SC16IS741A

Single UART with I²C-bus/SPI interface, 64 bytes of transmit and receive FIFOs, IrDA SIR built-in support

Rev. 1 — 18 March 2013 Product data sheet

1. General description

The SC16IS741A[1] is a slave I²C-bus/SPI interface to a single-channel high performance UART. It offers data rates up to 5 Mbit/s and guarantees low operating and sleeping current. The device comes in the TSSOP16 package, which makes it ideally suitable for handheld, battery operated applications. This device enables seamless protocol conversion from I²C-bus or SPI to and RS-232/RS-485 and are fully bidirectional.

The SC16IS741A's internal register set is backward-compatible with the widely used and widely popular 16C450. This allows the software to be easily written or ported from another platform.

The SC16IS741A also provides additional advanced features such as auto hardware and software flow control, automatic RS-485 support, and software reset. This allows the software to reset the UART at any moment, independent of the hardware reset signal.

2. Features and benefits

2.1 General features

- Single full-duplex UART
- Selectable I²C-bus or SPI interface
- 3.3 V or 2.5 V operation
- Industrial temperature range: −40 °C to +95 °C
- 64 bytes FIFO (transmitter and receiver)
- Fully compatible with industrial standard 16C450 and equivalent
- Baud rates up to 5 Mbit/s in 16× clock mode
- Auto hardware flow control using $\overline{\text{RTS}}$/$\overline{\text{CTS}}$
- Auto software flow control with programmable Xon/Xoff characters
- Single or double Xon/Xoff characters
- Automatic RS-485 support (automatic slave address detection)
- RS-485 driver direction control via $\overline{\text{RTS}}$ signal
- RS-485 driver direction control inversion
- Built-in IrDA encoder and decoder interface
- Software reset
- Transmitter and receiver can be enabled/disabled independent of each other
- Receive and Transmit FIFO levels
- Programmable special character detection

1. See Section 10.3.1 for the description of the differences between SC16IS741 and SC16IS741A.

- Fully programmable character formatting
  - 5-bit, 6-bit, 7-bit or 8-bit character
  - Even, odd, or no parity
  - 1, $1\frac{1}{2}$, or 2 stop bits
- Line break generation and detection
- Internal Loopback mode
- Sleep current less than 30 μA at 3.3 V
- Industrial and commercial temperature ranges
- Available in the TSSOP16 package

2.2 I²C-bus features

- Noise filter on SCL/SDA inputs
- 400 kbit/s maximum speed
- Compliant with I²C-bus fast speed
- Slave mode only

2.3 SPI features

- Slave mode only
- SPI Mode 0

3. Applications

- Factory automation and process control
- Portable and battery operated devices
- Cellular data devices

4. Ordering information

Table 1. Ordering information

Type number	Topside marking	Package		
		Name	Description	Version
SC16IS741AIPW	IS741A	TSSOP16	plastic thin shrink small outline package; 16 leads; body width 4.4 mm	SOT403-1

4.1 Ordering options

Table 2. Ordering options

Type number	Orderable part number	Package	Packing method	Minimum order quantity	Temperature
SC16IS741AIPW	SC16IS741AIPWJ	TSSOP16	Reel 13” Q1/T1 *standard mark SMD	2500	V_{DD} = 2.5 V ± 0.2 V, T_{amb} = −40 °C to +85 °C
					V_{DD} = 3.3 V ± 0.3 V, T_{amb} = −40 °C to +95 °C

5. Block diagram

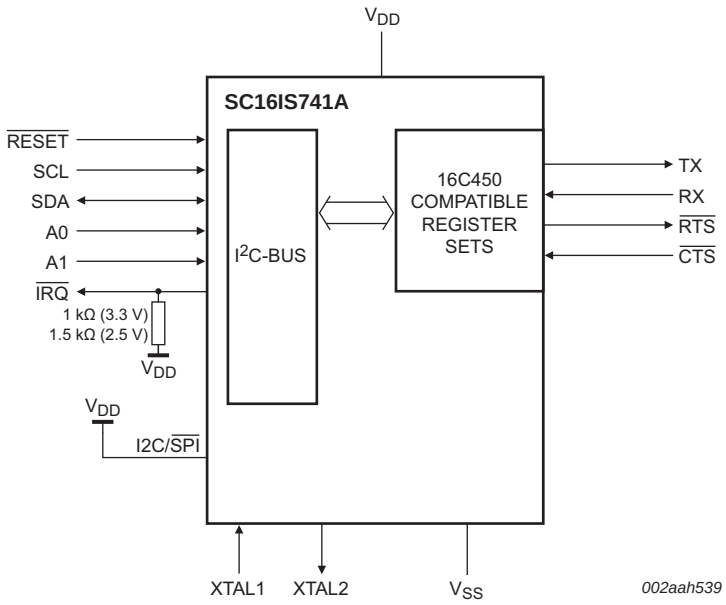


Fig 1. Block diagram of SC16IS741A I²C-bus interface

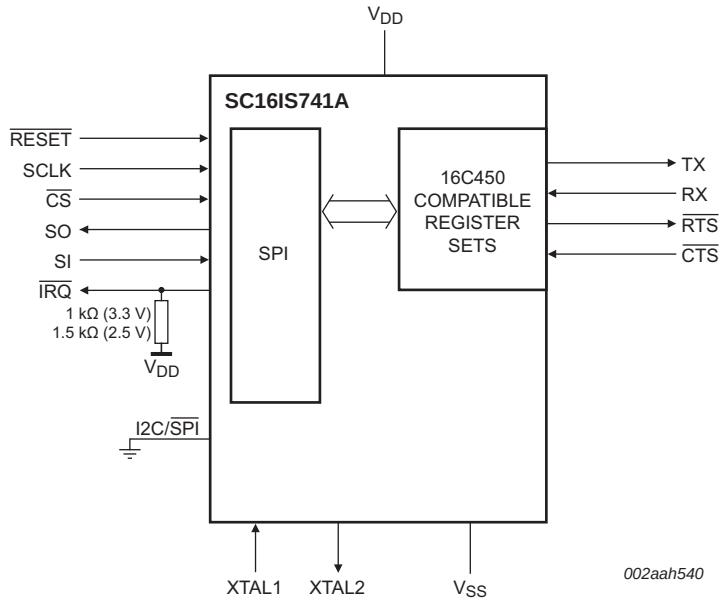
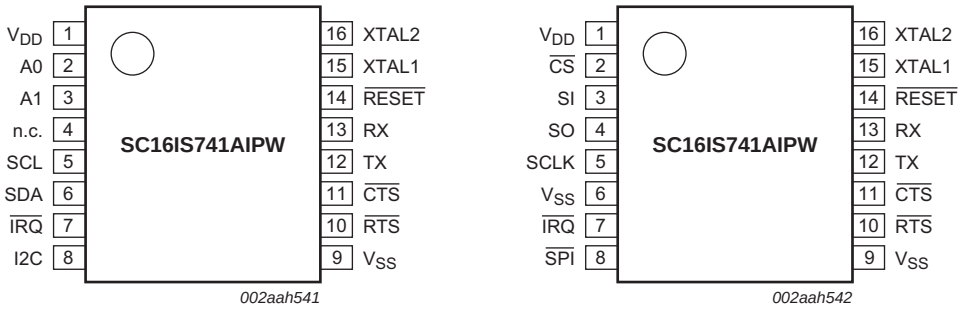


Fig 2. Block diagram of SC16IS741A SPI interface

6. Pinning information

6.1 Pinning



a. I²C-bus interface

b. SPI interface

Fig 3. Pin configuration for TSSOP16

6.2 Pin description

Table 3. Pin description

Symbol	Pin	Type	Description
V_{DD}	1	-	power supply
$\overline{CS}$/A0	2	I	SPI chip select or I²C-bus device address select A0. If SPI configuration is selected by I2C/$\overline{SPI}$ pin, this pin is the SPI chip select pin (Schmitt-trigger, active LOW). If I²C-bus configuration is selected by I2C/$\overline{SPI}$ pin, this pin along with A1 pin allows user to change the device's base address.
SI/A1	3	I	SPI data input pin or I²C-bus device address select A1. If SPI configuration is selected by I2C/$\overline{SPI}$ pin, this is the SPI data input pin. If I²C-bus configuration is selected by I2C/$\overline{SPI}$ pin, this pin along with A0 pin allows user to change the device's base address. To select the device address, please refer to Table 29.
SO	4	O	SPI data output pin. If SPI configuration is selected by I2C/$\overline{SPI}$ pin, this is a 3-stateable output pin. If I²C-bus configuration is selected by I2C/$\overline{SPI}$ pin, this pin function is undefined and must be left as n.c. (not connected).
SCL/SCLK	5	I	I²C-bus or SPI input clock.
SDA	6	I/O	I²C-bus data input/output, open-drain if I²C-bus configuration is selected by I2C/$\overline{SPI}$ pin. If SPI configuration is selected then this pin is an undefined pin and must be connected to V_{SS}.
$\overline{IRQ}$	7	O	Interrupt (open-drain, active LOW). Interrupt is enabled when interrupt sources are enabled in the Interrupt Enable Register (IER). Interrupt conditions include: change of state of the input pins, receiver errors, available receiver buffer data, available transmit buffer space, or when a modem status flag is detected. An external resistor (1 kΩ for 3.3 V, 1.5 kΩ for 2.5 V) must be connected between this pin and V_{DD}.
I2C/$\overline{SPI}$	8	I	I²C-bus or SPI interface select. I²C-bus interface is selected if this pin is at logic HIGH. SPI interface is selected if this pin is at logic LOW.

Table 3. Pin description *...continued*

Symbol	Pin	Type	Description
V_{SS}	9	-	ground
$\overline{RTS}$	10	O	UART request to send (active LOW). A logic 0 on the $\overline{RTS}$ pin indicates the transmitter has data ready and waiting to send. Writing a logic 1 in the modem control register MCR[1] will set this pin to a logic 0, indicating data is available. After a reset this pin is set to a logic 1. This pin only affects the transmit and receive operations when auto $\overline{RTS}$ function is enabled via the Enhanced Feature Register (EFR[6]) for hardware flow control operation.
$\overline{CTS}$	11	I	UART clear to send (active LOW). A logic 0 (LOW) on the $\overline{CTS}$ pin indicates the modem or data set is ready to accept transmit data from the SC16IS741A. Status can be tested by reading MSR[4]. This pin only affects the transmit and receive operations when auto $\overline{CTS}$ function is enabled via the Enhanced Feature Register EFR[7] for hardware flow control operation.
TX	12	O	UART transmitter output. During the local Loopback mode, the TX output pin is disabled and TX data is internally connected to the UART RX input.
RX	13	I	UART receiver input. During the local Loopback mode, the RX input pin is disabled and TX data is connected to the UART RX input internally.
$\overline{RESET}$	14	I	device hardware reset (active LOW)[1]
XTAL1	15	I	Crystal input or external clock input. Functions as a crystal input or as an external clock input. A crystal can be connected between XTAL1 and XTAL2 to form an internal oscillator circuit (see Figure 11). Alternatively, an external clock can be connected to this pin.
XTAL2	16	O	Crystal output or clock output. (See also XTAL1.) XTAL2 is used as a crystal oscillator output.

[1] See Section 7.4 “Hardware reset, Power-On Reset (POR) and software reset”

7. Functional description

The UART will perform serial-to-I²C conversion on data characters received from peripheral devices or modems, and I²C-to-serial conversion on data characters transmitted by the host. The complete status the SC16IS741A UART can be read at any time during functional operation by the host.

The SC16IS741A can be placed in an alternate mode (FIFO mode) relieving the host of excessive software overhead by buffering received/transmitted characters. Both the receiver and transmitter FIFOs can store up to 64 characters (including three additional bits of error status per character for the receiver FIFO) and have selectable or programmable trigger levels.

The SC16IS741A has selectable hardware flow control and software flow control. Hardware flow control significantly reduces software overhead and increases system efficiency by automatically controlling serial data flow using the $\overline{RTS}$ output and $\overline{CTS}$ input signals. Software flow control automatically controls data flow by using programmable Xon/Xoff characters.

The UART includes a programmable baud rate generator that can divide the timing reference clock input by a divisor between 1 and (2^{16} – 1).

7.1 Trigger levels

The SC16IS741A provides independently selectable and programmable trigger levels for both receiver and transmitter interrupt generation. After reset, both transmitter and receiver FIFOs are disabled and so, in effect, the trigger level is the default value of one character. The selectable trigger levels are available via the FCR. The programmable trigger levels are available via the TLR. If TLR bits are cleared then selectable trigger level in FCR is used. If TLR bits are not cleared then programmable trigger level in TLR is used.

7.2 Hardware flow control

Hardware flow control is comprised of auto $\overline{\text{CTS}}$ and auto $\overline{\text{RTS}}$ (see Figure 4). Auto $\overline{\text{CTS}}$ and auto $\overline{\text{RTS}}$ can be enabled/disabled independently by programming EFR[7:6].

With auto $\overline{\text{CTS}}$, $\overline{\text{CTS}}$ must be active before the UART can transmit data.

Auto $\overline{\text{RTS}}$ only activates the $\overline{\text{RTS}}$ output when there is enough room in the FIFO to receive data and de-activates the $\overline{\text{RTS}}$ output when the RX FIFO is sufficiently full. The halt and resume trigger levels in the TCR determine the levels at which $\overline{\text{RTS}}$ is activated/deactivated. If TCR bits are cleared then selectable trigger levels in FCR are used in place of TCR.

If both auto $\overline{\text{CTS}}$ and auto $\overline{\text{RTS}}$ are enabled, when $\overline{\text{RTS}}$ is connected to $\overline{\text{CTS}}$, data transmission does not occur unless the receiver FIFO has empty space. Thus, overrun errors are eliminated during hardware flow control. If not enabled, overrun errors occur if the transmit data rate exceeds the receive FIFO servicing latency.

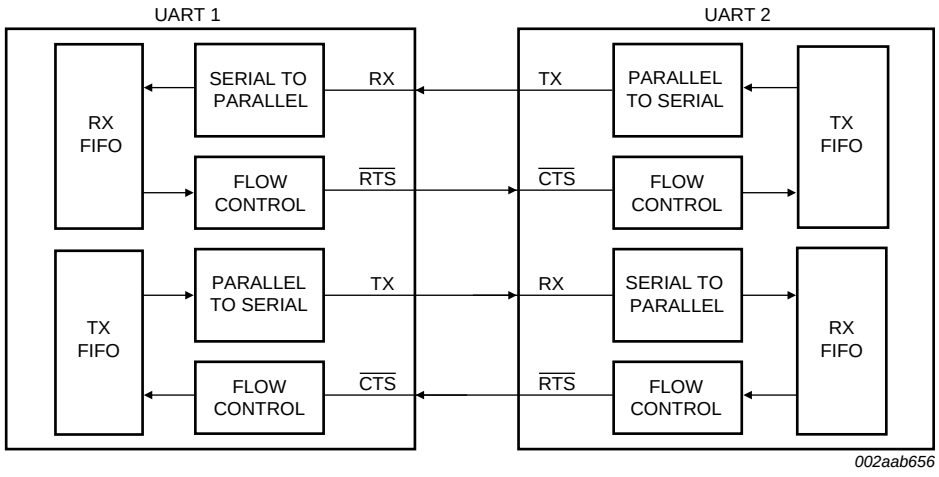
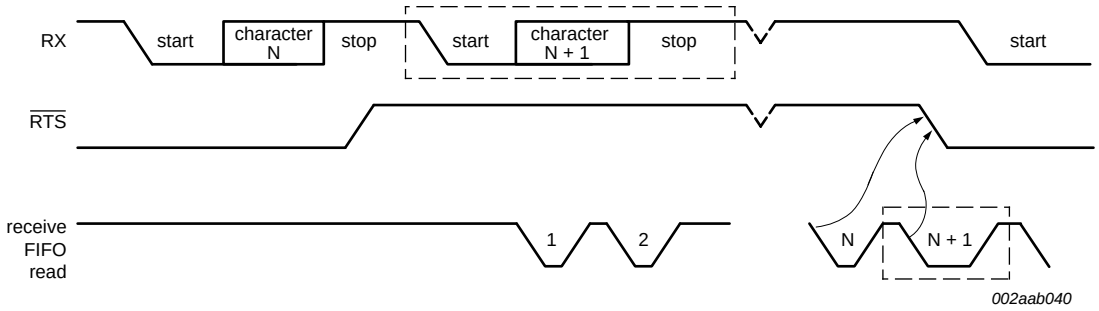


Fig 4. Autoflow control (auto $\overline{\text{RTS}}$ and auto $\overline{\text{CTS}}$) example

7.2.1 Auto $\overline{\text{RTS}}$

Figure 5 shows $\overline{\text{RTS}}$ functional timing. The receiver FIFO trigger levels used in auto $\overline{\text{RTS}}$ are stored in the TCR or FCR. $\overline{\text{RTS}}$ is active if the RX FIFO level is below the halt trigger level in TCR[3:0]. When the receiver FIFO halt trigger level is reached, $\overline{\text{RTS}}$ is de-asserted. The sending device (for example, another UART) may send an additional character after the trigger level is reached (assuming the sending UART has another character to send) because it may not recognize the de-assertion of $\overline{\text{RTS}}$ until it has begun sending the additional character. RTS is automatically re-asserted once the receiver FIFO reaches the resume trigger level programmed via TCR[7:4]. This re-assertion allows the sending device to resume transmission.



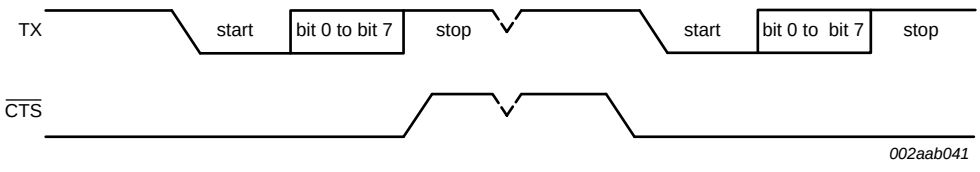
(1) N = receiver FIFO trigger level.

(2) The two blocks in dashed lines cover the case where an additional character is sent, as described in Section 7.2.1

Fig 5. $\overline{\text{RTS}}$ functional timing

7.2.2 Auto $\overline{\text{CTS}}$

Figure 6 shows $\overline{\text{CTS}}$ functional timing. The transmitter circuitry checks $\overline{\text{CTS}}$ before sending the next data byte. When $\overline{\text{CTS}}$ is active, the transmitter sends the next byte. To stop the transmitter from sending the following byte, $\overline{\text{CTS}}$ must be de-asserted before the middle of the last stop bit that is currently being sent. The auto $\overline{\text{CTS}}$ function reduces interrupts to the host system. When flow control is enabled, $\overline{\text{CTS}}$ level changes do not trigger host interrupts because the device automatically controls its own transmitter. Without auto $\overline{\text{CTS}}$, the transmitter sends any data present in the transmit FIFO and a receiver overrun error may result.



(1) When $\overline{\text{CTS}}$ is LOW, the transmitter keeps sending serial data out.

(2) When $\overline{\text{CTS}}$ goes HIGH before the middle of the last stop bit of the current character, the transmitter finishes sending the current character, but it does not send the next character.

(3) When $\overline{\text{CTS}}$ goes from HIGH to LOW, the transmitter begins sending data again.

Fig 6. $\overline{\text{CTS}}$ functional timing

7.3 Software flow control

Software flow control is enabled through the enhanced feature register and the Modem Control Register. Different combinations of software flow control can be enabled by setting different combinations of EFR[3:0]. Table 4 shows software flow control options.

Table 4. Software flow control options (EFR[3:0])

EFR[3]	EFR[2]	EFR[1]	EFR[0]	TX, RX software flow control
0	0	X	X	no transmit flow control
1	0	X	X	transmit Xon1, Xoff1
0	1	X	X	transmit Xon2, Xoff2
1	1	X	X	transmit Xon1 and Xon2, Xoff1 and Xoff2
X	X	0	0	no receive flow control
X	X	1	0	receiver compares Xon1, Xoff1
X	X	0	1	receiver compares Xon2, Xoff2
1	0	1	1	transmit Xon1, Xoff1 receiver compares Xon1 or Xon2, Xoff1 or Xoff2
0	1	1	1	transmit Xon2, Xoff2 receiver compares Xon1 or Xon2, Xoff1 or Xoff2
1	1	1	1	transmit Xon1 and Xon2, Xoff1 and Xoff2 receiver compares Xon1 and Xon2, Xoff1 and Xoff2
0	0	1	1	no transmit flow control receiver compares Xon1 and Xon2, Xoff1 and Xoff2

There are two other enhanced features relating to software flow control:

- **Xon Any function (MCR[5]):** Receiving any character will resume operation after recognizing the Xoff character. It is possible that an Xon1 character is recognized as an Xon Any character, which could cause an Xon2 character to be written to the RX FIFO.
- **Special character (EFR[5]):** Incoming data is compared to Xoff2. Detection of the special character sets the Xoff interrupt (IIR[4]) but does not halt transmission. The Xoff interrupt is cleared by a read of the IIR. The special character is transferred to the RX FIFO.

7.3.1 RX

When software flow control operation is enabled, the SC16IS741A will compare incoming data with Xoff1/Xoff2 programmed characters (in certain cases, Xoff1 and Xoff2 must be received sequentially). When the correct Xoff characters are received, transmission is halted after completing transmission of the current character. Xoff detection also sets IIR[4] (if enabled via IER[5]) and causes $\overline{\text{IRQ}}$ to go LOW.

To resume transmission, an Xon1/Xon2 character must be received (in certain cases Xon1 and Xon2 must be received sequentially). When the correct Xon characters are received, IIR[4] is cleared, and the Xoff interrupt disappears.

7.3.2 TX

Xoff1/Xoff2 character is transmitted when the RX FIFO has passed the HALT trigger level programmed in TCR[3:0] or the selectable trigger level in FCR[7:6]

Xon1/Xoff2 character is transmitted when the RX FIFO reaches the RESUME trigger level programmed in TCR[7:4] or RX FIFO falls below the lower selectable trigger level in FCR[7:6].

The transmission of Xoff/Xon(s) follows the exact same protocol as transmission of an ordinary character from the FIFO. This means that even if the word length is set to be 5, 6, or 7 bits, then the 5, 6, or 7 least significant bits of XOFF1/XOFF2 or XON1/XON2 will be transmitted. (Note that the transmission of 5, 6, or 7 bits of a character is seldom done, but this functionality is included to maintain compatibility with earlier designs.)

It is assumed that software flow control and hardware flow control will never be enabled simultaneously. Figure 7 shows an example of software flow control.

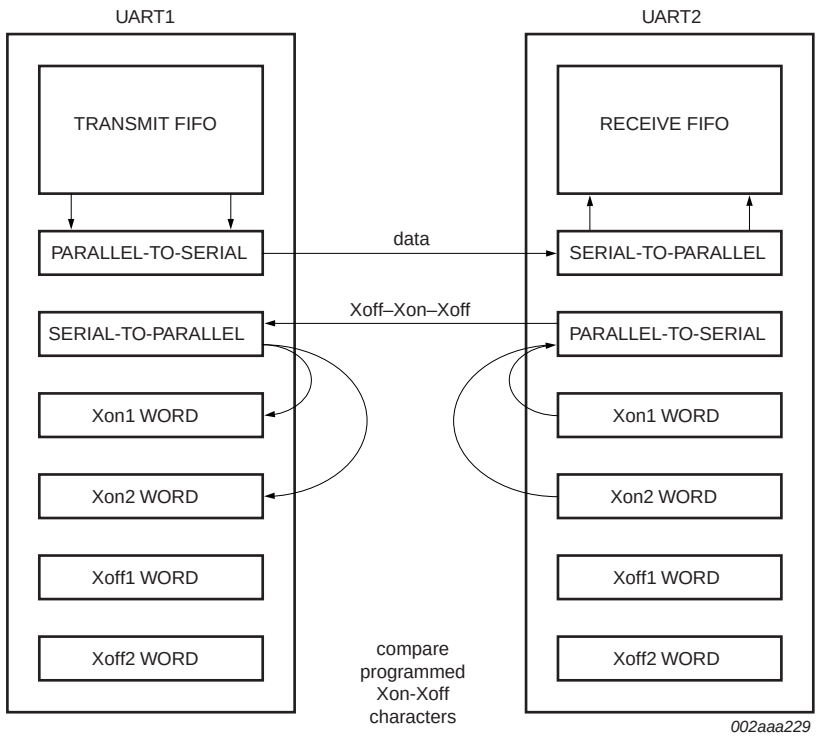


Fig 7. Example of software flow control

7.4 Hardware reset, Power-On Reset (POR) and software reset

These three reset methods are identical and will reset the internal registers as indicated in Table 5.

Table 5 summarizes the state of register.

Table 5. Register reset[1]

Register	Reset state
Interrupt Enable Register	all bits cleared
Interrupt Identification Register	bit 0 is set; all other bits cleared
FIFO Control Register	all bits cleared
Line Control Register	reset to 0001 1101 (0x1D)
Modem Control Register	all bits cleared
Line Status Register	bit 5 and bit 6 set; all other bits cleared
Modem Status Register	bits 0:3 cleared; bits 4:7 input signals
Enhanced Feature Register	all bits cleared
Receiver Holding Register	pointer logic cleared
Transmitter Holding Register	pointer logic cleared
Transmission Control Register	all bits cleared.
Trigger Level Register	all bits cleared.
Transmit FIFO level	reset to 0100 0000 (0x40)
Receive FIFO level	all bits cleared
Extra Feature Register	all bits cleared

[1] Registers DLL, DLH, SPR, XON1, XON2, XOFF1, XOFF2 are not reset by the top-level reset signal $\overline{\text{RESET}}$, POR or Software Reset, that is, they hold their initialization values during reset.

Table 6 summarizes the state of registers after reset.

Table 6. Output signals after reset

Signal	Reset state
TX	HIGH
$\overline{\text{RTS}}$	HIGH
$\overline{\text{IRQ}}$	HIGH by external pull-up

7.5 Interrupts

The SC16IS741A has interrupt generation and prioritization capability. The Interrupt Enable Register (IER) enables each of the interrupts and the $\overline{IRQ}$ signal in response to an interrupt generation. When an interrupt is generated, the IIR indicates that an interrupt is pending and provides the type of interrupt through IIR[5:0]. Table 7 summarizes the interrupt control functions.

Table 7. Summary of interrupt control functions

IIR[5:0]	Priority level	Interrupt type	Interrupt source
00 0001	none	none	none
00 0110	1	receiver line status	OE, FE, PE, or BI errors occur in characters in the RX FIFO
00 1100	2	RX time-out	Stale data in RX FIFO
00 0100	2	RHR interrupt	Receive data ready (FIFO disable) or RX FIFO above trigger level (FIFO enable)
00 0010	3	THR interrupt	Transmit FIFO empty (FIFO disable) or TX FIFO passes above trigger level (FIFO enable)
00 0000	4	Modem status	Change of state of modem input pins
01 0000	6	Xoff interrupt	Receive Xoff character(s)/ special character
10 0000	7	$\overline{CTS}$, $\overline{RTS}$	$\overline{RTS}$ pin or $\overline{CTS}$ pin change state from active (LOW) to inactive (HIGH)

It is important to note that for the framing error, parity error, and break conditions, LSR[7] generates the interrupt. LSR[7] is set when there is an error anywhere in the RX FIFO, and is cleared only when there are no more errors remaining in the FIFO. LSR[4:2] always represent the error status for the received character at the top of the RX FIFO. Reading the RX FIFO updates LSR[4:2] to the appropriate status for the new character at the top of the FIFO. If the RX FIFO is empty, then LSR[4:2] are all zeros.

For the Xoff interrupt, if an Xoff flow character detection caused the interrupt, the interrupt is cleared by an Xon flow character detection. If a special character detection caused the interrupt, the interrupt is cleared by a read of the IIR.

7.5.1 Interrupt mode operation

In Interrupt mode (if any bit of IER[3:0] is 1) the host is informed of the status of the receiver and transmitter by an interrupt signal, $\overline{IRQ}$. Therefore, it is not necessary to continuously poll the Line Status Register (LSR) to see if any interrupt needs to be serviced. Figure 8 shows Interrupt mode operation.

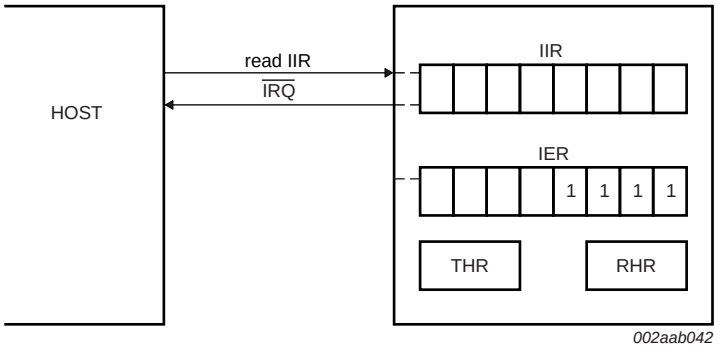


Fig 8. Interrupt mode operation

7.5.2 Polled mode operation

In Polled mode (IER[3:0] = 0000) the status of the receiver and transmitter can be checked by polling the Line Status Register (LSR). This mode is an alternative to the FIFO Interrupt mode of operation where the status of the receiver and transmitter is automatically known by means of interrupts sent to the CPU. Figure 9 shows FIFO Polled mode operation.

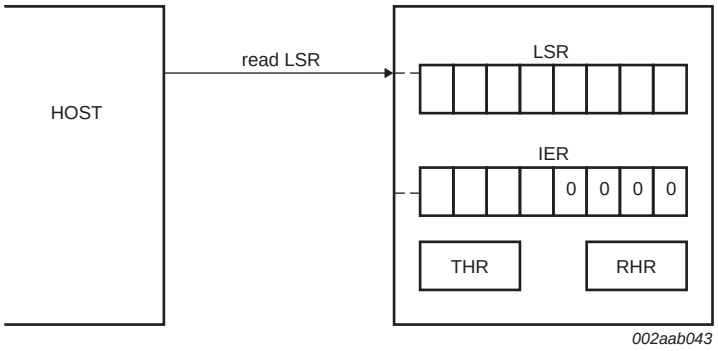


Fig 9. FIFO Polled mode operation

7.6 Sleep mode

Sleep mode is an enhanced feature of the SC16IS741A UART. It is enabled when EFR[4], the enhanced functions bit, is set and when IER[4] is set. Sleep mode is entered when:

- The serial data input line, RX, is idle (see Section 7.7 “Break and time-out conditions”).
- The TX FIFO and TX shift register are empty.
- There are no interrupts pending except THR.

Remark: Sleep mode will **not** be entered if there is data in the RX FIFO.

In Sleep mode, the clock to the UART is stopped. Since most registers are clocked using these clocks, the power consumption is greatly reduced. The UART will wake up when any change is detected on the RX line, when there is any change in the state of the modem input pins, or if data is written to the TX FIFO.

Remark: Writing to the divisor latches, DLL and DLH, to set the baud clock, must not be done during Sleep mode. Therefore, it is advisable to disable Sleep mode using IER[4] before writing to DLL or DLH.

7.7 Break and time-out conditions

When the UART receives a number of characters and these data are not enough to set off the receive interrupt (because they do not reach the receive trigger level), the UART will generate a time-out interrupt instead, 4 character times after the last character is received. The time-out counter will be reset at the center of each stop bit received or each time the receive FIFO is read.

A break condition is detected when the RX pin is pulled LOW for a duration longer than the time it takes to send a complete character plus Start, Stop and Parity bits. A break condition can be sent by setting LCR[6]. When this happens the TX pin will be pulled LOW until LSR[6] is cleared by the software.

7.8 Programmable baud rate generator

The SC16IS741A UART contains a programmable baud rate generator that takes any clock input and divides it by a divisor in the range between 1 and (2^{16} – 1). An additional divide-by-4 prescaler is also available and can be selected by MCR[7], as shown in Figure 10. The output frequency of the baud rate generator is 16 times the baud rate. The formula for the divisor is given in Equation 1:

$$divisor = \frac{\left(\frac{XTAL1\ crystal\ input\ frequency}{prescaler}\right)}{desired\ baud\ rate \times 16} \quad (1)$$

where:

prescaler = 1, when MCR[7] is set to ‘0’ after reset (divide-by-1 clock selected)

prescaler = 4, when MCR[7] is set to ‘1’ after reset (divide-by-4 clock selected).

Remark: The default value of prescaler after reset is divide-by-1.

Figure 10 shows the internal prescaler and baud rate generator circuitry.

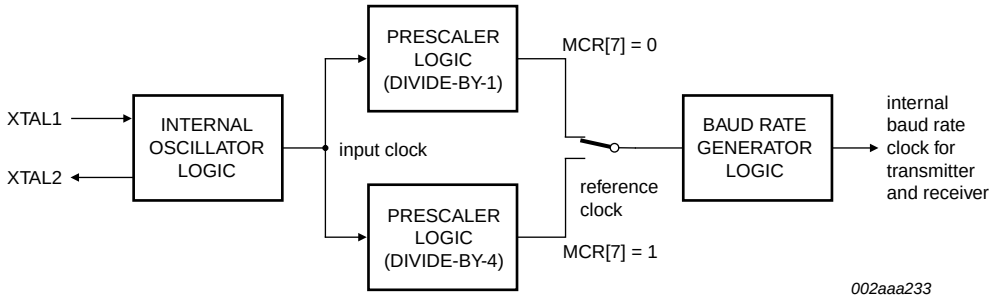


Fig 10. Prescaler and baud rate generator block diagram

DLL and DLH must be written to in order to program the baud rate. DLL and DLH are the least significant and most significant byte of the baud rate divisor. If DLL and DLH are both zero, the UART is effectively disabled, as no baud clock will be generated.

Remark: The programmable baud rate generator is provided to select both the transmit and receive clock rates.

Table 8 and Table 9 show the baud rate and divisor correlation for crystal with frequency 1.8432 MHz and 3.072 MHz, respectively.

Figure 11 shows the crystal clock circuit reference.

Table 8. Baud rates using a 1.8432 MHz crystal

Desired baud rate	Divisor used to generate 16× clock	Percent error difference between desired and actual
50	2304	0
75	1536	0
110	1047	0.026
134.5	857	0.058
150	768	0
300	384	0
600	192	0
1200	96	0
1800	64	0
2000	58	0.69
2400	48	0
3600	32	0
4800	24	0
7200	16	0
9600	12	0
19200	6	0
38400	3	0
56000	2	2.86

Table 9. Baud rates using a 3.072 MHz crystal

Desired baud rate	Divisor used to generate 16× clock	Percent error difference between desired and actual
50	2304	0
75	2560	0
110	1745	0.026
134.5	1428	0.034
150	1280	0
300	640	0
600	320	0
1200	160	0
1800	107	0.312
2000	96	0
2400	80	0
3600	53	0.628
4800	40	0
7200	27	1.23
9600	20	0
19200	10	0
38400	5	0

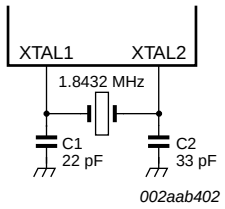


Fig 11. Crystal oscillator circuit reference

8. Register descriptions

The programming combinations for register selection are shown in Table 10.

Table 10. Register map - read/write properties

Register name	Read mode	Write mode
RHR/THR	Receive Holding Register (RHR)	Transmit Holding Register (THR)
IER	Interrupt Enable Register (IER)	Interrupt Enable Register
IIR/FCR	Interrupt Identification Register (IIR)	FIFO Control Register (FCR)
LCR	Line Control Register (LCR)	Line Control Register
MCR	Modem Control Register (MCR)[1]	Modem Control Register[1]
LSR	Line Status Register (LSR)	n/a
MSR	Modem Status Register (MSR)	n/a
SPR	Scratchpad Register (SPR)	Scratchpad Register
TCR	Transmission Control Register (TCR)[2]	Transmission Control Register[2]
TLR	Trigger Level Register (TLR)[2]	Trigger Level Register[2]
TXLVL	Transmit FIFO Level Register	n/a
RXLVL	Receive FIFO Level Register	n/a
EFCR	Extra Features Register	Extra Features Register
DLL	divisor latch LSB (DLL)[3]	divisor latch LSB[3]
DLH	divisor latch MSB (DLH)[3]	divisor latch MSB[3]
EFR	Enhanced Feature Register (EFR)[4]	Enhanced Feature Register[4]
XON1	Xon1 word[4]	Xon1 word[4]
XON2	Xon2 word[4]	Xon2 word[4]
XOFF1	Xoff1 word[4]	Xoff1 word[4]
XOFF2	Xoff2 word[4]	Xoff2 word[4]

[1] MCR[7] can only be modified when EFR[4] is set.

[2] Accessible only when ERF[4] = 1 and MCR[2] = 1, that is, EFR[4] and MCR[2] are read/write enables.

[3] Accessible only when LCR[7] is logic 1.

[4] Accessible only when LCR is set to 1011 1111b (0xBF).

Table 11. SC16IS741A internal registers

Register address	Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	R/W
General register set[1]										
0x00	RHR	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R
0x00	THR	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	W
0x01	IER	$\overline{\text{CTS}}$ interrupt enable[2]	$\overline{\text{RTS}}$ interrupt enable[2]	$\overline{\text{Xoff}}$[2]	Sleep mode[2]	modem status interrupt	receive line status interrupt	THR empty interrupt	RX data available interrupt	R/W
0x02	FCR	RX trigger level (MSB)	RX trigger level (LSB)	TX trigger level (MSB)[2]	TX trigger level (LSB)[2]	reserved[3]	TX FIFO reset[4]	RX FIFO reset[4]	FIFO enable	W
0x02	IIR[5]	FIFO enable	FIFO enable	interrupt priority bit 4[2]	interrupt priority bit 3[2]	interrupt priority bit 2	interrupt priority bit 1	interrupt priority bit 0	interrupt status	R
0x03	LCR	Divisor Latch Enable	set break	set parity	even parity	parity enable	stop bit	word length bit 1	word length bit 0	R/W
0x04	MCR	clock divisor[2]	IrDA mode enable[2]	Xon Any[2]	loopback enable	reserved[3]	TCR and TLR enable[2]	$\overline{\text{RTS}}$	reserved[3]	R/W
0x05	LSR	FIFO data error	THR and TSR empty	THR empty	break interrupt	framing error	parity error	overrun error	data in receiver	R
0x06	MSR	0	0	0	CTS	0	0	0	ΔCTS	R
0x07	SPR	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W
0x06	TCR[6]	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W
0x07	TLR[6]	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W
0x08	TXLVL	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R
0x09	RXLVL	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R
0x0D	reserved[3]	reserved[3]	reserved[3]	reserved[3]	reserved[3]	reserved[3]	reserved[3]	reserved[3]	reserved[3]	
0x0E	UART reset	reserved[3]	reserved[3]	reserved[3]	reserved[3]	UART software reset	reserved[3]	reserved[3]	reserved[3]	R/W
0x0F	EFCR	IrDA mode	reserved[3]	auto RS-485 RTS output inversion	auto RS-485 RTS direction control	reserved[3]	transmitter disable	receiver disable	9-bit mode enable	R/W
Special register set[7]										
0x00	DLL	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W
0x01	DLH	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W

Table 11. SC16IS741A internal registers *…continued*

Register address	Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	R/W
Enhanced register set[8]										
0x02	EFR	Auto $\overline{CTS}$	Auto $\overline{RTS}$	special character detect	enable enhanced functions	software flow control bit 3	software flow control bit 2	software flow control bit 1	software flow control bit 0	R/W
0x04	XON1	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W
0x05	XON2	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W
0x06	XOFF1	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W
0x07	XOFF2	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	R/W

[1] These registers are accessible only when LCR[7] = 0.

[2] These bits can only be modified if register bit EFR[4] is enabled.

[3] These bits are reserved and should be set to 0.

[4] After Receive FIFO or Transmit FIFO reset (through FCR[1:0]), the user must wait at least 2 × T_{clk} of XTAL1 before reading or writing data to RHR and THR, respectively.

[5] Burst reads on the serial interface (that is, reading multiple elements on the I²C-bus without a STOP or repeated START condition, or reading multiple elements on the SPI bus without de-asserting the $\overline{CS}$ pin), should not be performed on the IIR register.

[6] These registers are accessible only when MCR[2] = 1 and EFR[4] = 1.

[7] The special register set is accessible only when LCR[7] = 1 and not 0xBF.

[8] Enhanced Feature Registers are only accessible when LCR = 0xBF.

8.1 Receive Holding Register (RHR)

The receiver section consists of the Receiver Holding Register (RHR) and the Receiver Shift Register (RSR). The RHR is actually a 64-byte FIFO. The RSR receives serial data from the RX pin. The data is converted to parallel data and moved to the RHR. The receiver section is controlled by the Line Control Register. If the FIFO is disabled, location zero of the FIFO is used to store the characters.

8.2 Transmit Holding Register (THR)

The transmitter section consists of the Transmit Holding Register (THR) and the Transmit Shift Register (TSR). The THR is actually a 64-byte FIFO. The THR receives data and shifts it into the TSR, where it is converted to serial data and moved out on the TX pin. If the FIFO is disabled, the FIFO is still used to store the byte. Characters are lost if overflow occurs.

8.3 FIFO Control Register (FCR)

This is a write-only register that is used for enabling the FIFOs, clearing the FIFOs, setting transmitter and receiver trigger levels. Table 12 shows FIFO Control Register bit settings.

Table 12. FIFO Control Register bits description

Bit	Symbol	Description
7:6	FCR[7] (MSB), FCR[6] (LSB)	RX trigger. Sets the trigger level for the RX FIFO. 00 = 8 characters 01 = 16 characters 10 = 56 characters 11 = 60 characters
5:4	FCR[5] (MSB), FCR[4] (LSB)	TX trigger. Sets the trigger level for the TX FIFO. 00 = 8 spaces 01 = 16 spaces 10 = 32 spaces 11 = 56 spaces FCR[5:4] can only be modified and enabled when EFR[4] is set. This is because the transmit trigger level is regarded as an enhanced function.
3	FCR[3]	reserved
2	FCR[2][1]	reset TX FIFO logic 0 = no FIFO transmit reset (normal default condition) logic 1 = clears the contents of the transmit FIFO and resets the FIFO level logic (the Transmit Shift Register is not cleared or altered). This bit will return to a logic 0 after clearing the FIFO.
1	FCR[1][1]	reset RX FIFO logic 0 = no FIFO receive reset (normal default condition) logic 1 = clears the contents of the receive FIFO and resets the FIFO level logic (the Receive Shift Register is not cleared or altered). This bit will return to a logic 0 after clearing the FIFO.
0	FCR[0]	FIFO enable logic 0 = disable the transmit and receive FIFO (normal default condition) logic 1 = enable the transmit and receive FIFO

[1] FIFO reset requires at least two XTAL1 clocks, therefore, they cannot be reset without the presence of the XTAL1 clock.

8.4 Line Control Register (LCR)

This register controls the data communication format. The word length, number of stop bits, and parity type are selected by writing the appropriate bits to the LCR. Table 13 shows the Line Control Register bit settings.

Table 13. Line Control Register bits description

Bit	Symbol	Description
7	LCR[7]	divisor latch enable logic 0 = divisor latch disabled (normal default condition) logic 1 = divisor latch enabled
6	LCR[6]	Break control bit. When enabled, the break control bit causes a break condition to be transmitted (the TX output is forced to a logic 0 state). This condition exists until disabled by setting LCR[6] to a logic 0. logic 0 = no TX break condition (normal default condition). logic 1 = forces the transmitter output (TX) to a logic 0 to alert the communication terminal to a line break condition
5	LCR[5]	Set parity. LCR[5] selects the forced parity format (if LCR[3] = 1). logic 0 = parity is not forced (normal default condition). LCR[5] = logic 1 and LCR[4] = logic 0: parity bit is forced to a logical 1 for the transmit and receive data. LCR[5] = logic 1 and LCR[4] = logic 1: parity bit is forced to a logical 0 for the transmit and receive data.
4	LCR[4]	parity type select logic 0 = odd parity is generated (if LCR[3] = 1) logic 1 = even parity is generated (if LCR[3] = 1)
3	LCR[3]	parity enable logic 0 = no parity (normal default condition). logic 1 = a parity bit is generated during transmission and the receiver checks for received parity
2	LCR[2]	Number of stop bits. Specifies the number of stop bits. 0 to 1 stop bit (word length = 5, 6, 7, 8) 1 to 1.5 stop bits (word length = 5) 1 = 2 stop bits (word length = 6, 7, 8)
1:0	LCR[1:0]	Word length bits 1, 0. These two bits specify the word length to be transmitted or received; see Table 16.

Table 14. LCR[5] parity selection

LCR[5]	LCR[4]	LCR[3]	Parity selection
X	X	0	no parity
0	0	1	odd parity
0	1	1	even parity
1	0	1	forced parity '1'
1	1	1	forced parity '0'

Table 15. LCR[2] stop bit length

LCR[2]	Word length (bits)	Stop bit length (bit times)
0	5, 6, 7, 8	1
1	5	1½
1	6, 7, 8	2

Table 16. LCR[1:0] word length

LCR[1]	LCR[0]	Word length (bits)
0	0	5
0	1	6
1	0	7
1	1	8

8.5 Line Status Register (LSR)

Table 17 shows the Line Status Register bit settings.

Table 17. Line Status Register bits description

Bit	Symbol	Description
7	LSR[7]	FIFO data error. logic 0 = no error (normal default condition) logic 1 = at least one parity error, framing error, or break indication is in the receiver FIFO. This bit is cleared when no more errors are present in the FIFO.
6	LSR[6]	THR and TSR empty. This bit is the Transmit Empty indicator. logic 0 = transmitter hold and shift registers are not empty logic 1 = transmitter hold and shift registers are empty
5	LSR[5]	THR empty. This bit is the Transmit Holding Register Empty indicator. logic 0 = transmit hold register is not empty logic 1 = transmit hold register is empty. The host can now load up to 64 characters of data into the THR if the TX FIFO is enabled.
4	LSR[4]	break interrupt logic 0 = no break condition (normal default condition) logic 1 = a break condition occurred and associated character is 0x00, that is, RX was LOW for one character time frame
3	LSR[3]	framing error logic 0 = no framing error in data being read from RX FIFO (normal default condition). logic 1 = framing error occurred in data being read from RX FIFO, that is, received data did not have a valid stop bit
2	LSR[2]	parity error. logic 0 = no parity error (normal default condition) logic 1 = parity error in data being read from RX FIFO
1	LSR[1]	overrun error logic 0 = no overrun error (normal default condition) logic 1 = overrun error has occurred
0	LSR[0]	data in receiver logic 0 = no data in receive FIFO (normal default condition) logic 1 = at least one character in the RX FIFO

When the LSR is read, LSR[4:2] reflect the error bits (BI, FE, PE) of the character at the top of the RX FIFO (next character to be read). Therefore, errors in a character are identified by reading the LSR and then reading the RHR.

LSR[7] is set when there is an error anywhere in the RX FIFO, and is cleared only when there are no more errors remaining in the FIFO.

8.6 Modem Control Register (MCR)

The MCR controls the interface with the mode, data set, or peripheral device that is emulating the modem. Table 18 shows the Modem Control Register bit settings.

Table 18. Modem Control Register bits description

Bit	Symbol	Description
7	MCR[7][1]	clock divisor logic 0 = divide-by-1 clock input logic 1 = divide-by-4 clock input
6	MCR[6][1]	IrDA mode enable logic 0 = normal UART mode logic 1 = IrDA mode
5	MCR[5][1]	Xon Any logic 0 = disable Xon Any function logic 1 = enable Xon Any function
4	MCR[4]	enable loopback logic 0 = normal operating mode logic 1 = enable local Loopback mode (internal). In this mode the MCR[1:0] signals are looped back into MSR[4:5] and the TX output is looped back to the RX input internally.
3	MCR[3]	reserved
2	MCR[2]	TCR and TLR enable logic 0 = disable the TCR and TLR register. logic 1 = enable the TCR and TLR register.
1	MCR[1]	$\overline{\text{RTS}}$ logic 0 = force $\overline{\text{RTS}}$ output to inactive (HIGH) logic 1 = force $\overline{\text{RTS}}$ output to active ($\underline{\text{LOW}}$). In Loopback mode, controls MSR[4]. If Auto $\overline{\text{RTS}}$ is enabled, the $\overline{\text{RTS}}$ output is controlled by hardware flow control.
0	MCR[0]	reserved

[1] MCR[7:5] and MCR[2] can only be modified when EFR[4] is set, that is, EFR[4] is a write enable.

8.7 Modem Status Register (MSR)

This 8-bit register provides information about the current state of the control lines from the modem, data set, or peripheral device to the host. It also indicates when a control input from the modem changes state. Table 19 shows Modem Status Register bit settings.

Table 19. Modem Status Register bits description

Bit	Symbol	Description
7	MSR[7]	reserved
6	MSR[6]	reserved
5	MSR[5]	reserved
4	MSR[4]	CTS (active HIGH, logical 1). This bit is the complement of the $\overline{\text{CTS}}$ input.
3	MSR[3]	reserved
2	MSR[2]	reserved
1	MSR[1]	reserved
0	MSR[0]	ΔCTS. Indicates that $\overline{\text{CTS}}$ input has changed state. Cleared on a read.

8.8 Interrupt Enable Register (IER)

The Interrupt Enable Register (IER) enables each of the six types of interrupt, receiver error, RHR interrupt, THR interrupt, modem status, Xoff received, or $\overline{CTS}$/$\overline{RTS}$ change of state from LOW to HIGH. The $\overline{IRQ}$ output signal is activated in response to interrupt generation. Table 20 shows the Interrupt Enable Register bit settings.

Table 20. Interrupt Enable Register bits description

Bit	Symbol	Description
7	IER[7][1]	$\overline{CTS}$ interrupt enable logic 0 = disable the $\overline{CTS}$ interrupt (normal default condition) logic 1 = enable the $\overline{CTS}$ interrupt
6	IER[6][1]	$\overline{RTS}$ interrupt enable logic 0 = disable the $\overline{RTS}$ interrupt (normal default condition) logic 1 = enable the $\overline{RTS}$ interrupt
5	IER[5][1]	Xoff interrupt logic 0 = disable the Xoff interrupt (normal default condition) logic 1 = enable the Xoff interrupt
4	IER[4][1]	Sleep mode logic 0 = disable Sleep mode (normal default condition) logic 1 = enable Sleep mode. See Section 7.6 “Sleep mode” for details.
3	IER[3]	reserved
2	IER[2]	Receive Line Status interrupt logic 0 = disable the receiver line status interrupt (normal default condition) logic 1 = enable the receiver line status interrupt
1	IER[1]	Transmit Holding Register interrupt. logic 0 = disable the THR interrupt (normal default condition) logic 1 = enable the THR interrupt
0	IER[0]	Receive Holding Register interrupt. logic 0 = disable the RHR interrupt (normal default condition) logic 1 = enable the RHR interrupt

[1] IER[7:4] can only be modified if EFR[4] is set, that is, EFR[4] is a write enable. Re-enabling IER[1] will not cause a new interrupt if the THR is below the threshold.

8.9 Interrupt Identification Register (IIR)

The IIR is a read-only 8-bit register which provides the source of the interrupt in a prioritized manner. Table 21 shows Interrupt Identification Register bit settings.

Table 21. Interrupt Identification Register bits description

Bit	Symbol	Description
7:6	IIR[7:6]	mirror the contents of FCR[0]
5:1	IIR[5:1]	5-bit encoded interrupt. See Table 22.
0	IIR[0]	interrupt status logic 0 = an interrupt is pending logic 1 = no interrupt is pending

Table 22. Interrupt source

Priority level	IIR[5]	IIR[4]	IIR[3]	IIR[2]	IIR[1]	IIR[0]	Source of the interrupt
1	0	0	0	1	1	0	Receiver Line Status error
2	0	0	1	1	0	0	Receiver time-out interrupt
2	0	0	0	1	0	0	RHR interrupt
3	0	0	0	0	1	0	THR interrupt
4	0	0	0	0	0	0	modem interrupt
6	0	1	0	0	0	0	received Xoff signal/ special character
7	1	0	0	0	0	0	$\overline{CTS}$, $\overline{RTS}$ change of state from active (LOW) to inactive (HIGH)

8.10 Enhanced Features Register (EFR)

This 8-bit register enables or disables the enhanced features of the UART. Table 23 shows the enhanced feature register bit settings.

Table 23. Enhanced Features Register bits description

Bit	Symbol	Description
7	EFR[7]	$\overline{CTS}$ flow control enable logic 0 = $\overline{CTS}$ flow control is disabled (normal default condition) logic 1 = $\overline{CTS}$ flow control is enabled. Transmission will stop when a HIGH signal is detected on the $\overline{CTS}$ pin.
6	EFR[6]	$\overline{RTS}$ flow control enable. logic 0 = $\overline{RTS}$ flow control is disabled (normal default condition) logic 1 = $\overline{RTS}$ flow control is enabled. The $\overline{RTS}$ pin goes HIGH when the receiver FIFO **halt** trigger level TCR[3:0] is reached, and goes LOW when the receiver FIFO **resume** transmission trigger level TCR[7:4] is reached.
5	EFR[5]	Special character detect logic 0 = Special character detect disabled (normal default condition) logic 1 = Special character detect enabled. Received data is compared with Xoff2 data. If a match occurs, the received data is transferred to FIFO and IIR[4] is set to a logical 1 to indicate a special character has been detected.
4	EFR[4]	Enhanced functions enable bit logic 0 = disables enhanced functions and writing to IER[7:4], FCR[5:4], MCR[7:5]. logic 1 = enables the enhanced function IER[7:4], FCR[5:4], and MCR[7:5] so that they can be modified.
3:0	EFR[3:0]	Combinations of software flow control can be selected by programming these bits. See Table 4 “Software flow control options (EFR[3:0])”.

8.11 Division registers (DLL, DLH)

These are two 8-bit registers which store the 16-bit divisor for generation of the baud clock in the baud rate generator. DLH stores the most significant part of the divisor. DLL stores the least significant part of the divisor.

Remark: DLL and DLH can only be written to before Sleep mode is enabled, that is, before IER[4] is set.

8.12 Transmission Control Register (TCR)

This 8-bit register is used to store the RX FIFO threshold levels to stop/start transmission during hardware/software flow control. Table 24 shows Transmission Control Register bit settings.

Table 24. Transmission Control Register bits description

Bit	Symbol	Description
7:4	TCR[7:4]	RX FIFO trigger level to **resume**
3:0	TCR[3:0]	RX FIFO trigger level to **halt** transmission

TCR trigger levels are available from 0 to 60 characters with a granularity of four.

Remark: TCR can only be written to when EFR[4] = 1 and MCR[2] = 1. The programmer must program the TCR such that TCR[3:0] > TCR[7:4]. There is no built-in hardware check to make sure this condition is met. Also, the TCR must be programmed with this condition before auto $\overline{\text{RTS}}$ or software flow control is enabled to avoid spurious operation of the device.

8.13 Trigger Level Register (TLR)

This 8-bit register is used to store the transmit and received FIFO trigger levels used for interrupt generation. Trigger levels from 4 to 60 can be programmed with a granularity of 4. Table 25 shows trigger level register bit settings.

Table 25. Trigger Level Register bits description

Bit	Symbol	Description
7:4	TLR[7:4]	RX FIFO trigger levels (4 to 60), number of characters available.
3:0	TLR[3:0]	TX FIFO trigger levels (4 to 60), number of spaces available.

Remark: TLR can only be written to when EFR[4] = 1 and MCR[2] = 1. If TLR[3:0] or TLR[7:4] are logical 0, the selectable trigger levels via the FIFO Control Register (FCR) are used for the transmit and receive FIFO trigger levels. Trigger levels from 4 characters to 60 characters are available with a granularity of four. The TLR should be programmed for $^{N}/_{4}$, where N is the desired trigger level.

When the trigger level setting in TLR is zero, the SC16IS741A uses the trigger level setting defined in FCR. If TLR has non-zero trigger level value, the trigger level defined in FCR is discarded. This applies to both transmit FIFO and receive FIFO trigger level setting.

When TLR is used for RX trigger level control, FCR[7:6] should be left at the default state, that is, ‘00’.

8.14 Transmitter FIFO Level register (TXLVL)

This register is a read-only register, it reports the number of spaces available in the transmit FIFO.

Table 26. Transmitter FIFO Level register bits description

Bit	Symbol	Description
7	-	not used; set to zeros
6:0	TXLVL[6:0]	number of spaces available in TX FIFO, from 0 (0x00) to 64 (0x40)

8.15 Receiver FIFO Level register (RXLVL)

This register is a read-only register, it reports the fill level of the receive FIFO. That is, the number of characters in the RX FIFO.

Table 27. Receiver FIFO Level register bits description

Bit	Symbol	Description
7	-	not used; set to zeros
6:0	RXLVL[6:0]	number of characters stored in RX FIFO, from 0 (0x00) to 64 (0x40)

8.16 Extra Features Control Register (EFCR)

Table 28. Extra Features Control Register bits description

Bit	Symbol	Description
7	IRDA MODE	IrDA mode 0 = IrDA SIR, $^{3}/_{16}$ pulse ratio, data rate up to 115.2 kbit/s
6	-	reserved
5	RTSINVER	invert $\overline{\text{RTS}}$ signal in RS-485 mode 0: $\overline{\text{RTS}}$ = 0 during transmission and $\overline{\text{RTS}}$ = 1 during reception 1: $\overline{\text{RTS}}$ = 1 during transmission and $\overline{\text{RTS}}$ = 0 during reception
4	RTSCON	enable the transmitter to control the $\overline{\text{RTS}}$ pin 0 = transmitter does not control $\overline{\text{RTS}}$ pin 1 = transmitter controls $\overline{\text{RTS}}$ pin
3	-	reserved
2	TXDISABLE	Disable transmitter. UART does not send serial data out on the transmit pin, but the transmit FIFO will continue to receive data from host until full. Any data in the TSR will be sent out before the transmitter goes into disable state. 0: transmitter is enabled 1: transmitter is disabled
1	RXDISABLE	Disable receiver. UART will stop receiving data immediately once this bit set to a 1, and any data in the TSR will be sent to the receive FIFO. User is advised not to set this bit during receiving. 0: receiver is enabled 1: receiver is disabled
0	9-BIT MODE	Enable 9-bit or Multidrop mode (RS-485). 0: normal RS-232 mode 1: enables RS-485 mode

9. RS-485 features

9.1 Auto RS-485 $\overline{\text{RTS}}$ control

Normally the $\overline{\text{RTS}}$ pin is controlled by MCR bit 1, or if hardware flow control is enabled, the logic state of the $\overline{\text{RTS}}$ pin is controlled by the hardware flow control circuitry. EFCR register bit 4 will take the precedence over the other two modes; once this bit is set, the transmitter will control the state of the $\overline{\text{RTS}}$ pin. The transmitter automatically asserts the $\overline{\text{RTS}}$ pin (logic 0) once the host writes data to the transmit FIFO, and de-asserts $\overline{\text{RTS}}$ pin (logic 1) once the last bit of the data has been transmitted.

To use the auto RS-485 $\overline{\text{RTS}}$ mode the software would have to disable the hardware flow control function.

9.2 RS-485 $\overline{\text{RTS}}$ output inversion

EFCR bit 5 reverses the polarity of the $\overline{\text{RTS}}$ pin if the UART is in auto RS-485 $\overline{\text{RTS}}$ mode. When the transmitter has data to be sent it de-asserts the $\overline{\text{RTS}}$ pin (logic 1), and when the last bit of the data has been sent out the transmitter asserts the $\overline{\text{RTS}}$ pin (logic 0).

9.3 Auto RS-485

EFCR bit 0 is used to enable the RS-485 mode (multidrop or 9-bit mode). In this mode of operation, a ‘master’ station transmits an address character followed by data characters for the addressed ‘slave’ stations. The slave stations examine the received data and interrupt the controller if the received character is an address character (parity bit = 1).

To use the auto RS-485 mode the software would have to disable the hardware and software flow control functions.

9.3.1 Normal multidrop mode

The 9-bit Mode in EFCR (bit 0) is enabled, but not Special Character Detect (EFR bit 5). The receiver is set to Force Parity 0 (LCR[5:3] = 111) in order to detect address bytes.

With the receiver initially disabled, it ignores all the data bytes (parity bit = 0) until an address byte is received (parity bit = 1). This address byte will cause the UART to set the parity error. The UART will generate a line status interrupt (IER bit 2 must be set to ‘1’ at this time), and at the same time puts this address byte in the RX FIFO. After the controller examines the byte it must make a decision whether or not to enable the receiver; it should enable the receiver if the address byte addresses its ID address, and must not enable the receiver if the address byte does not address its ID address.

If the controller enables the receiver, the receiver will receive the subsequent data until being disabled by the controller after the controller has received a complete message from the ‘master’ station. If the controller does not disable the receiver after receiving a message from the ‘master’ station, the receiver will generate a parity error upon receiving another address byte. The controller then determines if the address byte addresses its ID address, if it is not, the controller then can disable the receiver. If the address byte addresses the ‘slave’ ID address, the controller takes no further action, the receiver will receive the subsequent data.

9.3.2 Auto address detection

If Special Character Detect is enabled (EFR[5] is set and the XOFF2 register contains the address byte) the receiver will try to detect an address byte that matches the programmed character in the XOFF2 register. If the received byte is a data byte or an address byte that does not match the programmed character in the XOFF2 register, the receiver will discard these data. Upon receiving an address byte that matches the Xoff2 character, the receiver will be automatically enabled if not already enabled, and the address character is pushed into the RX FIFO along with the parity bit (in place of the parity error bit). The receiver also generates a line status interrupt (IER[2] must be set to '1' at this time). The receiver will then receive the subsequent data from the 'master' station until being disabled by the controller after having received a message from the 'master' station.

If another address byte is received and this address byte does not match Xoff2 character, the receiver will be automatically disabled and the address byte is ignored. If the address byte matches Xoff2 character, the receiver will put this byte in the RX FIFO along with the parity bit in the parity error bit (LSR bit 2).

10. I²C-bus operation

The two lines of the I²C-bus are a serial data line (SDA) and a serial clock line (SCL). Both lines are connected to a positive supply via a pull-up resistor, and remain HIGH when the bus is not busy. Each device is recognized by a unique address whether it is a microcomputer, LCD driver, memory or keyboard interface and can operate as either a transmitter or receiver, depending on the function of the device. A device generating a message or data is a transmitter, and a device receiving the message or data is a receiver. Obviously, a passive function like an LCD driver could only be a receiver, while a microcontroller or a memory can both transmit and receive data.

10.1 Data transfers

One data bit is transferred during each clock pulse (see Figure 12). The data on the SDA line must remain stable during the HIGH period of the clock pulse in order to be valid. Changes in the data line at this time will be interpreted as control signals. A HIGH-to-LOW transition of the data line (SDA) while the clock signal (SCL) is HIGH indicates a START condition, and a LOW-to-HIGH transition of the SDA while SCL is HIGH defines a STOP condition (see Figure 13). The bus is considered to be busy after the START condition and free again at a certain time interval after the STOP condition. The START and STOP conditions are always generated by the master.

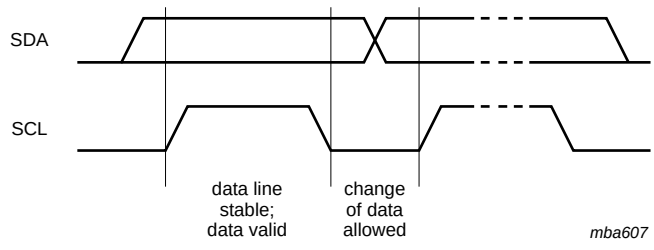


Fig 12. Bit transfer on the I²C-bus

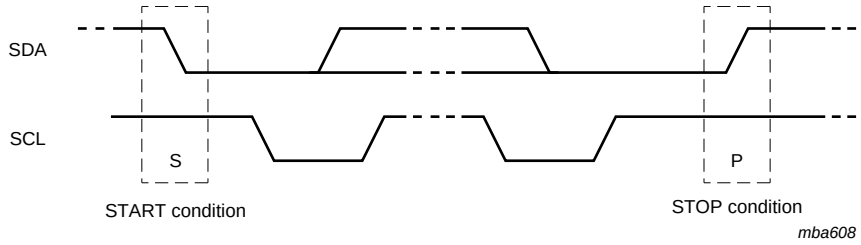


Fig 13. START and STOP conditions

The number of data bytes transferred between the START and STOP condition from transmitter to receiver is not limited. Each byte, which must be eight bits long, is transferred serially with the most significant bit first, and is followed by an acknowledge bit (see Figure 14). The clock pulse related to the acknowledge bit is generated by the master. The device that acknowledges has to pull down the SDA line during the acknowledge clock pulse, while the transmitting device releases this pulse (see Figure 15).

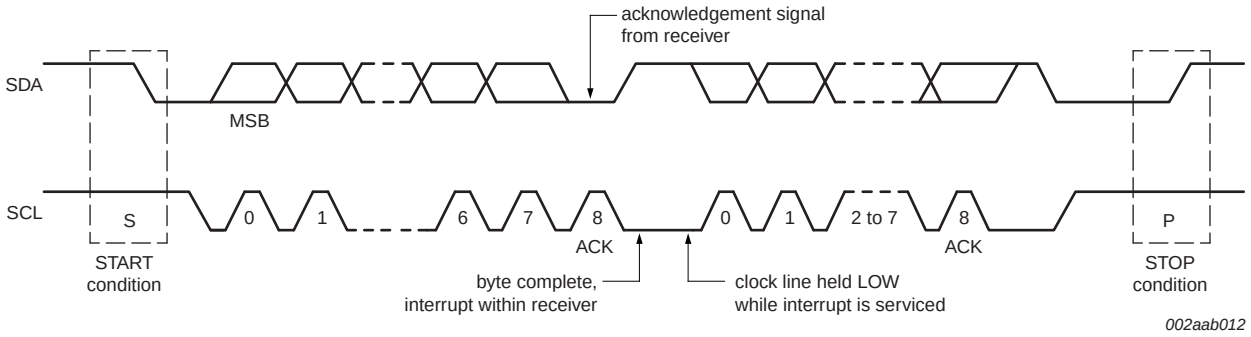


Fig 14. Data transfer on the I²C-bus

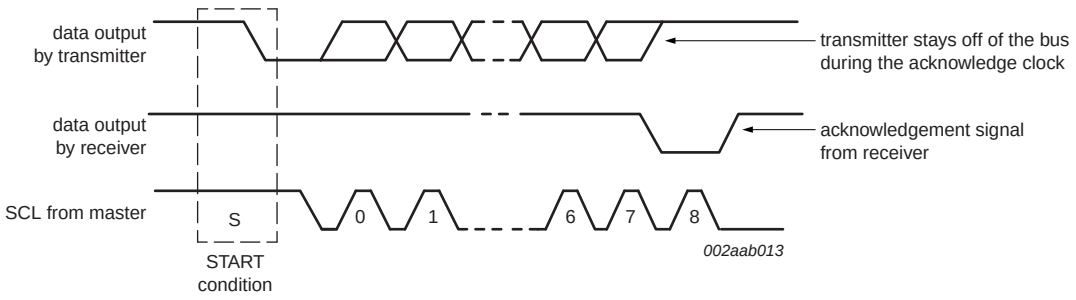


Fig 15. Acknowledge on the I²C-bus

A slave receiver must generate an acknowledge after the reception of each byte, and a master must generate one after the reception of each byte clocked out of the slave transmitter.

There are two exceptions to the ‘acknowledge after every byte’ rule. The first occurs when a master is a receiver: it must signal an end of data to the transmitter by **not** signaling an acknowledge on the last byte that has been clocked out of the slave. The acknowledge related clock, generated by the master should still take place, but the SDA line will not be pulled down. In order to indicate that this is an active and intentional lack of acknowledgement, we shall term this special condition as a ‘negative acknowledge’.

The second exception is that a slave will send a negative acknowledge when it can no longer accept additional data bytes. This occurs after an attempted transfer that cannot be accepted.

10.2 Addressing and transfer formats

Each device on the bus has its own unique address. Before any data is transmitted on the bus, the master transmits on the bus the address of the slave to be accessed for this transaction. A well-behaved slave with a matching address, if it exists on the network, should of course acknowledge the master's addressing. The addressing is done by the first byte transmitted by the master after the START condition.

An address on the network is seven bits long, appearing as the most significant bits of the address byte. The last bit is a direction (R/$\overline{W}$) bit. A ‘0’ indicates that the master is transmitting (write) and a ‘1’ indicates that the master requests data (read). A complete data transfer, comprised of an address byte indicating a ‘write’ and two data bytes is shown in Figure 16.

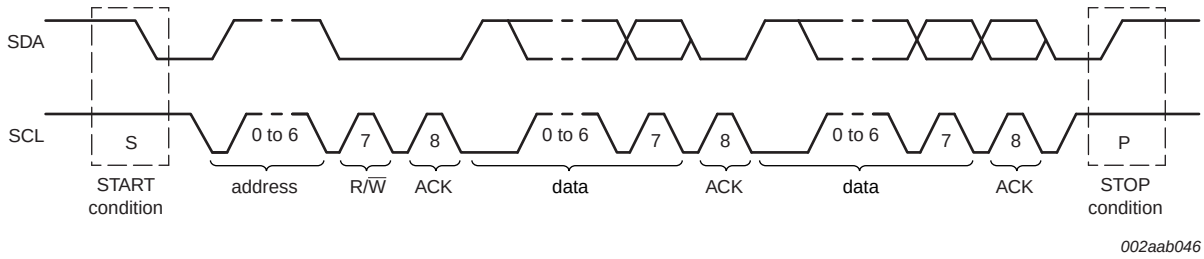


Fig 16. A complete data transfer

When an address is sent, each device in the system compares the first seven bits after the START with its own address. If there is a match, the device will consider itself addressed by the master, and will send an acknowledge. The device could also determine if in this transaction it is assigned the role of a slave receiver or slave transmitter, depending on the R/$\overline{W}$ bit.

Each node of the I2C-bus network has a unique seven-bit address. The address of a microcontroller is of course fully programmable, while peripheral devices usually have fixed and programmable address portions.

When the master is communicating with one device only, data transfers follow the format of Figure 16, where the R/$\overline{W}$ bit could indicate either direction. After completing the transfer and issuing a STOP condition, if a master would like to address some other device on the network, it could start another transaction by issuing a new START.

Another way for a master to communicate with several different devices would be by using a ‘repeated START’. After the last byte of the transaction was transferred, including its acknowledge (or negative acknowledge), the master issues another START, followed by address byte and data — without effecting a STOP. The master may communicate with a number of different devices, combining ‘reads’ and ‘writes’. After the last transfer takes place, the master issues a STOP and releases the bus. Possible data formats are demonstrated in Figure 17. Note that the repeated START allows for both change of a slave and a change of direction, without releasing the bus. We shall see later on that the change of direction feature can come in handy even when dealing with a single device.

In a single master system, the repeated START mechanism may be more efficient than terminating each transfer with a STOP and starting again. In a multimaster environment, the determination of which format is more efficient could be more complicated, as when a master is using repeated STARTs it occupies the bus for a long time and thus preventing other devices from initiating transfers.

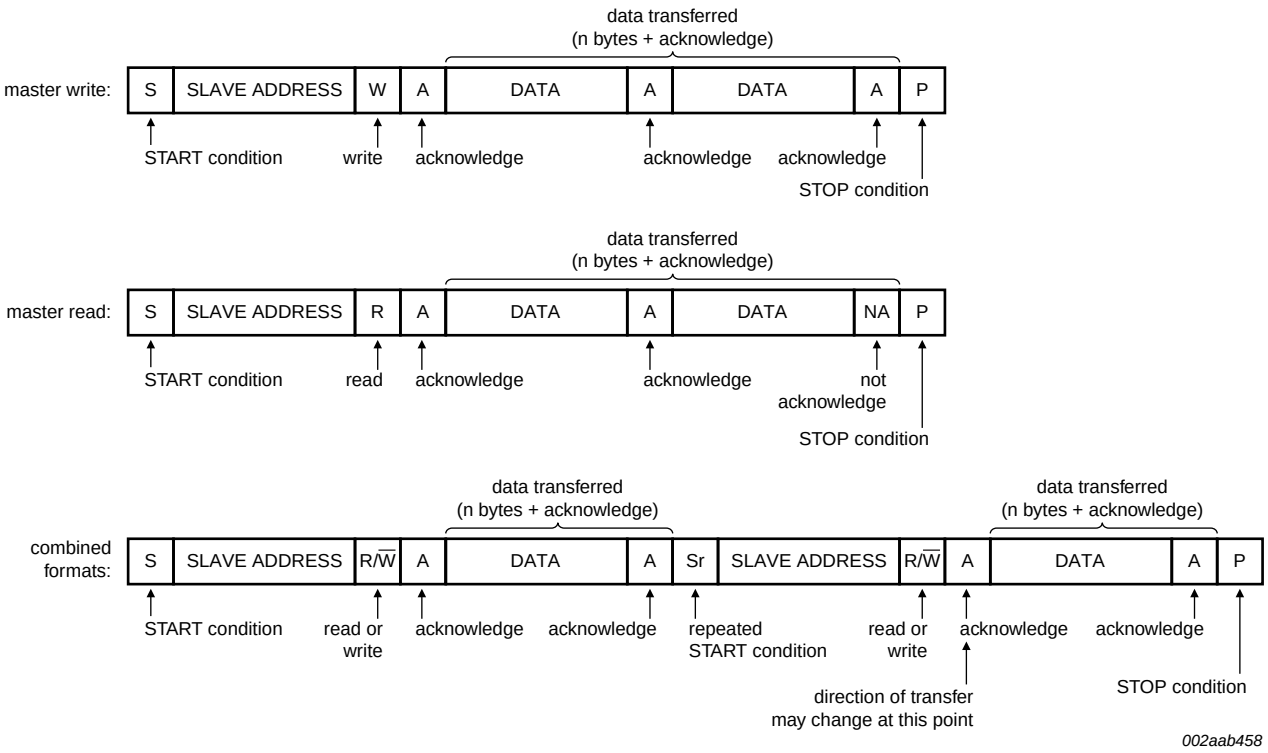


Fig 17. I²C-bus data formats

10.3 Addressing

Before any data is transmitted or received, the master must send the address of the receiver via the SDA line. The first byte after the START condition carries the address of the slave device and the read/write bit. Table 29 shows how the SC16IS741A's address can be selected by using A1 and A0 pins. For example, if these two pins are connected to V_{DD}, then the SC16IS741A's address is set to 0x90, and the master communicates with it through this address.

Table 29. SC16IS741A address map

A1	A0	SC16IS741A I²C addresses (hexadecimal)[1][2]
V_{DD}	V_{DD}	0x90 (1001 000X)
V_{DD}	V_{SS}	0x92 (1001 001X)
V_{DD}	SCL	0x94 (1001 010X)
V_{DD}	SDA	0x96 (1001 011X)
V_{SS}	V_{DD}	0x98 (1001 100X)
V_{SS}	V_{SS}	0x9A (1001 101X)
V_{SS}	SCL	0x9C (1001 110X)
V_{SS}	SDA	0x9E (1001 111X)
SCL	V_{DD}	0xA0 (1010 000X)
SCL	V_{SS}	0xA2 (1010 001X)
SCL	SCL	0xA4 (1010 010X)
SCL	SDA	0xA6 (1010 011X)
SDA	V_{DD}	0xA8 (1010 100X)
SDA	V_{SS}	0xAA (1010 101X)
SDA	SCL	0xAC (1010 110X)
SDA	SDA	0xAE (1010 111X)

[1] X = logic 0 for write cycle; X = logic 1 for read cycle.

[2] Consult Figure 23 and Figure 24 if A1 or A0 is connected to SCL or SDA.

10.3.1 SC16IS741A address decoder

In the SC16IS741, the I²C address decoder is reset by the on-board POR (Power-On Reset) or by the $\overline{\text{RESET}}$ pin. After reset, the address decoder is kept in reset until there is a valid START condition on the I²C-bus. When the START condition ends, the address decoder comes out of reset and continuously decodes the states of SCL, SDA, A1 and A0 pins. The decoding continues until there is a matching I²C slave address (configured by A1 and A0 pins) on the I²C-bus. The SC16IS741 locks into that address permanently until the device is reset by the $\overline{\text{RESET}}$ pin or by the on-board POR. If there are any glitches on the SDA line during the SCL LOW time, the SC16IS741 might not acknowledge (ACK) its configured I²C address or might ACK the wrong I²C address.

In the SC16IS741A, the I²C address decoder is reset by the on-board POR or by the $\overline{\text{RESET}}$ pin. After reset, the address decoder is kept in reset until there is a valid START condition on the I²C-bus. When the START condition ends, the address decoder comes out of reset and continuously decodes the states of SCL, SDA, A1 and A0 pins. The decoding would continue until the rising edge of SCL to create a STOP condition. This

rising edge of SCL will reset the I2C address decoder. As a result, the SC16IS741A locks into the I2C matching slave address (configured by A1 and A0) for only one I2C access cycle.

In the SC16IS741A, the I2C address decoder gets reset on every STOP condition and restarts after every START condition. Any glitches occurring on the SDA line during SCL LOW time between the START clock and the R/W clock are not allowed. These glitches might cause the SC16IS741A not to acknowledge its own I2C-bus address or to ACK the wrong I2C-bus address.

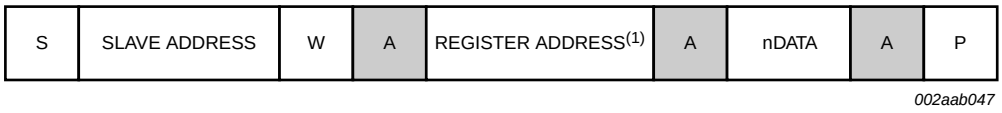
10.4 Use of subaddresses

When a master communicates with the SC16IS741A it must send a subaddress in the byte following the slave address byte. This subaddress is the internal address of the word the master wants to access for a single-byte transfer, or the beginning of a sequence of locations for a multi-byte transfer. A subaddress is an 8-bit byte. Unlike the device address, it does not contain a direction (R/$\overline{W}$) bit, and like any byte transferred on the bus it must be followed by an acknowledge.

Table 30 shows the breakdown of the subaddress (register address) byte. Bit 0 is not used, bits [2:1] are both set to zeroes, bits [6:3] are used to select one of the device's internal registers, and bit 7 is not used.

A register write cycle is shown in Figure 18. The START is followed by a slave address byte with the direction bit set to 'write', a subaddress byte, a number of data bytes, and a STOP signal. The subaddress indicates which register the master wants to access, and the data bytes which follow will be written one after the other to the subaddress location.

Table 30 and Table 31 show the bits' presentation at the subaddress byte for I2C-bus and SPI interfaces. Bit 0 is not used, bits 2:1 select the channel, bits 6:3 select one of the UART internal registers. Bit 7 is not used with the I2C-bus interface, but it is used by the SPI interface to indicate a read or a write operation.



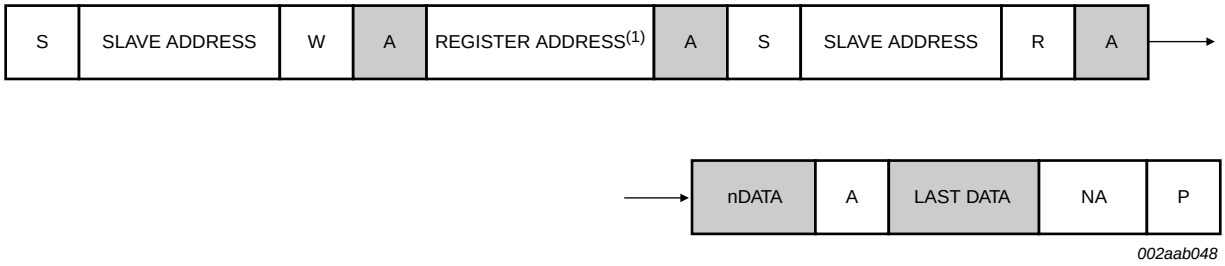
White block: host to SC16IS741A

Gray block: SC16IS741A to host

(1) See Table 30 for additional information.

Fig 18. Master writes to slave

The register read cycle (see Figure 19) commences in a similar manner, with the master sending a slave address with the direction bit set to 'write' with a following subaddress. Then, in order to reverse the direction of the transfer, the master issues a repeated START followed again by the device address, but this time with the direction bit set to 'read'. The data bytes starting at the internal subaddress will be clocked out of the device, each followed by a master-generated acknowledge. The last byte of the read cycle will be followed by a negative acknowledge, signaling the end of transfer. The cycle is terminated by a STOP signal.



White block: host to SC16IS741A

Gray block: SC16IS741A to host

(1) See Table 30 for additional information.

Fig 19. Master read from slave

Table 30. Register address byte (I2C)

Bit	Name	Function
7	-	not used
6:3	A[3:0]	UART's internal register select
2:1	CH1, CH0	channel select: CH1 = 0, CH0 = 0 Other values are reserved and should not be used.
0	-	not used

11. SPI operation

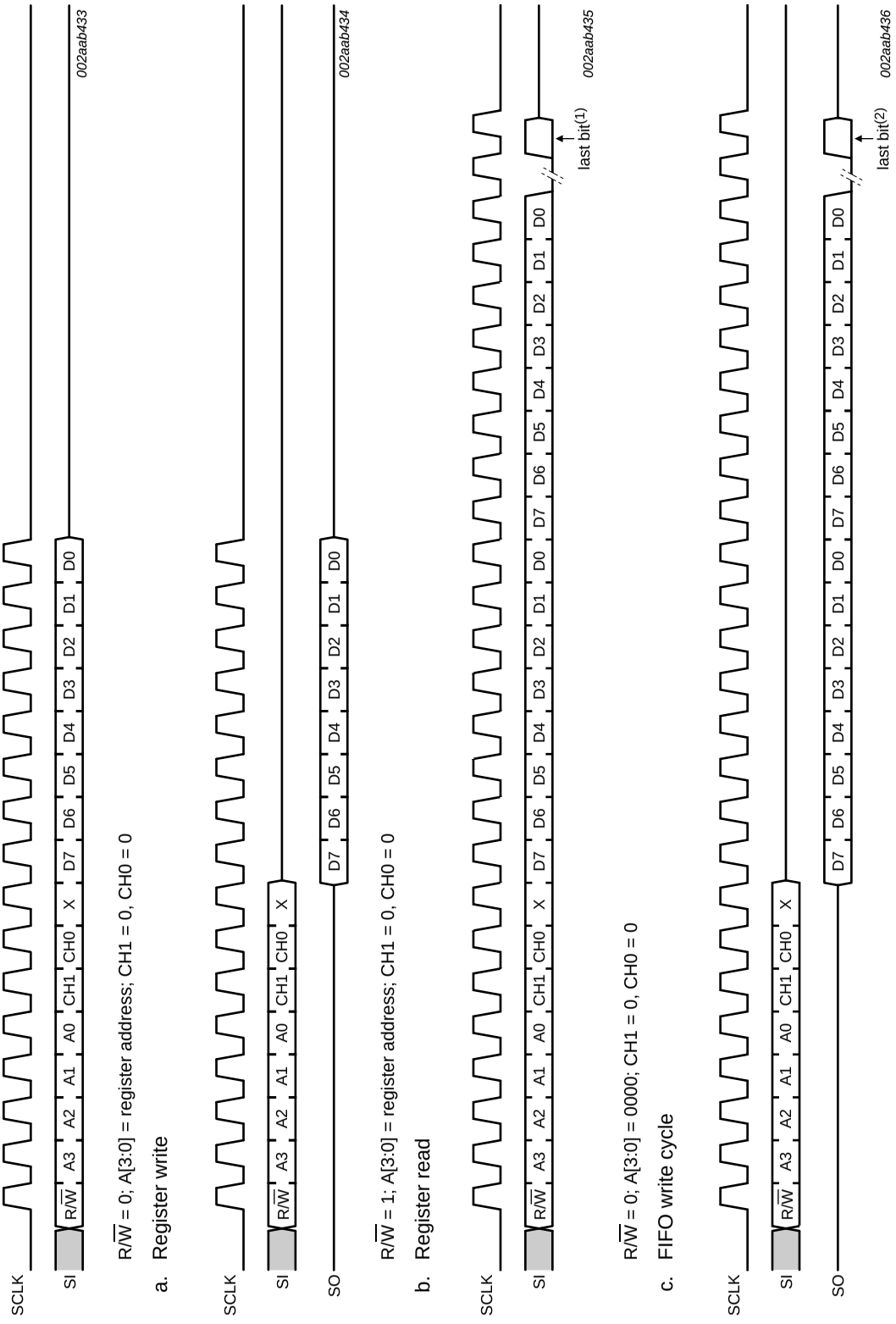


Fig 20. SPI operation

Table 31. Register address byte (SPI)

Bit	Name	Function
7	R/$\overline{W}$	1: read from UART 0: write to UART
6:3	A[3:0]	UART's internal register select
2:1	CH1, CH0	channel select: CH1 = 0, CH0 = 0 Other values are reserved and should not be used.
0	-	not used

12. Limiting values

Table 32. Limiting values

In accordance with the Absolute Maximum Rating System (IEC 60134).

Symbol	Parameter	Conditions	Min	Max	Unit
V_{DD}	supply voltage		–0.3	+4.6	V
V_I	input voltage	any input	–0.3	+5.5[1]	V
I_I	input current	any input	–10	+10	mA
I_O	output current	any output	–10	+10	mA
P_{tot}	total power dissipation		-	300	mW
P/out	power dissipation per output		-	50	mW
T_{amb}	ambient temperature	operating			
		V_{DD} = 2.5 V ± 0.2 V	–40	+85	°C
		V_{DD} = 3.3 V ± 0.3 V	–40	+95	°C
T_j	junction temperature		-	+125	°C
T_{stg}	storage temperature		–65	+150	°C

[1] 5.5 V steady state voltage tolerance on inputs and outputs is valid only when the supply voltage is present. 4.6 V steady state voltage tolerance on inputs and outputs when no supply voltage is present.

13. Static characteristics

Table 33. Static characteristics

V_{DD} = 2.5 V ± 0.2 V, T_{amb} = –40 °C to +85 °C; or V_{DD} = 3.3 V ± 0.3 V, T_{amb} = –40 °C to +95 °C; unless otherwise specified.

Symbol	Parameter	Conditions	V_{DD} = 2.5 V		V_{DD} = 3.3 V		Unit
			Min	Max	Min	Max	
Supplies							
V_{DD}	supply voltage		2.3	2.7	3.0	3.6	V
I_{DD}	supply current	operating; no load	-	6.0	-	6.0	mA
Inputs I2C/$\overline{SPI}$, RX, $\overline{CTS}$							
V_{IH}	HIGH-level input voltage		1.6	5.5[1]	2.0	5.5[1]	V
V_{IL}	LOW-level input voltage		-	0.6	-	0.8	V
I_L	leakage current	input; V_I = 0 V or 5.5 V[1]	-	1	-	1	µA
C_i	input capacitance		-	3	-	3	pF

Table 33. Static characteristics *...continued*

V_{DD} = 2.5 V ± 0.2 V, T_{amb} = −40 °C to +85 °C; or V_{DD} = 3.3 V ± 0.3 V, T_{amb} = −40 °C to +95 °C; unless otherwise specified.

Symbol	Parameter	Conditions	V_{DD} = 2.5 V		V_{DD} = 3.3 V		Unit
			Min	Max	Min	Max	
Outputs TX, $\overline{RTS}$, SO							
V_{OH}	HIGH-level output voltage	I_{OH} = −400 µA	1.85	-	-	-	V
		I_{OH} = −4 mA	-	-	2.4	-	V
V_{OL}	LOW-level output voltage	I_{OL} = 1.6 mA	-	0.4	-	-	V
		I_{OL} = 4 mA	-	-	-	0.4	V
C_o	output capacitance		-	4	-	4	pF
Output $\overline{IRQ}$							
V_{OL}	LOW-level output voltage	I_{OL} = 1.6 mA	-	0.4	-	-	V
		I_{OL} = 4 mA	-	-	-	0.4	V
C_o	output capacitance		-	4	-	4	pF
I²C-bus input/output SDA							
V_{IH}	HIGH-level input voltage		1.6	5.5[1]	2.0	5.5[1]	V
V_{IL}	LOW-level input voltage		-	0.6	-	0.8	V
V_{OL}	LOW-level output voltage	I_{OL} = 1.6 mA	-	0.4	-	-	V
		I_{OL} = 4 mA	-	-	-	0.4	V
I_L	leakage current	input; V_I = 0 V or 5.5 V[1]	-	10	-	10	µA
C_o	output capacitance		-	7	-	7	pF
I²C-bus inputs SCL, $\overline{CS}$/A0, SI/A1							
V_{IH}	HIGH-level input voltage		1.6	5.5[1]	2.0	5.5[1]	V
V_{IL}	LOW-level input voltage		-	0.6	-	0.8	V
I_L	leakage current	input; V_I = 0 V or 5.5 V[1]	-	10	-	10	µA
C_i	input capacitance		-	7	-	7	pF
Clock input XTAL1[2]							
V_{IH}	HIGH-level input voltage		1.8	5.5[1]	2.4	5.5[1]	V
V_{IL}	LOW-level input voltage		-	0.45	-	0.6	V
I_L	leakage current	input; V_I = 0 V or 5.5 V[1]	−30	+30	−30	+30	µA
C_i	input capacitance		-	3	-	3	pF
Sleep current							
$I_{DD(sleep)}$	sleep mode supply current	inputs are at V_{DD} or ground	-	30	-	30	µA

[1] 5.5 V steady state voltage tolerance on inputs and outputs is valid only when the supply voltage is present. 3.8 V steady state voltage tolerance on inputs and outputs when no supply voltage is present.

[2] XTAL2 should be left open when XTAL1 is driven by an external clock.

14. Dynamic characteristics

Table 34. I²C-bus timing specifications[1]

All the timing limits are valid within the operating supply voltage, ambient temperature range and output load; V_{DD} = 2.5 V ± 0.2 V, T_{amb} = −40 °C to +85 °C; or V_{DD} = 3.3 V ± 0.3 V, T_{amb} = −40 °C to +95 °C; and refer to V_{IL} and V_{IH} with an input voltage of V_{SS} to V_{DD}. All output load = 25 pF, except SDA output load = 400 pF.

Symbol	Parameter	Conditions	Standard mode I²C-bus		Fast mode I²C-bus		Unit
			Min	Max	Min	Max	
f_{SCL}	SCL clock frequency	[2]	0	100	0	400	kHz
t_{BUF}	bus free time between a STOP and START condition		4.7	-	1.3	-	µs
$t_{HD;STA}$	hold time (repeated) START condition		4.0	-	0.6	-	µs
$t_{SU;STA}$	set-up time for a repeated START condition		4.7	-	0.6	-	µs
$t_{SU;STO}$	set-up time for STOP condition		4.7	-	0.6	-	µs
$t_{HD;DAT}$	data hold time		0	-	0	-	ns
$t_{VD;ACK}$	data valid acknowledge time		-	0.6	-	0.6	µs
$t_{VD;DAT}$	data valid time	SCL LOW to data out valid	-	0.6	-	0.6	ns
$t_{SU;DAT}$	data set-up time		250	-	150	-	ns
t_{LOW}	LOW period of the SCL clock		4.7	-	1.3	-	µs
t_{HIGH}	HIGH period of the SCL clock		4.0	-	0.6	-	µs
t_f	fall time of both SDA and SCL signals		-	300	-	300	ns
t_r	rise time of both SDA and SCL signals		-	1000	-	300	ns
t_{SP}	pulse width of spikes that must be suppressed by the input filter		-	50	-	50	ns
$t_{d(int\_v)modem}$	modem interrupt valid delay time		0.2	-	0.2	-	µs
$t_{d(int\_clr)modem}$	modem interrupt clear delay time		0.2	-	0.2	-	µs
$t_{d(int\_v)rx}$	receive interrupt valid delay time		0.2	-	0.2	-	µs
$t_{d(int\_clr)rx}$	receive interrupt clear delay time		0.2	-	0.2	-	µs
$t_{d(int\_clr)tx}$	transmit interrupt clear delay time		1.0	-	0.5	-	µs
$t_{d(rst-SCL)}$	SCL delay time after reset	[3][4]	3	-	3	-	µs
$t_{d(SCL-A)}$	delay time from SCL to address		-	30	-	30	ns
$t_{d(SDA-A)}$	delay time from SDA to address		-	30	-	30	ns
$t_{d(A-SCL)}$	delay time from address to SCL		-	30	-	30	ns
$t_{d(A-SDA)}$	delay time from address to SDA		-	30	-	30	ns
$t_{w(rst)}$	reset pulse width		3	-	3	-	µs

[1] A detailed description of the I²C-bus specification, with applications, is given in user manual UM10204: *"I²C-bus specification and user manual"*. This may be found at www.nxp.com/documents/user_manual/UM10204.pdf.

[2] Minimum SCL clock frequency is limited by the bus time-out feature, which resets the serial bus interface if SDA is held LOW for a minimum of 25 ms.

[3] 2 XTAL1 clocks or 3 µs, whichever is less.

[4] The device will not acknowledge if an I²C-bus transaction occurs during the 'SCL delay time after reset'.

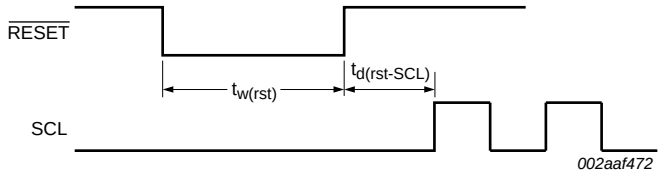
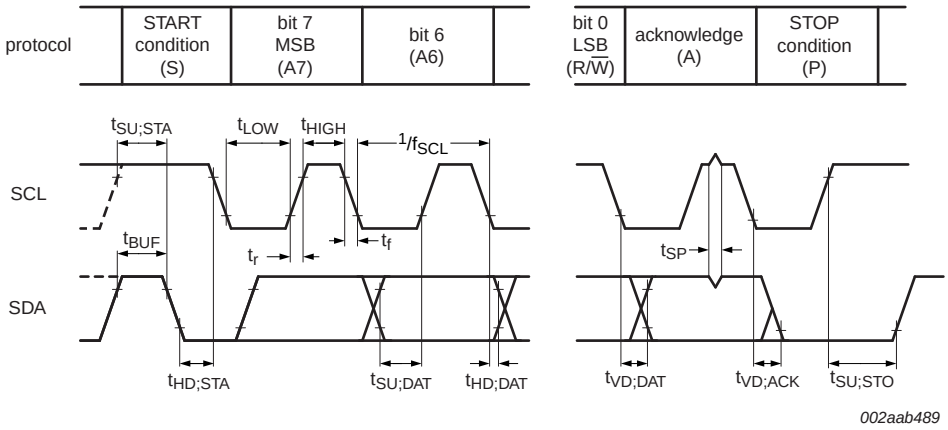


Fig 21. SCL delay after reset



Rise and fall times refer to V_{IL} and V_{IH}.

Fig 22. I²C-bus timing diagram

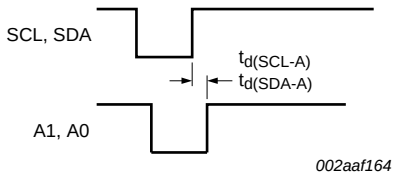


Fig 23. I²C-bus address selection A1/A0 delay

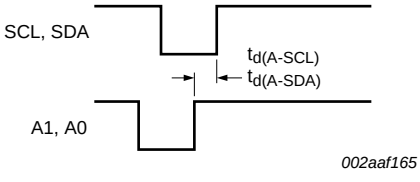


Fig 24. I²C-bus address selection SCL/SDA delay

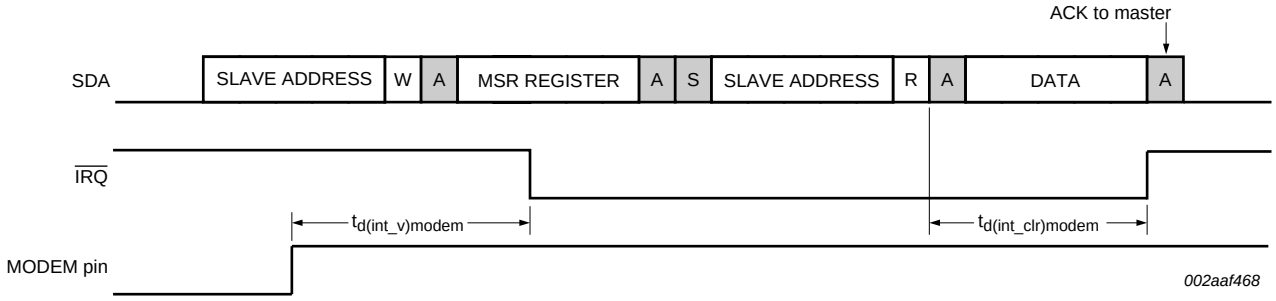


Fig 25. Modem input pin interrupt

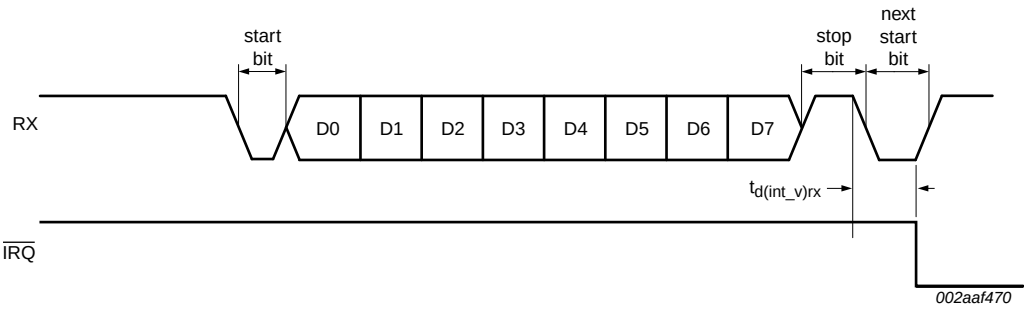


Fig 26. Receive interrupt

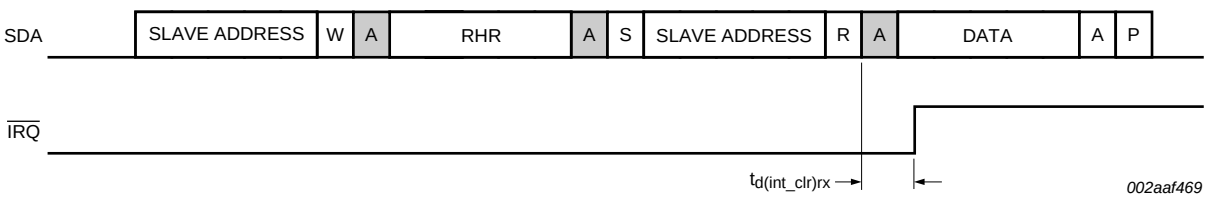


Fig 27. Receive interrupt clear

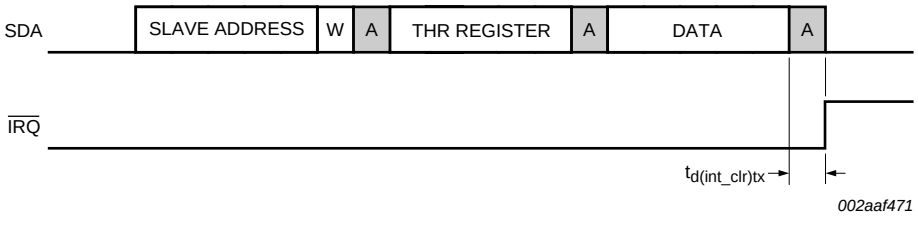


Fig 28. Transmit interrupt clear

Table 35. f_{XTAL} dynamic characteristics

V_{DD} = 2.5 V ± 0.2 V, T_{amb} = −40 °C to +85 °C; or V_{DD} = 3.3 V ± 0.3 V, T_{amb} = −40 °C to +95 °C

Symbol	Parameter	Conditions	V_{DD} = 2.5 V		V_{DD} = 3.3 V		Unit
			Min	Max	Min	Max	
t_{WH}	pulse width HIGH		10	-	6	-	ns
t_{WL}	pulse width LOW		10	-	6	-	ns
f_{XTAL}	frequency on pin XTAL	[1][2]	-	48	-	80	MHz

[1] Applies to external clock, crystal oscillator max. 24 MHz.

[2] $$f_{XTAL} = \frac{1}{t_{w(clk)}}$$

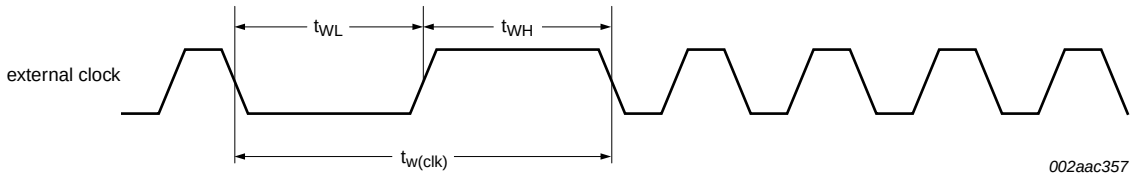


Fig 29. External clock timing

Table 36. SPI-bus timing specifications

All the timing limits are valid within the operating supply voltage, ambient temperature range and output load; V_{DD} = 2.5 V ± 0.2 V, T_{amb} = −40 °C to +85 °C; or V_{DD} = 3.3 V ± 0.3 V, T_{amb} = −40 °C to +95 °C; and refer to V_{IL} and V_{IH} with an input voltage of V_{SS} to V_{DD}. All output load = 25 pF, unless otherwise specified.

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
t_{TR}	$\overline{CS}$ HIGH to SO 3-state delay time	C_L = 100 pF	-	-	100	ns
t_{CSS}	$\overline{CS}$ to SCLK setup time		100	-	-	ns
t_{CSH}	$\overline{CS}$ to SCLK hold time		20	-	-	ns
t_{DO}	SCLK fall to SO valid delay time	C_L = 100 pF	-	-	100	ns
t_{DS}	SI to SCLK setup time		100	-	-	ns
t_{DH}	SI to SCLK hold time		20	-	-	ns
t_{CP}	SCLK period	$t_{CL} + t_{CH}$	250	-	-	ns
t_{CH}	SCLK HIGH time		100	-	-	ns
t_{CL}	SCLK LOW time		100	-	-	ns
t_{CSW}	$\overline{CS}$ HIGH pulse width		200	-	-	ns
$t_{d(int\_clr)tx}$	transmit interrupt clear delay time		200	-	-	ns
$t_{d(int\_clr)rx}$	receive interrupt clear delay time		200	-	-	ns
$t_{w(rst)}$	reset pulse width		3	-	-	µs

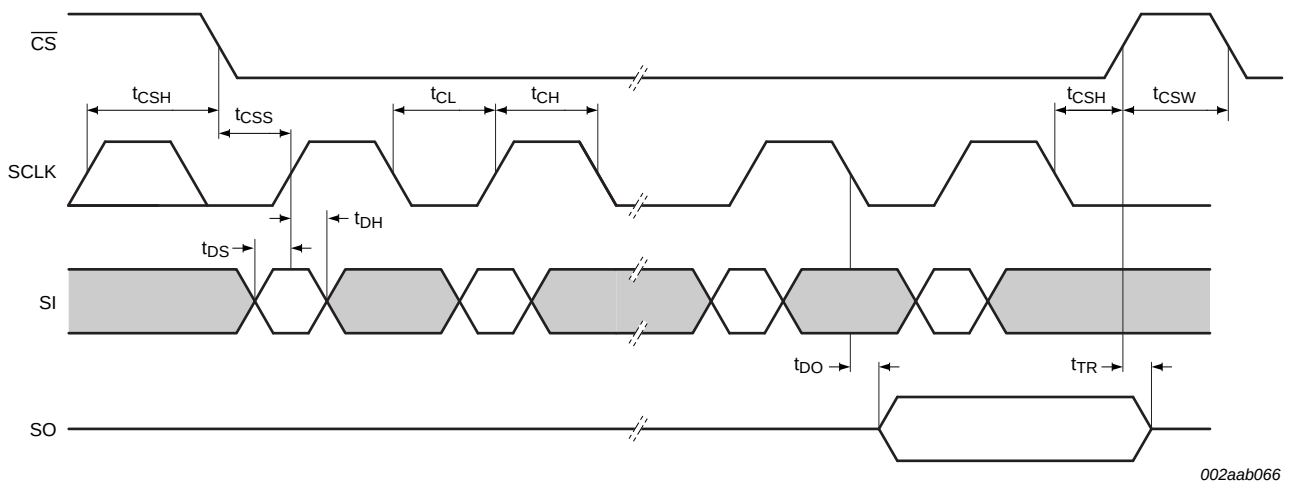
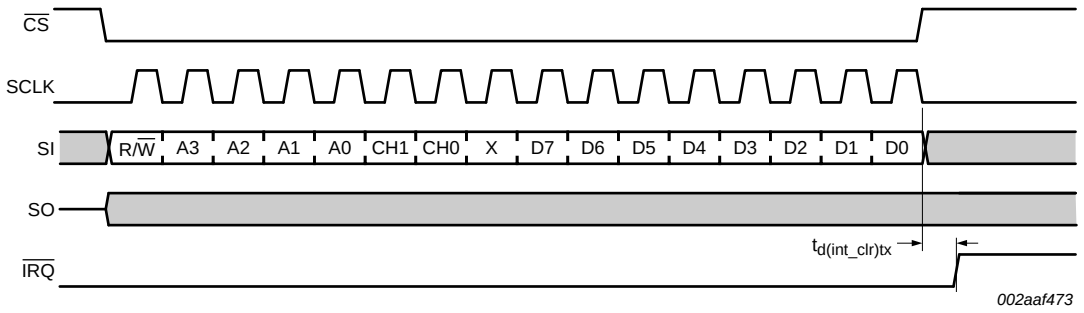
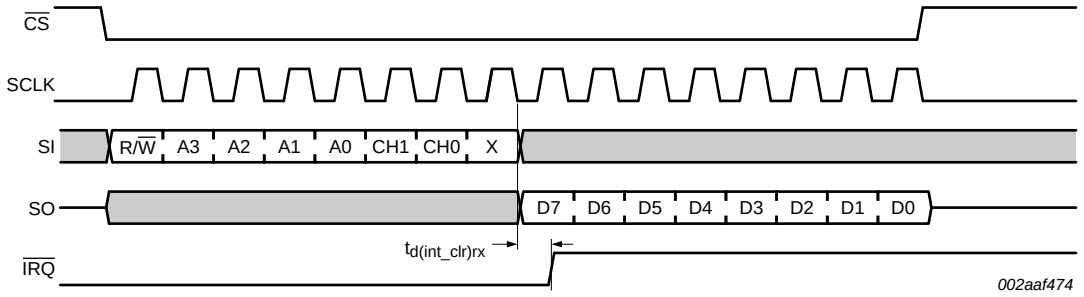


Fig 30. Detailed SPI-bus timing



R/W = 0; A[3:0] = THR (0x00); CH1 = 0; CH0 = 0

Fig 31. SPI write THR to clear TX INT

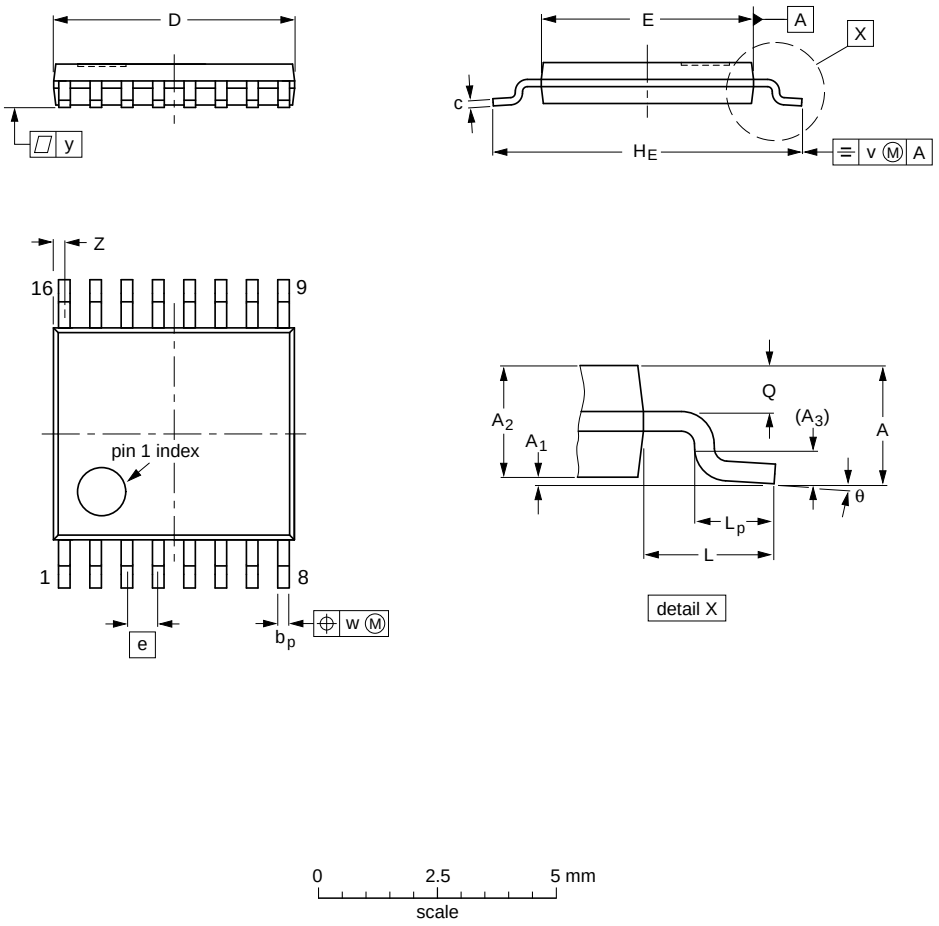


R/W = 1; A[3:0] = RHR (0x00); CH1 = 0; CH0 = 0

Fig 32. Read RHR to clear RX INT

15. Package outline

TSSOP16: plastic thin shrink small outline package; 16 leads; body width 4.4 mm **SOT403-1**



DIMENSIONS (mm are the original dimensions)

UNIT	A max.	A_1	A_2	A_3	b_p	c	D (1)	E (2)	e	H_E	L	L_p	Q	v	w	y	Z (1)	θ
mm	1.1	0.15 0.05	0.95 0.80	0.25	0.30 0.19	0.2 0.1	5.1 4.9	4.5 4.3	0.65	6.6 6.2	1	0.75 0.50	0.4 0.3	0.2	0.13	0.1	0.40 0.06	8° 0°

Notes

1. Plastic or metal protrusions of 0.15 mm maximum per side are not included.
2. Plastic interlead protrusions of 0.25 mm maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	JEITA			
SOT403-1		MO-153				~~99-12-27~~ 03-02-18

Fig 33. Package outline SOT403-1 (TSSOP16)

16. Handling information

All input and output pins are protected against ElectroStatic Discharge (ESD) under normal handling. When handling ensure that the appropriate precautions are taken as described in *JESD625-A* or equivalent standards.

17. Soldering of SMD packages

This text provides a very brief insight into a complex technology. A more in-depth account of soldering ICs can be found in Application Note *AN10365 "Surface mount reflow soldering description"*.

17.1 Introduction to soldering

Soldering is one of the most common methods through which packages are attached to Printed Circuit Boards (PCBs), to form electrical circuits. The soldered joint provides both the mechanical and the electrical connection. There is no single soldering method that is ideal for all IC packages. Wave soldering is often preferred when through-hole and Surface Mount Devices (SMDs) are mixed on one printed wiring board; however, it is not suitable for fine pitch SMDs. Reflow soldering is ideal for the small pitches and high densities that come with increased miniaturization.

17.2 Wave and reflow soldering

Wave soldering is a joining technology in which the joints are made by solder coming from a standing wave of liquid solder. The wave soldering process is suitable for the following:

- Through-hole components
- Leaded or leadless SMDs, which are glued to the surface of the printed circuit board

Not all SMDs can be wave soldered. Packages with solder balls, and some leadless packages which have solder lands underneath the body, cannot be wave soldered. Also, leaded SMDs with leads having a pitch smaller than ~0.6 mm cannot be wave soldered, due to an increased probability of bridging.

The reflow soldering process involves applying solder paste to a board, followed by component placement and exposure to a temperature profile. Leaded packages, packages with solder balls, and leadless packages are all reflow solderable.

Key characteristics in both wave and reflow soldering are:

- Board specifications, including the board finish, solder masks and vias
- Package footprints, including solder thieves and orientation
- The moisture sensitivity level of the packages
- Package placement
- Inspection and repair
- Lead-free soldering versus SnPb soldering

17.3 Wave soldering

Key characteristics in wave soldering are:

- Process issues, such as application of adhesive and flux, clinching of leads, board transport, the solder wave parameters, and the time during which components are exposed to the wave
- Solder bath specifications, including temperature and impurities

17.4 Reflow soldering

Key characteristics in reflow soldering are:

- Lead-free versus SnPb soldering; note that a lead-free reflow process usually leads to higher minimum peak temperatures (see Figure 34) than a SnPb process, thus reducing the process window
- Solder paste printing issues including smearing, release, and adjusting the process window for a mix of large and small components on one board
- Reflow temperature profile; this profile includes preheat, reflow (in which the board is heated to the peak temperature) and cooling down. It is imperative that the peak temperature is high enough for the solder to make reliable solder joints (a solder paste characteristic). In addition, the peak temperature must be low enough that the packages and/or boards are not damaged. The peak temperature of the package depends on package thickness and volume and is classified in accordance with Table 37 and 38

Table 37. SnPb eutectic process (from J-STD-020D)

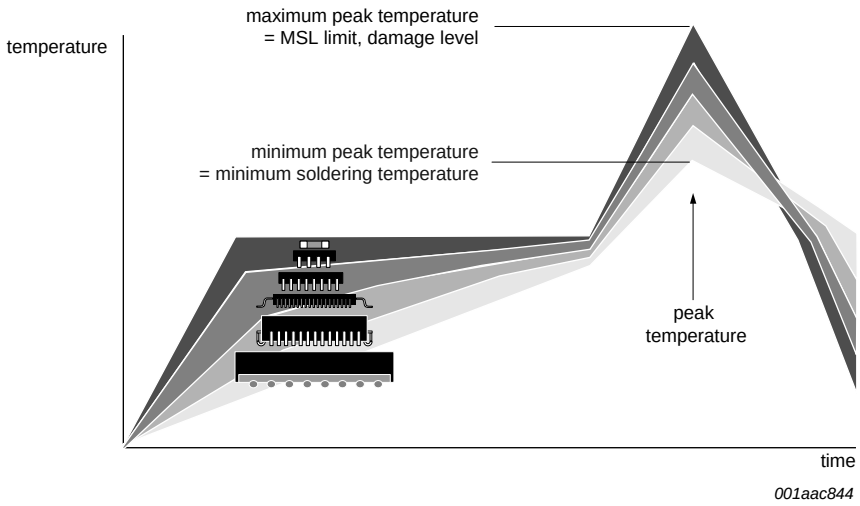
Package thickness (mm)	Package reflow temperature (°C)	
	Volume (mm³)	
	< 350	≥ 350
< 2.5	235	220
≥ 2.5	220	220

Table 38. Lead-free process (from J-STD-020D)

Package thickness (mm)	Package reflow temperature (°C)		
	Volume (mm³)		
	< 350	350 to 2000	> 2000
< 1.6	260	260	260
1.6 to 2.5	260	250	245
> 2.5	250	245	245

Moisture sensitivity precautions, as indicated on the packing, must be respected at all times.

Studies have shown that small packages reach higher temperatures during reflow soldering, see Figure 34.



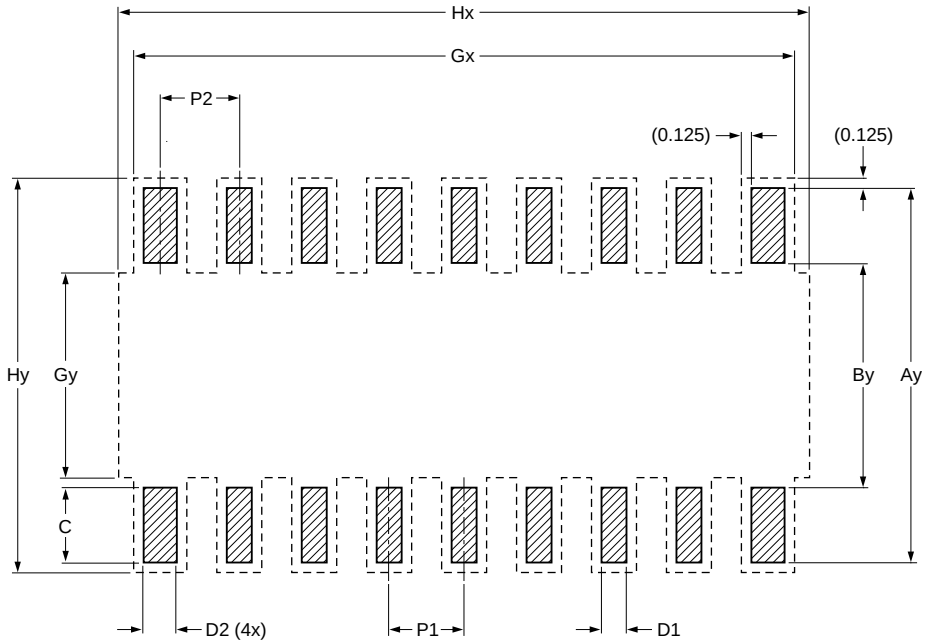
MSL: Moisture Sensitivity Level

Fig 34. Temperature profiles for large and small components

For further information on temperature profiles, refer to Application Note *AN10365 "Surface mount reflow soldering description"*.

18. Soldering: PCB footprints

Footprint information for reflow soldering of TSSOP16 package **SOT403-1**



Generic footprint pattern

Refer to the package outline drawing for actual layout

solder land

occupied area

DIMENSIONS in mm

P1	P2	Ay	By	C	D1	D2	Gx	Gy	Hx	Hy
0.650	0.750	7.200	4.500	1.350	0.400	0.600	5.600	5.300	5.800	7.450

sot403-1_fr

Fig 35. PCB footprint for SOT403-1 (TSSOP16); reflow soldering

19. Abbreviations

Table 39. Abbreviations

Acronym	Description
CPU	Central Processing Unit
FIFO	First In, First Out
I²C-bus	Inter-Integrated Circuit bus
IrDA	Infrared Data Association
LCD	Liquid Crystal Display
MIR	Medium InfraRed
POR	Power-On Reset
SIR	Serial InfraRed
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver/Transmitter

20. Revision history

Table 40. Revision history

Document ID	Release date	Data sheet status	Change notice	Supersedes
SC16IS741A v.1	20130318	Product data sheet	-	-

21. Legal information

21.1 Data sheet status

Document status[1][2]	Product status[3]	Definition
Objective [short] data sheet	Development	This document contains data from the objective specification for product development.
Preliminary [short] data sheet	Qualification	This document contains data from the preliminary specification.
Product [short] data sheet	Production	This document contains the product specification.

[1] Please consult the most recently issued document before initiating or completing a design.

[2] The term 'short data sheet' is explained in section "Definitions".

[3] The product status of device(s) described in this document may have changed since this document was published and may differ in case of multiple devices. The latest product status information is available on the Internet at URL http://www.nxp.com.

21.2 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

Short data sheet — A short data sheet is an extract from a full data sheet with the same product type number(s) and title. A short data sheet is intended for quick reference only and should not be relied upon to contain detailed and full information. For detailed and full information see the relevant full data sheet, which is available on request via the local NXP Semiconductors sales office. In case of any inconsistency or conflict with the short data sheet, the full data sheet shall prevail.

Product specification — The information and data provided in a Product data sheet shall define the specification of the product as agreed between NXP Semiconductors and its customer, unless NXP Semiconductors and customer have explicitly agreed otherwise in writing. In no event however, shall an agreement be valid in which the NXP Semiconductors product is deemed to offer functions and qualities beyond those described in the Product data sheet.

21.3 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the *Terms and conditions of commercial sale* of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) will cause permanent damage to the device. Limiting values are stress ratings only and (proper) operation of the device at these or any other conditions above those given in the Recommended operating conditions section (if present) or the Characteristics sections of this document is not warranted. Constant or repeated exposure to limiting values will permanently and irreversibly affect the quality and reliability of the device.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at http://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Non-automotive qualified products — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

21.4 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

I²C-bus — logo is a trademark of NXP B.V.

22. Contact information

For more information, please visit: **http://www.nxp.com**

For sales office addresses, please send an email to: **salesaddresses@nxp.com**

23. Contents

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 18 March 2013
Document identifier: SC16IS741A

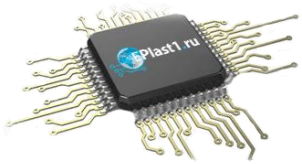
Компания «ЭлектроПласт» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

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.