
W = value of kn

RETURN **Return from Subroutine**

Syntax: RETURN {s}

Operands: s ∈ [0,1]

Operation: (TOS) → PC;
if s = 1,
(WS) → W,
(STATUSS) → STATUS,
(BSRS) → BSR,
PCLATU, PCLATH are unchanged

Status Affected: None

Encoding:

0000	0000	0001	001s

Description: Return from subroutine. The stack is popped and the top of the stack (TOS) is loaded into the Program Counter. If 's'= 1, the contents of the shadow registers WS, STATUSS and BSRS are loaded into their corresponding registers W, STATUS and BSR. If 's' = 0, no update of these registers occurs.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	POP PC from stack
No operation	No operation	No operation	No operation

Example: RETURN

After Instruction:
PC = TOS

RLCF **Rotate Left f through Carry**

Syntax: RLCF f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (f<n>) → dest<n + 1>,
(f<7>) → C,
(C) → dest<0>

Status Affected: C, N, Z

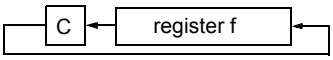
Encoding:

0011	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: RLCF REG, 0, 0

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 1100 1100
C = 1

RLNCF **Rotate Left f (No Carry)**

Syntax: RLNCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<n>) → dest<n + 1>,
(f<7>) → dest<0>

Status Affected: N, Z

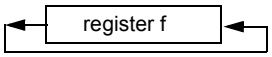
Encoding:

0100	01da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the left. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: RLNCF REG, 1, 0

Before Instruction
REG = 1010 1011
After Instruction
REG = 0101 0111

RRCF **Rotate Right f through Carry**

Syntax: RRCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<n>) → dest<n – 1>,
(f<0>) → C,
(C) → dest<7>

Status Affected: C, N, Z

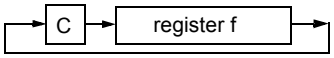
Encoding:

0011	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right through the Carry flag. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: RRCF REG, 0, 0

Before Instruction
REG = 1110 0110
C = 0
After Instruction
REG = 1110 0110
W = 0111 0011
C = 0

RRNCF **Rotate Right f (No Carry)**

Syntax: RRNCF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<n>) → dest<n – 1>,
(f<0>) → dest<7>

Status Affected: N, Z

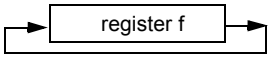
Encoding:

0100	00da	ffff	ffff

Description: The contents of register 'f' are rotated one bit to the right. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed back in register 'f'.

If 'a' is '0', the Access Bank will be selected, overriding the BSR value. If 'a' is '1', then the bank will be selected as per the BSR value.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.



Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: `RRNCF REG, 1, 0`

Before Instruction
REG = 1101 0111
After Instruction
REG = 1110 1011

Example 2: `RRNCF REG, 0, 0`

Before Instruction
W = ?
REG = 1101 0111
After Instruction
W = 1110 1011
REG = 1101 0111

SETF **Set f**

Syntax: SETF f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: FFh → f

Status Affected: None

Encoding:

0110	100a	ffff	ffff

Description: The contents of the specified register are set to FFh.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write register 'f'

Example: `SETF REG,1`

Before Instruction
REG = 5Ah
After Instruction
REG = FFh

SLEEP **Enter Sleep Mode**

Syntax: SLEEP

Operands: None

Operation: 00h → WDT,
0 → WDT postscaler,
1 → $\overline{TO}$,
0 → $\overline{PD}$

Status Affected: $\overline{TO}$, $\overline{PD}$

Encoding:

0000	0000	0000	0011

Description: The Power-Down Status bit ($\overline{PD}$) is cleared. The Time-out Status bit ($\overline{TO}$) is set. The Watchdog Timer and its postscaler are cleared.

The processor is put into Sleep mode with the oscillator stopped.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	Process Data	Go to Sleep

Example: SLEEP

Before Instruction
$\overline{TO}$ = ?
$\overline{PD}$ = ?
After Instruction
$\overline{TO}$ = 1 †
$\overline{PD}$ = 0

† If WDT causes wake-up, this bit is cleared.

SUBFWB **Subtract f from W with Borrow**

Syntax: SUBFWB f {,d {,a}}

Operands: 0 ≤ f ≤ 255
d ∈ [0,1]
a ∈ [0,1]

Operation: (W) – (f) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	01da	ffff	ffff

Description: Subtract register 'f' and Carry flag (borrow) from W (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1: SUBFWB REG, 1, 0

Before Instruction
REG = 3
W = 2
C = 1
After Instruction
REG = FF
W = 2
C = 0
Z = 0
N = 1 ; result is negative

Example 2: SUBFWB REG, 0, 0

Before Instruction
REG = 2
W = 5
C = 1
After Instruction
REG = 2
W = 3
C = 1
Z = 0
N = 0 ; result is positive

Example 3: SUBFWB REG, 1, 0

Before Instruction
REG = 1
W = 2
C = 0
After Instruction
REG = 0
W = 2
C = 1
Z = 1 ; result is zero
N = 0

SUBLW **Subtract W from Literal**

Syntax: SUBLW k

Operands: $0 \le k \le 255$

Operation: $k - (W) \rightarrow W$

Status Affected: N, OV, C, DC, Z

Encoding:

0000	1000	kkkk	kkkk

Description: W is subtracted from the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example 1:
```
SUBLW   02h
Before Instruction
    W   =   01h
    C   =   ?
After Instruction
    W   =   01h
    C   =   1     ; result is positive
    Z   =   0
    N   =   0
```

Example 2:
```
SUBLW   02h
Before Instruction
    W   =   02h
    C   =   ?
After Instruction
    W   =   00h
    C   =   1     ; result is zero
    Z   =   1
    N   =   0
```

Example 3:
```
SUBLW   02h
Before Instruction
    W   =   03h
    C   =   ?
After Instruction
    W   =   FFh   ; (2's complement)
    C   =   0     ; result is negative
    Z   =   0
    N   =   1
```

SUBWF **Subtract W from f**

Syntax: SUBWF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: $(f) - (W) \rightarrow$ dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	11da	ffff	ffff

Description: Subtract W from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1:
```
SUBWF   REG, 1, 0
Before Instruction
    REG   =   3
    W     =   2
    C     =   ?
After Instruction
    REG   =   1
    W     =   2
    C     =   1     ; result is positive
    Z     =   0
    N     =   0
```

Example 2:
```
SUBWF   REG, 0, 0
Before Instruction
    REG   =   2
    W     =   2
    C     =   ?
After Instruction
    REG   =   2
    W     =   0
    C     =   1     ; result is zero
    Z     =   1
    N     =   0
```

Example 3:
```
SUBWF   REG, 1, 0
Before Instruction
    REG   =   1
    W     =   2
    C     =   ?
After Instruction
    REG   =   FFh   ;(2's complement)
    W     =   2
    C     =   0     ; result is negative
    Z     =   0
    N     =   1
```

SUBWFB **Subtract W from f with Borrow**

Syntax: SUBWFB f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f) – (W) – ($\overline{C}$) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0101	10da	ffff	ffff

Description: Subtract W and the Carry flag (borrow) from register 'f' (2's complement method). If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example 1:
```
SUBWFB  REG, 1, 0
Before Instruction
    REG  =  19h   (0001 1001)
    W    =  0Dh   (0000 1101)
    C    =  1
After Instruction
    REG  =  0Ch   (0000 1011)
    W    =  0Dh   (0000 1101)
    C    =  1
    Z    =  0
    N    =  0     ; result is positive
```

Example 2:
```
SUBWFB  REG, 0, 0
Before Instruction
    REG  =  1Bh   (0001 1011)
    W    =  1Ah   (0001 1010)
    C    =  0
After Instruction
    REG  =  1Bh   (0001 1011)
    W    =  00h
    C    =  1
    Z    =  1     ; result is zero
    N    =  0
```

Example 3:
```
SUBWFB  REG, 1, 0
Before Instruction
    REG  =  03h   (0000 0011)
    W    =  0Eh   (0000 1101)
    C    =  1
After Instruction
    REG  =  F5h   (1111 0100)
                  ; [2's comp]
    W    =  0Eh   (0000 1101)
    C    =  0
    Z    =  0
    N    =  1     ; result is negative
```

SWAPF **Swap f**

Syntax: SWAPF f {,d {,a}}

Operands: $0 \le f \le 255$
$d \in [0,1]$
$a \in [0,1]$

Operation: (f<3:0>) → dest<7:4>,
(f<7:4>) → dest<3:0>

Status Affected: None

Encoding:

0011	10da	ffff	ffff

Description: The upper and lower nibbles of register 'f' are exchanged. If 'd' is '0', the result is placed in W. If 'd' is '1', the result is placed in register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example:
```
SWAPF   REG, 1, 0
Before Instruction
    REG  =  53h
After Instruction
    REG  =  35h
```

TBLRD **Table Read**

Syntax: TBLRD (*; *+; *-; +*)

Operands: None

Operation: if TBLRD *,
(Prog Mem (TBLPTR)) → TABLAT;
TBLPTR – No Change
if TBLRD *+,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) + 1 → TBLPTR
if TBLRD *-,
(Prog Mem (TBLPTR)) → TABLAT;
(TBLPTR) – 1 → TBLPTR
if TBLRD +*,
(TBLPTR) + 1 → TBLPTR;
(Prog Mem (TBLPTR)) → TABLAT

Status Affected: None

Encoding:

0000	0000	0000	10nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction is used to read the contents of Program Memory (P.M.). To address the program memory, a pointer called Table Pointer (TBLPTR) is used.

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range.

TBLPTR<0> = 0: Least Significant Byte of Program Memory Word

TBLPTR<0> = 1: Most Significant Byte of Program Memory Word

The TBLRD instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read Program Memory)	No operation	No operation (Write TABLAT)

TBLRD **Table Read (Continued)**

Example 1:
```
TBLRD   *+ ;
```
Before Instruction
TABLAT = 55h
TBLPTR = 00A356h
MEMORY(00A356h) = 34h
After Instruction
TABLAT = 34h
TBLPTR = 00A357h

Example 2:
```
TBLRD   +* ;
```
Before Instruction
TABLAT = AAh
TBLPTR = 01A357h
MEMORY(01A357h) = 12h
MEMORY(01A358h) = 34h
After Instruction
TABLAT = 34h
TBLPTR = 01A358h

TBLWT **Table Write**

Syntax: TBLWT (*; *+; *-; +*)

Operands: None

Operation: if TBLWT*,
(TABLAT) → Holding Register;
TBLPTR – No Change
if TBLWT*+,
(TABLAT) → Holding Register;
(TBLPTR) + 1 → TBLPTR
if TBLWT*-,
(TABLAT) → Holding Register;
(TBLPTR) – 1 → TBLPTR
if TBLWT+*,
(TBLPTR) + 1 → TBLPTR;
(TABLAT) → Holding Register

Status Affected: None

Encoding:

0000	0000	0000	11nn nn=0 * =1 *+ =2 *- =3 +*

Description: This instruction uses the 3 LSBs of TBLPTR to determine which of the 8 holding registers the TABLAT is written to. The holding registers are used to program the contents of Program Memory (P.M.). (Refer to **Section 6.0 "Memory Organization"** for additional details on programming Flash memory.)

The TBLPTR (a 21-bit pointer) points to each byte in the program memory. TBLPTR has a 2-Mbyte address range. The LSb of the TBLPTR selects which byte of the program memory location to access.

TBLPTR[0] = 0: Least Significant Byte of Program Memory Word

TBLPTR[0] = 1: Most Significant Byte of Program Memory Word

The TBLWT instruction can modify the value of TBLPTR as follows:

- no change
- post-increment
- post-decrement
- pre-increment

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	No operation
No operation	No operation (Read TABLAT)	No operation	No operation (Write to Holding Register)

TBLWT **Table Write (Continued)**

Example 1: TBLWT *+;

Before Instruction
TABLAT = 55h
TBLPTR = 00A356h
HOLDING REGISTER (00A356h) = FFh
After Instructions (table write completion)
TABLAT = 55h
TBLPTR = 00A357h
HOLDING REGISTER (00A356h) = 55h

Example 2: TBLWT +*;

Before Instruction
TABLAT = 34h
TBLPTR = 01389Ah
HOLDING REGISTER (01389Ah) = FFh
HOLDING REGISTER (01389Bh) = FFh
After Instruction (table write completion)
TABLAT = 34h
TBLPTR = 01389Bh
HOLDING REGISTER (01389Ah) = FFh
HOLDING REGISTER (01389Bh) = 34h

TSTFSZ **Test f, Skip if 0**

Syntax: TSTFSZ f {,a}

Operands: $0 \le f \le 255$
$a \in [0,1]$

Operation: skip if f = 0

Status Affected: None

Encoding:

0110	011a	ffff	ffff

Description: If 'f' = 0, the next instruction fetched during the current instruction execution is discarded and a NOP is executed, making this a 2-cycle instruction.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever $f \le 95$ (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1(2)
Note: 3 cycles if skip and followed by a 2-word instruction.

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	No operation

If skip:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation

If skip and followed by 2-word instruction:

Q1	Q2	Q3	Q4
No operation	No operation	No operation	No operation
No operation	No operation	No operation	No operation

Example:
```
HERE    TSTFSZ  CNT, 1
NZERO   :
ZERO    :
```
Before Instruction
PC = Address (HERE)
After Instruction
If CNT = 00h,
PC = Address (ZERO)
If CNT ≠ 00h,
PC = Address (NZERO)

XORLW **Exclusive OR Literal with W**

Syntax: XORLW k

Operands: $0 \le k \le 255$

Operation: (W) .XOR. k → W

Status Affected: N, Z

Encoding:

0000	1010	kkkk	kkkk

Description: The contents of W are XORed with the 8-bit literal 'k'. The result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example:
```
XORLW   0AFh
```
Before Instruction
W = B5h
After Instruction
W = 1Ah

XORWF	Exclusive OR W with f
Syntax:	XORWF f {,d {,a}}
Operands:	0 ≤ f ≤ 255 d ∈ [0,1] a ∈ [0,1]
Operation:	(W) .XOR. (f) → dest
Status Affected:	N, Z

Encoding:

0001	10da	ffff	ffff

Description: Exclusive OR the contents of W with register 'f'. If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in the register 'f'.

If 'a' is '0', the Access Bank is selected. If 'a' is '1', the BSR is used to select the GPR bank.

If 'a' is '0' and the extended instruction set is enabled, this instruction operates in Indexed Literal Offset Addressing mode whenever f ≤ 95 (5Fh). See **Section 29.2.3 "Byte-Oriented and Bit-Oriented Instructions in Indexed Literal Offset Mode"** for details.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example:

```
XORWF   REG, 1, 0
```

Before Instruction
REG = AFh
W = B5h
After Instruction
REG = 1Ah
W = B5h

29.2 Extended Instruction Set

In addition to the standard 75 instructions of the PIC18 instruction set, the PIC18FXXJ94 of devices also provides an optional extension to the core CPU functionality. The added features include eight additional instructions that augment Indirect and Indexed Addressing operations and the implementation of Indexed Literal Offset Addressing for many of the standard PIC18 instructions.

The additional features of the extended instruction set are enabled by default on unprogrammed devices. Users must properly set or clear the XINST Configuration bit during programming to enable or disable these features.

The instructions in the extended set can all be classified as literal operations, which either manipulate the File Select Registers, or use them for Indexed Addressing. Two of the instructions, ADDFSR and SUBFSR, each have an additional special instantiation for using FSR2. These versions (ADDULNK and SUBULNK) allow for automatic return after execution.

The extended instructions are specifically implemented to optimize re-entrant program code (that is, code that is recursive or that uses a software stack) written in high-level languages, particularly C. Among other things, they allow users working in high-level languages to perform certain operations on data structures more efficiently. These include:

- Dynamic allocation and deallocation of software stack space when entering and leaving subroutines
- Function Pointer invocation
- Software Stack Pointer manipulation
- Manipulation of variables located in a software stack

A summary of the instructions in the extended instruction set is provided in Table 29-3. Detailed descriptions are provided in **Section 29.2.2 "Extended Instruction Set"**. The opcode field descriptions in Table 29-1 (page 566) apply to both the standard and extended PIC18 instruction sets.

Note: The instruction set extension and the Indexed Literal Offset Addressing mode were designed for optimizing applications written in C; the user may likely never use these instructions directly in assembler. The syntax for these commands is provided as a reference for users who may be reviewing code that has been generated by a compiler.

29.2.1 EXTENDED INSTRUCTION SYNTAX

Most of the extended instructions use indexed arguments, using one of the File Select Registers and some offset to specify a source or destination register. When an argument for an instruction serves as part of Indexed Addressing, it is enclosed in square brackets ("[ ]"). This is done to indicate that the argument is used as an index or offset. The MPASM™ Assembler will flag an error if it determines that an index or offset value is not bracketed.

When the extended instruction set is enabled, brackets are also used to indicate index arguments in byte-oriented and bit-oriented instructions. This is in addition to other changes in their syntax. For more details, see **Section 29.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**.

Note: In the past, square brackets have been used to denote optional arguments in the PIC18 and earlier instruction sets. In this text and going forward, optional arguments are denoted by braces ("{ }").

TABLE 29-3: EXTENSIONS TO THE PIC18 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word				Status Affected
				MSb			LSb	
ADDFSR	f, k	Add Literal to FSR	1	1110	1000	ffkk	kkkk	None
ADDULNK	k	Add Literal to FSR2 and Return	2	1110	1000	11kk	kkkk	None
CALLW		Call Subroutine using WREG	2	0000	0000	0001	0100	None
MOVSF	z_s, f_d	Move z_s (source) to 1st word	2	1110	1011	0zzz	zzzz	None
		f_d (destination) 2nd word		1111	ffff	ffff	ffff	
MOVSS	z_s, z_d	Move z_s (source) to 1st word	2	1110	1011	1zzz	zzzz	None
		z_d (destination) 2nd word		1111	xxxx	xzzz	zzzz	
PUSHL	k	Store Literal at FSR2, Decrement FSR2	1	1110	1010	kkkk	kkkk	None
SUBFSR	f, k	Subtract Literal from FSR	1	1110	1001	ffkk	kkkk	None
SUBULNK	k	Subtract Literal from FSR2 and return	2	1110	1001	11kk	kkkk	None

29.2.2 EXTENDED INSTRUCTION SET

ADDFSR **Add Literal to FSR**

Syntax: ADDFSR f, k

Operands: $0 \le k \le 63$
$f \in [0, 1, 2]$

Operation: FSR(f) + k → FSR(f)

Status Affected: None

Encoding:	1110	1000	ffkk	kkkk

Description: The 6-bit literal 'k' is added to the contents of the FSR specified by 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to FSR

Example: ADDFSR 2, 23h

Before Instruction
FSR2 = 03FFh
After Instruction
FSR2 = 0422h

ADDULNK **Add Literal to FSR2 and Return**

Syntax: ADDULNK k

Operands: $0 \le k \le 63$

Operation: FSR2 + k → FSR2,
(TOS) → PC

Status Affected: None

Encoding:	1110	1000	11kk	kkkk

Description: The 6-bit literal 'k' is added to the contents of FSR2. A RETURN is then executed by loading the PC with the TOS.

The instruction takes two cycles to execute; a NOP is performed during the second cycle.

This may be thought of as a special case of the ADDFSR instruction, where f = 3 (binary '11'); it operates only on FSR2.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to FSR
No Operation	No Operation	No Operation	No Operation

Example: ADDULNK 23h

Before Instruction
FSR2 = 03FFh
PC = 0100h
After Instruction
FSR2 = 0422h
PC = (TOS)

Note: All PIC18 instructions may take an optional label argument preceding the instruction mnemonic for use in symbolic addressing. If a label is used, the instruction format then becomes: {label} instruction argument(s).

CALLW **Subroutine Call Using WREG**

Syntax: CALLW

Operands: None

Operation: (PC + 2) → TOS,
(W) → PCL,
(PCLATH) → PCH,
(PCLATU) → PCU

Status Affected: None

Encoding:

0000	0000	0001	0100

Description: First, the return address (PC + 2) is pushed onto the return stack. Next, the contents of W are written to PCL; the existing value is discarded. Then, the contents of PCLATH and PCLATU are latched into PCH and PCU, respectively. The second cycle is executed as a NOP instruction while the new next instruction is fetched.

Unlike CALL, there is no option to update W, STATUS or BSR.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read WREG	Push PC to stack	No operation
No operation	No operation	No operation	No operation

Example: HERE CALLW

Before Instruction
PC = address (HERE)
PCLATH = 10h
PCLATU = 00h
W = 06h
After Instruction
PC = 001006h
TOS = address (HERE + 2)
PCLATH = 10h
PCLATU = 00h
W = 06h

MOVSF **Move Indexed to f**

Syntax: MOVSF [z_s], f_d

Operands: $0 \le z_s \le 127$
$0 \le f_d \le 4095$

Operation: ((FSR2) + z_s) → f_d

Status Affected: None

Encoding:

1st word (source)	1110	1011	0zzz	$zzzz_s$
2nd word (destin.)	1111	ffff	ffff	$ffff_d$

Description: The contents of the source register are moved to destination register 'f_d'. The actual address of the source register is determined by adding the 7-bit literal offset 'z_s', in the first word, to the value of FSR2. The address of the destination register is specified by the 12-bit literal 'f_d' in the second word. Both addresses can be anywhere in the 4096-byte data space (000h to FFFh).

The MOVSF instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an Indirect Addressing register, the value returned will be 00h.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	No operation No dummy read	No operation	Write register 'f' (dest)

Example: MOVSF [05h], REG2

Before Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h
REG2 = 33h

MOVSS **Move Indexed to Indexed**

Syntax: MOVSS [z_s], [z_d]

Operands: $0 \le z_s \le 127$
$0 \le z_d \le 127$

Operation: ((FSR2) + z_s) → ((FSR2) + z_d)

Status Affected: None

Encoding:

1st word (source)	1110	1011	1zzz	$zzzz_s$
2nd word (dest.)	1111	xxxx	xzzz	$zzzz_d$

Description: The contents of the source register are moved to the destination register. The addresses of the source and destination registers are determined by adding the 7-bit literal offsets, 'z_s' or 'z_d', respectively, to the value of FSR2. Both registers can be located anywhere in the 4096-byte data memory space (000h to FFFh).

The MOVSS instruction cannot use the PCL, TOSU, TOSH or TOSL as the destination register.

If the resultant source address points to an Indirect Addressing register, the value returned will be 00h. If the resultant destination address points to an Indirect Addressing register, the instruction will execute as a NOP.

Words: 2

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Determine source addr	Determine source addr	Read source reg
Decode	Determine dest addr	Determine dest addr	Write to dest reg

Example:
```
MOVSS [05h], [06h]
```
Before Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 11h
After Instruction
FSR2 = 80h
Contents of 85h = 33h
Contents of 86h = 33h

PUSHL **Store Literal at FSR2, Decrement FSR2**

Syntax: PUSHL k

Operands: $0 \le k \le 255$

Operation: k → (FSR2),
FSR2 – 1 → FSR2

Status Affected: None

Encoding:

1111	1010	kkkk	kkkk

Description: The 8-bit literal 'k' is written to the data memory address specified by FSR2. FSR2 is decremented by 1 after the operation.

This instruction allows users to push values onto a software stack.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process data	Write to destination

Example:
```
PUSHL 08h
```
Before Instruction
FSR2H:FSR2L = 01ECh
Memory (01ECh) = 00h

After Instruction
FSR2H:FSR2L = 01EBh
Memory (01ECh) = 08h

SUBFSR **Subtract Literal from FSR**

Syntax: SUBFSR f, k

Operands: $0 \le k \le 63$

$f \in [0, 1, 2]$

Operation: FSRf – k → FSRf

Status Affected: None

Encoding:

1110	1001	ffkk	kkkk

Description: The 6-bit literal 'k' is subtracted from the contents of the FSR specified by 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: SUBFSR 2, 23h

Before Instruction
FSR2 = 03FFh

After Instruction
FSR2 = 03DCh

SUBULNK **Subtract Literal from FSR2 and Return**

Syntax: SUBULNK k

Operands: $0 \le k \le 63$

Operation: FSR2 – k → FSR2,
(TOS) → PC

Status Affected: None

Encoding:

1110	1001	11kk	kkkk

Description: The 6-bit literal 'k' is subtracted from the contents of the FSR2. A RETURN is then executed by loading the PC with the TOS.

The instruction takes two cycles to execute; a NOP is performed during the second cycle.

This may be thought of as a special case of the SUBFSR instruction, where f = 3 (binary '11'); it operates only on FSR2.

Words: 1

Cycles: 2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination
No Operation	No Operation	No Operation	No Operation

Example: SUBULNK 23h

Before Instruction
FSR2 = 03FFh
PC = 0100h

After Instruction
FSR2 = 03DCh
PC = (TOS)

29.2.3 BYTE-ORIENTED AND BIT-ORIENTED INSTRUCTIONS IN INDEXED LITERAL OFFSET MODE

Note: Enabling the PIC18 instruction set extension may cause legacy applications to behave erratically or fail entirely.

In addition to eight new commands in the extended set, enabling the extended instruction set also enables Indexed Literal Offset Addressing (**Section 6.6.1 "Indexed Addressing with Literal Offset"**). This has a significant impact on the way that many commands of the standard PIC18 instruction set are interpreted.

When the extended set is disabled, addresses embedded in opcodes are treated as literal memory locations: either as a location in the Access Bank (a = 0) or in a GPR bank designated by the BSR (a = 1). When the extended instruction set is enabled and a = 0, however, a file register argument of 5Fh or less is interpreted as an offset from the pointer value in FSR2 and not as a literal address. For practical purposes, this means that all instructions that use the Access RAM bit as an argument – that is, all byte-oriented and bit-oriented instructions, or almost half of the core PIC18 instructions – may behave differently when the extended instruction set is enabled.

When the content of FSR2 is 00h, the boundaries of the Access RAM are essentially remapped to their original values. This may be useful in creating backward compatible code. If this technique is used, it may be necessary to save the value of FSR2 and restore it when moving back and forth between C and assembly routines in order to preserve the Stack Pointer. Users must also keep in mind the syntax requirements of the extended instruction set (see **Section 29.2.3.1 "Extended Instruction Syntax with Standard PIC18 Commands"**).

Although the Indexed Literal Offset mode can be very useful for dynamic stack and pointer manipulation, it can also be very annoying if a simple arithmetic operation is carried out on the wrong register. Users who are accustomed to the PIC18 programming must keep in mind, that when the extended instruction set is enabled, register addresses of 5Fh or less are used for Indexed Literal Offset Addressing.

Representative examples of typical byte-oriented and bit-oriented instructions in the Indexed Literal Offset mode are provided on the following page to show how execution is affected. The operand conditions shown in the examples are applicable to all instructions of these types.

29.2.3.1 Extended Instruction Syntax with Standard PIC18 Commands

When the extended instruction set is enabled, the file register argument 'f' in the standard byte-oriented and bit-oriented commands is replaced with the literal offset value 'k'. As already noted, this occurs only when 'f' is less than or equal to 5Fh. When an offset value is used, it must be indicated by square brackets ("[ ]"). As with the extended instructions, the use of brackets indicates to the compiler that the value is to be interpreted as an index or an offset. Omitting the brackets, or using a value greater than 5Fh within the brackets, will generate an error in the MPASM™ Assembler.

If the index argument is properly bracketed for Indexed Literal Offset Addressing, the Access RAM argument is never specified; it will automatically be assumed to be '0'. This is in contrast to standard operation (extended instruction set disabled), when 'a' is set on the basis of the target address. Declaring the Access RAM bit in this mode will also generate an error in the MPASM Assembler.

The destination argument, 'd', functions as before.

In the latest versions of the MPASM Assembler, language support for the extended instruction set must be explicitly invoked. This is done with either the command-line option, /y, or the PE directive in the source listing.

29.2.4 CONSIDERATIONS WHEN ENABLING THE EXTENDED INSTRUCTION SET

It is important to note that the extensions to the instruction set may not be beneficial to all users. In particular, users who are not writing code that uses a software stack may not benefit from using the extensions to the instruction set.

Additionally, the Indexed Literal Offset Addressing mode may create issues with legacy applications written to the PIC18 assembler. This is because instructions in the legacy code may attempt to address registers in the Access Bank below 5Fh. Since these addresses are interpreted as literal offsets to FSR2 when the instruction set extension is enabled, the application may read or write to the wrong data addresses.

When porting an application to the PIC18FXXJ94, it is very important to consider the type of code. A large, re-entrant application that is written in C and would benefit from efficient compilation will do well when using the instruction set extensions. Legacy applications that heavily use the Access Bank will most likely not benefit from using the extended instruction set.

ADDWF **ADD W to Indexed (Indexed Literal Offset mode)**

Syntax: ADDWF [k] {,d}

Operands: $0 \le k \le 95$
$d \in [0,1]$

Operation: (W) + ((FSR2) + k) → dest

Status Affected: N, OV, C, DC, Z

Encoding:

0010	01d0	kkkk	kkkk

Description: The contents of W are added to the contents of the register indicated by FSR2, offset by the value 'k'.

If 'd' is '0', the result is stored in W. If 'd' is '1', the result is stored back in register 'f'.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process Data	Write to destination

Example: ADDWF [OFST],0

Before Instruction
W = 17h
OFST = 2Ch
FSR2 = 0A00h
Contents of 0A2Ch = 20h
After Instruction
W = 37h
Contents of 0A2Ch = 20h

BSF **Bit Set Indexed (Indexed Literal Offset mode)**

Syntax: BSF [k], b

Operands: $0 \le f \le 95$
$0 \le b \le 7$

Operation: 1 → ((FSR2) + k)<b>

Status Affected: None

Encoding:

1000	bbb0	kkkk	kkkk

Description: Bit 'b' of the register indicated by FSR2, offset by the value 'k', is set.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read register 'f'	Process Data	Write to destination

Example: BSF [FLAG_OFST], 7

Before Instruction
FLAG_OFST = 0Ah
FSR2 = 0A00h
Contents of 0A0Ah = 55h
After Instruction
Contents of 0A0Ah = D5h

SETF **Set Indexed (Indexed Literal Offset mode)**

Syntax: SETF [k]

Operands: $0 \le k \le 95$

Operation: FFh → ((FSR2) + k)

Status Affected: None

Encoding:

0110	1000	kkkk	kkkk

Description: The contents of the register indicated by FSR2, offset by 'k', are set to FFh.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read 'k'	Process Data	Write register

Example: SETF [OFST]

Before Instruction
OFST = 2Ch
FSR2 = 0A00h
Contents of 0A2Ch = 00h
After Instruction
Contents of 0A2Ch = FFh

29.2.5 SPECIAL CONSIDERATIONS WITH MICROCHIP MPLAB® IDE TOOLS

The latest versions of Microchip's software tools have been designed to fully support the extended instruction set for the PIC18F97J94 Family. This includes the MPLAB C18 C Compiler, MPASM assembly language and MPLAB Integrated Development Environment (IDE).

When selecting a target device for software development, MPLAB IDE will automatically set default Configuration bits for that device. The default setting for the XINST Configuration bit is '1', enabling the extended instruction set and Indexed Literal Offset Addressing. For proper execution of applications developed to take advantage of the extended instruction set, XINST must be set during programming.

To develop software for the extended instruction set, the user must enable support for the instructions and the Indexed Addressing mode in their language tool(s). Depending on the environment being used, this may be done in several ways:

- A menu option or dialog box within the environment that allows the user to configure the language tool and its settings for the project
- A command-line option
- A directive in the source code

These options vary between different compilers, assemblers and development environments. Users are encouraged to review the documentation accompanying their development systems for the appropriate information.

30.0 ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings(†)

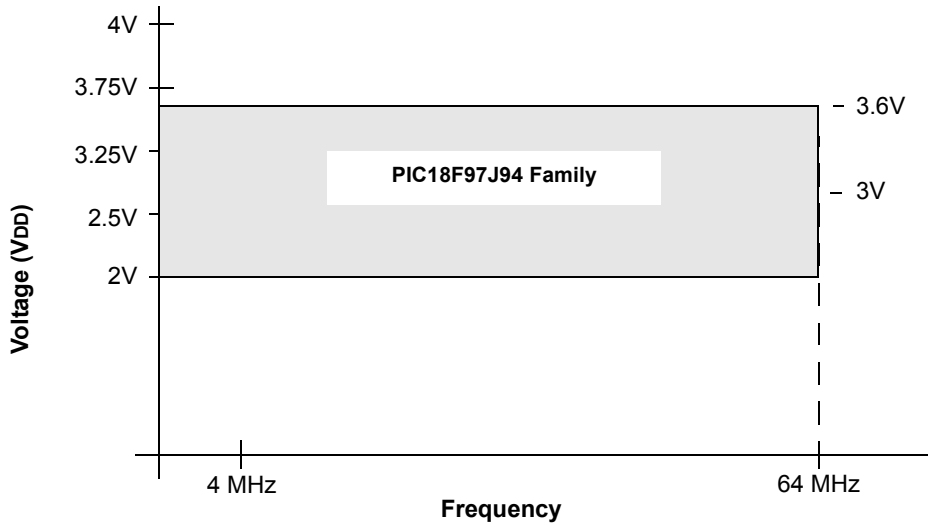
Ambient temperature under bias ..... -40°C to +100°C
Storage temperature ..... -65°C to +150°C
Voltage on $\overline{MCLR}$ with respect to V_{SS} ..... -0.3V to 5.5V
Voltage on any digital only I/O pin with respect to V_{SS} (except V_{DD}) ..... -0.3V to 5.5V
Voltage on any combined digital and analog pin with respect to V_{SS} (except V_{DD} and $\overline{MCLR}$) ..... -0.3V to (V_{DD} + 0.3V)
Voltage on V_{BAT} with respect to V_{SS} ..... -0.3V to 3.66V
Voltage on V_{USB3V3} with respect to V_{SS} ..... (V_{DD} – 0.3V) to +4.0V
Voltage on V_{DD} with respect to V_{SS} ..... -0.3V to 3.66V
Voltage on D+ or D- with respect to V_{SS} – 0W source impedance (**Note 2**) ..... -0.5V to (V_{USB3V3} + 0.5V)
Source impedance ≥ 28W, V_{USB3V3} ≥ 3.0V) ..... -1.0V to +4.6V
Total power dissipation **(Note 1)** 1W
Maximum current out of V_{SS} pin ..... 300 mA
Maximum current into V_{DD} pin ..... 250 mA
Input clamp current, I_{IK} (V_I < 0 or V_I > V_{DD}) ..... ±20 mA
Output clamp current, I_{OK} (V_O < 0 or V_O > V_{DD}) ..... ±20 mA
Maximum output current sunk by any I/O pins ..... 25 mA
Maximum output current sourced by any I/O pins ..... 25 mA
Maximum current sunk by all ports combined ..... 200 mA
Maximum current sourced by all ports combined ..... 200 mA

Note 1: Power dissipation is calculated as follows:
Pdis = V_{DD} x {I_{DD} – ∑ I_{OH}} + ∑ {(V_{DD} – V_{OH}) x I_{OH}} + ∑(V_{OL} x I_{OL})

2: The original "*USB 2.0 Specification*" indicated that USB devices should withstand 24-hour short circuits of D+ or D- to V_{BUS} voltages. This requirement was later removed in an engineering change notice (ECN) supplement to the USB specifications, which supersedes the original specifications. PIC18FXXJ94 family devices will typically be able to survive this short circuit test, but it is recommended to adhere to the absolute maximum specified here to avoid damaging the device.

† **NOTICE:** Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

FIGURE 30-1: VOLTAGE-FREQUENCY GRAPH, REGULATOR DISABLED (INDUSTRIAL)(1,2)



Note 1: When the USB module is enabled, VUSB3V3 and VDD should be connected together and provided 3.0V-3.6V. When the USB module is not enabled, VUSB3V3 and VDD should still be connected together.

2: VCAP (nominal on-chip regulator output voltage) = 1.8V.

TABLE 30-1: DC CHARACTERISTICS: SUPPLY VOLTAGE PIC18FXXJ94 (INDUSTRIAL)

PIC18FXXJ94 (Industrial)			Standard Operating Conditions: 2V to 3.6V (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C				
Param No.	**Symbol**	**Characteristic**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
D001	VDD	**Supply Voltage**	2.0	—	3.6	V	
D001C	AVDD	**Analog Supply Voltage**	VDD – 0.3	—	VDD + 0.3	V	
D001D	AVSS	**Analog Ground Potential**	VSS – 0.3	—	VSS + 0.3	V	
D001E	VUSB3V3	**USB Supply Voltage**	3	3.3	3.6	V	USB module enabled(3)
D002	VDR	**RAM Data Retention Voltage**(1)	1.2	—	—	V	
D003	VPOR	**VDD/VBAT Start Voltage** to Ensure Internal Power-on Reset Signal	—	—	0.7	V	See **Section 5.2 "Power-on Reset (POR)"** for details
D004	SVDD	**VDD/VBAT Rise Rate** to Ensure Internal Power-on Reset Signal	0.05	—	—	V/ms	See **Section 5.2 "Power-on Reset (POR)"** for details
D005	BVDD	**Brown-out Reset Voltage** BORV = 1(2) BORV = 0	 1.8 2.0	 1.88 2.05	 1.95 2.20	 V V	
D006	VVDDBOR		1.4V		2.0	V	
D007	VVBATBOR		1.4V		1.95	V	
D008	VDSBOR				1.8		

Note 1: This is the limit to which VDD can be lowered in Sleep mode, or during a device Reset, without losing RAM data.

2: The device will operate normally until Brown-out Reset occurs, even though VDD may be below VDDMIN.

3: VUSB3V3 should be connected to VDD.

TABLE 30-2: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18FXXJ94 (INDUSTRIAL)

PIC18FXXJ94 Family (Industrial)				**Standard Operating Conditions: 2V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial		
Param No.	**Typ.(1)**	**Max.**	**Units**	**Conditions**		
DC60	3.7	7.0	µA	-40°C	2.0V	Sleep(2)
	3.7	7.0	µA	+25°C		
	5.0	9.0	µA	+60°C		
	9.0	18	µA	+85°C		
	3.7	8.0	µA	-40°C	3.3V	
	3.7	8.0	µA	+25°C		
	5.0	11.0	µA	+60°C		
	10	20	µA	+85°C		
DC61	0.07	0.55	µA	-40°C	2.0V	Retention Sleep or Retention Deep Sleep(3)
	0.09	0.55	µA	+25°C		
	2.0	3.2	µA	+60°C		
	7.0	8.5	µA	+85°C		
	0.10	0.65	µA	-40°C	3.3V	
	0.15	0.65	µA	+25°C		
	2.0	3.5	µA	+60°C		
	7.2	9.0	µA	+85°C		
DC70	0.06	0.5	µA	-40°C	2.0V	Deep Sleep
	0.08	0.5	µA	+25°C		
	0.21	0.8	µA	+60°C		
	0.41	1.5	µA	+85°C		
	0.09	0.6	µA	-40°C	3.3V	
	0.11	0.6	µA	+25°C		
	0.42	1.2	µA	+60°C		
	0.8	4.8	µA	+85°C		
	0.4	3.0	µA	-40°C TO +85°C	0	RTCC with VBAT mode (LPRC or SOSC)(4)

Note 1: Data in the Typical column is at 3.3V, 25°C; typical parameters are for design guidance only and are not tested.

2: Retention regulator is disabled; SRETEN (RCON4<4>= 0), $\overline{\text{RETEN}}$ (CONFIG7L<0>= 1).

3: Retention regulator is enabled; SRETEN (RCON4<4> = 1), $\overline{\text{RETEN}}$ (CONFIG7L<0> = 0).

4: VBAT pin is connected to the battery and RTCC is running with VDD = 0.

TABLE 30-3: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18F97J94 FAMILY (INDUSTRIAL)

Param No.	**Device**	**Typ.**	**Max.**	**Units**	**Conditions**		
	Supply Current (IDD)						
	All Devices	22	55	µA	-40°C to +85°C	VDD = 2.0V	FOSC = 31 kHz, **RC_RUN**
		23	56	µA	-40°C to +85°C	VDD = 3.3V	
		21	54	µA	-40°C to +85°C	VDD = 2.0V	FOSC = 31 kHz, **RC_IDLE**
		22	55	µA	-40°C to +85°C	VDD = 3.3V	

TABLE 30-4: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18F97J94 FAMILY (INDUSTRIAL)

Param No.	Device	Typ.	Max.	Units	Conditions		
	Supply Current (I_{DD})						
	All Devices	22	55	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 32 kHz, **SEC_RUN**
		23	56	µA	-40°C to +85°C	V_{DD} = 3.3V	
		21	54	µA	-40°C to +85°C	V_{DD} =2.0V	F_{OSC} = 32 kHz, **SEC_IDLE**
		22	55	µA	-40°C to +85°C	V_{DD} = 3.3V	

TABLE 30-5: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18F97J94 FAMILY (INDUSTRIAL)

Param No.	Device	Typ.	Max.	Units	Conditions		
	Supply Current (I_{DD})						
	All Devices	325	430	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 1 MHz, **RC_RUN**
		325	430	µA	-40°C to +85°C	V_{DD} = 3.3V	
		540	700	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 4 MHz, **RC_RUN**
		540	700	µA	-40°C to +85°C	V_{DD} = 3.3V	
		820	1000	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 8 MHz, **RC_RUN**
		825	1000	µA	-40°C to +85°C	V_{DD} = 3.3V	
		275	370	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 1 MHz, **RC_IDLE**
		275	370	µA	-40°C to +85°C	V_{DD} = 3.3V	
		345	440	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 4 MHz, **RC_IDLE**
		345	440	µA	-40°C to +85°C	V_{DD} = 3.3V	
		435	620	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 8 MHz, **RC_IDLE**
		435	620	µA	-40°C to +85°C	V_{DD} = 3.3V	

TABLE 30-6: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18F97J94 FAMILY (INDUSTRIAL)

Param No.	Device	Typ.	Max.	Units	Conditions		
	Supply Current (I_{DD})						
	All Devices	100	150	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 1 MHz, **PRI_RUN** mode, EC Oscillator
		105	155	µA	-40°C to +85°C	V_{DD} = 3.3V	
		330	390	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 4 MHz, **PRI_RUN** mode, EC Oscillator
		340	405	µA	-40°C to +85°C	V_{DD} = 3.3V	
		5.0	5.5	mA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 64 MHz, **PRI_RUN** mode, EC Oscillator
		5.0	5.5	mA	-40°C to +85°C	V_{DD} = 3.3V	
		5.7	6.5	mA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 64 MHz, **PRI_RUN** mode, 8 MHz EC Oscillator with 96 MHz or 8X PLL
		5.7	7.0	mA	-40°C to +85°C	V_{DD} = 3.3V	
		52	90	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 1 MHz, **PRI_IDLE** mode, EC Oscillator
		66	95	µA	-40°C to +85°C	V_{DD} = 3.3V	
		135	185	µA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 4 MHz, **PRI_IDLE** mode, EC Oscillator
		145	195	µA	-40°C to +85°C	V_{DD} = 3.3V	
		1.8	2.6	mA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 64 MHz, **PRI_IDLE** mode, EC Oscillator
		2.0	2.8	mA	-40°C to +85°C	V_{DD} = 3.3V	
		2.3	2.9	mA	-40°C to +85°C	V_{DD} = 2.0V	F_{OSC} = 64 MHz, **PRI_IDLE** mode, 8 MHz EC Oscillator with 96 MHz or 8X PLL
		2.4	3.0	mA	-40°C to +85°C	V_{DD} = 3.3V	

TABLE 30-7: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18F97J94 FAMILY (INDUSTRIAL)

Param No.	Device	Typ.(1)	Max.	Units	Conditions		
Module Differential Currents (ΔIWDT, ΔIBOR, ΔIHLVD, ΔIDSBOR, ΔIDSWDT, ΔIOSCB, ΔIADRC, ΔILCD, ΔIUSB)							
D020 (ΔIWDT)	**Watchdog Timer**	0.4	1	µA	-40°C to +85°C	VDD = 2.0V	
		0.4	1	µA	-40°C to +85°C	VDD = 3.3V	
D021 (ΔIBOR)	**Brown-out Reset**	4	8	µA	-40°C to +85°C	VDD = 2.0V	High-Power BOR
		5	9	µA	-40°C to +85°C	VDD = 3.3V	
D022 (ΔIHLVD)	**High/Low-Voltage Detect**	4	8	µA	-40°C to +85°C	VDD = 2.0V	
		5	9	µA	-40°C to +85°C	VDD = 3.3V	
D023 (ΔIDSBOR)	**Deep Sleep BOR**	135	480	nA	-40°C to +85°C	VDD = 2.0V to 3.3V	ΔDeep Sleep BOR(2)
D024 (ΔIDSWDT)	**Deep Sleep Watchdog Timer**	290	480	nA	-40°C to +85°C	VDD = 2.0V to 3.3V	ΔDeep Sleep WDT(2)
D025 (ΔIOSCB)	**Real-Time Clock/ Calendar with Timer1 Oscillator**	0.38	1	µA	-40°C to +85°C	VDD = 2.0V	Sleep mode 32.768 kHz, T1OSCEN = 1, LPT1OSC = 0
		0.55	1	µA	-40°C to +85°C	VDD = 3.3V	
D027 (ΔILCD)	**LCD Module**	0.6	4	µA	-40°C to +85°C	VDD = 3.3V	ΔLCD External/Internal, 1/8 MUX, 1/3 Bias(2,3)
		6	30	µA	-40°C to +85°C	VDD = 2.0V	ΔLCD Charge Pump, 1/8 MUX, 1/3 Bias(2,4)
		7	40	µA	-40°C to +85°C	VDD = 3.3V	
D028 (ΔIADRC)	**A/D with RC**	330	500	µA	-40°C to +85°C	VDD = 2.0V	
		385	500	µA	-40°C to +85°C	VDD = 3.3V	
D028 (ΔIUSB)	**USB Module**	1	2	mA	-40°C to +85°C	VDD and VUSB3V3 = 3.3V	USB enabled, no cable connected; traffic makes a large difference(5)

Note 1: Data in the Typical column is at 3.3V, 25°C unless otherwise stated. Parameters are for design guidance only and are not tested.

2: Incremental current while the module is enabled and running.

3: LCD is enabled and running, no glass is connected; the resistor ladder current is not included.

4: LCD is enabled and running, no glass is connected.

5: This is the module differential current when the USB module is enabled and clocked at 48 MHz, but with no USB cable attached. When the USB cable is attached, or data is being transmitted, the current consumption may be much higher (see **Section 27.6.4 "USB Transceiver Current Consumption"**). During USB Suspend mode (USBEN = 1, SUSPND = 1, bus in Idle state), the USB module current will be dominated by the D+ or D- pull-up resistor. The integrated pull-up resistors use "resistor switching" according to the resistor_ecn supplement to the "*USB 2.0 Specification*" and therefore, may be as low as 900Ω during Idle conditions.

TABLE 30-8: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18F97J94 FAMILY (INDUSTRIAL)

DC CHARACTERISTICS			**Standard Operating Conditions: 3.0V < VDD < 3.6V** -40°C ≤ TA ≤ +85°C for Industrial (unless otherwise stated)				
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
	VBT	Operating Voltage	2.0	—	3.6	V	Battery connected to VBAT pin
	VBTADC	VBAT A/D Monitoring Voltage Specification(1)	1.6	—	3.6	V	A/D monitoring the VBAT pin using the internal A/D channel

Note 1: Measure A/D value using the A/D represented by the equation (Measured Voltage = ((VBAT/2)/VDD) * 1024) for 10-bit A/D; Measured Voltage = ((VBAT/2)/VDD) * 4096) for 12-bit A/D.

TABLE 30-9: DC CHARACTERISTICS: POWER-DOWN AND SUPPLY CURRENT PIC18F97J94 FAMILY (INDUSTRIAL)

DC CHARACTERISTICS			Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial			
Param No.	**Sym.**	**Characteristic**	**Min.**	**Max.**	**Units**	**Conditions**
	VIL	**Input Low Voltage**				
		All I/O Ports:				
D031		Schmitt Trigger Buffer	Vss	0.2 VDD	V	2V ≤ VDD ≤ 3.6V
D031A		RC3 and RC4	Vss	0.3 VDD	V	I2C enabled
D031B			Vss	0.8	V	SMBus enabled
D032		$\overline{MCLR}$	Vss	0.2 VDD	V	
D033		OSC1	Vss	0.2 VDD	V	LP, MS, HS modes
D033A		OSC1	Vss	0.2 VDD	V	EC modes
D034		SOSCI	Vss	0.3 VDD	V	
	VIH	**Input High Voltage**				
		All I/O Ports:				
D041		Schmitt Trigger Buffer	0.8 VDD	VDD	V	2V ≤ VDD ≤ 3.6V
D041A		RC3 and RC4	0.7 VDD	VDD	V	I2C enabled
D041B			2.1	VDD	V	SMBus enabled
D042		$\overline{MCLR}$	0.8 VDD	VDD	V	
D043		OSC1	0.9 VDD	VDD	V	RC mode
D043A		OSC1	0.7 VDD	VDD	V	HS mode
D044		SOSCI	0.7 VDD	VDD	V	
	IIL	**Input Leakage Current**(1)				
D060		I/O Ports	±50	±500	nA	Vss ≤ VPIN ≤ VDD Pin at high-impedance
D061		$\overline{MCLR}$	—	±500	nA	Vss ≤ VPIN ≤ VDD
D063		OSC1	—	1	µA	Vss ≤ VPIN ≤ VDD
D070	IPU	**Weak Pull-up Current**				
		Weak Pull-up Current	50	400	µA	VDD = 3.6V, VPIN = Vss
	VOL	**Output Low Voltage**				
D080		I/O Ports:	—	0.4	V	IOL = 6.6 mA, VDD = 3.6V
		All Ports	—	0.4	V	IOL = 5.0 mA, VDD = 2V
D083		OSC2/CLKO	—	0.4	V	IOL = 6.6 mA, VDD = 3.6V
		(EC modes)	—	0.4	V	IOL = 5.0 mA, VDD = 2V
	VOH	**Output High Voltage**(1)				
D090		I/O Ports:	3.0	—	V	IOH = -3.0 mA, VDD = 3.6V
		All Ports	2.4	—	V	IOH = -6.0 mA, VDD = 3.6V
			1.6	—	V	IOH = -1.0 mA, VDD = 2V
D092		OSC2/CLKO	1.4	—	V	IOH = -3.0 mA, VDD = 2V
		(INTOSC, EC modes)				
			2.4	—	V	IOH = -6.0 mA, VDD = 3.6V
			1.4	—	V	IOH = -1.0 mA, VDD = 2V
		Capacitive Loading Specs on Output Pins				
D100	COSC2	OSC2 Pin	—	20	pF	In HS mode when external clock is used to drive OSC1
D101	CIO	All I/O Pins and OSC2	—	50	pF	To meet the AC Timing Specifications
D102	CB	SCLx, SDAx	—	400	pF	I2C Specification

Note 1: Negative current is defined as current sourced by the pin.

TABLE 30-10: DC CHARACTERISTICS: CTMU CURRENT SOURCE SPECIFICATIONS

DC CHARACTERISTICS			Standard Operating Conditions: 2V to 3.6V Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ.(1)**	**Max.**	**Units**	**Conditions**
	IOUT1	CTMU Current Source, Base Range	—	550	—	nA	CTMUCON1<1:0> = 01
	IOUT2	CTMU Current Source, 10x Range	—	5.5	—	µA	CTMUCON1<1:0> = 10
	IOUT3	CTMU Current Source, 100x Range	—	55	—	µA	CTMUCON1<1:0> = 11

Note 1: Nominal value at center point of current trim range (CTMUCON1<7:2> = 000000).

TABLE 30-11: MEMORY PROGRAMMING REQUIREMENTS

DC CHARACTERISTICS			Standard Operating Conditions Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	**Sym.**	**Characteristic**	**Min.**	**Typ†**	**Max.**	**Units**	**Conditions**
		Internal Program Memory Programming Specifications(1)					
D110	VPP	Voltage on $\overline{MCLR}$/VPP Pin	VDD + 1.5	—	10	V	**(Note 2, Note 3)**
D113	IDDP	Supply Current During Programming	—	—	10	mA	
		Program Flash Memory					
D130	EP	Cell Endurance	1K	20K	—	E/W	-40°C to +85°C
D131	VPR	VDD for Read	2	—	3.6	V	
D132B	VPEW	Voltage for Self-Timed Erase or Write Operations VDD	2	—	3.6	V	PIC18FXXKXX devices
D133A	TIW	Self-Timed Write Cycle Time	—	2	—	ms	
D133B	TIE	Self-Timed Block Erased Cycle Time	—	33	—	ms	
D134	TRETD	Characteristic Retention	10	—	—	Year	Provided no other specifications are violated
D135	IDDP	Supply Current during Programming	—	—	10	mA	
D140	TWE	Writes per Erase Cycle	—	—	1		For each physical address

† Data in "Typ" column is at 3.3V, 25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

Note 1: These specifications are for programming the on-chip program memory through the use of table write instructions.

2: Required only if single-supply programming is disabled.

3: The MPLAB® ICD 2 does not support variable VPP output. Circuitry to limit the MPLAB ICD 2 VPP voltage must be placed between the MPLAB ICD 2 and the target system when programming or debugging with the MPLAB ICD 2.

TABLE 30-12: COMPARATOR SPECIFICATIONS

Operating Conditions: 2.0V ≤ VDD ≤ 3.6V, -40°C ≤ TA ≤ +85°C							
Param No.	**Sym.**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Comments**
D300	VIOFF	Input Offset Voltage	—	±5.0	40	mV	
D301	VICM	Input Common-Mode Voltage	0	—	AVDD	V	
D302	CMRR	Common-Mode Rejection Ratio	55	—	—	dB	
D303	TRESP	Response Time(1)	—	150	400	ns	
D304	TMC2OV	Comparator Mode Change to Output Valid*	—	—	10	μs	

Note 1: Response time is measured with one comparator input at (AVDD – 1.5)/2, while the other input transitions from VSS to VDD.

TABLE 30-13: COMPARATOR VOLTAGE REFERENCE SPECIFICATIONS

Operating Conditions: 2.0V ≤ VDD ≤ 3.6V, -40°C ≤ TA ≤ +85°C							
Param No.	**Sym.**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Comments**
D310	VRES	Resolution	VDD/32	—	VDD/32	LSb	
D311	VRAA	Absolute Accuracy	—	—	3/4	LSb	
D312	VRUR	Unit Resistor Value (R)	—	2k	—	Ω	
D313	TSET	Settling Time(1)	—	—	10	μs	

Note 1: Settling time measured while CVRR = 1 and CVR<3:0> transitions from '0000' to '1111'.

TABLE 30-14: INTERNAL VOLTAGE REGULATOR SPECIFICATIONS

Operating Conditions: -40°C ≤ TA ≤ +85°C							
Param No.	**Sym.**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Comments**
	VRGOUT	Regulator Output Voltage	—	1.8	—	V	
	CEFC	External Filter Capacitor Value	4.7	10	—	μF	Capacitor must be low-ESR, a low series resistance (< 5Ω)

TABLE 30-15: RC OSCILLATOR START-UP TIME

AC CHARACTERISTICS		**Standard Operating Conditions: 2V to 3.6V (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for Industrial				
Param No.	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Comments**
	TFRC	—	15	—	μs	
	TLPRC	—	10	—	μs	

TABLE 30-16: USB MODULE SPECIFICATIONS

Operating Conditions: -40°C <T_A < +85°C							
Param No.	**Sym.**	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Comments**
D313	V_{USB3V3}	USB Voltage	3	—	3.6	V	Voltage on V_{USB3V3} pin must be in this range for proper USB operation
D314	I_{IL}	Input Leakage on Pin	—	—	±1	µA	V_{SS} < V_{PIN} < V_{DD} pin at high-impedance
D318	V_{DIFS}	Differential Input Sensitivity	—	—	0.2	V	The difference between D+ and D- must exceed this value while V_{CM} is met
D319	V_{CM}	Differential Common-Mode Range	0.8	—	2.5	V	
D320	Z_{OUT}	Driver Output Impedance(1)	28	—	44	Ω	
D321	V_{OL}	Voltage Output Low	0	—	0.3	V	1.5 kΩ load connected to 3.6V
D322	V_{OH}	Voltage Output High	2.8	—	3.6	V	1.5 kΩ load connected to ground

Note 1: The D+ and D- signal lines have built-in impedance matching resistors. No external resistors, capacitors or magnetic components are necessary on the D+/D- signal paths between the PIC18F97J94 family device and a USB cable.

30.1 AC (Timing) Characteristics

30.1.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created following one of the following formats:

TABLE 30-17: TIMING PARAMETER SYMBOLS

1. TppS2ppS
2. TppS
3. TCC:ST (I^2C specifications only)
4. Ts (I^2C specifications only)

T			
F	Frequency	T	Time

Lowercase letters (pp) and their meanings:

pp			
cc	CCP1	osc	OSC1
ck	CLKO	rd	$\overline{RD}$
cs	$\overline{CS}$	rw	$\overline{RD}$ or $\overline{WR}$
di	SDI	sc	SCK
do	SDO	ss	$\overline{SS}$
dt	Data in	t0	T0CKI
io	I/O port	t1	T1CKI
mc	$\overline{MCLR}$	wr	$\overline{WR}$

Uppercase letters and their meanings:

S			
F	Fall	P	Period
H	High	R	Rise
I	Invalid (High-impedance)	V	Valid
L	Low	Z	High-impedance
I^2C only			
AA	output access	High	High
BUF	Bus free	Low	Low

TCC:ST (I^2C specifications only)

CC			
HD	Hold	SU	Setup
ST			
DAT	DATA input hold	STO	Stop condition
STA	Start condition		

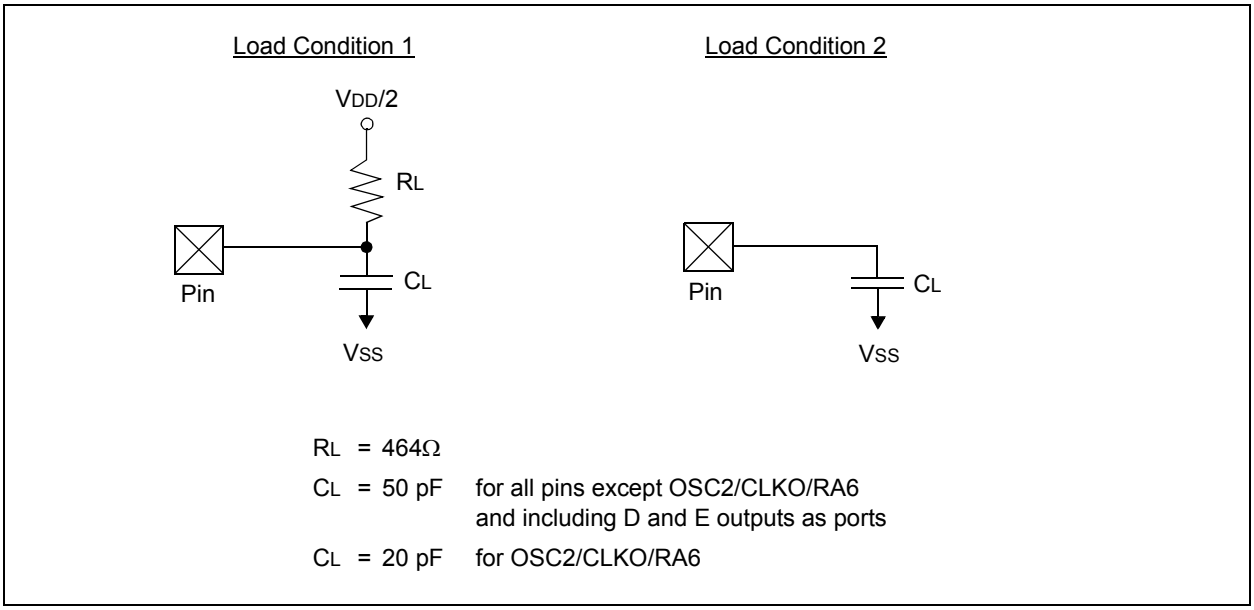
30.1.2 TIMING CONDITIONS

The temperature and voltages specified in Table 30-18 apply to all timing specifications unless otherwise noted. Figure 30-2 specifies the load conditions for the timing specifications.

TABLE 30-18: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC

AC CHARACTERISTICS	**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ TA ≤ +85°C for industrial Operating voltage VDD range as described in **Section TABLE 30-1:** and **Section** .

FIGURE 30-2: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS



RL = 464Ω

CL = 50 pF for all pins except OSC2/CLKO/RA6 and including D and E outputs as ports

CL = 20 pF for OSC2/CLKO/RA6

30.1.3 TIMING DIAGRAMS AND SPECIFICATIONS

FIGURE 30-3: EXTERNAL CLOCK TIMING

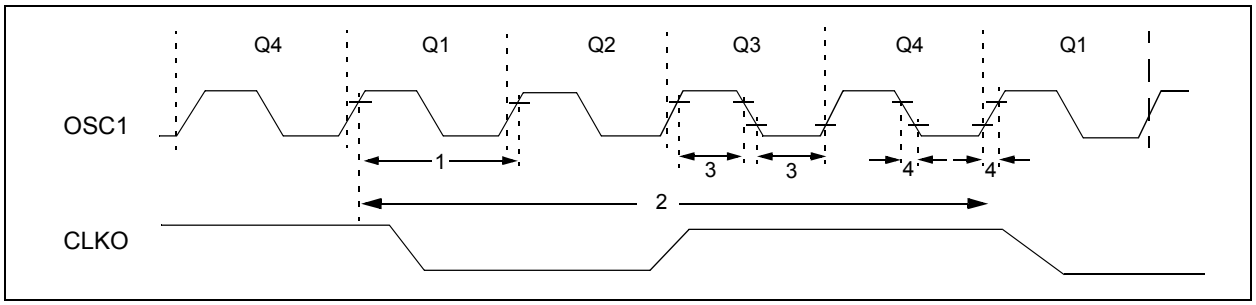


TABLE 30-19: EXTERNAL CLOCK TIMING REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
1A	FOSC	External CLKIN Frequency(1)	DC	64	MHz	EC Oscillator mode
		Oscillator Frequency(1)	4	16	MHz	HS Oscillator mode
			4	16	MHz	HS + PLL Oscillator mode
1	TOSC	External CLKIN Period(1)	15.6	—	ns	EC, ECIO Oscillator mode
		Oscillator Period(1)	40	250	ns	HS Oscillator mode
			62.5	250	ns	HS+PLL Oscillator mode
2	TCY	Instruction Cycle Time(1)	62.5	—	ns	TCY = 4/FOSC
3	TOSL, TOSH	External Clock in (OSC1) High or Low Time	10	—	ns	HS Oscillator mode
4	TOSR, TOSF	External Clock in (OSC1) Rise or Fall Time	—	7.5	ns	HS Oscillator mode

Note 1: Instruction cycle period (TCY) equals four times the input oscillator time base period for all configurations except PLL. All specified values are based on characterization data for that particular oscillator type under standard operating conditions with the device executing code. Exceeding these specified limits may result in an unstable oscillator operation and/or higher than expected current consumption. All devices are tested to operate at “min.” values with an external clock applied to the OSC1/CLKIN pin. When an external clock input is used, the “max.” cycle time limit is “DC” (no clock) for all devices.

TABLE 30-20: 4/6/8x PLL CLOCK TIMING SPECIFICATIONS (VDD = 2.0V TO 3.6V)(1)

Param No.	Sym.	Characteristic	Min.	Typ.	Max.	Units	Conditions
F10	FOSC	Oscillator Frequency Range	4	—	16	MHz	VDD = 2.0-3.6V, -40°C to +85°C
F11	FSYS	On-Chip VCO System Frequency	16	—	64	MHz	VDD = 2.0-3.6V, -40°C to +85°C
F12	t_{rc}	PLL Start-up Time (Lock Time)	—	—	2	ms	
F13	ΔCLK	CLKOUT Stability (Jitter)	-2	—	+2	%	

Note 1: These specifications are for x96 PLL or x8 PLL.

TABLE 30-21: 96 MHZ PLL CLOCK TIMING SPECIFICATIONS (V_{DD} = 2.0V TO 3.6V)

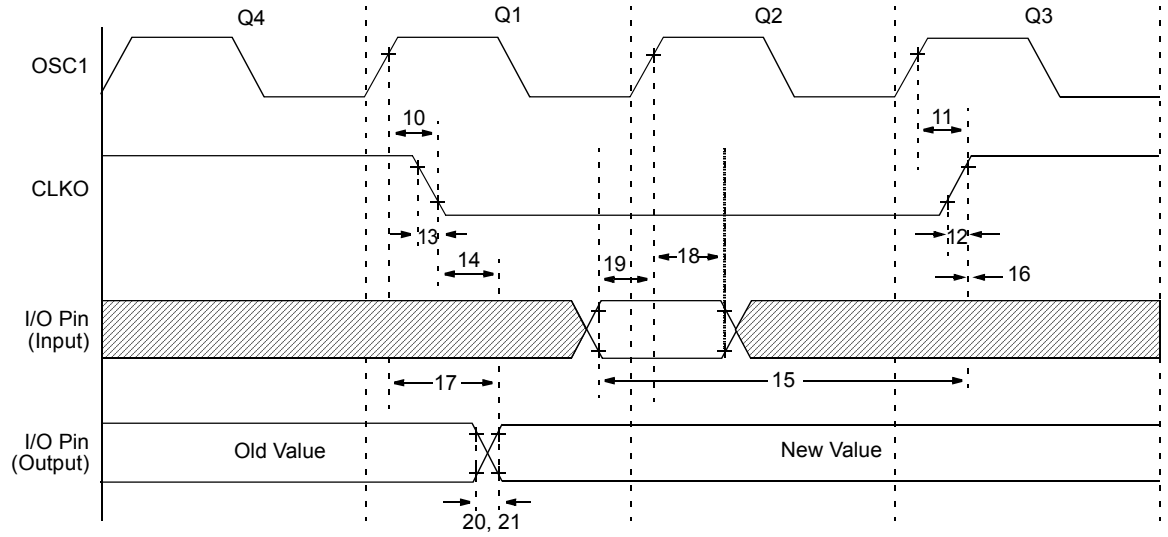
Sym.	Characteristic	Min.	Typ.	Max.	Units	Conditions
F_{PLLIN}	PLL Input Frequency Range (after prescaling)	3.94	4	4.06	MHz	V_{DD} = 2.0-3.6V, -40°C to +85°C
F_{SYS}	On-Chip VCO System Frequency	—	96	—	MHz	V_{DD} = 2.0-3.6V, -40°C to +85°C
t_{rc}	PLL Start-up Time (Lock Time)	—	—	200	µs	
ΔCLK	CLKOUT Stability (Jitter)	-0.25	—	+0.25	%	

TABLE 30-22: INTERNAL RC ACCURACY (FRC)

PIC18FXXJ94		**Standard Operating Conditions (unless otherwise stated)** Operating temperature -40°C ≤ T_A ≤ +85°C					
Param No.	**Characteristics**	**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**	
OA1	**FRC Accuracy @ Freq = 8 MHz, 4 MHz, 2 MHz, 1 MHz, 500 kHz, 250 kHz, 125 kHz, 31.25 kHz(1)**						
		-0.5	—	+0.5	%	+25°C	V_{DD} = 3.0-3.6V
		-1.5	—	+1.5	%	-40°C to +85°C	V_{DD} = 2.0-3.6V
OA2	**LPRC Accuracy @ Freq = 31 kHz**						
		-20	—	20	%	-40°C to +85°C	V_{DD} = 2.0-3.6V

Note 1: Frequency is calibrated at +25°C. OSCTUNE register can be used to compensate for temperature drift.

FIGURE 30-4: CLKO AND I/O TIMING



Note: Refer to Figure 30-2 for load conditions.

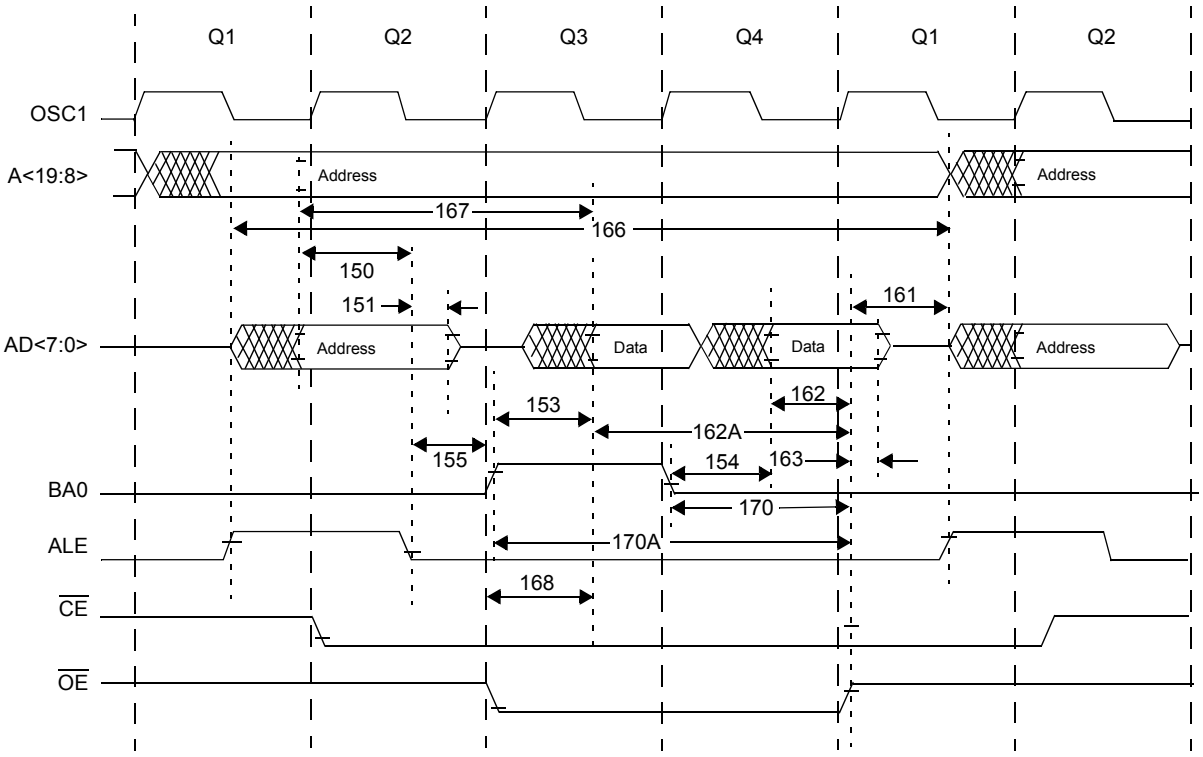
TABLE 30-23: CLKO AND I/O TIMING REQUIREMENTS

Param No.	Symbol	Characteristic	Min.	Typ.	Max.	Units	Conditions
10	TOSH2CKL	OSC1 ↑ to CLKO ↓	—	75	200	ns	**(Note 1)**
11	TOSH2CKH	OSC1 ↑ to CLKO ↑	—	75	200	ns	**(Note 1)**
12	TCKR	CLKO Rise Time	—	15	30	ns	**(Note 1)**
13	TCKF	CLKO Fall Time	—	15	30	ns	**(Note 1)**
14	TCKL2IOV	CLKO ↓ to Port Out Valid	—	—	0.5 TCY + 20	ns	
15	TIOV2CKH	Port In Valid before CLKO ↑	0.25 TCY + 25	—	—	ns	
16	TCKH2IOI	Port In Hold after CLKO ↑	0	—	—	ns	
17	TOSH2IOV	OSC1 ↑ (Q1 cycle) to Port Out Valid	—	50	150	ns	
18	TOSH2IOI	OSC1 ↑ (Q2 cycle) to Port Input Invalid (I/O in hold time)	100	—	—	ns	
19	TIOV2OSH	Port Input Valid to OSC1 ↑ (I/O in setup time)	0	—	—	ns	
20	TIOR	Port Output Rise Time	—	10	25	ns	
21	TIOF	Port Output Fall Time	—	10	25	ns	
22†	TINP	INTx Pin High or Low Time	20	—	—	ns	
23†	TRBP	RB<7:4> Change INTx High or Low Time	TCY	—	—	ns	

† These parameters are asynchronous events not related to any internal clock edges.

Note 1: Measurements are taken in EC mode, where CLKO output is 4 x TOSC.

FIGURE 30-5: PROGRAM MEMORY FETCH TIMING DIAGRAM (8-BIT)

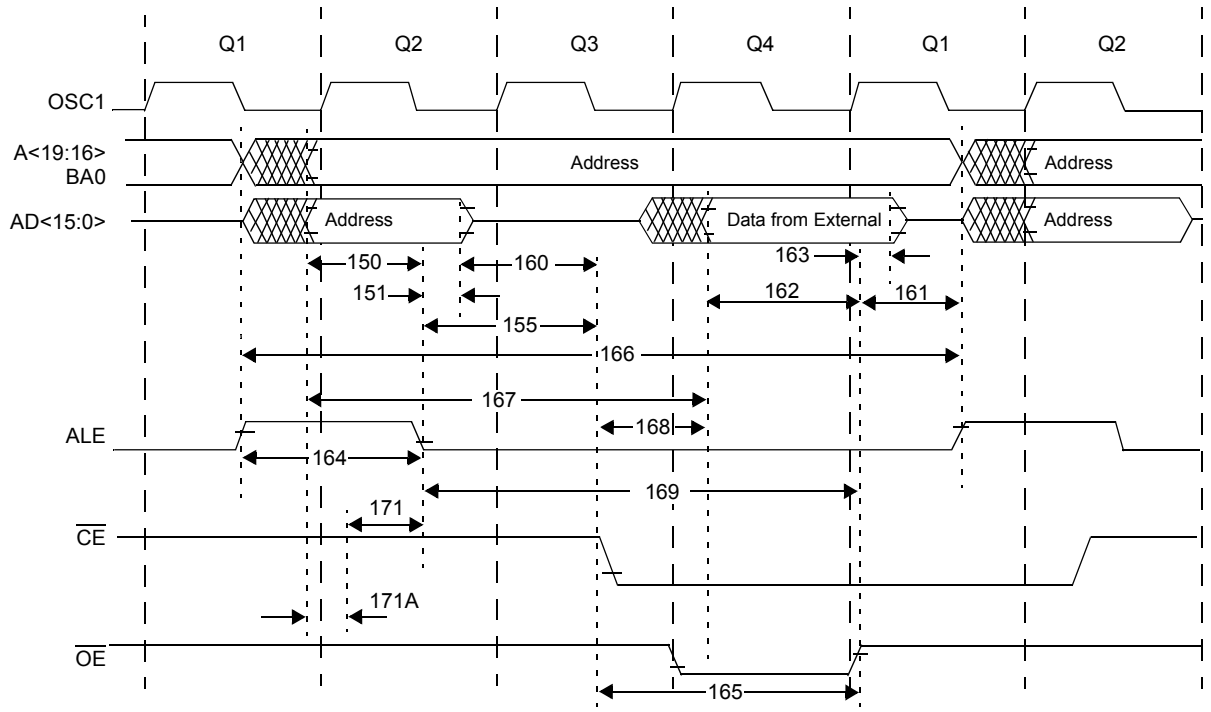


Note: Fmax = 25 MHz in 8-Bit External Memory mode.

TABLE 30-24: PROGRAM MEMORY FETCH TIMING REQUIREMENTS (8-BIT)

Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units
150	TadV2alL	Address Out Valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ to Address Out Invalid (address hold time)	5	—	—	ns
153	BA01	BA0 ↑ to Most Significant Data Valid	0.125 TCY	—	—	ns
154	BA02	BA0 ↓ to Least Significant Data Valid	0.125 TCY	—	—	ns
155	TalL2oeL	ALE ↓ to $\overline{OE}$ ↓	0.125 TCY	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ to A/D Driven	0.125 TCY – 5	—	—	ns
162	TadV2oeH	Least Significant Data Valid Before $\overline{OE}$ ↑ (data setup time)	20	—	—	ns
162A	TadV2oeH	Most Significant Data Valid Before $\overline{OE}$ ↑ (data setup time)	0.25 TCY + 20	—	—	ns
163	ToeH2adI	$\overline{OE}$ ↑ to Data in Invalid (data hold time)	0	—	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	TCY	—	ns
167	TACC	Address Valid to Data Valid	0.5 TCY – 10	—	—	ns
168	Toe	$\overline{OE}$ ↓ to Data Valid	—	—	0.125 TCY + 5	ns
170	TubH2oeH	BA0 = 0 Valid Before $\overline{OE}$ ↑	0.25 TCY	—	—	ns
170A	TubL2oeH	BA0 = 1 Valid Before $\overline{OE}$ ↑	0.5 TCY	—	—	ns

FIGURE 30-6: PROGRAM MEMORY READ TIMING DIAGRAM

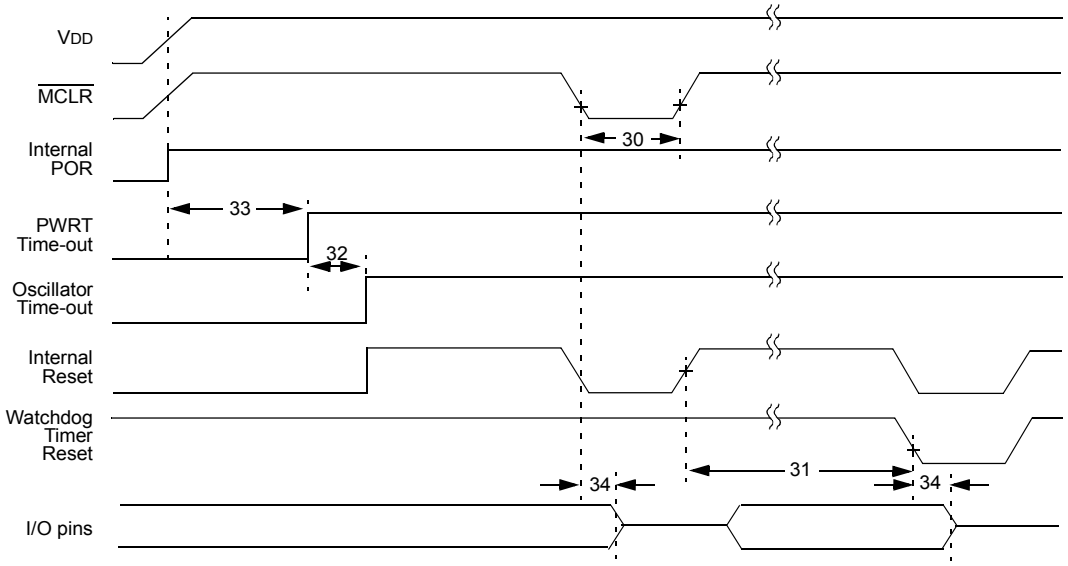


Operating Conditions: 2.0V < VCC < 3.6V, -40°C < TA < +125°C unless otherwise stated.

TABLE 30-25: CLKO AND I/O TIMING REQUIREMENTS

Param. No.	Symbol	Characteristics	Min.	Typ.	Max.	Units
150	TadV2alL	Address Out Valid to ALE ↓ (address setup time)	0.25 TCY – 10	—	—	ns
151	TalL2adl	ALE ↓ to Address Out Invalid (address hold time)	5	—	—	ns
155	TalL2oeL	ALE ↓ to $\overline{OE}$ ↓	10	0.125 TCY	—	ns
160	TadZ2oeL	A/D High-Z to $\overline{OE}$ ↓ (bus release to $\overline{OE}$)	0	—	—	ns
161	ToeH2adD	$\overline{OE}$ ↑ to A/D Driven	0.125 TCY – 5	—	—	ns
162	TadV2oeH	LS Data Valid before $\overline{OE}$ ↑ (data setup time)	20	—	—	ns
163	ToeH2adl	$\overline{OE}$ ↑ to Data In Invalid (data hold time)	0	—	—	ns
164	TalH2alL	ALE Pulse Width	—	0.25 TCY	—	ns
165	ToeL2oeH	$\overline{OE}$ Pulse Width	0.5 TCY – 5	0.5 TCY	—	ns
166	TalH2alH	ALE ↑ to ALE ↑ (cycle time)	—	TCY	—	ns
167	Tacc	Address Valid to Data Valid	0.75 TCY – 25	—	—	ns
168	Toe	$\overline{OE}$ ↓ to Data Valid		—	0.5 TCY – 25	ns
169	TalL2oeH	ALE ↓ to $\overline{OE}$ ↑	0.625 TCY – 10	—	0.625 TCY + 10	ns
171	TalH2csL	Chip Enable Active to ALE ↓	0.25 TCY – 20	—	—	ns
171A	TubL2oeH	A/D Valid to Chip Enable Active	—	—	10	ns

FIGURE 30-7: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER AND POWER-UP TIMER TIMING



Note: Refer to Figure 30-2 for load conditions.

FIGURE 30-8: BROWN-OUT RESET TIMING

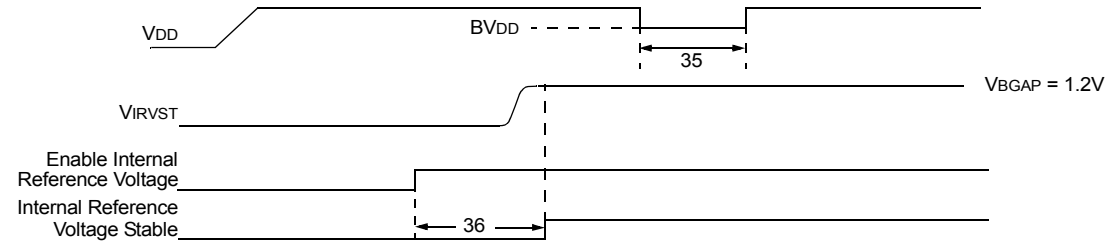


TABLE 30-26: RESET, WATCHDOG TIMER, OSCILLATOR START-UP TIMER, POWER-UP TIMER AND BROWN-OUT RESET REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Typ.	Max.	Units	Conditions
30	TmcL	$\overline{\text{MCLR}}$ Pulse Width (low)	2	—	—	µs	
31	TWDT	Watchdog Timer Time-out Period (no postscaler)	—	4.00	—	ms	
32	TOST	Oscillation Start-up Timer Period	1024 TOSC	—	1024 TOSC	—	TOSC = OSC1 period
33	TPWRT	Power-up Timer Period	—	300 µs	—	µs	
34	TIOZ	I/O High-Impedance from $\overline{\text{MCLR}}$ Low or Watchdog Timer Reset	—	2	—	µs	
35	TBOR	Brown-out Reset Pulse Width	200	—	—	µs	VDD ≤ BVDD (see D005)
36	TIRVST	Time for Internal Reference Voltage to become Stable	—	25	—	µs	
37	THLVD	High/Low-Voltage Detect Pulse Width	200	—	—	µs	VDD ≤ VHLVD
38	TCSD	CPU Start-up Time	5	—	10	µs	
39	TIOBST	Time for INTOSC to Stabilize	—	1	—	µs	

FIGURE 30-9: HIGH/LOW-VOLTAGE DETECT CHARACTERISTICS

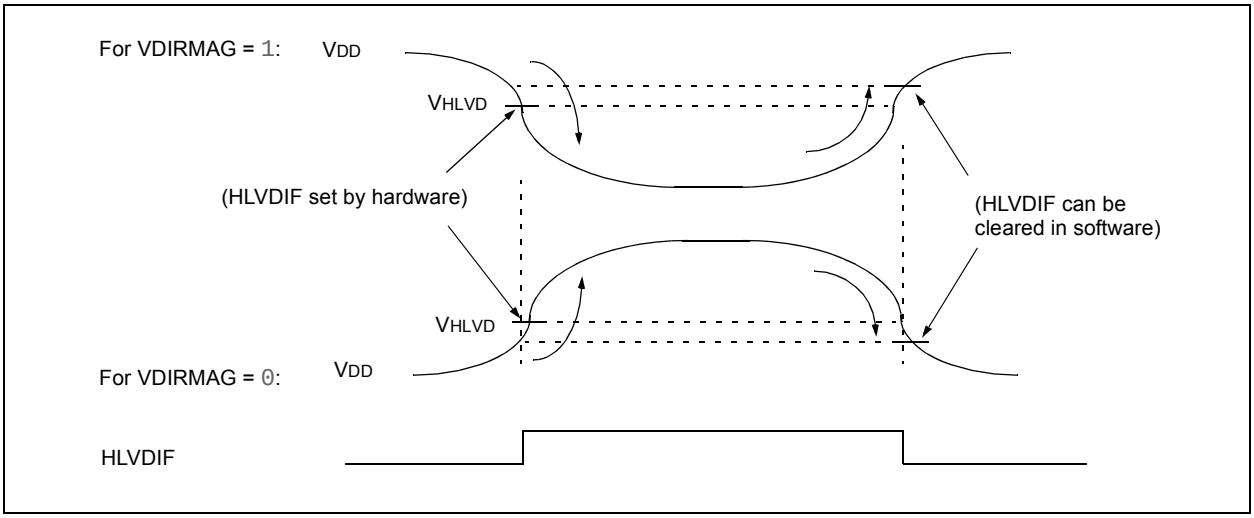
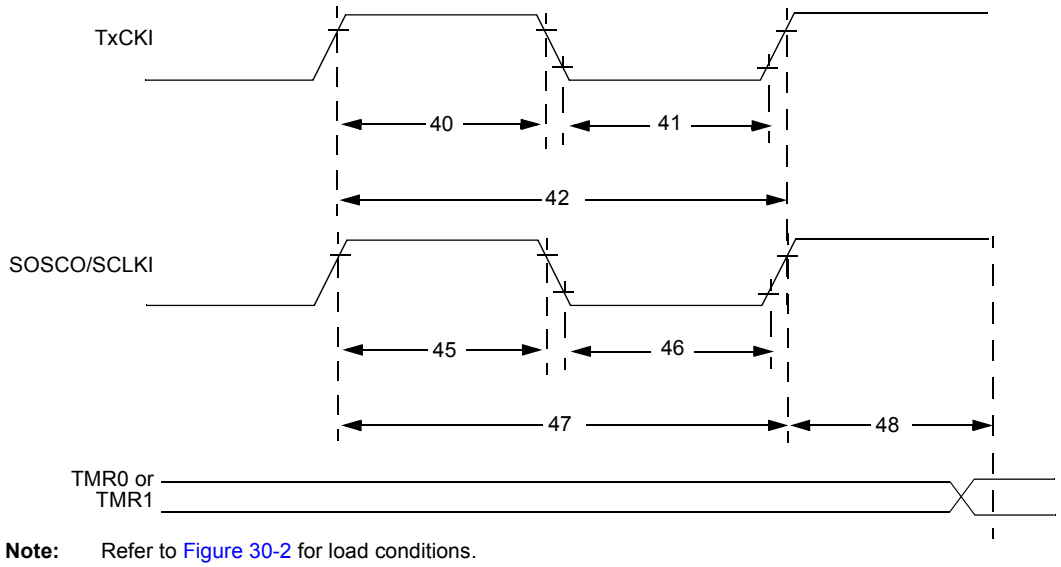


TABLE 30-27: HIGH/LOW-VOLTAGE DETECT CHARACTERISTICS

Standard Operating Conditions (unless otherwise stated) Operating temperature -40°C ≤ TA ≤ +85°C for industrial								
Param. No.	**Sym.**	**Characteristic**		**Min.**	**Typ.**	**Max.**	**Units**	**Conditions**
D420		HLVD Voltage on VDD Transition High-to-Low	HLVDL<3:0> = 0100	2.0	—	2.2	V	
			HLVDL<3:0> = 0101	2.1	—	2.3	V	
			HLVDL<3:0> = 0110	2.2	—	2.4	V	
			HLVDL<3:0> = 0111	2.3	—	2.5	V	
			HLVDL<3:0> = 1000	2.4	—	2.6	V	
			HLVDL<3:0> = 1001	2.5	—	2.75	V	
			HLVDL<3:0> = 1010	2.7	—	2.95	V	
			HLVDL<3:0> = 1011	2.8	—	3.1	V	
			HLVDL<3:0> = 1100	3.0	—	3.3	V	
			HLVDL<3:0> = 1101	3.3	—	3.6	V	
			HLVDL<3:0> = 1110	3.45	—	3.75	V	

FIGURE 30-10: TIMER0 AND TIMER1 EXTERNAL CLOCK TIMINGS

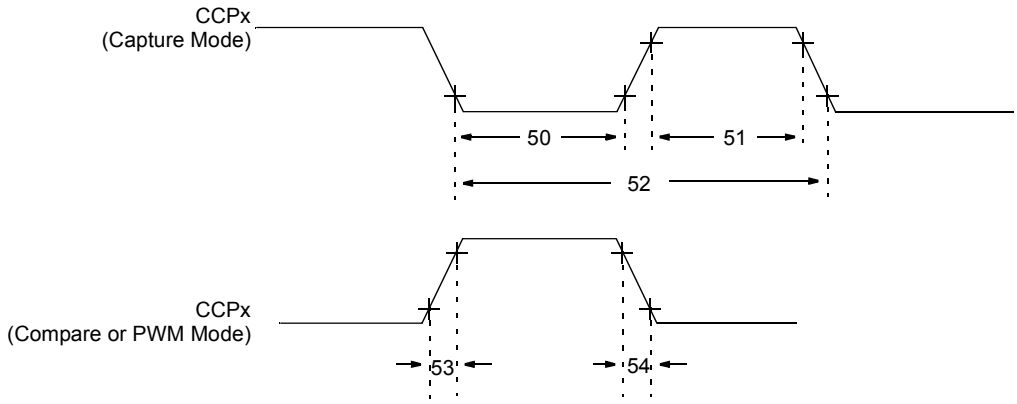


Note: Refer to Figure 30-2 for load conditions.

TABLE 30-28: TIMER0 AND TIMER1 EXTERNAL CLOCK REQUIREMENTS

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
40	TT0H	T0CKI High Pulse Width	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
41	TT0L	T0CKI Low Pulse Width	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
42	TT0P	T0CKI Period	No prescaler	TCY + 10	—	ns	
			With prescaler	Greater of: 20 ns or (TCY + 40)/N	—	ns	N = prescale value (1, 2, 4,..., 256)
45	TT1H	T1CKI High Time	Synchronous, no prescaler	0.5 TCY + 20	—	ns	
			Synchronous, with prescaler	10	—	ns	
			Asynchronous	30	—	ns	
46	TT1L	T1CKI Low Time	Synchronous, no prescaler	0.5 TCY + 5	—	ns	
			Synchronous, with prescaler	10	—	ns	
			Asynchronous	30	—	ns	
47	TT1P	T1CKI Input Period	Synchronous	Greater of: 20 ns or (TCY + 40)/N	—	ns	N = prescale value (1, 2, 4, 8)
			Asynchronous	60	—	ns	
	FT1	T1CKI Oscillator Input Frequency Range		DC	50	kHz	
48	TCKE2TMRI	Delay from External T1CKI Clock Edge to Timer Increment		2 TOSC	7 TOSC	—	

FIGURE 30-11: CAPTURE/COMPARE/PWM TIMINGS (CCP1, CCP2 MODULES)

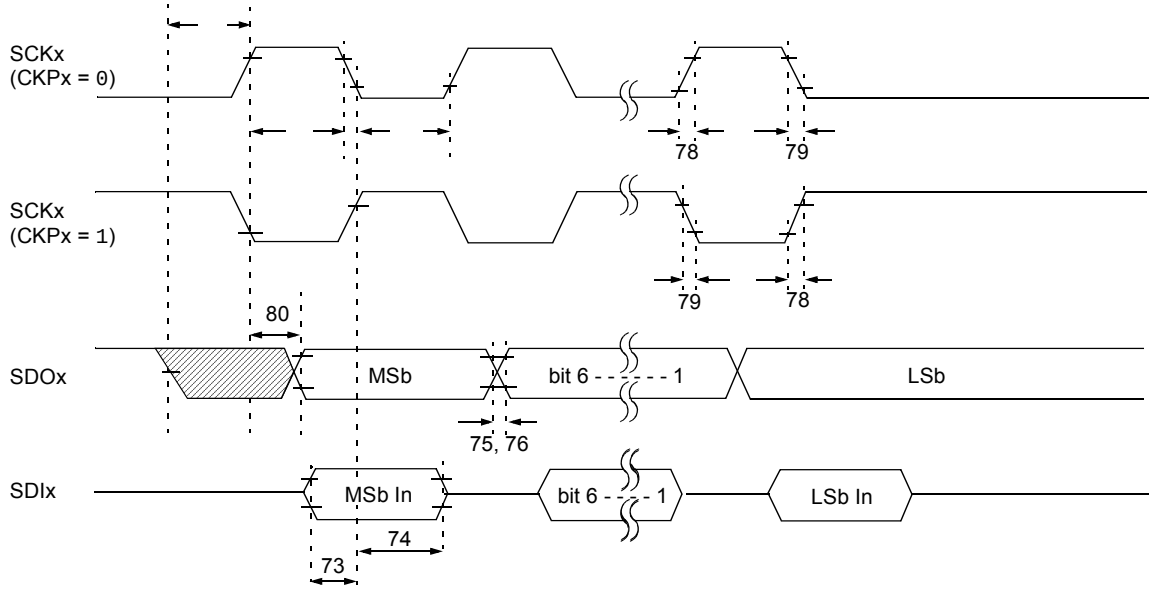


Note: Refer to Figure 30-2 for load conditions.

TABLE 30-29: CAPTURE/COMPARE/PWM REQUIREMENTS (CCP1, CCP2 MODULES)

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
50	TccL	CCPx Input Low Time	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
51	TccH	CCPx Input High Time	No prescaler	0.5 TCY + 20	—	ns	
			With prescaler	10	—	ns	
52	TccP	CCPx Input Period		$\frac{3\ T_{CY} + 40}{N}$	—	ns	N = prescale value (1, 4 or 16)
53	TccR	CCPx Output Fall Time		—	25	ns	
54	TccF	CCPx Output Fall Time		—	25	ns	

FIGURE 30-12: EXAMPLE SPI MASTER MODE TIMING (CKE = 0)

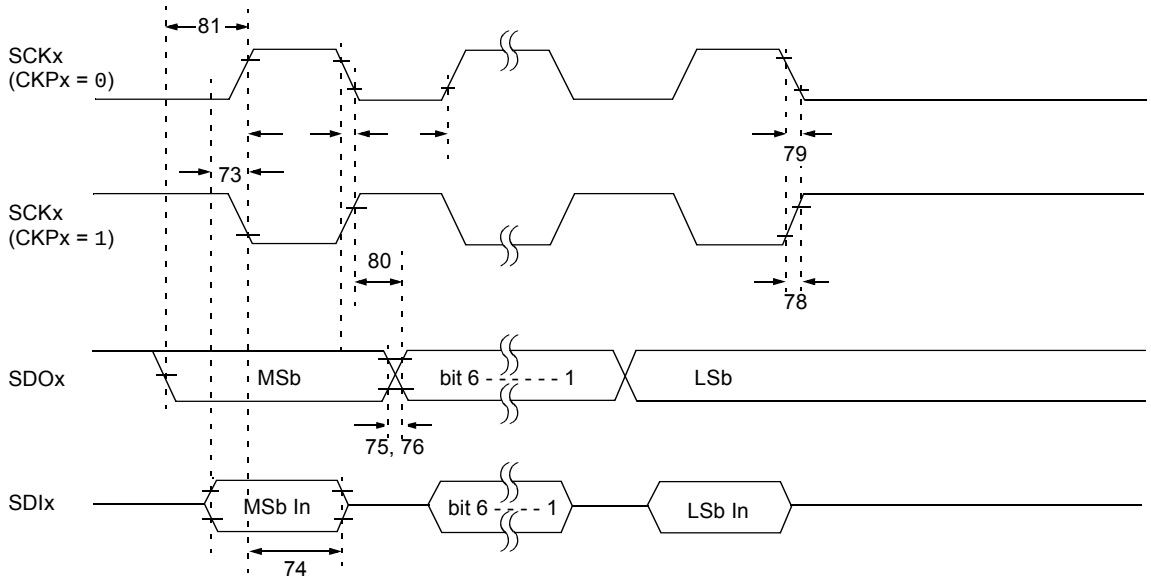


Note: Refer to Figure 30-2 for load conditions.

TABLE 30-30: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 0)

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
73	TDIV2SCH, TDIV2SCL	Setup Time of SDIx Data Input to SCKx Edge	20	—	ns	
73A	TB2B	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2	1.5 TCY + 40	—	ns	
74	TSCH2DIL, TSCL2DIL	Hold Time of SDIx Data Input to SCKx Edge	40	—	ns	
75	TDOR	SDOx Data Output Rise Time	—	25	ns	
76	TDOF	SDOx Data Output Fall Time	—	25	ns	
78	TSCR	SCKx Output Rise Time (Master mode)	—	25	ns	
79	TSCF	SCKx Output Fall Time (Master mode)	—	25	ns	
80	TSCH2DOV, TSCL2DOV	SDOx Data Output Valid after SCKx Edge	—	50	ns	

FIGURE 30-13: EXAMPLE SPI MASTER MODE TIMING (CKE = 1)

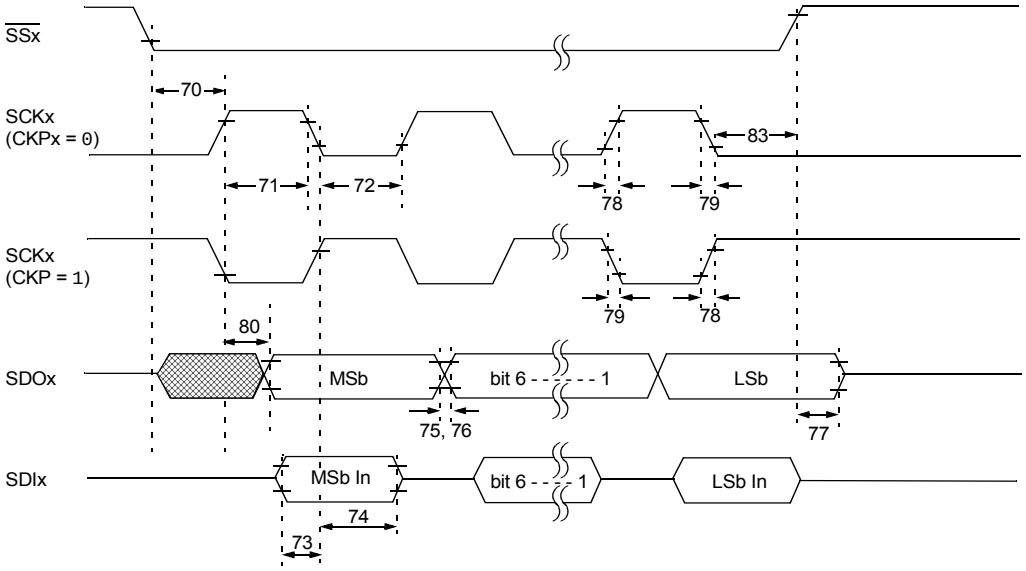


Note: Refer to Figure 30-2 for load conditions.

TABLE 30-31: EXAMPLE SPI MODE REQUIREMENTS (MASTER MODE, CKE = 1)

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
73	TDIV2SCH, TDIV2SCL	Setup Time of SDIx Data Input to SCKx Edge	20	—	ns	
73A	TB2B	Last Clock Edge of Byte 1 to the 1st Clock Edge of Byte 2	1.5 TCY + 40	—	ns	
74	TSCH2DIL, TSCL2DIL	Hold Time of SDIx Data Input to SCKx Edge	40	—	ns	
75	TDOR	SDOx Data Output Rise Time	—	25	ns	
76	TDOF	SDOx Data Output Fall Time	—	25	ns	
78	TSCR	SCKx Output Rise Time (Master mode)	—	25	ns	
79	TSCF	SCKx Output Fall Time (Master mode)	—	25	ns	
80	TSCH2DOV, TSCL2DOV	SDOx Data Output Valid after SCKx Edge	—	50	ns	
81	TDOV2SCH, TDOV2SCL	SDOx Data Output Setup to SCKx Edge	TCY	—	ns	

FIGURE 30-14: EXAMPLE SPI SLAVE MODE TIMING (CKE = 0)



Note: Refer to Figure 30-2 for load conditions.

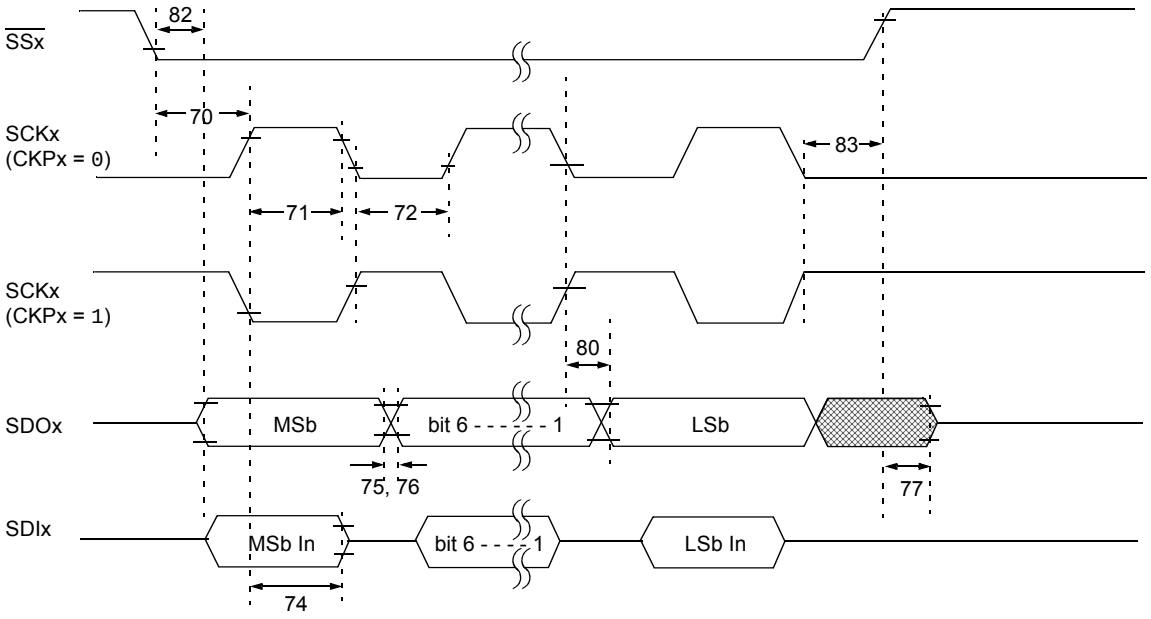
TABLE 30-32: EXAMPLE SPI MODE REQUIREMENTS (SLAVE MODE TIMING, CKE = 0)

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SSx}$ ↓ to SCKx ↓ or SCKx ↑ Input		3 TCY	—	ns	
70A	TssL2WB	$\overline{SSx}$ to write to SSPBUF		3 TCY	—	ns	
71	TscH	SCKx Input High Time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCKx Input Low Time (Slave mode)	Continuous	1.25 TCY + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73	TdiV2scH, TdiV2scL	Setup Time of SDIx Data Input to SCKx Edge		20	—	ns	
73A	Tb2b	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2		1.5 TCY + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge		40	—	ns	
75	TdoR	SDOx Data Output Rise Time		—	25	ns	
76	TdoF	SDOx Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SSx}$ ↑ to SDOx Output High-impedance		10	50	ns	
78	TscR	SCKx Output Rise Time (Master mode)		—	25	ns	
79	TscF	SCKx Output Fall Time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge		—	50	ns	
83	TscH2ssH, TscL2ssH	$\overline{SSx}$ ↑ after SCKx Edge		1.5 TCY + 40	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 30-15: EXAMPLE SPI SLAVE MODE TIMING (CKE = 1)



Note: Refer to Figure 30-2 for load conditions.

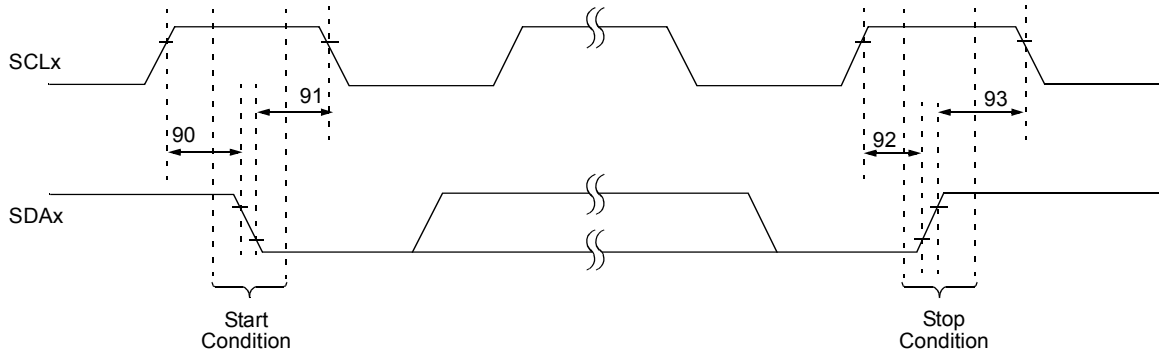
TABLE 30-33: EXAMPLE SPI SLAVE MODE REQUIREMENTS (CKE = 1)

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
70	TssL2scH, TssL2scL	$\overline{SSx}$ ↓ to SCKx ↓ or SCKx ↑ Input		3 Tcy	—	ns	
70A	TssL2WB	$\overline{SSx}$ to Write to SSPBUF		3 Tcy	—	ns	
71	TscH	SCKx Input High Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns	
71A			Single Byte	40	—	ns	**(Note 1)**
72	TscL	SCKx Input Low Time (Slave mode)	Continuous	1.25 Tcy + 30	—	ns	
72A			Single Byte	40	—	ns	**(Note 1)**
73A	Tb2b	Last Clock Edge of Byte 1 to the First Clock Edge of Byte 2		1.5 Tcy + 40	—	ns	**(Note 2)**
74	TscH2diL, TscL2diL	Hold Time of SDIx Data Input to SCKx Edge		40	—	ns	
75	TdoR	SDOx Data Output Rise Time		—	25	ns	
76	TdoF	SDOx Data Output Fall Time		—	25	ns	
77	TssH2doZ	$\overline{SSx}$ ↑ to SDOx Output High-Impedance		10	50	ns	
78	TscR	SCKx Output Rise Time (Master mode)		—	25	ns	
79	TscF	SCKx Output Fall Time (Master mode)		—	25	ns	
80	TscH2doV, TscL2doV	SDOx Data Output Valid after SCKx Edge		—	50	ns	
82	TssL2doV	SDOx Data Output Valid after $\overline{SSx}$ ↓ Edge		—	50	ns	
83	TscH2ssH, TscL2ssH	$\overline{SSx}$ ↑ after SCKx Edge		1.5 Tcy + 40	—	ns	

Note 1: Requires the use of Parameter #73A.

2: Only if Parameter #71A and #72A are used.

FIGURE 30-16: I²C BUS START/STOP BITS TIMING

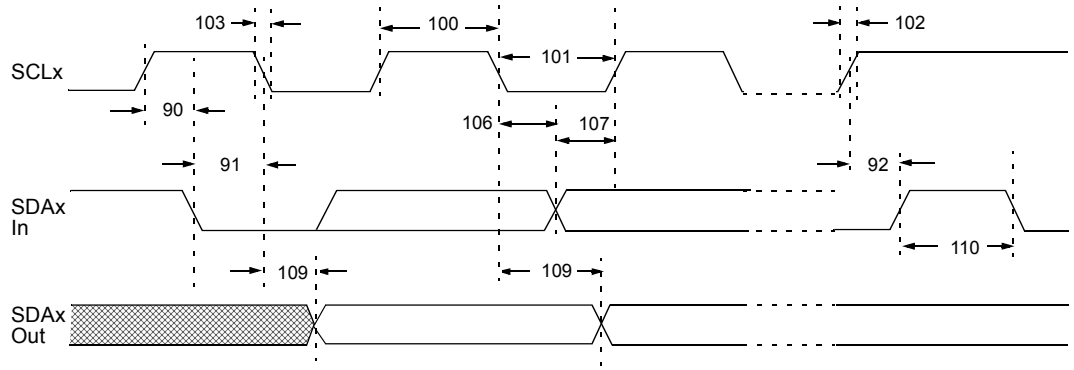


Note: Refer to Figure 30-2 for load conditions.

TABLE 30-34: I²C BUS START/STOP BITS REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4700	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	600	—		
91	THD:STA	Start Condition Hold Time	100 kHz mode	4000	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	600	—		
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	4700	—	ns	
			400 kHz mode	600	—		
93	THD:STO	Stop Condition Hold Time	100 kHz mode	4000	—	ns	
			400 kHz mode	600	—		

FIGURE 30-17: I²C BUS DATA TIMING



Note: Refer to Figure 30-2 for load conditions.

TABLE 30-35: I²C BUS DATA REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	4.0	—	µs	
			400 kHz mode	0.6	—	µs	
			MSSPx module	1.5 TCY	—		
101	TLOW	Clock Low Time	100 kHz mode	4.7	—	µs	
			400 kHz mode	1.3	—	µs	
			MSSPx module	1.5 TCY	—		
102	TR	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
103	TF	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns	
			400 kHz mode	20 + 0.1 CB	300	ns	CB is specified to be from 10 to 400 pF
90	TSU:STA	Start Condition Setup Time	100 kHz mode	4.7	—	µs	Only relevant for Repeated Start condition
			400 kHz mode	0.6	—	µs	
91	THD:STA	Start Condition Hold Time	100 kHz mode	4.0	—	µs	After this period, the first clock pulse is generated
			400 kHz mode	0.6	—	µs	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	ns	
			400 kHz mode	0	0.9	µs	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	4.7	—	µs	
			400 kHz mode	0.6	—	µs	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	**(Note 1)**
			400 kHz mode	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	µs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	µs	

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCLx to avoid unintended generation of Start or Stop conditions.

2: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCLx line is released.

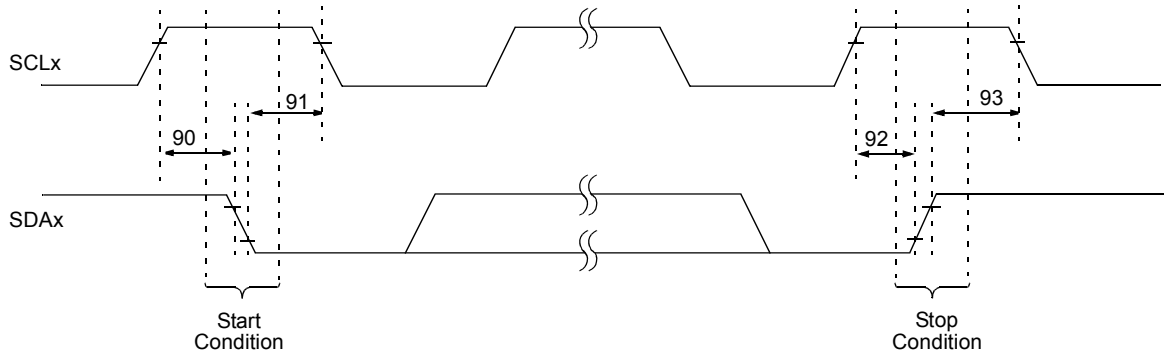
TABLE 30-35: I²C BUS DATA REQUIREMENTS (SLAVE MODE)

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
D102	CB	Bus Capacitive Loading	—	400	pF	

Note 1: As a transmitter, the device must provide this internal minimum delay time to bridge the undefined region (min. 300 ns) of the falling edge of SCLx to avoid unintended generation of Start or Stop conditions.

2: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but the requirement, TSU:DAT ≥ 250 ns, must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, TR max. + TSU:DAT = 1000 + 250 = 1250 ns (according to the Standard mode I²C bus specification), before the SCLx line is released.

FIGURE 30-18: MSSPx I²C BUS START/STOP BITS TIMING WAVEFORMS



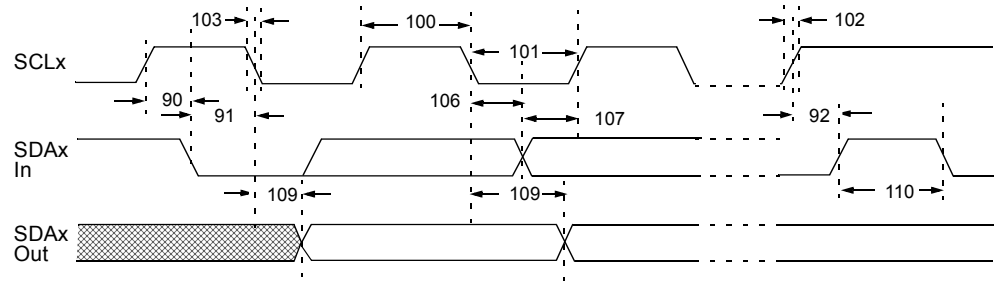
Note: Refer to Figure 30-2 for load conditions.

TABLE 30-36: MSSPx I²C BUS START/STOP BITS REQUIREMENTS

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		
93	THD:STO	Stop Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	ns	
			400 kHz mode	2(TOSC)(BRG + 1)	—		
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—		

Note 1: Maximum pin capacitance = 10 pF for all I²C pins.

FIGURE 30-19: MSSPx I²C BUS DATA TIMING



Note: Refer to Figure 30-2 for load conditions.

TABLE 30-37: MSSPx I²C BUS DATA REQUIREMENTS

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
100	THIGH	Clock High Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
101	TLOW	Clock Low Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
102	TR	SDAx and SCLx Rise Time	100 kHz mode	—	1000	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	300	ns	
103	TF	SDAx and SCLx Fall Time	100 kHz mode	—	300	ns	CB is specified to be from 10 to 400 pF
			400 kHz mode	20 + 0.1 CB	300	ns	
			1 MHz mode(1)	—	100	ns	
90	TSU:STA	Start Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	Only relevant for Repeated Start condition
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
91	THD:STA	Start Condition Hold Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	After this period, the first clock pulse is generated
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
106	THD:DAT	Data Input Hold Time	100 kHz mode	0	—	—	
			400 kHz mode	0	0.9	μs	
			1 MHz mode(1)	—	μs	ns	
107	TSU:DAT	Data Input Setup Time	100 kHz mode	250	—	ns	**(Note 2)**
			400 kHz mode	100	—	ns	
			1 MHz mode(1)	—	—	ns	

Note 1: Maximum pin capacitance = 10 pF for all II²C pins.

2: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but Parameter #107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, Parameter #102 + Parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCLx line is released.

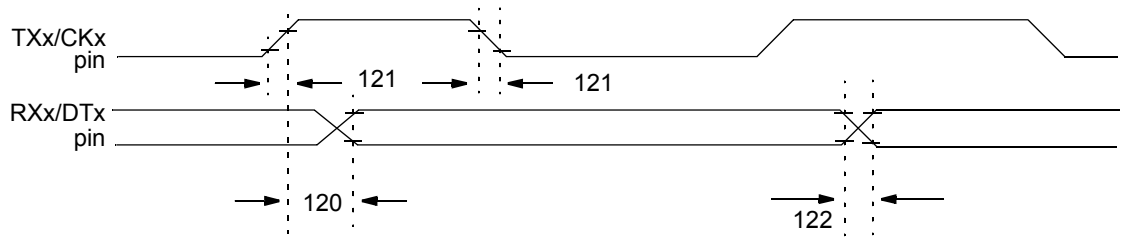
TABLE 30-37: MSSPx I²C BUS DATA REQUIREMENTS

Param. No.	Symbol	Characteristic		Min.	Max.	Units	Conditions
92	TSU:STO	Stop Condition Setup Time	100 kHz mode	2(TOSC)(BRG + 1)	—	—	
			400 kHz mode	2(TOSC)(BRG + 1)	—	—	
			1 MHz mode(1)	2(TOSC)(BRG + 1)	—	—	
109	TAA	Output Valid from Clock	100 kHz mode	—	3500	ns	
			400 kHz mode	—	1000	ns	
			1 MHz mode(1)	—	—	ns	
110	TBUF	Bus Free Time	100 kHz mode	4.7	—	µs	Time the bus must be free before a new transmission can start
			400 kHz mode	1.3	—	µs	
			1 MHz mode(1)	—	—	µs	
D102	CB	Bus Capacitive Loading		—	400	pF	

Note 1: Maximum pin capacitance = 10 pF for all II²C pins.

2: A Fast mode I²C bus device can be used in a Standard mode I²C bus system, but Parameter #107 ≥ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCLx signal. If such a device does stretch the LOW period of the SCLx signal, it must output the next data bit to the SDAx line, Parameter #102 + Parameter #107 = 1000 + 250 = 1250 ns (for 100 kHz mode), before the SCLx line is released.

FIGURE 30-20: EUSARTx SYNCHRONOUS TRANSMISSION (MASTER/SLAVE) TIMING

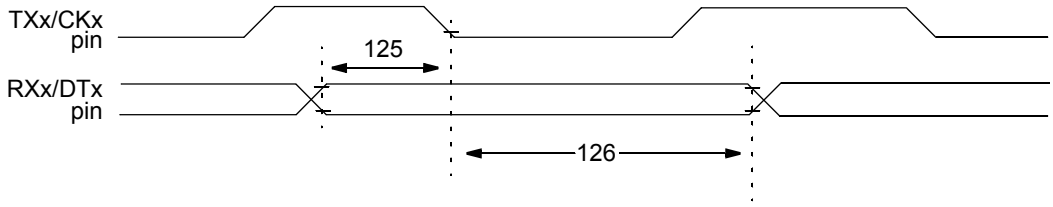


Note: Refer to Figure 30-2 for load conditions.

TABLE 30-38: EUSARTx/AUSARTx SYNCHRONOUS TRANSMISSION REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
120	TCKH2DTV	SYNC XMIT (MASTER and SLAVE) Clock High to Data Out Valid	—	40	ns	
121	TCKRF	Clock Out Rise Time and Fall Time (Master mode)	—	20	ns	
122	TDTRF	Data Out Rise Time and Fall Time	—	20	ns	

FIGURE 30-21: EUSARTx/AUSARTx SYNCHRONOUS RECEIVE (MASTER/SLAVE) TIMING



Note: Refer to Figure 30-2 for load conditions.

TABLE 30-39: EUSARTx/AUSARTx SYNCHRONOUS RECEIVE REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
125	TDTV2CKL	SYNC RCV (MASTER and SLAVE) Data Hold before CKx ↓ (DTx hold time)	10	—	ns	—
126	TCKL2DTL	Data Hold after CKx ↓ (DTx hold time)	15	—	ns	—

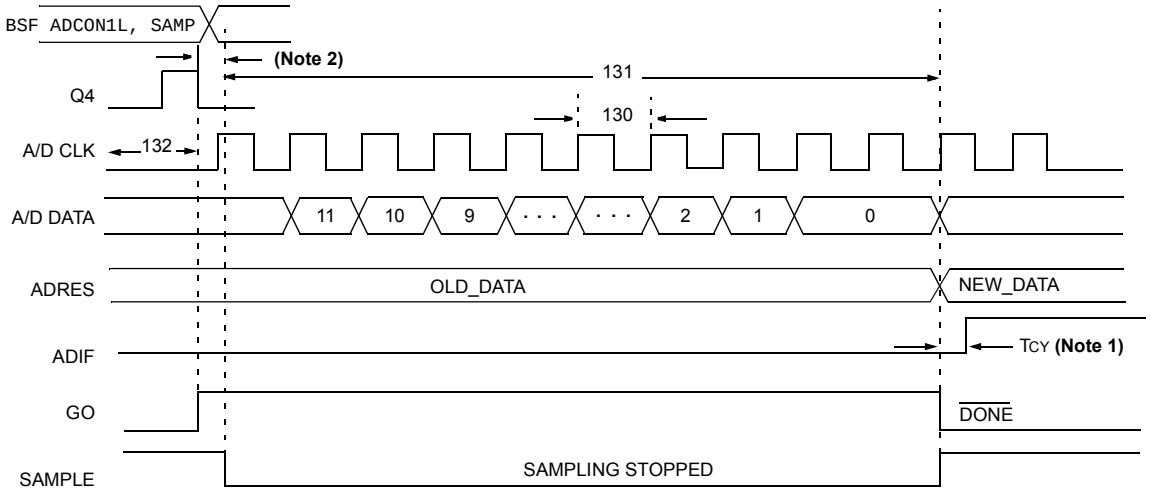
TABLE 30-40: A/D CONVERTER CHARACTERISTICS:PIC18FXXJ94 (INDUSTRIAL)

Param. No.	Sym.	Characteristic	Min.	Typ.	Max.	Units	Conditions
A01	NR	Resolution	—	—	12	bit	ΔVREF ≥ 2.0V
A03	EIL	Integral Linearity Error	—	<±1	±2.0	LSB	VDD = 3.0V (ΔVREF ≥ 2.0V)
A04	EDL	Differential Linearity Error	—	<±1	+2.0/-1.0	LSB	VDD = 3.0V (ΔVREF ≥ 2.0V)
A06	EOFF	Offset Error	—	<±1	±5	LSB	VDD = 3.0V (ΔVREF ≥ 2.0V)
A07	EGN	Gain Error	—	<±1	±5	LSB	VDD = 3.0V (ΔVREF ≥ 2.0V)
A10	—	Monotonicity(1)				—	VSS ≤ VAIN ≤ VREF
A20	ΔVREF	Reference Voltage Range (VREFH – VREFL)	2	—	VDD – VSS	V	For 12-bit resolution
A21	VREFH	Reference Voltage High	AVSS + 2.0V	—	AVDD + 0.3V	V	For 12-bit resolution
A22	VREFL	Reference Voltage Low	AVSS – 0.3V	—	AVDD – 2.0V	V	For 12-bit resolution
A25	VAIN	Analog Input Voltage	VREFL	—	VREFH	V	
A28	AVDD	Analog Supply Voltage	VDD – 0.3	—	VDD + 0.3	V	
A29	AVSS	Analog Supply Voltage	VSS – 0.3	—	VSS + 0.3	V	
A30	ZAIN	Recommended Impedance of Analog Voltage Source	—	—	2.5	kΩ	
A50	IREF	VREF Input Current(2)	— —	— —	5 150	μA μA	During VAIN acquisition. During A/D conversion cycle.

Note 1: The A/D conversion result never decreases with an increase in the input voltage.

2: VREFH current is from the RA3/AN3/VREF+ pin or VDD, whichever is selected as the VREFH source. VREFL current is from the RA2/AN2/VREF-/CVREF pin or VSS, whichever is selected as the VREFL source.

FIGURE 30-22: A/D CONVERSION TIMING



Note 1: If the A/D clock source is selected as RC, a time of TCY is added before the A/D clock starts. This allows the SLEEP instruction to be executed.

2: This is a minimal RC delay (typically 100 ns), which also disconnects the holding capacitor from the analog input.

TABLE 30-41: A/D CONVERSION REQUIREMENTS

Param. No.	Symbol	Characteristic	Min.	Max.	Units	Conditions
		Sample Start Delay from Setting SAMP	2	3	TAD	
130	TAD	A/D Clock Period	300	—	ns	
			250	—	ns	A/D RC mode
131	TCNV	Conversion Time (not including acquisition time)(2)	14	15	TAD	
132	TACQ	Acquisition Time(3)	—	750	ns	-40°C to +85°C(5)
135	TSWC	Switching Time from Convert → Sample	—	(Note 4)		
	TDIS	Discharge Time	1	—	TAD	-40°C to +85°C
		A/D Stabilization Time (from setting ADON to setting SAMP)	300	—	ns	

Note 1: The time of the A/D clock period is dependent on the device frequency and the TAD clock divider.

2: ADRES registers may be read on the following TCY cycle.

3: The time for the holding capacitor to acquire the "New" input voltage when the voltage changes full scale after the conversion (VDD to VSS or VSS to VDD). The source impedance (*RS*) on the input channels is 50Ω.

4: On the following cycle of the device clock.

5: The time for the holding capacitor to acquire the "New" input voltage when the voltage changes full scale after the conversion (AVDD to AVSS or AVSS to AVDD).

31.0 DEVELOPMENT SUPPORT

The PIC® microcontrollers (MCU) and dsPIC® digital signal controllers (DSC) are supported with a full range of software and hardware development tools:

- Integrated Development Environment
  - MPLAB® X IDE Software
- Compilers/Assemblers/Linkers
  - MPLAB XC Compiler
  - MPASM™ Assembler
  - MPLINK™ Object Linker/ MPLIB™ Object Librarian
  - MPLAB Assembler/Linker/Librarian for Various Device Families
- Simulators
  - MPLAB X SIM Software Simulator
- Emulators
  - MPLAB REAL ICE™ In-Circuit Emulator
- In-Circuit Debuggers/Programmers
  - MPLAB ICD 3
  - PICkit™ 3
- Device Programmers
  - MPLAB PM3 Device Programmer
- Low-Cost Demonstration/Development Boards, Evaluation Kits and Starter Kits
- Third-party development tools

31.1 MPLAB X Integrated Development Environment Software

The MPLAB X IDE is a single, unified graphical user interface for Microchip and third-party software, and hardware development tool that runs on Windows®, Linux and Mac OS® X. Based on the NetBeans IDE, MPLAB X IDE is an entirely new IDE with a host of free software components and plug-ins for high-performance application development and debugging. Moving between tools and upgrading from software simulators to hardware debugging and programming tools is simple with the seamless user interface.

With complete project management, visual call graphs, a configurable watch window and a feature-rich editor that includes code completion and context menus, MPLAB X IDE is flexible and friendly enough for new users. With the ability to support multiple tools on multiple projects with simultaneous debugging, MPLAB X IDE is also suitable for the needs of experienced users.

Feature-Rich Editor:

- Color syntax highlighting
- Smart code completion makes suggestions and provides hints as you type
- Automatic code formatting based on user-defined rules
- Live parsing

User-Friendly, Customizable Interface:

- Fully customizable interface: toolbars, toolbar buttons, windows, window placement, etc.
- Call graph window

Project-Based Workspaces:

- Multiple projects
- Multiple tools
- Multiple configurations
- Simultaneous debugging sessions

File History and Bug Tracking:

- Local file history feature
- Built-in support for Bugzilla issue tracker

31.2 MPLAB XC Compilers

The MPLAB XC Compilers are complete ANSI C compilers for all of Microchip's 8, 16, and 32-bit MCU and DSC devices. These compilers provide powerful integration capabilities, superior code optimization and ease of use. MPLAB XC Compilers run on Windows, Linux or MAC OS X.

For easy source level debugging, the compilers provide debug information that is optimized to the MPLAB X IDE.

The free MPLAB XC Compiler editions support all devices and commands, with no time or memory restrictions, and offer sufficient code optimization for most applications.

MPLAB XC Compilers include an assembler, linker and utilities. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. MPLAB XC Compiler uses the assembler to produce its object file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

31.3 MPASM Assembler

The MPASM Assembler is a full-featured, universal macro assembler for PIC10/12/16/18 MCUs.

The MPASM Assembler generates relocatable object files for the MPLINK Object Linker, Intel® standard HEX files, MAP files to detail memory usage and symbol reference, absolute LST files that contain source lines and generated machine code, and COFF files for debugging.

The MPASM Assembler features include:

- Integration into MPLAB X IDE projects
- User-defined macros to streamline assembly code
- Conditional assembly for multipurpose source files
- Directives that allow complete control over the assembly process

31.4 MPLINK Object Linker/ MPLIB Object Librarian

The MPLINK Object Linker combines relocatable objects created by the MPASM Assembler. It can link relocatable objects from precompiled libraries, using directives from a linker script.

The MPLIB Object Librarian manages the creation and modification of library files of precompiled code. When a routine from a library is called from a source file, only the modules that contain that routine will be linked in with the application. This allows large libraries to be used efficiently in many different applications.

The object linker/library features include:

- Efficient linking of single libraries instead of many smaller files
- Enhanced code maintainability by grouping related modules together
- Flexible creation of libraries with easy module listing, replacement, deletion and extraction

31.5 MPLAB Assembler, Linker and Librarian for Various Device Families

MPLAB Assembler produces relocatable machine code from symbolic assembly language for PIC24, PIC32 and dsPIC DSC devices. MPLAB XC Compiler uses the assembler to produce its object file. The assembler generates relocatable object files that can then be archived or linked with other relocatable object files and archives to create an executable file. Notable features of the assembler include:

- Support for the entire device instruction set
- Support for fixed-point and floating-point data
- Command-line interface
- Rich directive set
- Flexible macro language
- MPLAB X IDE compatibility

31.6 MPLAB X SIM Software Simulator

The MPLAB X SIM Software Simulator allows code development in a PC-hosted environment by simulating the PIC MCUs and dsPIC DSCs on an instruction level. On any given instruction, the data areas can be examined or modified and stimuli can be applied from a comprehensive stimulus controller. Registers can be logged to files for further run-time analysis. The trace buffer and logic analyzer display extend the power of the simulator to record and track program execution, actions on I/O, most peripherals and internal registers.

The MPLAB X SIM Software Simulator fully supports symbolic debugging using the MPLAB XC Compilers, and the MPASM and MPLAB Assemblers. The software simulator offers the flexibility to develop and debug code outside of the hardware laboratory environment, making it an excellent, economical software development tool.

31.7 MPLAB REAL ICE In-Circuit Emulator System

The MPLAB REAL ICE In-Circuit Emulator System is Microchip's next generation high-speed emulator for Microchip Flash DSC and MCU devices. It debugs and programs all 8, 16 and 32-bit MCU, and DSC devices with the easy-to-use, powerful graphical user interface of the MPLAB X IDE.

The emulator is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with either a connector compatible with in-circuit debugger systems (RJ-11) or with the new high-speed, noise tolerant, Low-Voltage Differential Signal (LVDS) interconnection (CAT5).

The emulator is field upgradable through future firmware downloads in MPLAB X IDE. MPLAB REAL ICE offers significant advantages over competitive emulators including full-speed emulation, run-time variable watches, trace analysis, complex breakpoints, logic probes, a ruggedized probe interface and long (up to three meters) interconnection cables.

31.8 MPLAB ICD 3 In-Circuit Debugger System

The MPLAB ICD 3 In-Circuit Debugger System is Microchip's most cost-effective, high-speed hardware debugger/programmer for Microchip Flash DSC and MCU devices. It debugs and programs PIC Flash microcontrollers and dsPIC DSCs with the powerful, yet easy-to-use graphical user interface of the MPLAB IDE.

The MPLAB ICD 3 In-Circuit Debugger probe is connected to the design engineer's PC using a high-speed USB 2.0 interface and is connected to the target with a connector compatible with the MPLAB ICD 2 or MPLAB REAL ICE systems (RJ-11). MPLAB ICD 3 supports all MPLAB ICD 2 headers.

31.9 PICkit 3 In-Circuit Debugger/ Programmer

The MPLAB PICkit 3 allows debugging and programming of PIC and dsPIC Flash microcontrollers at a most affordable price point using the powerful graphical user interface of the MPLAB IDE. The MPLAB PICkit 3 is connected to the design engineer's PC using a full-speed USB interface and can be connected to the target via a Microchip debug (RJ-11) connector (compatible with MPLAB ICD 3 and MPLAB REAL ICE). The connector uses two device I/O pins and the Reset line to implement in-circuit debugging and In-Circuit Serial Programming™ (ICSP™).

31.10 MPLAB PM3 Device Programmer

The MPLAB PM3 Device Programmer is a universal, CE compliant device programmer with programmable voltage verification at V_{DDMIN} and V_{DDMAX} for maximum reliability. It features a large LCD display (128 x 64) for menus and error messages, and a modular, detachable socket assembly to support various package types. The ICSP cable assembly is included as a standard item. In Stand-Alone mode, the MPLAB PM3 Device Programmer can read, verify and program PIC devices without a PC connection. It can also set code protection in this mode. The MPLAB PM3 connects to the host PC via an RS-232 or USB cable. The MPLAB PM3 has high-speed communications and optimized algorithms for quick programming of large memory devices, and incorporates an MMC card for file storage and data applications.

31.11 Demonstration/Development Boards, Evaluation Kits, and Starter Kits

A wide variety of demonstration, development and evaluation boards for various PIC MCUs and dsPIC DSCs allows quick application development on fully functional systems. Most boards include prototyping areas for adding custom circuitry and provide application firmware and source code for examination and modification.

The boards support a variety of features, including LEDs, temperature sensors, switches, speakers, RS-232 interfaces, LCD displays, potentiometers and additional EEPROM memory.

The demonstration and development boards can be used in teaching environments, for prototyping custom circuits and for learning about various microcontroller applications.

In addition to the PICDEM™ and dsPICDEM™ demonstration/development board series of circuits, Microchip has a line of evaluation kits and demonstration software for analog filter design, KEELOQ® security ICs, CAN, IrDA®, PowerSmart battery management, SEEVAL® evaluation system, Sigma-Delta ADC, flow rate sensing, plus many more.

Also available are starter kits that contain everything needed to experience the specified device. This usually includes a single application and debug capability, all on one board.

Check the Microchip web page (www.microchip.com) for the complete list of demonstration, development and evaluation kits.

31.12 Third-Party Development Tools

Microchip also offers a great collection of tools from third-party vendors. These tools are carefully selected to offer good value and unique functionality.

- Device Programmers and Gang Programmers from companies, such as SoftLog and CCS
- Software Tools from companies, such as Gimpel and Trace Systems
- Protocol Analyzers from companies, such as Saleae and Total Phase
- Demonstration Boards from companies, such as MikroElektronika, Digilent® and Olimex
- Embedded Ethernet Solutions from companies, such as EZ Web Lynx, WIZnet and IPLogika®

32.0 DC AND AC CHARACTERISTICS GRAPHS AND CHARTS

The graphs and tables provided in this section are for **design guidance** and are **not tested**.

In some graphs or tables, the data presented are **outside specified operating range** (i.e., outside specified V_{DD} range). This is for **information only** and devices are ensured to operate properly only within the specified range.

Unless otherwise noted, all graphs apply to both the L and LF devices.

Note:	The graphs and tables provided following this note are a statistical summary based on a limited number of samples and are provided for informational purposes only. The performance characteristics listed herein are not tested or guaranteed. In some graphs or tables, the data presented may be outside the specified operating range (e.g., outside specified power supply range) and therefore, outside the warranted range.

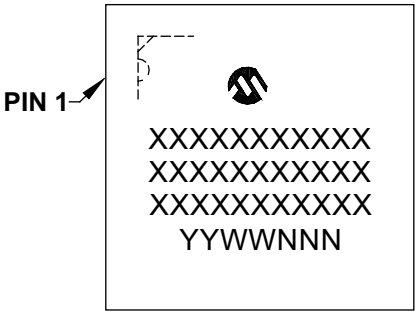
"Typical" represents the mean of the distribution at 25°C. "Maximum", "Max.", "Minimum" or "Min." represents (mean + 3σ) or (mean - 3σ) respectively, where σ is a standard deviation, over each temperature range.

Charts and graphs are not available at this time.

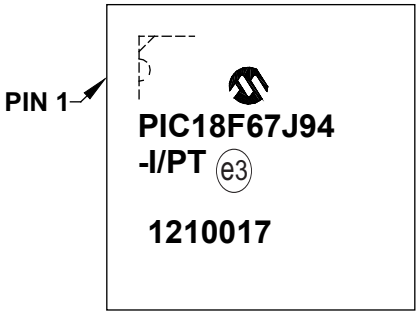
33.0 PACKAGING INFORMATION

33.1 Package Marking Information

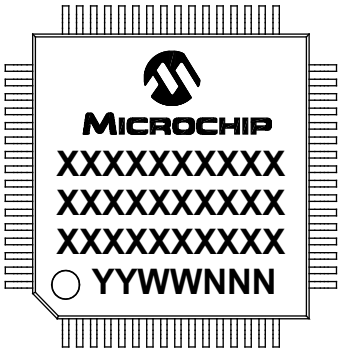
64-Lead QFN (9x9x0.9 mm)



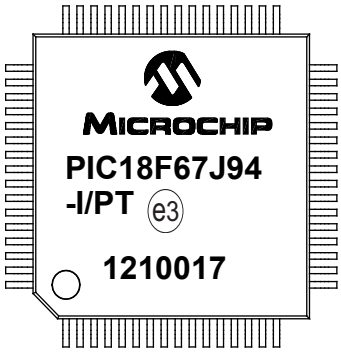
Example



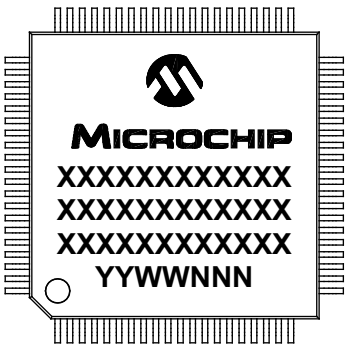
64-Lead TQFP (10x10x1 mm)



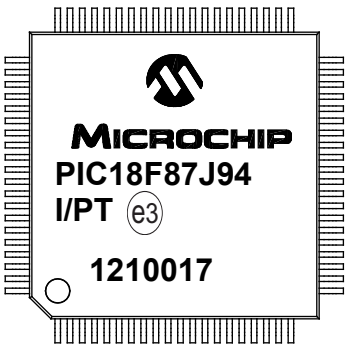
Example



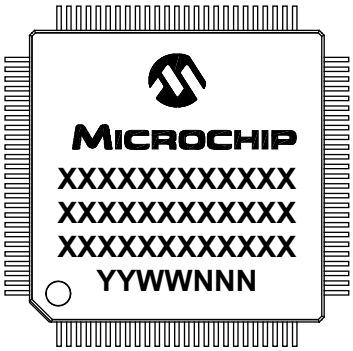
80-Lead TQFP (12x12x1 mm)



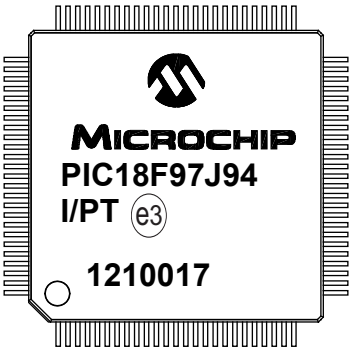
Example



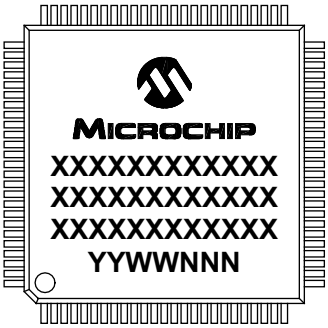
100-Lead TQFP (12x12x1 mm)



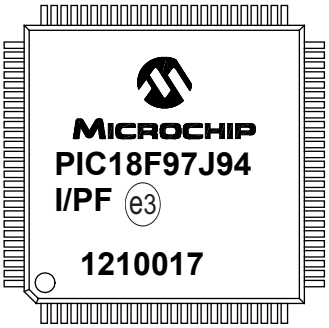
Example



100-Lead TQFP (14x14x1 mm)



Example



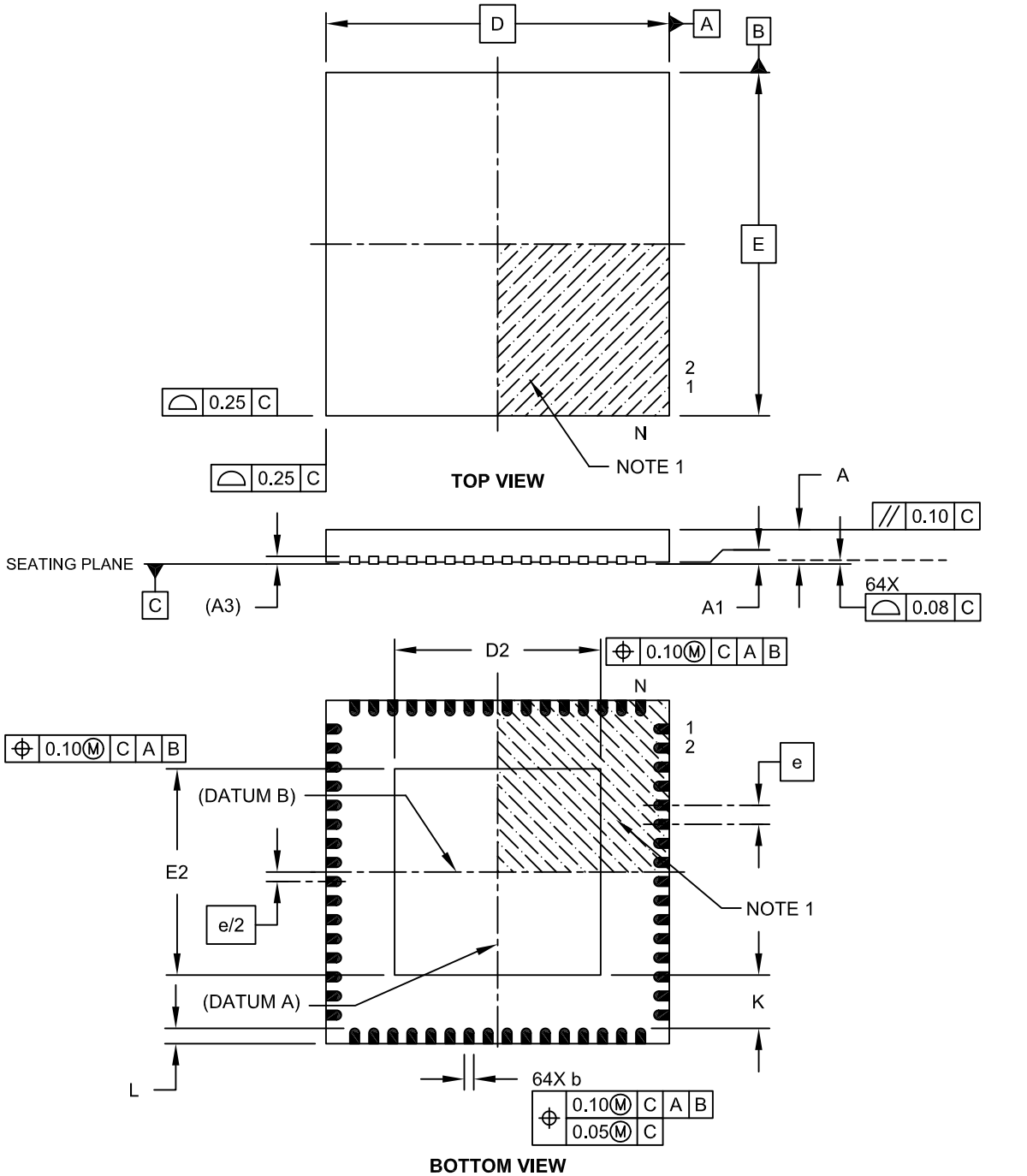
Legend:	XX...X	Customer-specific information
	Y	Year code (last digit of calendar year)
	YY	Year code (last 2 digits of calendar year)
	WW	Week code (week of January 1 is week ‘01’)
	NNN	Alphanumeric traceability code
	(e3)	Pb-free JEDEC® designator for Matte Tin (Sn)
	*	This package is Pb-free. The Pb-free JEDEC® designator ((e3)) can be found on the outer packaging for this package.
Note:	In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.	

33.2 Package Details

The following sections give the technical details of the packages.

64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body with 5.40 x 5.40 Exposed Pad [QFN]

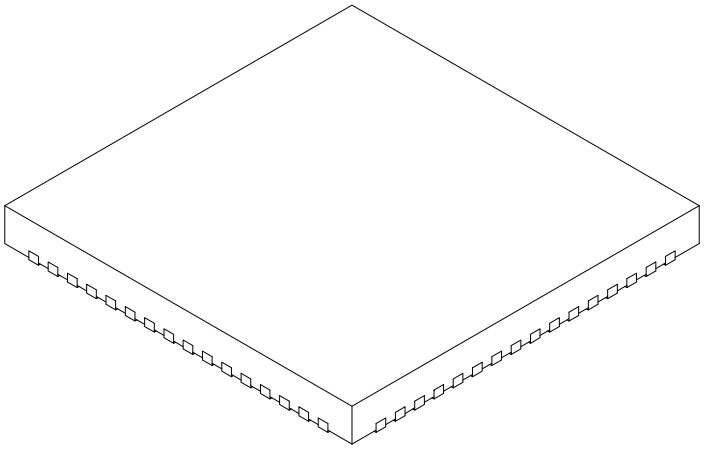
Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Microchip Technology Drawing C04-154A Sheet 1 of 2

64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body with 5.40 x 5.40 Exposed Pad [QFN]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		64	
Pitch	e		0.50 BSC	
Overall Height	A	0.80	0.90	1.00
Standoff	A1	0.00	0.02	0.05
Contact Thickness	A3		0.20 REF	
Overall Width	E		9.00 BSC	
Exposed Pad Width	E2	5.30	5.40	5.50
Overall Length	D		9.00 BSC	
Exposed Pad Length	D2	5.30	5.40	5.50
Contact Width	b	0.20	0.25	0.30
Contact Length	L	0.30	0.40	0.50
Contact-to-Exposed Pad	K	0.20	-	-

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Package is saw singulated.
3. Dimensioning and tolerancing per ASME Y14.5M.

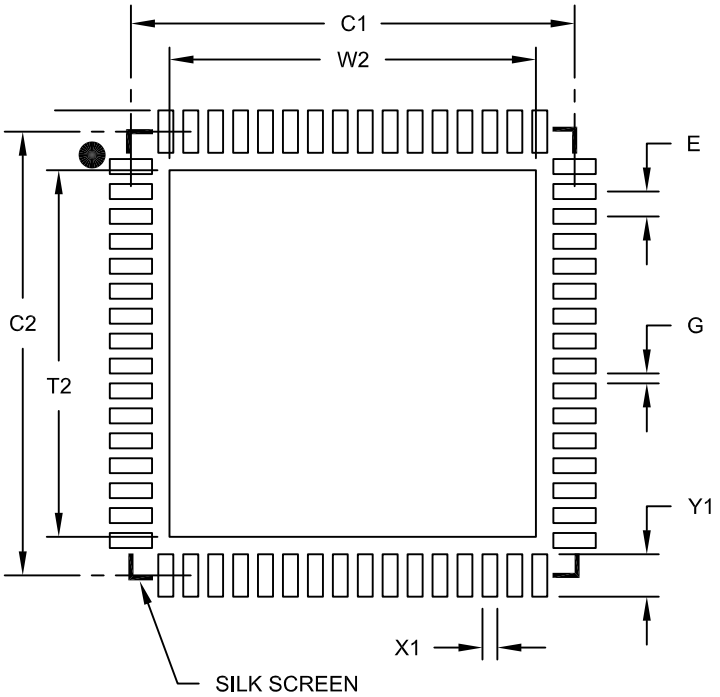
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-154A Sheet 2 of 2

64-Lead Plastic Quad Flat, No Lead Package (MR) – 9x9x0.9 mm Body [QFN]
With 0.40 mm Contact Length

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Optional Center Pad Width	W2			7.35
Optional Center Pad Length	T2			7.35
Contact Pad Spacing	C1		8.90	
Contact Pad Spacing	C2		8.90	
Contact Pad Width (X64)	X1			0.30
Contact Pad Length (X64)	Y1			0.85
Distance Between Pads	G	0.20		

Notes:

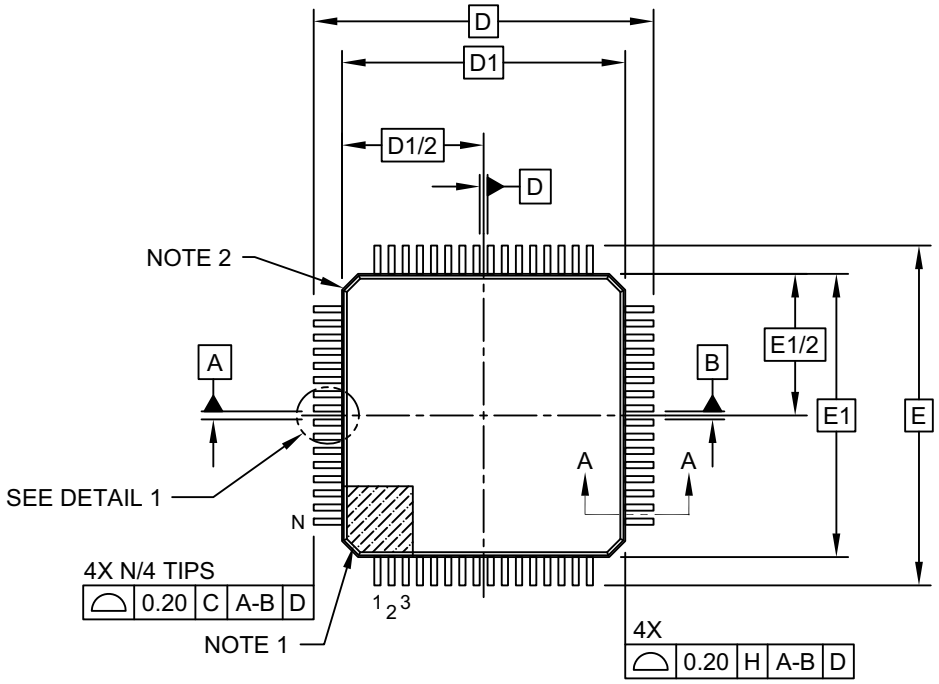
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

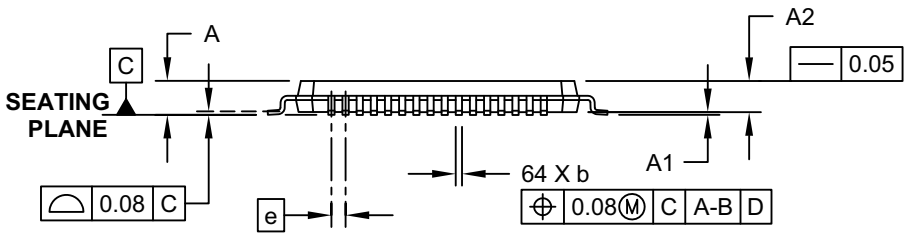
Microchip Technology Drawing No. C04-2149A

64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note:	For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW

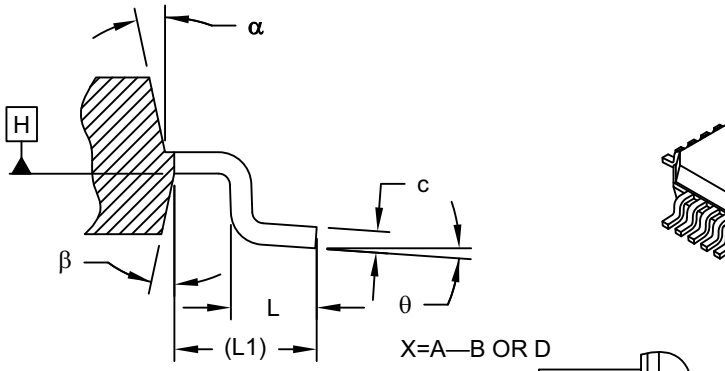


SIDE VIEW

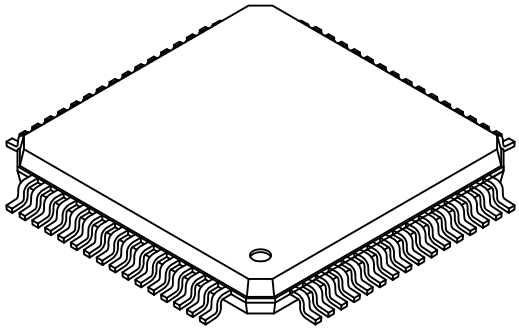
Microchip Technology Drawing C04-085C Sheet 1 of 2

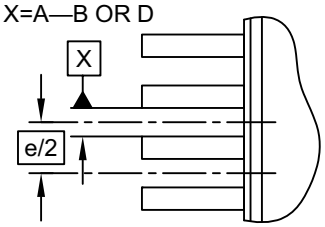
64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



SECTION A-A





DETAIL 1

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		64	
Lead Pitch	e		0.50 BSC	
Overall Height	A	-	-	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	-	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		12.00 BSC	
Overall Length	D		12.00 BSC	
Molded Package Width	E1		10.00 BSC	
Molded Package Length	D1		10.00 BSC	
Lead Thickness	c	0.09	-	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M

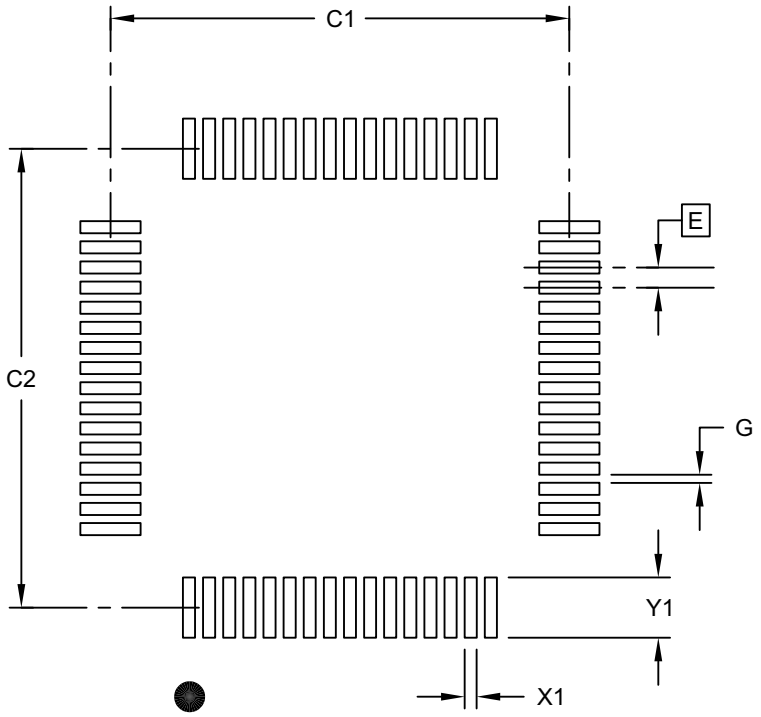
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-085C Sheet 2 of 2

64-Lead Plastic Thin Quad Flatpack (PT)-10x10x1 mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Contact Pad Spacing	C1		11.40	
Contact Pad Spacing	C2		11.40	
Contact Pad Width (X28)	X1			0.30
Contact Pad Length (X28)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

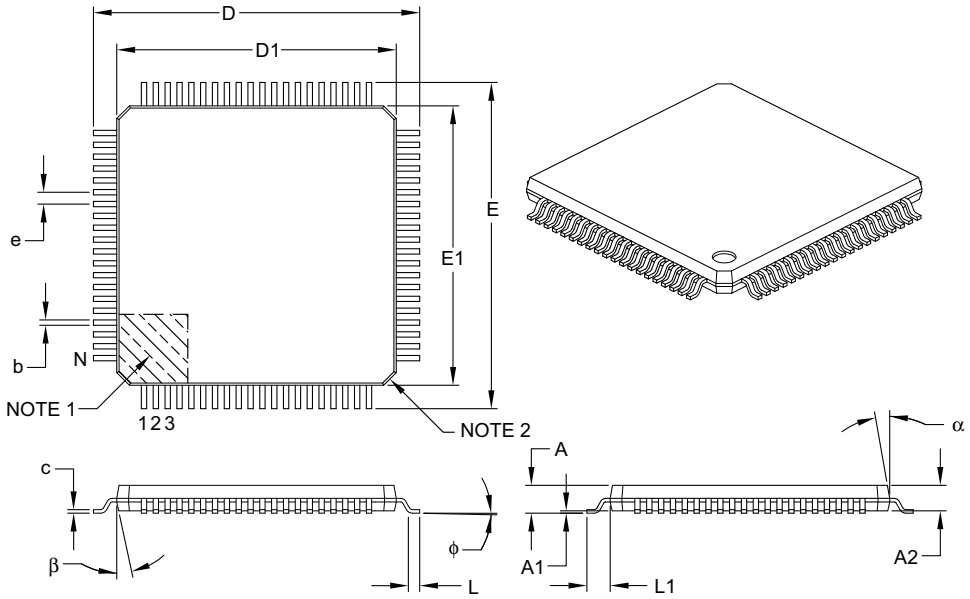
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2085B Sheet 1 of 1

80-Lead Plastic Thin Quad Flatpack (PT) – 12x12x1 mm Body, 2.00 mm [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		80	
Lead Pitch	e		0.50 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		14.00 BSC	
Overall Length	D		14.00 BSC	
Molded Package Width	E1		12.00 BSC	
Molded Package Length	D1		12.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

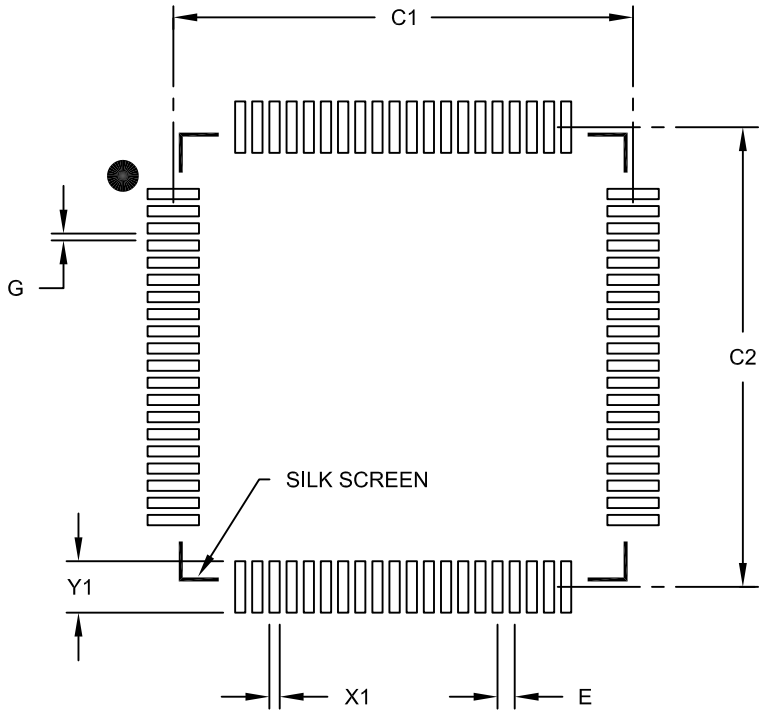
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-092B

80-Lead Plastic Thin Quad Flatpack (PT) - 12x12x1mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Contact Pad Spacing	C1		13.40	
Contact Pad Spacing	C2		13.40	
Contact Pad Width (X80)	X1			0.30
Contact Pad Length (X80)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

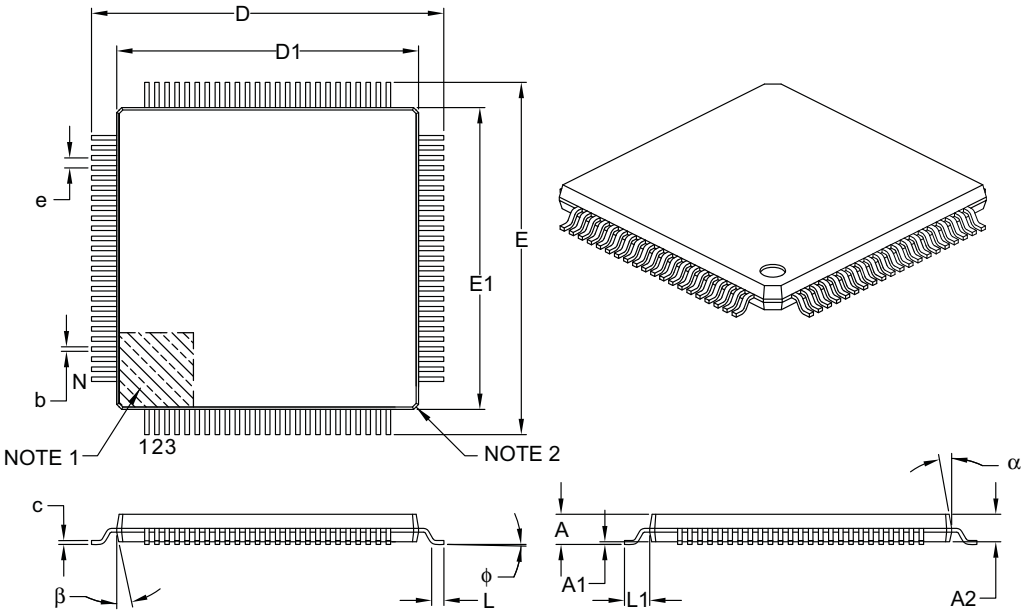
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2092B

100-Lead Plastic Thin Quad Flatpack (PT) – 12x12x1 mm Body, 2.00 mm [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		100	
Lead Pitch	e		0.40 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		14.00 BSC	
Overall Length	D		14.00 BSC	
Molded Package Width	E1		12.00 BSC	
Molded Package Length	D1		12.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.13	0.18	0.23
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

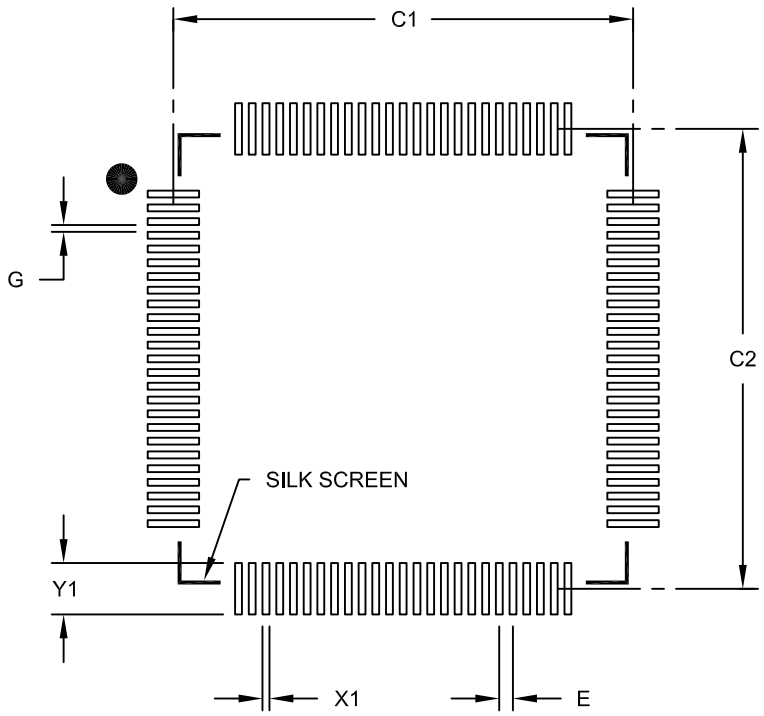
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-100B

100-Lead Plastic Thin Quad Flatpack (PT)-12x12x1mm Body, 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.40 BSC	
Contact Pad Spacing	C1		13.40	
Contact Pad Spacing	C2		13.40	
Contact Pad Width (X100)	X1			0.20
Contact Pad Length (X100)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

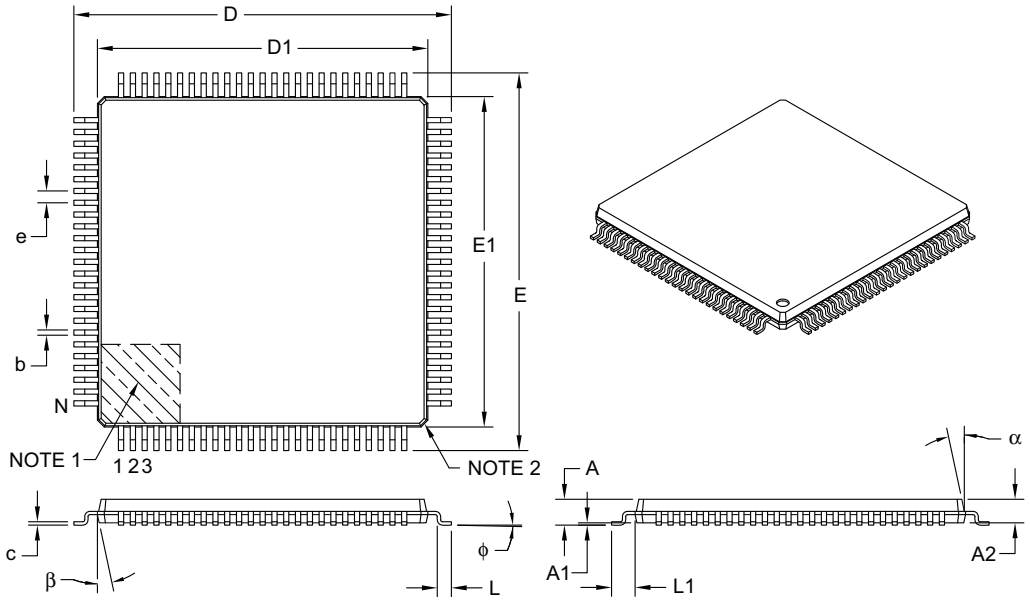
1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2100B

100-Lead Plastic Thin Quad Flatpack (PF) – 14x14x1 mm Body, 2.00 mm [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Number of Leads	N		100	
Lead Pitch	e		0.50 BSC	
Overall Height	A	–	–	1.20
Molded Package Thickness	A2	0.95	1.00	1.05
Standoff	A1	0.05	–	0.15
Foot Length	L	0.45	0.60	0.75
Footprint	L1		1.00 REF	
Foot Angle	ϕ	0°	3.5°	7°
Overall Width	E		16.00 BSC	
Overall Length	D		16.00 BSC	
Molded Package Width	E1		14.00 BSC	
Molded Package Length	D1		14.00 BSC	
Lead Thickness	c	0.09	–	0.20
Lead Width	b	0.17	0.22	0.27
Mold Draft Angle Top	α	11°	12°	13°
Mold Draft Angle Bottom	β	11°	12°	13°

Notes:

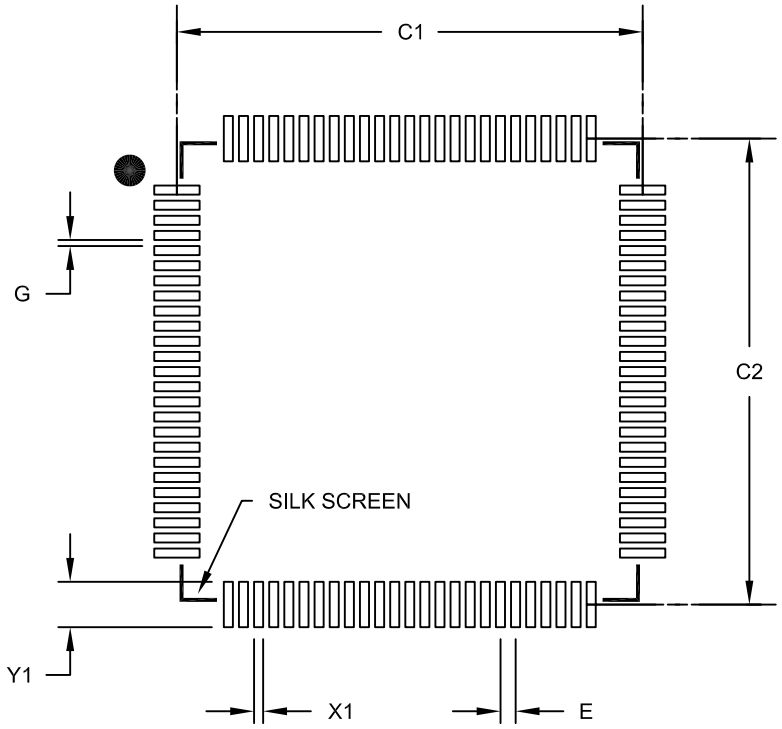
1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Chamfers at corners are optional; size may vary.
3. Dimensions D1 and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

BSC: Basic Dimension. Theoretically exact value shown without tolerances.
REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-110B

100-Lead Plastic Thin Quad Flatpack (PF) - 14x14x1 mm Body 2.00 mm Footprint [TQFP]

Note: For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



RECOMMENDED LAND PATTERN

Units		MILLIMETERS		
Dimension Limits		MIN	NOM	MAX
Contact Pitch	E		0.50 BSC	
Contact Pad Spacing	C1		15.40	
Contact Pad Spacing	C2		15.40	
Contact Pad Width (X100)	X1			0.30
Contact Pad Length (X100)	Y1			1.50
Distance Between Pads	G	0.20		

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M

BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2110B

APPENDIX A: REVISION HISTORY

Revision A (October 2012)

This is the initial release of the document.

Revision B (08/2016)

Updated data sheet to new format. Added corrections as per *PIC18F97J94 Family Silicon Errata and Data Sheet Clarifications* (DS8000551D), as follows: Updated Table 1; Added Table 2, Table 3 and Table 4; Updated Tables 1-4, 3-3, 6-2, 11-3, 11-6, 22-1; Added Table 30-18; Updated Register 4-8; Added Register 22-26; Updated Figures 11-7 and 22-1; Updated Examples 11-6, 22-1 and 22-2; Updated Equation 22-1. Update Packaging Information chapter. Other corrections.

Revision C (08/2016)

Remove Preliminary status from data sheet.

THE MICROCHIP WEBSITE

Microchip provides online support via our website site at www.microchip.com. This website is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the website contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip website at www.microchip.com. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the website at: http://microchip.com/support.

PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, refer to the factory or the listed sales office.

PART NO.	[X](1)	X	/XX	XXX
Device	Tape and Reel Option	Temperature Range	Package	Pattern

Device:	PIC18F97J94, PIC18F96J94, PIC18F95J94, PIC18F87J94, PIC18F86J94, PIC18F85J94, PIC18F67J94, PIC18F66J94, PIC18F65J94 VDD range 2.0 to 3.6V
Tape and Reel Option:	Blank = Standard packaging (tube or tray) T = Tape and Reel(1)
Temperature Range:	I = -40°C to +85°C (Industrial)
Package:	PT = TQFP (Thin Quad Flatpack) PF = TQFP (100-Pin Thin Quad, 14x14x1 Body)
Pattern:	QTP, SQTP, Code or Special Requirements (blank otherwise)

Examples:

a) PIC18F97J94-I/PT = Industrial temp., TQFP package, QTP pattern #301.

b) PIC18F87J94-I/PT = Industrial temp., TQFP package.

Note 1: Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Trademarks

ISBN: 978-1-5224-0896-3

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7828

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Dusseldorf
Tel: 49-2129-3766400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Venice
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Poland - Warsaw
Tel: 48-22-3325737

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

06/23/16

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Microchip:

PIC18F87J94-I/PT PIC18F67J94-I/PT PIC18F85J94-I/PT PIC18F65J94-I/PT PIC18F97J94-I/PF PIC18F97J94-I/PT PIC18F66J94-I/PT PIC18F65J94-I/MR PIC18F66J94-I/MR PIC18F65J94T-I/PT PIC18F67J94T-I/MR PIC18F66J94T-I/PT PIC18F96J94-I/PT PIC18F96J94T-I/PF PIC18F96J94T-I/PT PIC18F97J94T-I/PF PIC18F86J94-I/PT PIC18F97J94T-I/PT PIC18F65J94T-I/MR PIC18F96J94-I/PF PIC18F95J94T-I/PF PIC18F87J94T-I/PT PIC18F67J94T-I/PT PIC18F67J94-I/MR PIC18F95J94T-I/PT PIC18F85J94T-I/PT PIC18F95J94-I/PT PIC18F95J94-I/PF PIC18F86J94T-I/PT PIC18F66J94T-I/MR

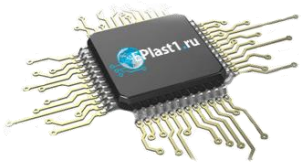
Компания «ЭлектроПласт» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Оперативные поставки широкого спектра электронных компонентов отечественного и импортного производства напрямую от производителей и с крупнейших мировых складов;
- Поставка более 17-ти миллионов наименований электронных компонентов;
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- Лицензия ФСБ на осуществление работ с использованием сведений, составляющих государственную тайну;
- Поставка специализированных компонентов (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Aeroflex, Peregrine, Syfer, Eurofarad, Texas Instrument, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits,General Dynamics и др.);

Помимо этого, одним из направлений компании «ЭлектроПласт» является направление «Источники питания». Мы предлагаем Вам помощь Конструкторского отдела:

- Подбор оптимального решения, техническое обоснование при выборе компонента;
- Подбор аналогов;
- Консультации по применению компонента;
- Поставка образцов и прототипов;
- Техническая поддержка проекта;
- Защита от снятия компонента с производства.



Как с нами связаться

Телефон: 8 (812) 309 58 32 (многоканальный)
Факс: 8 (812) 320-02-42
Электронная почта: org@eplast1.ru
Адрес: 198099, г. Санкт-Петербург, ул. Калинина, дом 2, корпус 4, литера А.