PCI Target Interface Controller

SB16C1053APCI

Revision 1.06

SystemBase Co., Ltd.

CONTENTS

1. Description

SB16C1053APCI is a PCI Target Interface Controller with Dual-UART, single Parallel Port and MIO Bus™. It offers easy PCI Target Card Adapter implementation for serial port and parallel port. SB16C1053APCI provides high performance serial and parallel port communication. With a built-in two SB16C1050A Core that has built-in 256-byte FIFO, SB16C1053APCI decreases CPU load, is stronger at errors such as Overrun error and works well with simultaneous use of multiple ports. Furthermore, it is capable of waking up PC that is powered off through interrupts or Wake-up Requests with PCI Power Management implemented. SB16C1053APCI supports up to 6 serial ports extension, provides RS422/485 Auto Toggling function and Global Interrupt Function to the built-in UART allowing a more convenient handling of serial communication at driver level. Finally, with SB16C1053APCI, it is easy to design various modes of Serial and Parallel Multi-Port as below:

- 1-port Serial Multi-Port PCI card adapter (1S mode)
- 2-port Serial Multi-Port card adapter (2S mode)
- 4-port Serial Multi-Port card adapter (4S mode)
- 6-port Serial Multi-Port card adapter (6S mode)
- 1-port Parallel Multi-Port card adapter (1P mode)
- 2-port Serial and 1-port Parallel Multi-Port card adapter (2S1P mode)

SB16C1053APCI offers TQFP128 packages.

2. Features

2.1 PCI Interface

- Compliant with PCI Local Bus Specification 2.3
- Supports 32-bit Bus / 33MHz and 66MHz
- Supports data transmission of max. 264MB/sec
- Supports PCI Power Management 1.2
- Supports CompactPCI and CompactPCI Hot Swap
- Download Configuration Data from external serial EEPROM

- 3.3V Operation
- 5V Tolerant Inputs

2.2 Internal Dual-UART

- 2 Channel High Performance UART with 16C1050A core
- Up to 5.3 Mbps Baud Rate (Up to 85 MHz Oscillator Input Clock)
- 256-byte Transmit FIFO
- 256-byte Receive FIFO with Error Flags
- Programmable and Selectable Transmit and Receive FIFO Trigger Levels for Interrupt Generation
- 9-bit Communication (Multi-drop with Auto Address Detection)

- Software (Xon/Xoff) / Hardware (RTS#/CTS#) Flow Control
  - Programmable Xon/Xoff Characters
  - Programmable Auto-RTS and Auto-CTS
- Interrupt Poll Control
- Optional Data Flow Resume by Xon Any Character Control
- Optional Data Flow Additional Halt by Xoff Re-transmit Control
- Control pins for RS-422 Point to Point/Multi-Drop Auto Control
- Control pins for RS-485 Echo/Non Echo Auto Control
- Programmable Auto Control signal assertion width of TXEN and RXEN#
- Software Selectable Baud Rate Generator
- Prescaler Provides Additional Divide-by-4 Function
- Programmable Sleep Mode
- Programmable Serial Interface Characteristics
  - 5, 6, 7, or 8-bit Characters
  - Even, Odd, or No Parity Bit Generation and Detection
  - 1, 1.5, or 2 Stop Bit Generation
- False Start Bit Detection
- Line Break Generation and Detection
- Fully Prioritized Interrupt System Controls
- Modem Control Functions (RTS#, CTS#, DTR#, DSR#, DCD#, and RI#)

2.3 Parallel Interface

Parallel port used to transfer data between a host PC and a peripheral such as a printer.

- Support All IEEEStandard 1284 Protocols (Compatibility Mode, Nibble Mode, Byte Mode, EPP Mode and ECP Mode)
- 16-byte FIFO for SPP/ECP mode

2.4 MIO Bus Interface

SB16C1053APCI offers 8-bit Legacy Bus for additional 4 serial ports. SystemBase revised effective 8-bit data bus for from 1-port up to 32-port expandable serial Multi-Port. The name is MIO(Multi-Port I/O) Bus™.

SB16C1053APCI supports 4-port serial mode and 6-port serial mode using the MIO Bus™.

3. Ordering Information

Table 3–1: Ordering Information

Part Number	Package	Operating Temperature Range	Device Status
SB16C1053APCI-TQ	128-Pin TQFP	-40 ℃ to +85 ℃	Active

4. Block Diagram

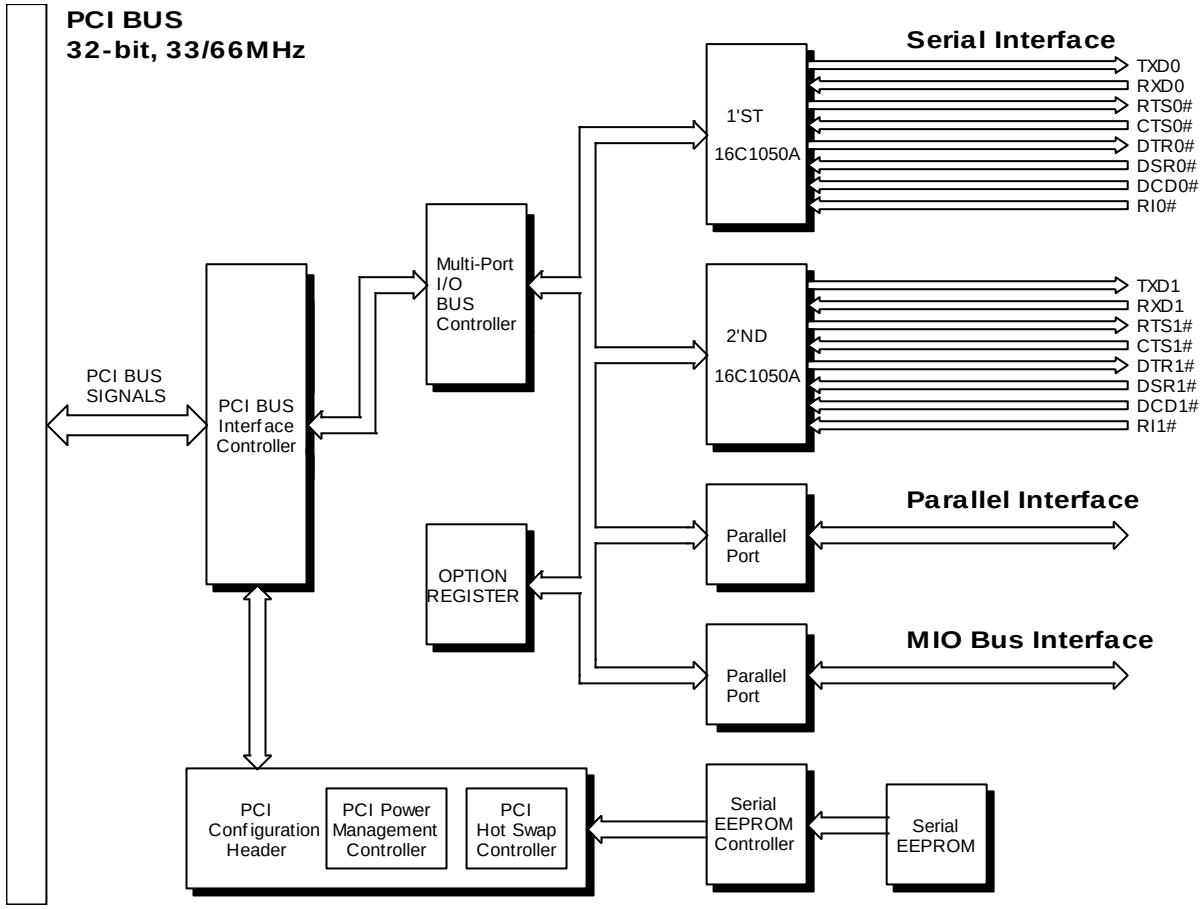


Figure 4–1: Block Diagram

5. Applications

5.1 Serial 1-port (1S)

Serial 1-port is generally made with SB16C1053APCI (called by 1S Mode). Special logic is not needed to make one serial port since Single-UART is built inside the SB16C1053APCI. Depending on Serial Interface, Transceiver IC of the RS232, RS422 or RS485 needs to be attached for long distance transmission. SB16C1053APCI is used as a PC add-in card type in Figure 5–1

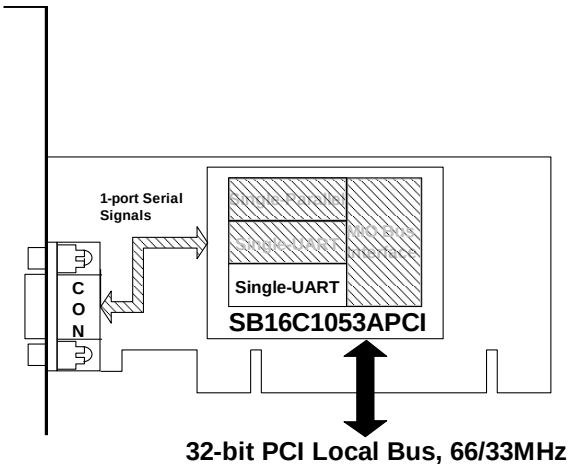


Figure 5–1: Serial 1-port Mode Application Block Diagram

5.2 Serial 2-port (2S)

Serial 2-port is generally made with SB16C1053APCI (called by 2S Mode). Special logic is not needed to make two serial ports since Dual-UART is built inside the SB16C1053APCI. Depending on Serial Interface, Transceiver IC of the RS232, RS422 or RS485 needs to be attached for long distance transmission. SB16C1053APCI is used as a PC add-in card type in Figure 5–2

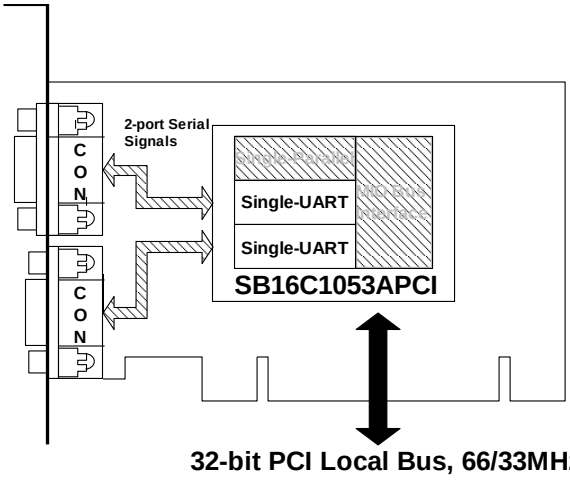


Figure 5–2: Serial 2-port Mode Application Block Diagram

5.3 Serial 2-port and Parallel 1-port (2S 1P)

Serial 2-port and Parallel 1-port (2S1P) is generally made with SB16C1053APCI (called by 2S1P Mode). Special logic is not needed to make one serial port since Dual-UART and IEEE1284 compatible Parallel Port is built inside the SB16C1053APCI. The Parallel Port supports all IEEE1284 standard protocols – Compatibility, Nibble, Byte, EPP and ECP mode). SB16C1053APCI is used as a PC add-in card type in Figure 5–3

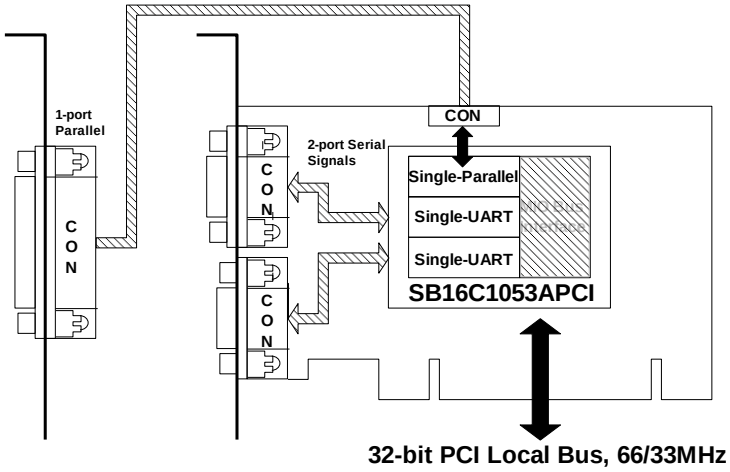


Figure 5–3: Serial 2-port and Parallel 1-port Mode Application Block Diagram

5.4 Parallel 1-port (1P)

Parallel 1-port is generally made with SB16C1053APCI (called by 1P Mode). Special logic is not needed to make one parallel port since IEEE1284 compatible Parallel Port is built inside the SB16C1053APCI. SB16C1053APCI is used as a PC add-in card type in Figure 5–4

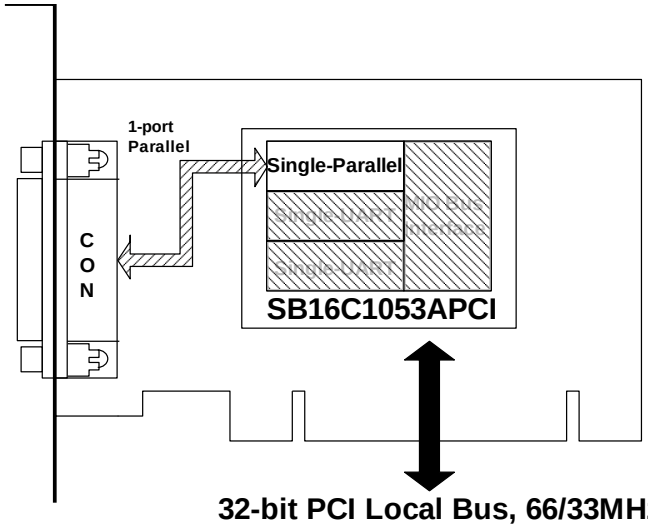


Figure 5–4: Parallel 1-port Mode Application Block Diagram

5.5 Serial 4-port (4S)

Serial 4-port is generally made with SB16C1053APCI (called by 4S Mode). Especially SB16C1053APCI support 8-bit legacy local bus for extension of additional serial ports. SystemBase revised the *MIO BUS*™(Multi-Port I/O Bus) for the extension. Basically Dual-UART is built inside the SB16C1053APCI. So we can make 4-port Serial add-in card using external Dual-UART through the MIO BUS™.

SB16C1053APCI is used as a PC add-in card type in Figure 5–5

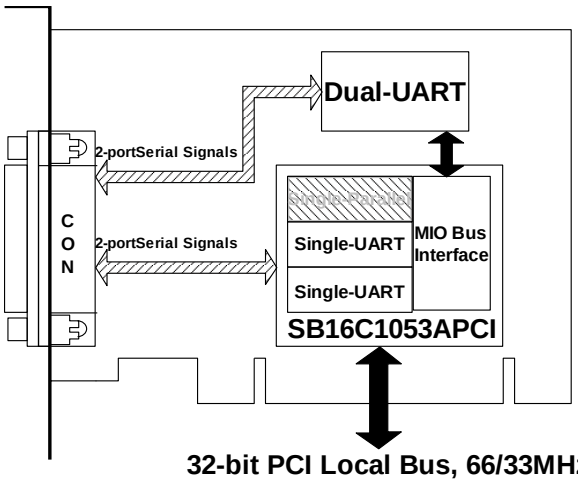


Figure 5–5: Serial 4-port Mode Application Block Diagram

5.6 Serial 6-port (6S)

Serial 6-port is generally made with SB16C1053APCI (called by 6S Mode). Especially SB16C1053APCI support 8-bit legacy local bus for extension of additional serial ports. SystemBase revised the *MIO BUS*™(Multi-Port I/O Bus) for the extension. Basically Dual-UART is built inside the SB16C1053APCI. So we can make 6-port Serial add-in card using external Quad-UART through the MIO BUS™.

SB16C1053APCI is used as a PC add-in card type in Figure 5–6

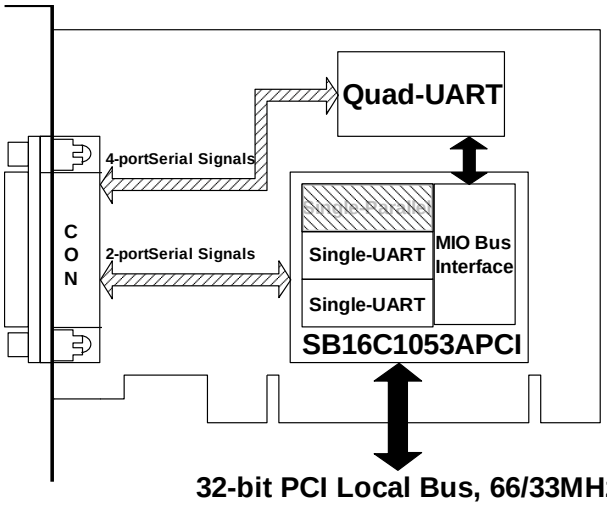


Figure 5–6: Serial 6-port Mode Application Block Diagram

6. Pin Configuration

6.1 Pin Configuration for 128-Pin TQFP Package

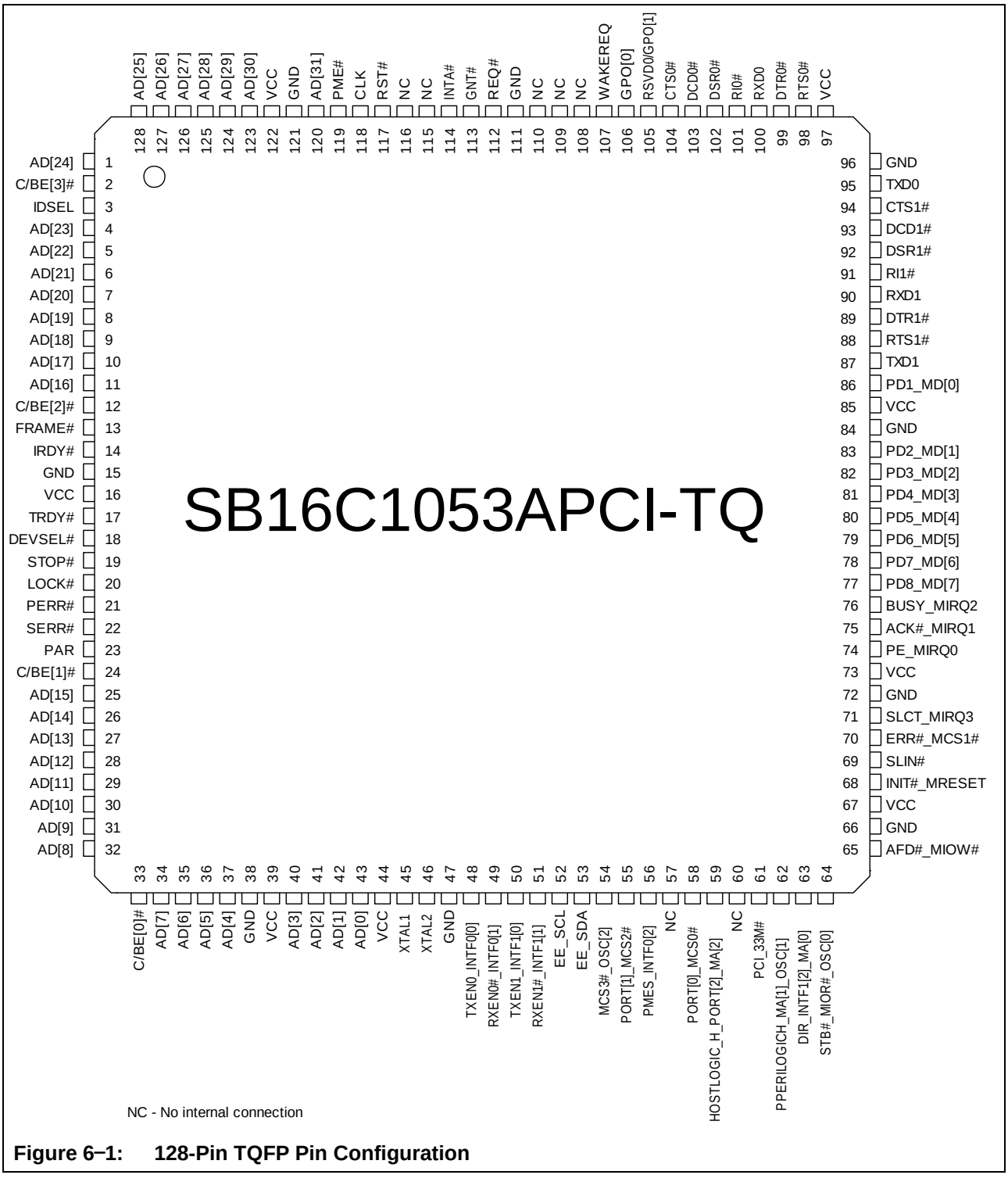


Figure 6–1: 128-Pin TQFP Pin Configuration

6.2 Pin Description

Table 6–1: Pin Description

Modem and Serial I/O Interface			
Name	**Pin**	**Type**	**Description**
TXD0 TXD1	95 87	O O	**Transmit Data:** These pins are individual transmit data output. During the local loop-back mode, the TXD output pin is disabled and TXD data is internally connected to the RXD input.
RXD0 RXD1	100 90	I I	**Receive Data:** These pins are individual receive data input. During the local loop-back mode, the RXD input pin is disabled and RXD data is internally connected to the TXD output.
RTS0# RTS1#	98 88	O O	**Request to Send (active low):** These pins indicate that the UART is ready to send data to the modem, and affect transmit and receive operations only when Auto-RTS function is enabled.
CTS0# CTS1#	104 94	I I	**Clear to Send (active low):** These pins indicate the modem is ready to accept transmitted data from the UART, and affect transmit and receive operations only when Auto-CTS function is enabled.
DTR0# DTR1#	99 89	O O	**Data Terminal Ready (active low):** These pins indicate UART is ready to transmit or receive data.
DSR0# DSR1#	102 92	I I	**Data Set Ready (active low):** These pins indicate modem is powered-on and is ready for data exchange with UART.
DCD0# DCD1#	103 93	I I	**Carrier Detect (active low):** These pins indicate that a carrier has been detected by modem.
RI0# RI1#	101 91	I I	**Ring Indicator (active low):** These pins indicate the modem has received a ringing signal from telephone line. A low to high transition on these input pins generates a modem status interrupt, if enabled.
TXEN0_INTF00 TXEN1_INTF10	48 50	I/O I/O	These pin are dual mode pins. After power is supplied to the chip, the pin is set to input mode for a while and receive INTFx[2:0] input. After that, the pins are set to output mode and outputs TXENx as auto toggle. **TX Enable:** These pins are for auto tri-state control of the RS422 or RS485 communication. When serial data is transmitted to TXD, the value set on ATR[5] is transmitted. These pins eliminate additional glue logic outside. **Line Interface Type Select:** These pins are used to select the type of Line Transceiver interfaced in Serial port Mode. The inputted value from these pins is shown in IIR0[5:4] of the Option register. In 1S, 2S ALL+ mode, INTF0[2:0] are for 1st serial port and INTF1[2:0] are for 2nd serial port. In 4S and 6S ALL+ mode, INTF0[2:0] are for 1st, 3rd, 5th serial ports and INTF1[2:0] are for 2nd, 4th, 6th serial port. In this case, the chip read the interface settings sequentially using GPO[1:0] output. 000b: RS232 interface is selected. 100b: RS422 1:1 interface is selected. 101b: RS422 Multi-Drop interface is selected. 110b: RS485 Non-Echo interface is selected.

			111b: RS485 Echo interface is selected.
RXEN0#_INTF01 RXEN1#_INTF11	49 51	I/O I/O	These pin are dual mode pins. After power is supplied to the chip, the pin is set to input mode for a while and receive INTFx[2:0] input. After that, the pins are set to output mode and outputs RXEN#x as auto toggle. **RX Enable:** This pins are for auto tri-state control of the RS422 or RS485 communication. When serial date is transmitted to TXD, the value set on ATR[7] is transmitted. These pins eliminate additional glue logic outside. **Line Interface Type Select:** These pins are used to select the type of Line Transceiver interfaced in Serial port Mode. The inputted value from these pins is shown in IIR0[5:4] of the Option register. 000b: RS232 interface is selected. 100b: RS422 1:1 interface is selected. 101b: RS422 Multi-Drop interface is selected. 110b: RS485 Non-Echo interface is selected. 111b: RS485 Echo interface is selected.

Parallel Port Interfaces

Name	Pin	Type	Description
PD1_MD0 PD2_MD1 PD3_MD2 PD4_MD3 PD5_MD4 PD6_MD5 PD7_MD6 PD8_MD7	86 83 82 81 80 79 78 77	I/O	These pins works as the data I/O of Parallel Port in 1P and 2S1P mode and the 8-bit data I/O of MIO Bus™ in 4S and 6S mode. **Parallel Port Data Bus:** These pins are data bus of Parallel Port in 1P and 2S1P mode. **MIO Bus Data Bus:** These pins are data bus of MIO Bus™ in 4S and 6S mode.
BUSY_MIRQ2	76	I	These pins works as BUSY in 1P and 2S1P mode and MIRQ2 of MIO Bus™ in 4S and 6S mode. **Parallel Port Busy:** This pin is asserted by peripheral when the peripheral isn't ready to receive data. **MIO Bus Interrupt Request 2:** This pin is an interrupt request for 3th UART channel among the external Quad-UART channel in 4S and 6S mode.
ACK#_MIRQ1	75	I	These pins works as ACK# in 1P and 2S1P mode and MIRQ1 of MIO Bus™ in 4S and 6S mode. **Parallel Port Acknowledge:** This pin is asserted by printer to indicate that data transfer is succeeded and it is ready for new data. **MIO Bus Interrupt Request 1:** This pin is an interrupt request for 2nd UART channel among the external Quad-UART channel in 4S and 6S mode.
PE_MIRQ0	74	I	These pins works as PE in 1P and 2S1P mode and MIRQ0 of MIO Bus™ in 4S and 6S mode. **Parallel Port Paper Empty:** This pin is activated by printer when it runs out of paper. **MIO Bus Interrupt Request 0:** This pin is an interrupt request for 1st UART channel among the external Quad-UART channel in 4S and 6S mode.

Table 6–1: Pin Description…*continued*

Function Configuration Interfaces			
Name	**Pin**	**Type**	**Description**
SLCT_MIRQ3	71	I	These pins works as SLCT in 1P and 2S1P mode and MIRQ3 of MIO Bus™ in 4S and 6S mode. **Parallel Port Peripheral Selected:** This pin is asserted by peripheral when the peripheral is selected (the peripheral device is on line). **MIO Bus Interrupt Request 3:** This pin is an interrupt request for 4th UART channel among the external Quad-UART channel in 4S and 6S mode.
ERR#_MCS1#	70	I/O	These pins works as ERR# in 1P and 2S1P mode and MCS1# of MIO Bus™ in 4S and 6S mode. **Parallel Port Error:** This input pin is held by peripheral when peripheral is in error condition. **MIO Bus Chip Select 1:** This output pin is a chip select signal of MIO Bus™ for the 2nd external UART channel.
SLIN#	69	O	**Parallel Port Select:** This pin is asserted by host to select the peripheral.
INIT#_MRESET	68	O	These pins works as INIT# in 1P and 2S1P mode and MRESET of MIO Bus™ in 4S and 6S mode. **Parallel Port Initialize:** This output pin is asserted by host to initialize the peripheral. **MIO Bus Reset:** This pin is the reset signal of MIO Bus™ for external UART channels.
AFD#_MIOW#	65	O	These pins works as AFD# in 1P and 2S1P mode and MIOW# of MIO Bus™ in 4S and 6S mode. **Parallel Port Auto Feed:** This output pin is asserted by host to put peripheral device into auto-line fedd mode. When software asserts AFD#, the printer is instructed to advance the paper one line for each carriage return encountered. **MIO Bus I/O Write:** This pin is the I/O Write signal of MIO Bus™ for external UART channels.
STB#_MIOR#_OSC0	64	I/O	This pin is dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive OSCx[2:0] input. After that, the pin is set to output mode and outputs STB# or MIOR#. This pin works as STB# in 1P and 2S1P mode and MIOR# of MIO Bus™ in 4S and 6S mode. **Parallel Port Strobe:** This pin is asserted by peripheral to latch the available data on PD[7:0]. **MIO Bus I/O Read:** This pin is the I/O Read signal of MIO Bus™ for external UART channels. **Oscillator Setting 0:** This pin is working in input mode for a while after power is supplied to the chipset. In this time, oscillator setting is set and the value is adopted to Device Information Register in option register space.

Table 6–1: Pin Description…*continued*

Function Configuration Interfaces			
Name	**Pin**	**Type**	**Description**
DIR_INTF12_MA0	63	I/O	These pin are dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive INTFx[2:0] input. After that, the pin is set to output mode and outputs DIR or MA0. This pin works as DIR in 1P and 2S1P mode and MA0 of MIO Bus™ in 4S and 6S mode. **Parallel Port Direction:** This pin indicates the data direction of parallel port data. When it has '0', it means forward direction. When it has '1', it means reverse direction. **MIO Bus Address Bus:** This pin is the address bus of MIO Bus™ for external UART channels. **Interface Setting:** This pin is working in input mode for a while after power is supplied to the chipset. In this time, interface setting is set and the value is adopted to Interface Information Register in option register space.
PERILOGICH_MA1_OSC1	62	I/O	This pin is dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive OSCx[2:0] input. After that, the pin is set to PERILOGICH(input) or MA1(output). This pin works as PERILOGICH in 1P and 2S1P mode and MA1 of MIO Bus™ in 4S and 6S mode. **Parallel Port Peripheral Logic High:** This input pin is set to indicate that the peripheral is in valid state. But when this pin is reset to indicate that peripheral is in power off state or invalid state. **MIO Bus Address Bus:** This pin is the address bus of MIO Bus™ for external UART channels. **Oscillator Setting 1:** This pin is working in input mode for a while after power is supplied to the chipset. In this time, oscillator setting is set and the value is adopted to Device Information Register in option register space.
HOSTLOGICH_PORT2_MA2	59	I/O	This pin is dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive PORTx[2:0] input. After that, the pin is set to HOSTLOGICH(output) or MA2(output). This pin works as HOSTLOGICH in 1P and 2S1P mode and MA2 of MIO Bus™ in 4S and 6S mode. **Parallel Port Host Logic High:** This output pin is set to indicate that the host is in valid state. But when this pin is reset to indicate that host is in power off state or invalid state. **MIO Bus Address Bus:** This output pin is the address bus of MIO Bus™ for external UART channels. **Port Setting 2:** This input pin is a port setting for various operating modes.

Table 6–1: Pin Description…*continued*

I²C Serial EEPROM Interfaces			
Name	**Pin**	**Type**	**Description**
SCL	52	O	**Serial EEPROM Clock Output:** Connected to SCL of serial EEPROM.
SDA	53	I/O	**Serial EEPROM Data Input/Output:** Connected to SDA of serial EEPROM.

PCI Interfaces			
Name	**Pin**	**Type**	**Description**
CLK	118	I	**PCI Clock:** PCI clock provides timing for all transaction on SB16C1053APCI.
RST#	117	I	**PCI Reset:** Reset the SB16C1053APCI. The inputted signal indicates when the applied main power is within the specified tolerance and stable. This signal is asynchronous to CLK when asserted or deasserted.
INTA#	114	O/D	**Interrupt A:** Interrupt A is used to request an interrupt. Interrupts on PCI are defined as “level sensitive”, asserted low, using open drain output drivers. The assertion and deassertion of INTA# is asynchronous to CLK.
PME#	119	O/D	**Power Management Event:** This signal can be used by SB16C1053APCI to request a change in the SB16C1053APCI or main system power state. The assertion and deassertion of PME# is asynchronous to CLK. This signal has additional electrical requirements over and above standard open drain signals that allow it to be shared between devices that are powered off and those that are powered on. The use of this pin is specified in the PCI Bus Power Management Interface Specification.
PMES_INTF02	56	I/O	This pin is dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive INTFx[2:0] input. After that, the pin is set to output mode and outputs PMES as Power Management Status. **PME Status:** This signal indicates PM state. If PM state is in D0 state, it is set to 1b and if PM state is in D3 state, it is cleared to 0b. **Line Interface Type Select:** These pins are used to select the type of Line Transceiver interfaced in Serial port Mode. The inputted value from these pins is shown in IIR0[5:4] of the Option register.
PCI_33M#	61	I	**PCI Operational Speed:** This input is for selecting the operational speed of PCI Bus. This chip is operated at 33MHz PCI Bus when this pin is cleared to 0b. And it is operated at 66MHz PCI Bus when this pin is setted to 1b.
PAR	23	T/S	**Parity:** Parity is even parity across AD[31:00] and C/BE[3:0]#.
FRAME#	13	S/T/S	**Cycle Frame:** This signal is driven by the master of main system to indicate the beginning and duration of an access.
IRDY#	14	S/T/S	**Initiator Ready:** This signal indicates the initiating agent’s(main system’s) ability to complete the current data phase of the transaction.
TRDY#	17	S/T/S	**Target Ready:** This signal indicates the target agent’s(SB16C1053APCI’s) ability to complete the current data phase of the transaction.
STOP#	19	S/T/S	**Stop:** This signal indicates that the current target(SB16C1053APCI) is requesting the master to stop current transaction.

Table 6–1: Pin Description…*continued*

PCI Interfaces			
Name	**Pin**	**Type**	**Description**
AD[31] AD[30] AD[29] AD[28] AD[27] AD[26] AD[25] AD[24] AD[23] AD[22] AD[21] AD[20] AD[19] AD[18] AD[17] AD[16] AD[15] AD[14] AD[13] AD[12] AD[11] AD[10] AD[09] AD[08] AD[07] AD[06] AD[05] AD[04] AD[03] AD[02] AD[01] AD[00]	120, 123 124 125 126 127 128 1 4 5 6 7 8 9 10 11 25 26 27 28 29 30 31 32 34 35 36 37 40 41 42 43	T/S	**Address and Data:** These signals are multiplexed on the same pins. A Bus transaction consists of an address phase followed by a data phase. SB16C1053APCI does no support both read and write bursts.
C/BE[3]# C/BE[2]# C/BE[1]# C/BE[0]#	2 12 24 33	T/S	**Bus Command and Byte Enables:** These signals are multiplexed on same pins. During the address phase of transaction, C/BE[3:0]# define the bus command. During the data phase, C/BE[3:0]# are used as Byte Enables.
LOCK#	20	S/T/S	**Lock:** This signal provides for exclusive use of a resource. SB16C1053APCI may be locked by one master at a time. See the PCI Local Bus Specification for the detail operation of lock function.

IDSEL	3	I	**Initialization Device Select:** This is used for chip selection during configuration read and write transaction.

Table 6–1: Pin Description…*continued*

PCI Interfaces			
Name	**Pin**	**Type**	**Description**
DEVSEL#	18	S/T/S	**Device Select:** This signal indicates that the driving device has decoded its address as the target of the current access. As an input, DEVSEL# indicates whether any device on the bus has been selected.
PERR#	21	S/T/S	**Parity Error:** This signal is only for reporting data parity errors during all PCI transactions except Special Cycle.
SERR#	22	S/T/S	**System Error:** This signal is for reporting address parity errors, data parity errors on the Special Cycle command, or any other system error where the result will be catastrophic.
REQ#	112	O	**Request:** This signal indicates to the arbiter that SB16C1053APCI want to use of the PCI bus.
GNT#	113	I	**Grant:** This signal indicates to the SB16C1053APCI from the arbiter that access to the PCI bus has been granted.

Other Interfaces			
Name	**Pin**	**Type**	**Description**
XTAL1	45	I	**Crystal or External Clock Input:** This clock input of serial channel.
XTAL2	46	O	**Crystal or Buffed Clock Output:** This output level is 3.3V.
WAKEREQ	107	I	**WAKE Request:** PM state of PCI Device goes from D3 state to D0 state with the Wake Up Event. This pin receives the event signal needed for the transition from D3 state to D0 state.
MCS3#_OSC2	54	I/O	This pin is dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive OSCx[2:0] input. After that, the pin is set to output mode and outputs MCS3# of MIO Bus™ in 4S and 6S mode. **Oscillator Setting 2:** This pin is working in input mode for a while after power is supplied to the chipset. In this time, oscillator setting is set and the value is adopted to Device Information Register in option register space. **MIO Bus Chip Select 3:** This output pin is a chip select signal of MIO Bus™ for the 4th external UART channel.
MCS2#_PORT1	55	I/O	This pin is dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive PORTx[2:0] input. After that, the pin is set to MCS2#(output) of MIO Bus™ in 4S and 6S mode. **MIO Bus Chip Select 1:** This output pin is a chip select signal of MIO Bus™ for the 2rndxternal UART channel. **Port Setting 1:** This input pin is a port setting for various operating modes.

MCS0#_PORT0	58	I/O	This pin is dual mode pin. After power is supplied to the chip, the pin is set to input mode for a while and receive PORTx[2:0] input. After that, the pin is set to MCS0# (output) of MIO Bus™ in 4S and 6S mode. **MIO Bus Chip Select 0:** This output pin is a chip select signal of MIO Bus™ for the 1st external UART channel. **Port Setting 0:** This input pin is a port setting for various operating modes.

Table 6–1: Pin Description…*continued*

Other Interfaces

Name	Pin	Type	Description
GPO[1] GPO[0]	105 106	O O	**General Purpose Output:** These output pins are controlled by GPOCR and GPODR of the Option register.

Power and Ground

Name	Pin	Type	Description
VCC	16, 39, 44, 67, 73, 85, 97, 122	PWR	**Power Supply:** Connect to +3.3V and to Core Ground through 0.1uF capacitors.
GND	15, 38, 47, 66, 72, 84, 96, 111, 121	GND	**Ground:** Connect to ground.
NC	57, 60, 108, 109, 110, 115, 116		**No Connect**

7. Configuration Loader

SB16C1053APCI can perform system initialization by reading PCI Configuration header data from Internal MIO registers or external serial EEPROM. It is decided to download configuration header data through exist of external serial ROM.

SB16C1053APCI use a 256K I^2C serial EEPROM (24C256). We recommend customers use ATMEL AT24C256 or CATALYST CA24C256 for external serial ROM

When SB16C1053APCI is reset after power is granted, Configuration Loader inside SB16C1053APCI loads Configuration header data and etc from internal MIO registers or external serial EEPROM depending on the existence of external EXT_LOAD pin. If external serial ROM exist, it reads saved data from external serial EEPROM and performs configuration. If external serial ROM do not exist, it reads data from internal MIO registers to perform the configuration.

7.1 MIO Register

When configuring SB16C1053APCI through MIO Register, Vendor ID is fixed to 14A1h which is the Vendor ID of SystemBase. Device ID varies depending on port inputs PORT[2:0] of SB16C1053APCI.

Table 7–1: SystemBase PCI Device ID

SystemBase PCI Device ID		
PORT[2:0]	Name	Device ID
000b (0h)	Serial 2-port Mode (2S)	4D02h
001b (1h)	Serial 6-port Mode (6S)	4D06h
011b (3h)	Parallel 1-port Mode (1P)	4301h
100b (4h)	Serial 1-port Mode (1S)	4D01h
110b (6h)	Serial 2-port, Parallel 1-port Mode (2S1P)	4303h
111b (7h)	Serial 4-port Mode (4S)	4D04h
others	Reserved	-

7.2 Serial EEPROM Information Table

When the Auto Load Tag of the external serial EEPROM have hexadecimal 55 (55h), SB16C1053APCI start to load configuration data from serial ROM. SB16C1053APCI can recognize whether external serial EEPROM exist or not using the AUTO Load Tag. Even though the external serial EEPROM is attached, if the value of AUTO Load Tag is not 55h, SB16C1053APCI think there is no external serial EEPROM on I^2C bus.

When the Auto Load Tag value is not 55h or there is no serial ROM, SB16C1053APCI didn't load any data from serial ROM. But SB16C1053APCI load the configuration data from MIO Register instead of external serial EEPROM.

Table 7–2: Serial EEPROM Information Table

Address	Description
00h	Auto Load Tag (55h)
01h	Vendor ID Low Byte
02h	Vendor ID High Byte
03h	Device ID Low Byte
04h	Device ID High Byte
05h	Revision ID
06h	Sub Vendor ID Low Byte
07h	Sub Vendor ID High Byte
08h	Sub System ID Low Byte
09h	Sub System ID High Byte
0Ah~	Reserved

Vendor ID : Represents manufacturer of device. It is a unique ID given by PCI SIG and must be downloaded from external serial EEPROM. If you do not own Vendor ID, you can use 14A1h given to SystemBase by PCI SIG with permission.

Device ID : A unique ID of each device and is assigned at manufacturer's discretion and must be downloaded from serial EEPROM. If you do not prepare Device ID, you can use SystemBase's OEM Device ID (please contact SystemBase's technical support).

Revision ID : It is a value representing device revision and must be downloaded from serial EEPROM. You have to use B0h.

Sub Vendor ID : Shows information about Subsystem manufacturer. Generally, Vendor ID or Device ID is information about Controller chip and Sub Vendor ID or Sub System ID is information about manufacturer who made the product with the chip.
It must be downloaded from serial EEPROM. If you do not prepare Sub Vendor ID, you can use 14A1h given to SystemBase by PCI SIG with permission.

Sub System ID : You can think of this as a Subsystem manufacturer's own Device ID.
It must be downloaded from serial EEPROM. If you do not prepare Device ID, you can use SystemBase's OEM Device ID (please contact SystemBase's technical support).

8. PCI Configuration Space

PCI Configuration offers one type of Configuration Space access method.

- PCI Compatible Configuration method

PCI Compatible Configuration method is compatible with PCI version 2.3 and higher and supports 100% binary compatibility to software for operating system agreed bus list and organization.

From 0 byte up to 256 bytes is called PCI Compatible Configuration Space

8.1 Configuration Space Map of SB16C1053APCI

Table 8–1: Configuration Space Map

	Bit[31:24]	Bit[23:16]	Bit[15:8]	Bit[7:0]
Reg00	Device ID		Vendor ID	
Reg04	Status Register		Command Register	
Reg08	Class Code			Revision
Reg0C	BIST	Header Type	Latency Timer	Cache Line Size
Reg10	BAR0 (1st UART)			
Reg14	BAR1 (2nd UART)			
Reg18	BAR2 (Parallel Basic Register or External UARTs)			
Reg1C	BAR3 (Parallel Extended Register or External UARTs)			
Reg20	BAR4 (Option Register)			
Reg24	BAR5 (Reserved)			
Reg28	CardBus CIS Pointer			
Reg2C	Subsystem ID		Subsystem Vendor ID	
Reg30	Expansion ROM BAR			
Reg34	Reserved			Cap. Pointer
Reg38	Reserved			
Reg3C	Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line
Reg40	Power Management Capability			
Reg44	Power Management Control & Status			
Reg48~FF	Reserved			

Configuration Space of SB16C1053APCI can be divided into 2 following functions.

- PCI Compatible Configuration Registers
- Power Management Registers

PCI Compatible Configuration Registers are from Reg00 to Reg3C and these parts are compatible with existing PCI Configuration Registers. Power Management Registers are from Reg40 to Reg44.

SB16C1053APCI uses Configuration Register of Header Type0 which is used as Endpoint.

Following is a detailed description of PCI Compatible Configuration Register.

8.1.1 Vendor ID

A 16-bit register which represents the manufacturer of the device.
It is a unique ID given by PCI SIG after membership registration. If you do not own a Vendor ID, it is fine to use 14A1h given to SystemBase by PCI SIG.

8.1.2 Device ID

A 16-bit unique ID of each device given by the Function Manufacturer which can be assigned by the manufacturer freely. It is related to software driver installation/recognition. Please refer the Device ID for each mode in Table 8-2.

Table 8–2: Command Register

PORT[2:0]	100	011	000	110	111	001
MODE	1S	1P	2S	2S1P	4S	6S
Device ID	4D01h	4301h	4D02h	4303h	4D04h	4D06h

8.1.3 Command Register

Table 8–3: Command Register

Bit	Type	Description
15:11	RO	**Reserved:** Hardwired to 00000b
10	RW	**Interrupt Disable:** This bit controls PCI function's INTx interrupt signal creation ability. When it is 0b, function can assert INTx interrupt signal. When it is 1b, function cannot assert INTx interrupt signal. Default value of this bit is 0b.
9	RO	Hardwired to 0b.
8	RW	**SERR Enable:** If this bit is set and the function detects a non-fatal error and a fatal error, error reporting is executed to Root Complex. You can set the kind of errors to report to Device Control Register. Default value is 0b.
7	RO	Hardwired to 0b.
6	RW	**Parity Error Response:** A Root Complex Integrated Endpoint that is not associated with a Root Complex Event Collector is permitted to hardwire this bit to 0b. Default value is 0b.
5	RO	Hardwired to 0b.
4	RO	Hardwired to 0b.
3	RO	Hardwired to 0b.
2	RW	Hardwired to 0b.
1	RW	**Memory Address Space Decoder Enable:** When this is 0b, Memory Decoder is disabled and Memory Transactions arriving to this device are responded with Completion of Unsupported Request state. When it is 1b, Memory Decoder is enabled and memory transactions arriving to this device are accepted and handled.
0	RW	**IO Address Space Decoder Enable:** When this is 0b, IO decoder is disabled and IO transactions arriving to this device are responded with Completion of Unsupported Request state. When it is 1b, IO decoder is enabled and IO transactions arriving to this device are accepted and handled.

8.1.4 Status Register

Table 8–4: Status Register

Bit	Type	Description
15	RW	**Detected Parity Error:** This bit must be set by the device whenever it detects a parity error, even if parity error handling is disabled. Default value of this bit is 0b.
14	RW	**Signaled System Error:** This bit must be set whenever the device asserts SERR#. Devices that will never assert SERR# do not need to implement this bit.
13	RW	**Received Master Abort:** This bit must be set by a master device whenever its transaction is terminated with Master-Abort. Default value of this bit is 0b.
12	RW	**Received Target Abort:** This bit must be set by a master device whenever its transaction is terminated with Target-Abort. Default value of this bit is 0b.
11	RW	**Signaled Target Abort:** This bit must be set by a target device whenever it terminates a transaction with Target-Abort. Default value of this bit is 0b.
10:9	RO	**DEVSEL Timing:** These bits encode the timing of DEVSEL#. These are encoded as 00b for fast, 01b for medium, and 10b for slow (11b is reserved). Hardwired to 01b.
8	RW	**Master Data Parity Error:** Set when three conditions are met: 1) the bus agent asserted PERR# itself (on a read) or observed PERR# asserted (on a write); 2) the agent setting the bit acted as the bus master for the operation in which the error occurred; and 3) the Parity Error Response bit (Command register) is set. Default value of this bit is 0b.
7	RO	**Fast Back-to-Back Capable:** Hardwired to 0b.
6	RO	**Reserved.** Hardwired to 0b.
5	RO	**66MHz-Capable:** Indicates whether or not this device is capable of running at 66MHz and hardwired to 1b.
4	RO	**Capabilities List:** Hardwired to 0b.
3	RO	**Interrupt Status:** Indicates that the function has an interrupt request that has not been processed yet. (Function is waiting to be serviced after asserting an interrupt signal. In other words, it is 1b when INTx# signal is asserted.)
2:0	RO	**Reserved.** Hardwired to 000b.

8.1.5 Revision

This register shows device revision. Manufacturer can assign it freely. It is also related to software device driver installation.

8.1.6 Class Code

This register contains descriptions on functions the device implements. It is divided as Base Class, Sub Class and Programming Interface in bytes. It must be set to the values provided by PCI Bus Specification.

When SB16C1053APCI works as a serial card on 1S, 2S, 2S1P, 4S and 6S mode, it gets 07_00_02h since it is a serial communication card adaptor. But when SB16C1053APCI works as a parallel card on 1P mode, it gets 07_01_03h since it is a parallel communication card adaptor.

Base Class Code is 07h(communication controller), Sub Class Code is 00h(serial controller) and Programming Interface is 02h(16C550 compatible).

Base Class Code is 07h(communication controller), Sub Class Code is 01h(parallel controller) and Programming Interface is 03h(IEEE 1284 controller).

8.1.7 Cache Line Size

This register assigns size of system's Cache Line. It is implemented as [RW] for compatibility with existing PCI. It is not supported and hardwired to 0000_0000b.

8.1.8 Latency Timer

This register assigns latency clock related to bus master which does burst access. It is not supported and hardwired to 0000_0000b.

8.1.9 Header Type

Configuration Space Header type and [RO]

Bit[7] : Shows whether device is Multi Function or Single Function. This product has default value 0b since it only supports Single Function.

Bit[6:0] : Assign header type after 10h. 00h is target device, 01h is PCI-to-PCI Bridge and 02h is CardBus bridge. This product has default value 00 since it is a target device.

8.1.10 BIST(Built-In Self Test)

Table 8–5: BIST

Bit	Type	Description
7	RO	**BIST Capable:** Hardwired to 0b.
6	RO	**Start BIST:** Hardwired to 0b.
5:4	RO	**Reserved:** Hardwired to 00b.
3:0	RO	**Completion Code:** Hardwired to 0000b.

8.1.11 Base Address Registers

These are spaces for assigning Base Address for accessing I/O device or memory on PCI Local Bus. There are 6 spaces Base Address Register from 0 to 5, but the Base Address Register 5 is set as unused and reserved area.

When Base Address Register Bit[0] is 0b, the space is used as Memory space and when 1b, it is used as I/O space. Therefore, Bit[0] of all used Base Address Register from 0 to 4 are all set to value 1b. All of these Base Address Register spaces are used as space for I/O.

Each Base Address Register spaces are assigned to internal UART, external UART or Parallel Port Register for each operational mode – 1P, 1S, 2S, 2S1P, 4S and 6S mode. *Please refer 'Table 8-6: Base Address Registers for each operational mode' for more detail.*

Table 8–6: Base Address Registers

PORT[2:0]	100	011	000	110	111	001
MODE	1S	1P	2S	2S1P	4S	6S
BAR0	UART0	-	UART0	UART0	UART0	UART0
BAR1	-	-	UART1	UART1	UART1	UART1
BAR2	-	Parallel0	-	Parallel0	UART2	UART2/3
BAR3	-	Parallel1	-	Parallel1	UART3	UART4/5
BAR4	OPTION	OPTION	OPTION	OPTION	OPTION	OPTION

8.1.11.1 Base Address Register 0

SB16C1053APCI has six operating modes. In 1S, 2S, 2S1P, 4S and 6S mode.

When SB16C1053APCI is in 1S, 2S, 2S1P, 4S and 6S mode, Base Address Register0(BAR0) automatically sets the size of the Address Space of internal 1st UART of SB16C1053APCI. The I/O address space is 00~07h.

When SB16C1053APCI is in 1P mode, BAR0 is not used.

8.1.11.2 Base Address register 1

When SB16C1053APCI is in 2S, 2S1P, 4S and 6S mode, Base Address Register1(BAR1) automatically sets the size of the Address Space of internal 2nd UART of SB16C1053APCI. The I/O address space is 00~07h.

When SB16C1053APCI is in 1S and 1P mode, BAR1 is not used.

8.1.11.3 Base Address register 2

When SB16C1053APCI is in 1P and 2S1P mode, Base Address Register2(BAR2) automatically sets the size of the Address Space of Parallel Port Register. The I/O address space is 00~07h.

When SB16C1053APCI is in 4S mode, Base Address Register2(BAR2) automatically sets the size of the Address Space of an external UART (3rd UART) Register. The I/O address space is 00~07h.

When SB16C1053APCI is in 6S mode, Base Address Register2(BAR2) automatically sets the size of the Address Space of two external UART(3rd & 4th UART) Register. The I/O address space is 00~0Fh.

When SB16C1053APCI is in 1S and 2S mode, BAR2 is not used.

8.1.11.4 Base Address register 3

When SB16C1053APCI is in 1P and 2S1P mode, Base Address Register3(BAR3) automatically sets the size of the Address Space of extended Parallel Port Register in ECP mode. The I/O address space is 00~07h.

When SB16C1053APCI is in 4S mode, Base Address Register3(BAR3) automatically sets the size of the Address Space of an external UART (4th UART) Register. The I/O address space is 00~07h.

When SB16C1053APCI is in 6S mode, Base Address Register3(BAR3) automatically sets the size of the Address Space of two external UART (5th & 6th UART) Register. The I/O address space is 00~0Fh.

When SB16C1053APCI is in 1S and 2S mode, BAR3 is not used.

8.1.11.5 Base Address register 4

SB16C1053APCI contains Option Registers area which controls overall operations of the SB16C1053APCI. The Option Registers are made by SystemBase for users to control and manage Serial Multi-Port more easily and conveniently. Users and software developers can easily control Serial Interface of Multi-Port with them.

SB16C1053APCI sets this area with Base Address Register4(BAR4). I/O Address space size of the Option I/O Register is from 00 to 1Fh. Option Registers indicate information on the six operating modes, the Line Interface through the pin, Oscillator, Interrupt and SB16C1053APCI. See *'**11. Option I/O Space**'* for more details.

8.1.11.6 Base Address register 5

In SB16C1053APCI, Base Address Register5(BAR5) is not used and reserved area.

8.2 Power Management Registers of SB16C1053APCI

Sometimes control over power is needed on PCI Bus applied systems. Especially in cases when system uses independent power source like mobile system or when PCI device uses a lot of power, the system must limit power supply to PCI device when it is not in use to make a power efficient system. For this reason, *'PCI specification provides Power Management Interface Specification'* making Power Management more convenient. SB16C1053APCI supports PCI Power Management Interface Specification Revision 1.2. Content of registers that are implemented at Configuration Space Header is shown below. See *'Power Management Spec. Rev 1.2'* for more details.

Table 8–7: Power Management Register Block

Reg40	Power management Capabilities (PMC)		Next Item Ptr	Capability ID
Reg44	Data	PMCSR_BSE Bridge Support Extensions	Power Management Control/Status Register (PMCSR)	

8.2.1 Capability ID (40h)

Capability ID regarding Power Management Interface and default value is 01h.

8.2.2 Pointer to Next Capability (41h)

A pointer that stores address of register which has information about next Capability. SB16C1053APCI does not have additional capabilities.
Hardwired to 0000_0000b.

8.2.3 Power Management Capabilities (42~43h)

Table 8–8: Power Management Capabilities

Bit	Type	Description
15:11	RO	**PME_Support:** These bits indicate the power states in which the function may assert PME#. SB16C1053APCI can assert PME# signal in $D3_{hot}$ and $D3_{cold}$ states and has value of 1_1000b.
10	RO	**D2_Support:** Tells if D2 Power Management State is supported. This device does not support D2 state and the bit is set to 0b.
9	RO	**D1_Support:** Tells if D1 Power Management State is supported. This device does not support D1 state and the bit is set to 0b.
8:6	RO	**Aux Current:** Report 3.3Vaux auxiliary current requirements for this device. This device is configured to require 375mA which is the maximum support capacity of an electric current supply and the bits are set to 111b.
5	RO	**Device-Specific Initialization (DSI):** Shows the need of DSI after transition from D3 to D0 uninitialized state. It should be set to 0b since initial value configuration for UART's communication is not needed here.
4	RO	**Reserved**
3	RO	**PME Clock:** Requires PCI clock to generate PME# signal and set to 0b.
2:0	RO	**Version:** Compatible with PCI PM Specification V1.2 and set to 011b.

8.2.4 Power Management Control/Status Register (44~45h)

This 16bit Register manages PCI Function's Power Management state and it is also used to enable and monitor PME.

Table 8–9: Power Management Control/Status Register

Bit	Type	Description
15	RW	**PME_Status:** This bit is set when the function would assert the PME# signal independent of the state of the PME_En bit. Writing 1b to this bit will clear it and cause the function to stop asserting a PME# (if enabled). Writing 0b has no effect. This bit is sticky and must be explicitly cleared by the OS each time the OS is initially loaded since this device supports PME# from $D3_{cold}$,
14:13	RO	**Data Scale:** This device did not implement Data Scale and hardwired to 00b.
12:9	RO	**Data Select:** This device did not implement Data Select and hardwired to 0000b
8	RW	**PME_En:** If PME_En is set to 1b, then the function can assert PME#. If PME_En is cleared to 0b, then the function do not assert PME#. This bit is sticky and must be explicity cleared by the OS each time the device is initially loaded since this device is supported to PME# from $D3_{cold}$.
7:4	RO	**Reserved:** Hardwired to 0b.
3	RO	**No_Soft_Reset:** Device does not execute internal reset when changing from $D3_{hot}$ to D0 through software control of PowerState bits. It's because full Re-Initialization is not needed for device to return to D0. Hardwired to 0b.
2	RO	**Reserved:** Hardwired to 0b.
1:0	RW	**PowerState:** PM S/W can decide Power Management state by configuring this section. PowerState = 00b means D0 state PowerState = 01b means D1 state PowerState = 10b means D2 state PowerState = 11b means $D3_{hot}$ state

9. Power Management

PCI was the most famous and useful bus since it was introduced in 1992. It is used in various computer systems from Laptops to Servers. It supported high performance applications by offering large bandwidth and efficiently supporting multiple masters. Also, it offers efficient power management through Power Management and various types of Form Factor modules and Applications.

PCI-PM defines four different Power States regarding PCI or PCI Express and interface for controlling these Power States. This device defines two different Power States.

Refer to '*PCI Bus Power Management Interface Specification Revision 1.2'* for more information on Power Management.

9.1 PCI Power Management

9.1.1 PCI Function Power State

4 power states are defined for PCI function. These are D0, D1, D2 and D3; D0 is maximum power consumption state and D3 is minimum power consumption state. D1 and D2 are middle states between D0(Power On) and D3(Power Off) and power consumption decreases as state changes to D3. As device changes from D0 to D3, it consumes lesser power and stores lesser Context information about current state. As a result, waiting time needed for the device to return to D0 increases.

D3 Power State organizes Special Category of Power Management State and Function can change to D3 state by physically removing Power from PCI device. D3 is classified into two states depending on existence or absence of Vcc. Those states are $D3_{hot}$ and $D3_{cold.}$

$D3_{hot}$ is the state where Vcc exist and it goes to maximum power-saving mode when both power and reference clock are supplied. When software writes D0 state on function's PMCSR register to get out of this mode, it can change into D0 state.

$D3_{cold}$ is classified into Power Off and Sleep state depending on existence of Vaux power.

At Power Off state, device's main power and Vaux are cut off and execution of Wake event is not possible. It is $D3_{cold}$ state. To get out of this state, push power button to start system.

At Sleep state, device's main power is cut off and only Vaux is supplied. It is $D3_{hot}$ state and Wake event can be executed. To get out of this state, assert PME# signal to Root Complex. The system wakeup by this can change to D0 state by re-assigning Vcc to this device and assigning RST#.

D0 state is classified into $D0_{uninitialized}$ and $D0_{active}$. $D0_{uninitialized}$ state is before system is initialized after Power has been supplied and $D0_{active}$ state is after system has been initialized.

All PCI function must support D0, $D3_{hot}$ and $D3_{cold.}$ SB16C1053APCI also support D0, $D3_{hot}$ and $D3_{cold}$ and do not support D1 and D2.

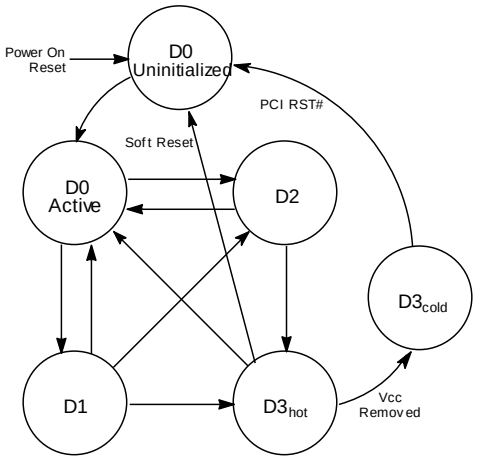


Figure 9–1: PCI Function Power Management State Transition

Cf. Hibernate state is variation of shutdown state. In this state, all states of computer is saved on disk and thus when power comes back, it can be started as current session.

9.2 SB16C1053APCI Power Management Pins and Functions

9.2.1 SB16C1053APCI Pins for Power Management

Table 9–1: SB16C1053APCI Pin Table for PM

Pin Name	Type	Description
WAKEREQ	I	Input of Wake Event Request. For example, it receives wakeup event signal generated by Ring Indicator.
PME#	O	Side band signal that Wakes Root Complex up to restore main power and reference clock that have been removed to implement Wakeup Event.
PME_S	O	Indicates PM state. It can be used for auto-power down function and it will be connected to FORCEOFF# pin on MAX3243.

9.2.2 SB16C1053APCI Power Management Wakeup implementation

Below figure is Logic Diagram of MAX3243. As you can see from this figure, RIN2 input signal (this pin is mainly prepared to be used by Ring Indicator.) is forked to reversed output signal called ROUT2 and output called ROUT2B. Among these, ROUT2B signal is not influenced by FORCEOFF# signal and input/output of buffer is not restricted. RIn input is screened by FORCEOFF# but above ROUT2B logic is Open when FORCEOFF# so RIn signal becomes an input without reversion. This signal is connected to WAKEREQ of SB16C1053APCI and handled as Wake Event. And if it is in $D3_{cold}$ state, this signal is asserted as PME# side band signal.

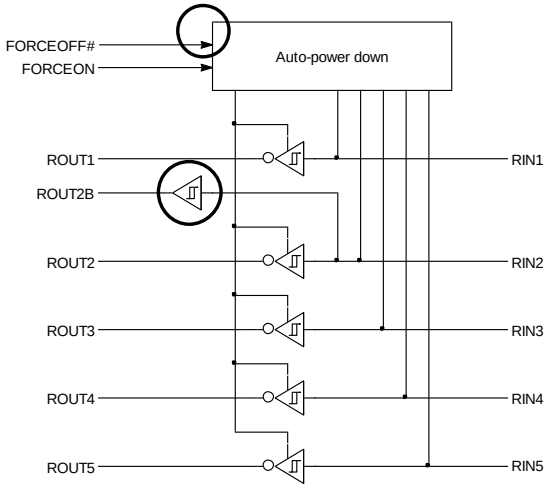
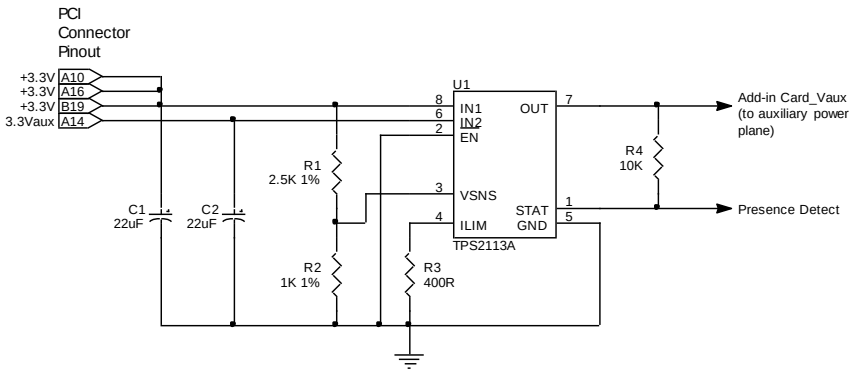


Figure 9–2: Logic Diagram of MAX3243

9.2.3 3.3Vaux Presence Detection & Power Routing

PCI Add-In Card that implements a function which can generate Power Management Event from D3cold must decide existence of 3.3V on Pin B10(3.3Vaux) of PCI Bus. When weak pull-down attached to Pin B10 is implemented on system board that does not support supply of 3.3Vaux, it should be there to make logic low reference and must be implemented in all Add-In Card. On systems that do not supply 3.3Vaux through Pin B10, PCI Add-In Card must use any voltage source that Add-In Card can supply to provide supply to Aux Power of its own. So depending on existence of 3.3Vaux of Pin B10, design a circuit that supplies Power to its Aux Power as shown above.



VSNS > 0.8V	3.3Vaux (IN2) > +3.3V (IN1)	Add-in Card_Vaux (OUT)
Yes	X	+3.3V
No	No	+3.3V
No	Yes	3.3Vaux

Figure 9–3: Sample Circuit for Aux Power Supply

10. Option I/O Space

Option I/O Space is determined by Base Address Register 4 (20h ~ 23h) from PCI Configuration Space.
Contents of I/O register which is installed in this area include basic information about PCI Multi Port hardware. The size of this area is 64(00h ~ 3Fh) bytes total.
This Option Registers are made by SystemBase for users to control and manage Serial Multi-Port more easily and conveniently. Users and software developers can easily control Serial Interface of Multi-Port with them.

Table 10-1: Option I/O Register Map

I/O Address	Register Name	I/O
00h	GIR0 (General Information Register 0 – Serial Port Number)	RO
01h	GIR1 (General Information Register 1 – Product Version)	RO
02h	GIR2 (General Information Register 2 – Parallel Port Number)	RO
03h	GIR3 (General Information Register 3 – PCI Core Version)	RO
03h	SRR (Software Reset Register)	WO
04h	DIR (Port1 ~ Port2, Device Information Register)	RO
05 ~ 07h	Reserved	-
08h	IIR0 (Port1 ~ Port2, Interface Information Register) IIR0 (Port1, Interface Information Register) for ALL+ mode	RW
09h	IIR1 (Port2, Interface Information Register) for ALL+ mode	RW
0A ~ 0Bh	Reserved	-
0Ch	IMR (Port1 ~ Port2, Interrupt Mask Register)	RW
0D ~ 0Fhh	Reserved	-
10h	IPR (Port1 ~ Port2, Interrupt Poll Register)	RO
11 ~ 13h	Reserved	-
14h	PPFTTR (Parallel Port FIFO TX Threshold Register)	RW
15h	PPFRTR (Parallel Port FIFO RX Threshold Register)	RW
16h	ATPSR (Auto Toggle Pin Select Register)	RW
17h	PPISR (Parallel Port Interrupt Status Register)	RO
18h	PPMRR (PM_PME Message Resource Register in D3hot)	RW
19 ~ 1Bh	Reserved	-
1Ch ~ 1Fh	Reserved	-
20h	GPOCR (General Purpose Output Control Register)	RW
21h	GPODR General Purpose Output Data Register)	RW
22h	Reserved	-
23h	PAFR (Parallel Additional Function Register)	RW
24 ~ 3Fh	Reserved	-

Cf. RO – Read Only
WO – Write Only
RW – Read/Write

10.1 General Information Register 0 – Serial Port Number (GIR0, BAR4+0h)

Serial Port Number: Shows how many ports are installed on current Serial Multi-Port.
1S mode has 01h(1 port). 2S and 2S1P mode have 02h(2 ports). 4S mode have 04h (4 ports). 6S mode have 06h (6 ports). Other mode except upper cases have 00h.

10.2 General Information Register 1 – Product Version (GIR1, BAR4+1h)

General Information Register1 indicates the version of PCI Target Controller.
GIR1 is C0h Currently and meaning ver. 12.0

10.3 General Information Register 2 – Parallel Port Number (GIR2, BAR4+2h)

General Information Register2 indicates the number of operational parallel port in SB16C1053APCI.
In case of 1P and 2S1P mode, it have 01h. It means SB16C1053APCI have one active parallel port.
In other case, it have 00h. It means SB16C1053APCI have none active parallel port.

10.4 General Information Register 3 – Core Version (GIR3, BAR4+3h)

General Information Register3 indicates SystemBase's PCI Core version (Currently 24h meaning 2.4)

10.5 Software Reset Register (SRR, BAR4+3h)

If 52h("R") is written on SRR, Reset is outputted to Serial Multi-Port I/O Bus and this means PCI UART goes to Reset state. If values other than 52h are written on SRR, Reset state is cleared.

10.6 Device Information Register (DIR, BAR4+4h)

DIR: Device information of Port1 ~ Port2

Table 10–2: Device Information Register Description

Bit	Symbol	Description
7:4	DIR[7:4]	**UART Select**: Content of U[2:0] shows type of UART. 0000b: 16C550 compatible UART (not supported any more) 0001b: 16C1050 compatible UART (not supported any more) 0010b: 16C1050A (9-bit and Enhanced Auto Toggle) 0011 ~ 1111b: Reserved
3:0	DIR[3:0]	**Oscillator Frequency Select:** O[3:0] shows frequency (max. communication speed) of clock source that is used. 0000b: 1.8432MHz (up to 115.2Kbps) 0001b: 3.6864MHz (up to 230.4Kbps) 0010b: 7.3728MHz (up to 460.8Kbps) 0011b: 14.7456MHz (up to 921.6Kbps) 0100b: 29.4912MHz (up to 1,8432Mbps) 0101 ~ 1111b: Reserved

10.7 Interface Information Register 0, 1 (IIR0, BAR4+8h/ IIR1, BAR4+9h)

IIR0 indicates interface information of serial ports in the Serial 1-port, Serial 2-port, Serial 4-port and Serial 6-port modes. In these modes, IIR1 is not used.

In Serial 1-port and Serial 2-port ALL mode, IIR0 indicates interface information of 1st Serial port and IIR1 indicates interface information of 2nd Serial port.

Table 10–3: Interface Information Register 0 Description

Bit	Symbol	Description
7	IIR0[7]	Hardwired to '0'
6:4	IIR0[6:4]	**Interface Type Indicator:** This interface information is set by INTF0[2:0] inputs. 000b: RS232 interface is selected. 100b: RS422 1:1 interface is selected. 101b: RS422 Multi-Drop interface is selected. 110b: RS485 Non-Echo interface is selected. 111b: RS485 Echo interface is selected. These fields indicate the interface type of all serial ports in the Serial 1-port, Serial 2-port, Serial 4-port and Serial 6-port normal modes. But they indicate the interface type of 1st serial port, 3rd serial port and 5th serial port in the Serial 1-port and Serial 2-port, Serial 4-port and Serial 6-port ALL modes by GPO[1:0]. This field is changed by GPO[1:0] control output values. Please refer the below Table.
3:0	IIR0[3:0]	Reserved. Hardwired to 0000b.

Table 10–4: Interface Information Register 0 in 4S and 6S ALL mode.

GPO[1:0] = 00b & 11b	GPO[1:0] = 01b	GPO[1:0] = 10b
It receives interface setting value for 1st serial port in 4S and 6S ALL mode.	It receives interface setting value for 3rd serial port in 4S and 6S ALL mode.	It receives interface setting value for 5th serial port in 6S ALL mode.

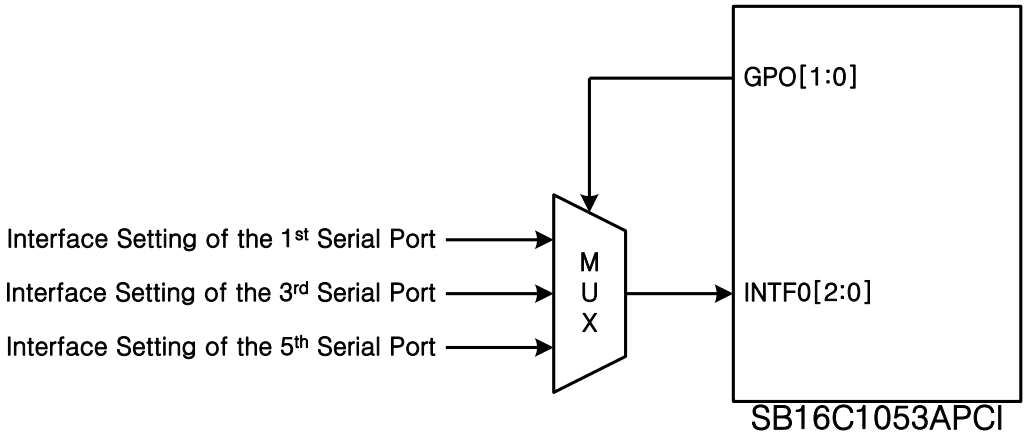


Figure 10–1: Selecting Interface Information, INTF0[2:0] using GPO[1:0]

Table 10–5: Interface Information Register 1 Description

Bit	Symbol	Description
7	IIR1[7]	Hardwired to '0'
6:4	IIR1[6:4]	**Interface Type Indicator:** This interface information is set by INTF1[2:0] inputs. 000b: RS232 interface is selected. 100b: RS422 1:1 interface is selected. 101b: RS422 Multi-Drop interface is selected. 110b: RS485 Non-Echo interface is selected. 111b: RS485 Echo interface is selected. These fields indicate the interface type of all serial ports in the Serial 1-port, Serial 2-port, Serial 4-port and Serial 6-port normal modes. But they indicate the interface type of 2nd serial port, 4th serial port and 6th serial port in the Serial 1-port and Serial 2-port, Serial 4-port and Serial 6-port ALL modes by GPO[1:0]. This field is changed by GPO[1:0] control output values. Please refer the below Table.
3:0	IIR1[3:0]	Reserved. Hardwired to 0000b.

Table 10–6: Interface Information Register 1 in 4S and 6S ALL mode.

GPO[1:0] = 00b & 11b	GPO[1:0] = 01b	GPO[1:0] = 10b
It receives interface setting value for 2nd serial port in 4S and 6S ALL mode.	It receives interface setting value for 4th serial port in 4S and 6S ALL mode.	It receives interface setting value for 6th serial port in 6S ALL mode.

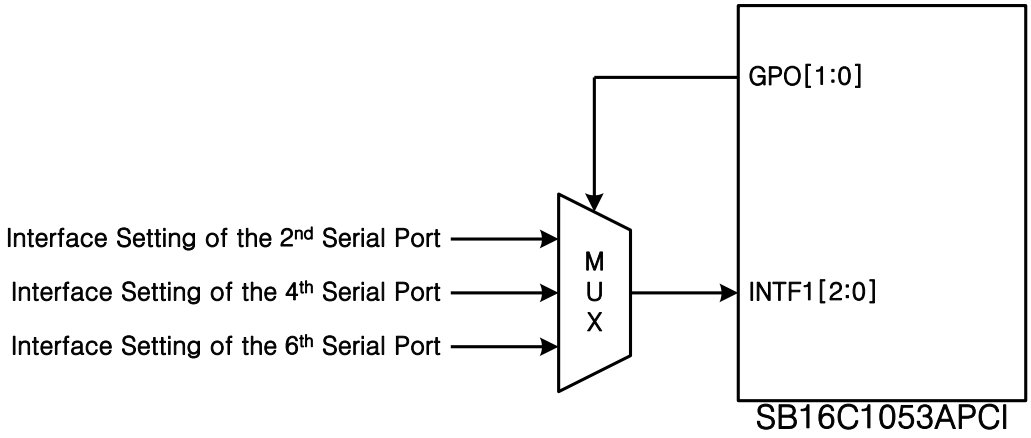


Figure 10–2: Selecting Interface Information, INTF1[2:0] using GPO[1:0]

10.8 Interrupt Mask Register (IMR, BAR4+Ch)

IMR enables or disables interrupt of serial ports and parallel port.

Table 10–7: Interrupt Mask Register Description

Bit	Symbol	Description
7	IMR[7]	Not used.
6	IMR[6]	**Interrupt Masking Bit for Parallel Port** 0b: Disables Parallel Port interrupt. (default) 1b: Enables Parallel Port interrupt.
5	IMR[5]	**Interrupt Masking Bit for 6th Serial Port** 0b: Disables Serial Port6(4th external UART) interrupt. (default) 1b: Enables Serial Port6(4th external UART) interrupt.
4	IMR[4]	**Interrupt Masking Bit for 5th Serial Port** 0b: Disables Serial Port5(3rd external UART) interrupt. (default) 1b: Enables Serial Port5(3rd external UART) interrupt.
3	IMR[3]	**Interrupt Masking Bit for 4th Serial Port** 0b: Disables Serial Port4(2nd external UART) interrupt. (default) 1b: Enables Serial Port4(2nd external UART) interrupt.
2	IMR[2]	**Interrupt Masking Bit for 3rd Serial Port** 0b: Disables Serial Port3(1st external UART) interrupt. (default) 1b: Enables Serial Port3(1st external UART) interrupt.
1	IMR[1]	**Interrupt Masking Bit for 2nd Serial Port** 0b: Disables Serial Port2(2nd internal UART) interrupt. (default) 1b: Enables Serial Port2(2nd internal UART) interrupt.
0	IMR[0]	**Interrupt Masking Bit for 1st Serial Port** 0b: Disables Serial Port1(1st internal UART) interrupt. (default) 1b: Enables Serial Port1(1st internal UART) interrupt.

10.9 Interrupt Poll Register (IPR, BAR4+10h)

IPR indicates interrupt generation state of Port 1 ~ Port 2.

Table 10-8: Interrupt Poll Register Description

Bit	Symbol	Description
7	IPR[7]	Not used.
6	IPR[6]	**Interrupt Polling Bit for Parallel Port** 0b: Parallel Port interrupt has occurred. 1b: Parallel Port interrupt has not occurred.
5	IPR[5]	**Interrupt Polling Bit for 6th Serial Port** 0b: Serial Port6(4th external UART) interrupt has occurred. 1b: Serial Port6(4th external UART) interrupt has not occurred.
4	IPR[4]	**Interrupt Polling Bit for 5th Serial Port** 0b: Serial Port5(3rd external UART) interrupt has occurred. 1b: Serial Port5(3rd external UART) interrupt has not occurred.
3	IPR[3]	**Interrupt Polling Bit for 4th Serial Port** 0b: Serial Port4(2nd external UART) interrupt has occurred. 1b: Serial Port4(2nd external UART) interrupt has not occurred.
2	IPR[2]	**Interrupt Polling Bit for 3rd Serial Port** 0b: Serial Port3(1st external UART) interrupt has occurred. 1b: Serial Port3(1st external UART) interrupt has not occurred.
1	IPR[1]	**Interrupt Polling Bit for 2nd Serial Port** 0b: Serial Port2(2nd internal UART) interrupt has occurred. 1b: Serial Port2(2nd internal UART) interrupt has not occurred.
0	IPR[0]	**Interrupt Polling Bit for 1st Serial Port** 0b: Serial Port1(1st internal UART) interrupt has occurred. 1b: Serial Port1(1st internal UART) interrupt has not occurred.

10.10 Parallel Port FIFO TX Threshold Register (PPFTTR, BAR4+14h)

Select the TX FIFO threshold of Parallel Port in ECP mode.

Table 10–9: Parallel Port FIFO TX Threshold Register Description

Bit	Symbol	Description
7:4	PPFTTR[7:4]	Reserved.
3:0	PPFTTR[3:0]	Select TX FIFO threshold of Parallel Port in ECP from 0 to 15.

10.11 Parallel Port FIFO RX Threshold Register (PPFRTR, BAR4+15h)

Select the RX FIFO threshold of Parallel Port in ECP mode.

Table 10–10: Parallel Port FIFO RX Threshold Register Description

Bit	Symbol	Description
7:4	PPFRTR[7:4]	Reserved.
3:0	PPFRTR[3:0]	Select RX FIFO threshold of Parallel Port in ECP from 0 to 15.

10.12 Auto Toggle Pin Select Register (ATPSR, BAR4+16h)

Select a pin for auto toggle function among DTR#, RTS# and TXEN/RXEN#.

Table 10–11: Auto Toggle Pin Select Register Description

Bit	Symbol	Description
7:2	ATPSR[7:2]	Reserved. Hardwired to '000000'
1:0	ATPSR[1:0]	00b : Auto Toggle function is disabled. (default) In this case, DTR# works as Data Terminal Ready, RTS# as Request To Send, TXEN as TXRDY#(TX Data Ready) and RXEN# as RXRDY#(RX Data Ready). 01b : Use DTR# as Auto Toggle pin for RS422 and RS485. In this case, DTR# works as the auto toggle pin, TXEN. TXEN works as TXRDY#(TX Data Ready) and RXEN# as RXRDY#(RX Data Ready). 10b : Use RTS# as Auto Toggle pin for RS422 and RS485. In this case, RTS# works as the auto toggle pin, TXEN. TXEN works as TXRDY#(TX Data Ready) and RXEN# as RXRDY#(RX Data Ready). 11b : Use TXEN and RXEN# as Auto Toggle pin for RS422 and RS485. These two pins are dedicated pins for auto toggle function. In this case, DTR# works as Data Terminal Ready, RTS# as Request To Send.

10.13 Parallel Port Interrupt Status Register (PPISR, BAR4+17h)

It indicates interrupt status of parallel port.

Table 10–12: Parallel Port Interrupt Status Register Description

Bit	Symbol	Description
7:5	PPISR[7:5]	Reserved. Hardwired to 000b.
4	PPISR[4]	**COMP_NACK** (This bit is adopted in Compatibility mode) When CTRL[4] (Control Register Bit 4 of Parallel Port) is 1b, if NACK is 0b, it is set to 1b. It indicates NACK interrupt occurred. When NACK is 1b or CTRL[4] is 0b, it is cleared to 0b. If parallel port isn't in COMP mode or CTRL[4] is 0b in COMP mode, this bit always have 0b.
3	PPISR[3]	**ECP_NACK** (This bit is adopted in ECP mode) When CTRL[4] (Control Register Bit 4 of Parallel Port) is 1b, if NACK is changed from 0b to 1b, it is set to 1b. It indicates NACK interrupt occurred. When PPISR is read or CTRL[4] is wrote by 0b, it is cleared to 0b. If parallel port isn't in ECP mode or CTRL[4] is 0b in ECP mode, this bit always have 0b.
2	PPISR[2]	**ECP_NERR** (This bit is adopted in ECP mode) When ECR[4] (Extended Control Register Bit 4 of Parallel Port) is 0b, if NERR is changed from 1b to 0b, it is set to 1b. It indicates NERR interrupt occurred. When PPISR is read or ECR[4] is wrote by 1b, it is cleared to 0b. If parallel port isn't in ECP mode or ECR[4] is 1b in ECP mode, this bit always have 0b.
1	PPISR[1]	**ECP_RX** (This bit is adopted in ECP mode) When ECR[2] (Extended Control Register Bit 2 of Parallel Port) is 0b, if the stacked data in RX FIFO is more than RX FIFO trigger level, this bit is set to 1b. It indicates RX interrupt occurred. When PPISR or ECR is read or ECR[2] is wrote by 1b, it is cleared to 0b. If parallel port isn't in ECP mode or ECR[2] is 1b in ECP mode, this bit always have 0b.
0	PPISR[0]	**ECP_TX** (This bit is adopted in ECP mode) When ECR[2] (Extended Control Register Bit 2 of Parallel Port) is 0b, if the stacked data in TX FIFO is more than TX FIFO trigger level, this bit is set to 1b. It indicates TX interrupt occurred. When PPISR or ECR is read or ECR[2] is wrote by 1b, it is cleared to 0b. If parallel port isn't in ECP mode or ECR[2] is 1b in ECP mode, this bit always have 0b.

10.14 PM_PME Message Resource Register (PPMRR, BAR4+18h)

Select event signal to wakeup Root Complex in $D3_{hot}$ state.

Table 10–13: PME# Signal Resource Register Description

Bit	Symbol	Description
1	PSRR[1]	**D3hot-I** 0b: Interrupt is not selected as Wakeup Event for waking up Root Complex (default). 1b: Interrupt is selected as Wakeup Event for waking up Root Complex. Whether interrupt is generated or not is determined by IMR. That is, some port can only generate interrupt or any ports among all ports can generate interrupt. When interrupt occurs, asserts PME# signal to Root Complex.
0	PSRR[0]	**D3hot-W** 0b: WAKEREQ pin is not selected as Wakeup Event for waking up Root Complex (default). 1b: WAKEREQ pin is selected as Wakeup Event for waking up Root Complex. When 1b is received by any logic, asserts PME# signal to Root Complex. If PSRR[1:0] is set as 11b which means both $D3_{hot}$-Interrupt and $D3_{hot}$-WAKEREQ are set, PME# signal is asserted when only one of both events occurs.

10.15 General Purpose Outputs Control Register (GPOCR, BAR4+20h)

GPOCR enables or disables GPO[7:0] to output ports respectively.

Table 10–14: GPIO Output Enable Register Description

Bit	Symbol	Description
7	GPOCR[7]	0b: DIR or MA0 isn't output through DIR_INTF12_MA0 pin. (default). 1b: DIR or MA0 is output through DIR_INTF12_MA0 pin. It is decided by PORT[2:0] which signal is output.
6	GPOCR[6]	0b: RXEN1# isn't output through RXEN1#_INTF11 pin. (default). 1b: RXEN1# is output through RXEN1#_INTF11 pin.
5	GPOCR[5]	0b: TXEN1 isn't output through TXEN1_INTF10 pin. (default). 1b: TXEN1 is output through TXEN1_INTF10 pin.
4	GPOCR[4]	0b: PMES isn't output through PMES_INTF02 pin. (default). 1b: PMES is output through PMES_INTF02 pin.
3	GPOCR[3]	0b: RXEN0# isn't output through RXEN0#_INTF01 pin. (default). 1b: RXEN0# is output through RXEN0#_INTF01 pin.
2	GPOCR[2]	0b: TXEN0 isn't output through TXEN0_INTF00 pin. (default). 1b: TXEN0 is output through TXEN0_INTF00 pin.
1	GPOCR[1]	0b: GPOD1 isn't output through OEM#_GPO1 pin. (default). 1b: GPOD1 is output through OEM#_GPO1 pin.
0	GPOCR[0]	0b: GPOD0 isn't output through GPO0 pin. (default). 1b: GPOD0 is output through OEM#_GPO0 pin.

Cf. GPOCR[1:0] are outputs as General Purpose Outputs Control for selecting Serial Interface Information. Please refer Table 10-4 and Table 10-5.

10.16 General Purpose Outputs Data Register (GPODR, BAR4+21h)

GPODR sets output value of GPO[1:0] respectively.

Table 10–15: General Purpose Outputs Data Register Description

Bit	Symbol	Description
1	GPODR[1]	The output value of GPO1 pin.
0	GPODR[0]	The output value of GPO0 pin.

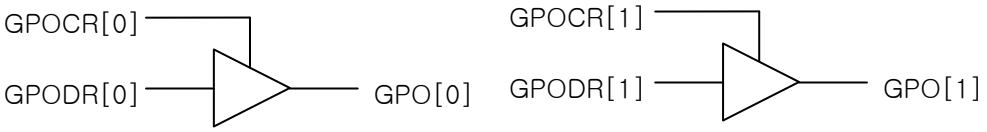


Figure 10-3: GPO[1:0] outputs are controlled by GPOCR and GPODR

10.17 Parallel Additional Function Register (PAFR, BAR4+23h)

Controls the Host Logic High output pin and monitors Peripheral Logic High input pin.

Table 10–16: Parallel Additional Function Register Description

Bit	Symbol	Description
1	PAFR[1]	**P_LOGIC_H** : Peripheral Logic High This bit indicates status of the peripheral device. (Read Only) 0b : It means the peripheral device isn't in the stable status and have some problems after power is supplied. 1b : It means the peripheral device is in the stable status after power is supplied.
0	PAFR[0]	**H_LOGIC_H** : Host Logic High This bit indicates status of the host device. (Read/Write) When the host don't go into the stable status and have some problems after power is supplied, host output 0b through H_LOGIC_H pin. When the host goes into the stable status after power is supplied, host output 1b through H_LOGIC_H pin.

11. UART(SB16C1050A) Functional Description

SB16C1050A offers 16C450 and 16C650 modes. When FIFO is enabled, it has a register configuration compatible with 64-byte FIFO and 16C650, so it becomes compatible with 16C650. If you enable 256-byte FIFO, you use the unique supreme function that SB16C1050A offers. It offers communication speed up to 5.3Mbps and more enhanced functions that other UARTs with 128-byte FIFO do not.
SB16C1050A can select hardware/software flow control. Hardware flow control significantly reduces software overhead and increases system efficiency by automatically controlling serial data flow using the RTS# output and CTS# input signals. Software flow control automatically controls data flow by using programmable Xon/Xoff characters.

UART I/O Space is determined by Base Address Register 0 (10h ~ 13h) from PCI Configuration Space. This is BAR0 area of Configuration Space and is for accessing actual physical UARTs.

11.1 UART I/O Address Map

8 bytes per Port are assigned since the type of installed UART is 16C550 compatible device.
Each BAR is assigned for each UART port in the SB16C1053APCI. Each UART can be assigned each COM Port number. For example, 1st UART Port can be assigned to COM11 and 2nd UART Port can be assigned to COM15. But this port remap is not possible in the 6S mode of SB16C1053APCI. Because BAR2 and BAR3 include each two UART ports in the space.
For more information, please refer *'Table 8-6, Base Address Registers'*.

Table 11–1: UART I/O Address Map

BAR	I/O Address	1-serial Mode	2-serial Mode
BAR0	00 ~ 07h	UART0	UART0
BAR1	00 ~ 07h	-	UART1

11.1 FIFO Operation

SB16C1050A's FIFO has two modes, 64-byte FIFO mode and 256-byte FIFO mode. Setting FCR[0] to 1b enables FIFO, and if AFR[0] is set to 0b, it operates in 64-byte FIFO mode(default). In this mode, Transmit Data FIFO, Receive Data and Receive Status FIFO are 64 bytes. 64-byte FIFO mode allows you to select the Transmit Interrupt Trigger Level from 8, 16, 32, or 56. You can verify this Interrupt Trigger Level by TTR and RTR. In this mode TTR and RTR are Read Only.

And by FCR[5:4], XOFF Trigger Level can be selected to either 8, 16, 56, or 60, and XON Trigger Level to either 0, 8, 16, or 56 by FCR[7:6]. You can verify XON and XOFF Trigger Level by FUR and FLR. In 64-byte FIFO mode TTR and RTR are Read Only.

If you select 256-byte FIFO mode, you can experience more powerful features of SB16C1050A. Setting both FCR[0] and AFR[0] to 1b will enable this mode. In this mode, Transmit Data FIFO, Receive Data and Receive Status FIFO are 256 bytes. Interrupt Trigger Level and XON, XOFF Trigger Level are controlled by TTR, RTR, FUR and FLR, not by FCR[7:4]. That is, TTR, RTR, FUR and FLR can both read and write. You can verify free space of Transmit FIFO and the number of characters received in Receive FIFO by TCR, RCR and ISR[7:6].

While TX FIFO is full, the value sent to THR by CPU disappears. And while RX FIFO is full, the data coming from external devices disappear as well, provided that flow control function is not used.

For more information, refer to Register Description.

11.2 Hardware Flow Control

Hardware flow control is done by Auto-RTS and Auto-CTS. Auto-RTS and Auto-CTS can be enabled/disabled independently by programming EFR[7:6]. If Auto-RTS is enabled, it reports that it cannot receive more data by asserting RTS# when the amount of received data in RX FIFO exceeds the written value in FUR. Then after the data stored in RX FIFO is read by CPU, it reports that it can receive new data by deasseting RTS# when the amount of existing data in RX FIFO is less than the written value in FLR. When Auto-CTS is enabled and CTS# is cleared to 0b, transmitting data to TX FIFO has to be suspended because external device has reported that it cannot accept more data. When data transmission has been suspended and CTS# is set to 1b, data in TX FIFO is retransmitted because external device has reported that it can accept more data. These operations prevent overrun during communication and if hardware flow control is disabled and transmit data rate exceeds RX FIFO service latency, overrun error occurs.

11.2.1 Auto-RTS

To enable Auto-RTS, EFR[6] should be set to 1b. Once enabled, RTS# outputs 0b. If the number of received data in RX FIFO is larger than the value stored in FUR, RTS# will be changed to 1b and if not, holds 0b. This state indicates that RX FIFO can accept more data. After RTS# changed to 1b and reported to the CPU that it cannot accept more data, the CPU reads the data in RX FIFO and then the amount of data in RX FIFO reduces. When the amount of data in RX FIFO equals the value written in FLR, RTS# changes to 0b and reports that it can accept more data. That is, if RTS# is 0b now, RTS# is not changed to 1b until the amount in RX FIFO exceeds the value set in FUR. But if RTS# is 1b now, RTS# is not changed to 0b until the amount in RX FIFO equals the value written in FLR.

The value of FUR and FLR is determined by FIFO mode. If FCR[7:6] holds 00b, ’01’, ‘10’, and 11b, FUR stores 8, 16, 56, and 60, respectively. And if FCR[5:4] holds 00b, ’01’, ‘10’, and 11b, FLR stores 0, 8, 16, and 56, respectively in 64-byte FIFO. In 256-byte FIFO mode, users can write FUR and FLR values as they want and use them. But the value of FUR must be larger than that of FLR. While Auto-RTS is enabled, you can verify if RTS# is 0b or 1b by FSR[5]. If FSR[5] is 0b, RTS# is 0b and if 1b, RTS# is

1b, too.

When IER[6] is set to 1b and RTS# is changed from 0b to 1b by Auto-RTS function, interrupt occurs and it is displayed on ISR[5:0]. Interrupts by Auto-RTS function are removed if MSR is read. RTS# is changed from 0b to 1b after the first STOP bit is received. Figure 11−1 shows the RTS# timing chart while Auto-RTS is enabled.

In Figure 11−1, Data Byte n-1 is received and RTS# is deasserted when the amount of data in RX FIFO is larger than the value written in FUR. UART completes transmitting new data (DATA BYTE n) which has started being transmitted even though external UART recognizes RTS# has been deasserted. After that, the device stops transmitting more data. If CPU reads data of RX FIFO, the value of RCR decreases and then if that value equals that of FLR, RTS# is asserted for external UART to transmit new data.

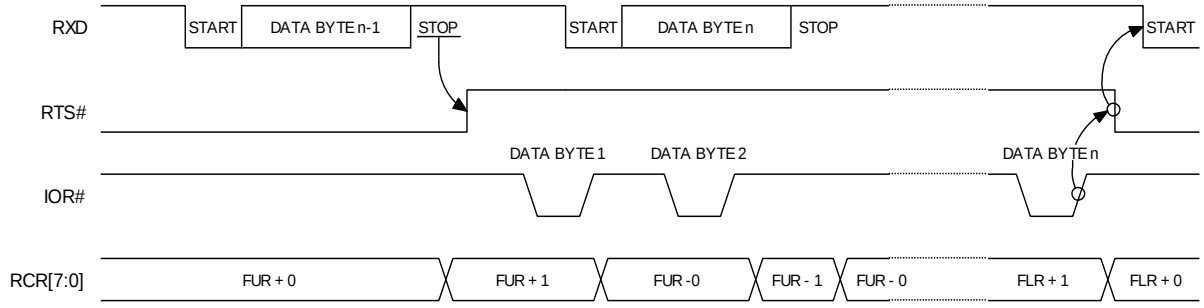


Figure 11−1: RTS# Functional Timing

11.2.2 Auto-CTS

Setting EFR[7] to 1b enables Auto-RTS. If enabled, data in TX FIFO are determined to be transmitted or suspended by the value of CTS#. If 0b, it means external UART can receive new data and data in TX FIFO are transmitted through TXD pin. If 1b, it means external UART can not accept more data and data in TX FIFO are not transmitted. But data being transmitted by then complete transmission. These procedures are performed irrespective of FIFO modes. While Auto-CTS is enabled, you can verify the input value of CTS# by FSR[1]. If 0b, CTS# is 0b and it means external UART can accept new data, If 1b, CTS# is 1b and it means external UART can not accept more data and data in TX FIFO are not being transmitted. If IER[7] is set to 1b, interrupt is generated by Auto-CTS when the input of CTS# is changed from 0b to 1b, and it is shown on ISR[5:0]. Interrupts generated by Auto-CTS are removed if MSR is read.

11.3 Software Flow Control

Software flow control is performed by Xon and Xoff character transmitting/accepting. Software flow control is enabled/disabled independently by programming EFR[3:0] and MCR[6:5, 2]. If TX software flow control is enabled by EFR[3:2], Xoff character is transmitted to report that data can not be accepted when the stored amount of data in RX FIFO exceeds the value in FUR. After the CPU reads the data in RX FIFO and if the read amount is less than the value in FLR, Xon character is transmitted to report that more data can be accepted. If TX software flow control is enabled by EFR[1:0] and Xoff character is inputted through RXD pin, it means no more data can be accepted, and data transmission is suspended even though data are in TX FIFO. If Xon character is received through RXD pin while data transmission is suspended, it means more data can be accepted, and therefore data in TX FIFO are re-transmitted. These procedures prevent overruns during communication. If software flow control is disabled, overrun occurs when the transmit data rate exceeds RX FIFO service latency. Different combinations of software flow control can be enabled by setting different combinations of EFR[3:0] . Table 11−2 shows software flow control options.

Table 11−2: Software flow control options (EFR[3:0])

EFR[3]	EFR[2]	EFR[1]	EFR[0]	TX, RX software flow controls
0	0	X	X	No transmit control
1	0	X	X	Transmit Xon1/Xoff1
0	1	X	X	Transmit Xon2/Xoff2
1	1	X	X	Transmit Xon1, Xon2/Xoff1, 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
X	X	1	1	Receiver compares Xon1, Xon2/Xoff1, Xoff2
0	0	0	0	No transmit control, No receive flow control
0	0	1	0	No transmit control, Receiver compares Xon1/Xoff1
0	0	0	1	No transmit control, Receiver compares Xon2/Xoff2
0	0	1	1	No transmit control, Receiver compares Xon1, Xon2/Xoff1, Xoff2
1	0	0	0	Transmit Xon1/Xoff1, No receive flow control
1	0	1	0	Transmit Xon1/Xoff1, Receiver compares Xon1/Xoff1
1	0	0	1	Transmit Xon1/Xoff1, Receiver compares Xon2/Xoff2
1	0	1	1	Transmit Xon1/Xoff1, Receiver compares Xon1, Xon2/Xoff1, Xoff2
0	1	0	0	Transmit Xon2/Xoff2, No receive flow control
0	1	1	0	Transmit Xon2/Xoff2, Receiver compares Xon1/Xoff1
0	1	0	1	Transmit Xon2/Xoff2, Receiver compares Xon2/Xoff2
0	1	1	1	Transmit Xon2/Xoff2, Receiver compares Xon1, Xon2/Xoff1, Xoff2
1	1	0	0	Transmit Xon2/Xoff2, No receive flow control
1	1	1	0	Transmit Xon2/Xoff2, Xoff2, Receiver compares Xon1/Xoff1
1	1	0	1	Transmit Xon1, Xon2/Xoff1, Xoff2, Receiver compares Xon2/Xoff2
1	1	1	1	Transmit Xon1, Xon2/Xoff1, Xoff2, Receiver compares Xon1, Xon2/Xoff1, Xoff2

11.3.1 Transmit Software Flow Control

To make Transmit Software Flow Control enabled, EFR[3:2] must be set to 01b, 10b or 11b. Unlike Auto-RTS in which 0b is outputted on RTS# when TX software flow control function is enabled, Xon character is not transmitted at first. If the amount of data in RX FIFO (written in ISR[6] and RCR) is less than the value in FUR, Xon character is not transmitted because Xon is in initial state. But if the amount of data in RX FIFO exceeds the value in FUR, Xoff character is transmitted immediately. Transmitting Xoff character means no more data can be accepted and after CPU reads data in RX FIFO, data in RX FIFO decreases. When the amount of data in RX FIFO is same as the value of FLR, Xon character is transmitted and it means reporting to external UART that it can accept more data. After transmitting Xoff character, Xon character is not transmitted until the amount of data in RX FIFO is same as the value of FLR.

The value of FLR is determined by FIFO mode. If FCR[7:6] is 00b, 01, 10 , and 11b, FUR is 8, 16, 56, and 60, respectively. And if FCR[5:4] is 00b, 01b, 10b, and 11b, FLR is 0, 8, 16, and 56, respectively in 64-byte FIFO. In 256-byte FIFO mode, users can input values in FUR and FLR as they want and use them. But the value in FUR must be larger than that of FLR. While TX software flow control is active, its status (if Xon or Xoff) can be verified by FSR[4]. If FSR[4] is 0b, the status is Xon and if 1b, the status is Xoff. It can be verified by FSR[4] only. And for there is no condition to generate interrupt, interrupt doesn't occur. It is different from that interrupt is generated by IER[5] when RX software flow control is enabled.

11.3.2 Receive Software Flow Control

To make Receive Software Flow Control enabled, EFR[1:0] must be set to 01b, 10b or 11b. When enabled, data in TX FIFO are determined to be transmitted or suspended by incoming Xon/Xoff characters. If Xon character is received, it means external UART can accept new data, and data in TX FIFO are transmitted through TXD pin. If Xoff character is received, it means external UART can not accept more data, and data in TX FIFO are not transmitted. But data being transmitted by that time are completely transmitted. These procedures are performed irrespective of FIFO modes. While Receive Software Flow Control is enabled, you can verify if the RX Software Flow Control status is XON or XOFF by FSR[0]. If it is 0b, RX Software Flow Control status is XON and it means external UART can accept new data. If 1b, RX Software Flow Control status is XOFF and it means external UART can not accept more data and data in TX FIFO are not being transmitted. If IER[5] is set to 1b, interrupt is generated when Xoff character is received and it is shown on ISR[5:0]. Interrupts generated by RX Software Flow Control are removed if ISR is read or Xon character is received.

General problems in using XON/XOFF function and tips for using Xon/Xoff character as one character are as follows.

- When RX Software Flow Control and Auto-CTS are enabled, LSR's Transmit Empty Bit and Transmit Holding Empty Bit are not affected even though RX Flow Control status is XOFF or 1b is inputted on CTS# pin, so data in TX FIFO are suspended. That is, these two bits are set to 1b if there is space available in TX FIFO.
- Xon/Xoff character which generated parity error are treated as normal Xon/Xoff

character.

- If Xon and Xoff character are set to same, both characters are treated as Xon character.

Tips for using Xon/Xoff character as two characters are as follows.

- If received characters are Xon1, Xon1 and Xon2, RX flow control status becomes XON and previous Xon1 is ignored.
- If received characters are Xoff1, Xoff1 and Xoff2, RX flow control status becomes XOFF and previous Xoff1 is ignored.
- If received characters are repeated as Xon1 Xoff1, Xon1 and Xoff1, there is no effect in RX flow control status and these characters are not treated as data. But if received characters are Xon1 Xoff1, Xon1, Xoff1, Xon1 and Xon2, RX flow control status becomes XON.
- If received characters are Xon1 Xoff1, Xon1, Xoff1 and Xoff2, RX flow control status becomes XOFF.
- If Xon1 and Xoff1 characters do not precede Xon2 and Xoff2, Xon2 and Xoff2 are treated as data and stored in RX FIFO.
- If Xon1 is not accompanied with Xon2 or Xoff1 character, it is treated as data and stored in RX FIFO.
- If Xoff1 is not accompanied with Xoff2 or Xon1 character, it is treated as data and stored in RX FIFO.

  As seen before, if received characters are Xon1, Xoff2, Xon2 or Xoff1, Xon2, Xoff2, these characters are all treated as data and stored in RX FIFO.

If characters are arrived continuously like Xon1, Xon2 or Xoff1, Xoff2, descriptions are as follows.

- If Xon1, Xon2 characters and Xoff1, Xoff2 characters are same with each other, all characters are treated as normal XON and XOFF characters.
- If Xon1, Xoff1 characters and Xon2, Xoff2 characters are same with each other, these are treated as normal XON characters.
- If Xon1, Xon2, Xoff1 characters are same and Xoff2 is different, these are treated as normal XON, XOFF characters.
- If Xon1, Xon2, Xoff2 characters are same and Xoff1 is different, these are treated as normal XON, XOFF characters.
- If Xon2, Xoff1, Xoff2 characters are same and Xon1 is different, these are treated as normal XON, XOFF characters.
- If Xon1, Xoff1, Xoff2 characters are same and Xon2 is different, these are treated as normal XON, XOFF characters.
- If Xon2, Xoff1 characters are same and Xon1, Xoff2 are different, these are treated as normal XON, XOFF characters.
- If Xon1, Xon2, Xoff1, Xoff2 are all same, these are treated only as normal XON characters.

  In all these cases no XON/XOFF characters are treated as data.
  Refer to Table 11−3 below.

Table 11−3: Xon/Xoff Character Recognition Logic Table

Xon1 Char.	Xon2 Char.	Xoff1 Char.	Xoff2 Char.	Recognition of Xon Char.	Recognition of Xoff Char.
11h	11h	13h	13h	Yes	Yes
11h	13h	11h	13h	Yes	No
11h	11h	11h	13h	Yes	Yes
11h	11h	13h	11h	Yes	Yes
11h	13h	13h	13h	Yes	Yes
11h	13h	11h	11h	Yes	Yes
11h	13h	13h	14h	Yes	Yes
11h	11h	11h	11h	Yes	No

When XON/XOFF software flow control function and Xon Any function is enabled, descriptions are as follows.

If Xon, Xoff characters are used as one character,

- If Xoff character arrives during XON status, status changes to XOFF.
- If Xon character arrives during XOFF status, status changes to XON.
- If Xoff character arrives during XOFF status, status changes to XON but Xoff character is not treated as data.

If Xon, Xoff characters are used as two characters,

- If only Xon1 or Xon1 + Xon2 character arrives during Xoff status, status changes to Xon and all characters are not treated as data.
- If only Xon2 character arrives during Xoff status, status changes to Xon and Xon2 character is treated as data and stored in RX FIFO.
- If Xoff1 + Xoff2 character arrives during XON status, status changes to XON.
- If Xoff1 + Xoff2 character arrives during XOFF status, status is changed to XON by Xoff1 and changed to XOFF again by Xoff2.

When Software flow control function and Special character function is enabled, descriptions are as follows.

- If Xoff1 character is used as Software flow control character, character in Xoff2 Register is recognized as Special character.
- If Xoff2 character is used as Software flow control character, it is not recognized as Special character but as Xoff character because both are same.
- If Xoff1, Xoff2 character is sequential and Xoff1 + Xoff2 character is used as Software flow control character, it is not recognized as Special character but as Xoff2 character because both are same.
- If Xoff1 + Xoff2 character is used as Software flow control character and Xoff2 character which does not follow after Xoff1 character arrives, it is not recognized as Xoff2 character but as Special character even though both are same.

11.3.3 Xon Any Function

While RX Software flow control function is enabled, data in TX FIFO are transmitted when received Xon character and transmission is suspended when Xoff character is received. This status is called 'XOFF status'. Transmission is re-started when status changes to 'XON status' by incoming Xon character or Xon Any function that changes status when any data arrives. Xon Any function is enabled if MCR[5] is set to 1b. While it is enabled, XOFF status changes to XON status though Xoff character arrives.
Details about it are described in 11.3.2 Receive Software Flow Control.

11.3.4 Xoff Re-transmit Function

While TX Software flow control function is active, Xoff character is transmitted when the amount of data in RX FIFO exceeds the value of FUR. Though it received Xoff character, external UART may not recognize this character for some reason and continue to transmit data. Under TX Software flow control, because Xoff character had been transmitted once before, it is not transmitted again though more data arrive. In this situation, overflow may occur in RX FIFO. Conventional UARTs can not deal this situation but SB16C1050A does with Xoff Re-transmit function.

Xoff Re-transmit function transmits Xoff character again when more data arrives from external UART though it transmitted Xoff character before. By this function the external UART can recognize Xoff character and stop transmitting data though it didn't recognize the Xoff character before.

There are four Xoff Re-transmitting settings by XRCR[1:0]. Xoff character can be re-transmitted when every 1, 4, 8 or 16 data arrives in XOFF status.

If XRCR[1:0] is 00b, Xoff character is re-transmitted whenever 1 more data arrives in XOFF status. If XRCR[1:0] is '01', Xoff character is re-transmitted whenever 4 more data arrives in XOFF status. If '10', 8 more data and if 11b, 16 more data. If the value of FUR is approaching the FIFO size, 256-byte, it is good to write XRCR[1:0] 00b. If the 256-FUR value is small, it is good to select 00b of XRCR and if large, it is good to select 11b.

Xoff Re-transmit function is enabled by MCR[6] and MCR[2]. Change MCR[2] from OP1# function to Xoff Re-transmit function by setting MCR[6] to 1b and set MCR[2] to 1b again. Then Xoff Re-transmit function is enabled. When disabling it, first set MCR[6] to 1b and then clear MCR[2] to 0b.

11.4 Sleep Mode with Auto Wake-Up

The SB16C1050A provides sleep mode operation to reduce its power consumption when sleep mode is activated. Sleep mode is enabled when EFR[4] and IER[4] are set to 1b.

Sleep mode is activated when:

- RXD input is in idle state.
- CTS#, DSR#, DCD#, and RI# are not toggling.
- The TX FIFO and TSR are in empty state.
- No interrupt is pending except THR and time-out interrupts.

In sleep mode, the SB16C1050A clock and baud rate clock are stopped. Since most registers are clocked using these clocks, the power consumption is greatly reduced.

Normal operation is resumed when:

- RXD input receives the data start bit transition.
- Data byte is loaded to the TX FIFO or THR.
- CTS#, DSR#, DCD#, and RI# inputs are changed.

11.5 Programmable Baud Rate Generator

The SB16C1050A has a programmable baud rate generator with a prescaler. The prescaler is controlled by MCR[7], as shown in Figure 12−2. The MCR[7] sets the prescaler to divide the clock frequency by 1 or 4. The baud rate generator further divides this clock frequency by a programmable divisor (DLL and DLM) between 1 and (2^{16} – 1) to obtain a 16X sampling rate clock of the serial data rate. The sampling rate clock is used by transmitter for data bit shifting and receiver for data sampling.

The divisor of the baud rate generator is:

$$\text{Divisor} = \frac{\left(\frac{\text{XTAL1 Crystal Input Frequency}}{\text{Prescaler}}\right)}{(\text{Desired Baud Rate} \times 16)}$$

MCR[7] is cleared to 0b (prescaler = 1), when CLKSEL input is in high state after reset.

MCR[7] is set to 1b (prescaler = 4), when CLKSEL input is in low state after reset.

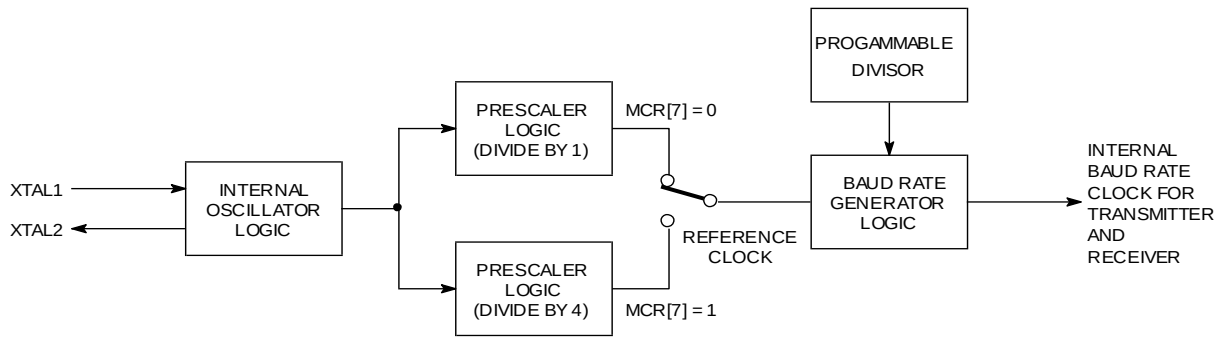


Figure 11−2: Prescaler and Baud Rate Generator Block Diagram

DLL and DLM must be written in order to program the baud rate. DLL and DLM are the least and most significant byte of the baud rate divisor, respectively. If DLL and DLM are both zero, the SB16C1050A is effectively disabled, as no baud clock will be generated.

Table 11−4 shows the baud rate and divisor value for prescaler with divide by 1 as well as crystal with frequency 1.8432MHz, 3.6864MHz, 7.3728MHz, and 14.7456MHz, respectively.

Figure 11−3 shows the crystal clock circuit reference.

Table 11–4: Baud Rates

Desired Baud Rate	16X Digit Divisor for Prescaler with Divide by 1			
	1.8432MHz	3.6864MHz	7.3728MHz	14.7456MHz
50	0900h	1200h	2400h	4800h
75	0600h	0C00h	1800h	3000h
150	0300h	0600h	0C00h	1800h
300	0180h	0300h	0600h	0C00h
600	00C0h	0180h	0300h	0600h
1200	0060h	00C0h	0180h	0300h
1800	0040h	0080h	0100h	0200h
2000	003Ah	0074h	00E8h	01D0h
2400	0030h	0060h	00C0h	0180h
3600	0020h	0040h	0080h	0100h
4800	0018h	0030h	0060h	00C0h
7200	0010h	0020h	0040h	0080h
9600	000Ch	0018h	0030h	0060h
19.2K	0006h	000Ch	0018h	0030h
38.4K	0003h	0006h	000Ch	0018h
57.6K	0002h	0004h	0008h	0010h
115.2K	0001h	0002h	0004h	0008h
230.4K	–	0001h	0002h	0004h
460.8K	–	–	0001h	0002h
921.6K	–	–	–	0001h

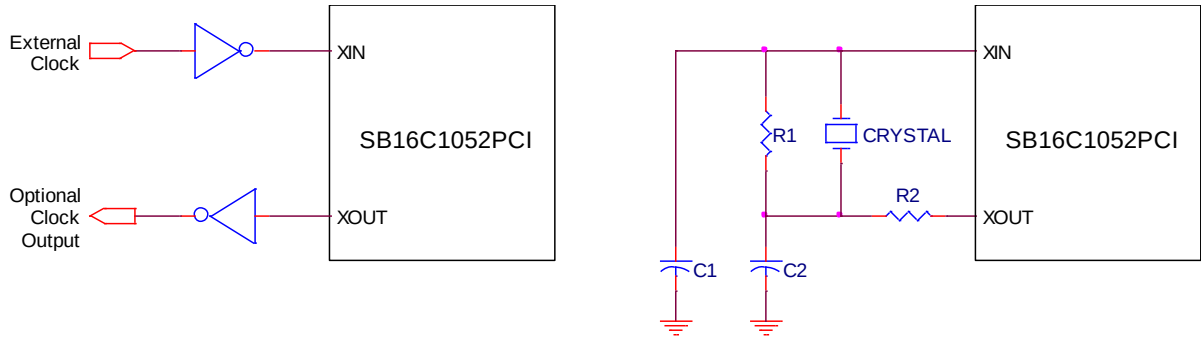


Figure 11–3: Crystal Clock Circuit Reference Diagram

Table 11–5: Component Values

Frequency Range (MHz)	C1 (pF)	C2 (pF)	R1 (Ω)	R2(Ω)
1.8~8	22	68	220K	470 ~ 1.5K
8~16	33~68	33 ~ 68	220K ~ 2.2M	470 ~ 1.5K

11.6 Break and Time-out Conditions

Break Condition:

Break Condition occurs when TXD signal outputs 0b and sustains for more than one character.

It occurs if LCR[6] is set to 1b and deleted if 0b. If break condition occurs when normal data are transmitted on TXD, break signal is transmitted and internal serial data are also transmitted, but they are not outputted to external TXD pin. When Break condition is deleted, then they are transmitted to TXD pin.

Figure 11–4 below shows the Break Condition Block Diagram.

Time-out Condition:

When serial data is received from external UART, characters are stored in RX FIFO. When the number of characters in RX FIFO reaches the trigger level, interrupt is generated for the CPU to treat characters in RX FIFO. But when the number of characters in RX FIFO does not reach the trigger level and no more data arrives from external device, interrupt is not generated and therefore CPU cannot recognize it. SB16C1050A offers time-out function for this situation. Time-out function generates an interrupt and reports to CPU when the number of RX FIFO is less than trigger level and no more data receives for four character time.

Time-out interrupt is enabled when IER[2] is set to 1b and can be verified by ISR.

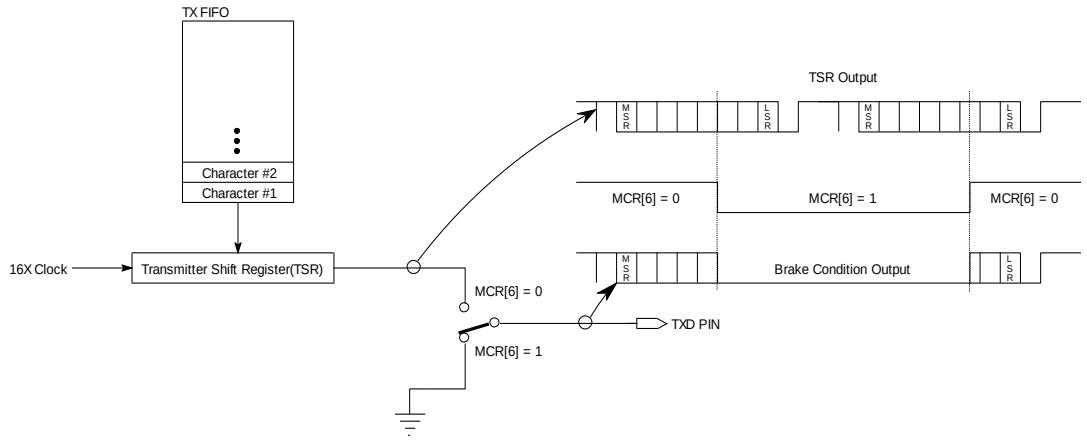


Figure 11–4: Break Condition Block Diagram

11.7 Multi Drop Mode (9-bit Data Transmission)

Some micro-controllers use 9-bit serial communication for several years. It is started when Intel's 8051 micro-controller used 9-bit serial communication, and then, it has been spread out in the market. Serial communication transmits 8-bit data generally. But, 9-bit Serial communication is that transmitting 9-bit data literally.

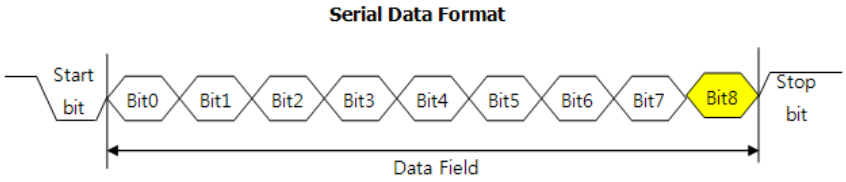


Figure 11–5: Serial Data Format

A multi-drop system consists of master device and slave device in RS422 or RS485 communication. That is because several slave devices are used common data bus line for common. In multi-drop mode, each slave uses distinct address (ID) and a master can communicate with each slave separately using this address.

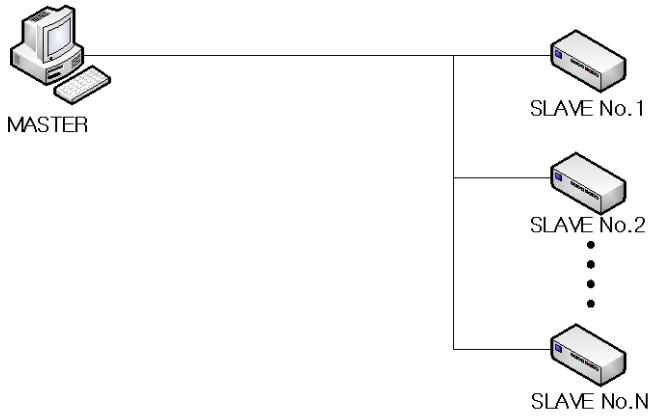


Figure 11–6: Multi-Drop System Network

The slave reads all data from bus and compares all data with address itself. If it is equal to address of slave, the slave accept the data. Then, because the slave doesn't know what data from slave is address or data, comparison between data and its own address is operated to find a packet indicated its own among the data from bus.

Because of this, the processing time is consumed in a lot slave and the performance of device is reduced. In order to resolve this problem, 9-bit communication is used to discriminate between address and data. If 9-bit data from master is '1', it means address. Also, 9-bit data from master is '0, it' means data. The processing time can be reduced because the address is compared to its own address if 9-bit is '1'. If the address is same to its own address, a slave accepts data. But, it is not same to its own address, a slave ignores data.

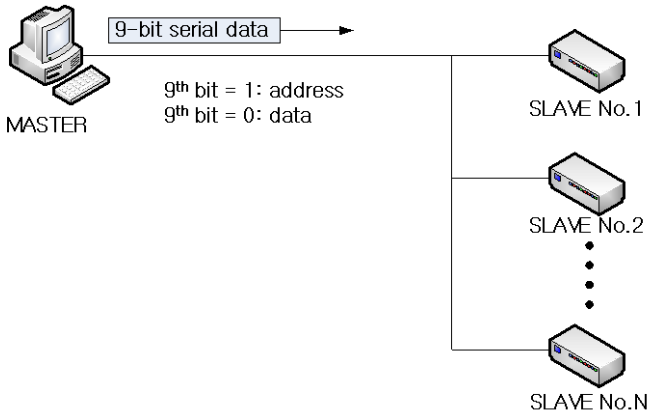


Figure 11−7: 9-bit Communication on Multi-Drop System Network

When RS422/RS485 communication is worked, slaves check the data from the master. If the address is same to its own address, the slave accepts following data and it is not same to its own address, the slave doesn't accept the data. Then, because it is not precise when address is transmitted, a slave device should check RS422 or RS485 Bus's data continually.

But, if 9-bit communication is used, 9-bit data in serial data has only to '1', it is recognized for address. Data doesn't have to be checked because the data is judged that transmitted for its own after the address is compared to its own address. Because of this, the slave device doesn't have to do unnecessary comparison process. As a result, the performance is improved because device's overhead is reduced and the device can be used in many ways.

SB16C1050A UART Core developed in SystemBase is supported 9-bit communication and 3 convenience functions are offered additionally.

12.7.1 Transmit 9-bit Address Register (TAR) / Transmit 9-bit Data Register(TDR)

In SB16C1050A, By utilizing for SPR(Scratch Pad Register, 7h) that is used for program buffer and is not effective to UART operation, 9-bit serial data is transmitted more convenient. If SB16C1050A is operated as Multi-Drop mode, 7h is operated as not SPR but TAR(Transmit 9-bit Address Register, 7h). As a result of this, 9-bit serial data is transmitted conveniently. Also, THR(Transmit Holding Register, 0h) is operated as TDR(Transmit 9-bit Data Register, 0h), it helps that 9-bit serial data is transmitted. In other words, if Multi-Drop mode is set, TDR(0h) has only to be written byte data, it transmitted '0' in 9^{th} bit automatically and TAR(7h) has only to be written byte data, it transmitted '1' in 9^{th} bit automatically.

The below picture is UART transmission using THR like the existing 16C550 UART.

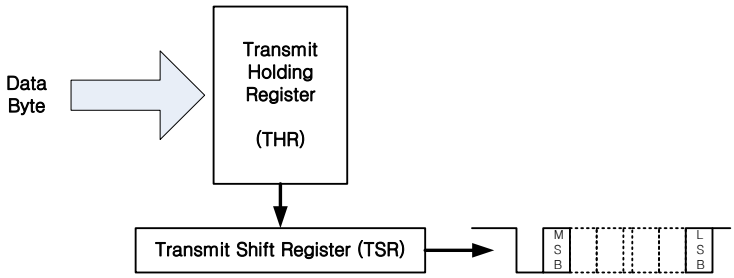


Figure 11–8: THR & TSR of transmit part

But MDR(Multi-Drop mode Register)'s MDE(Multi-Drop mode Enable) bit is set, SB16C1050A is worked as Multi-Drop mode, this can 9-bit serial data transmission. In order to transmit 9-bit address, TAR(Transmit 9-bit Address Register) is active and 9-bit serial information can be transmitted external through TSR(Transmit Shift Register) like below picture.

When 9th bit can be selected following NPS(Ninth-bit Polarity Select) bit information of MDR(Multi-Drop mode Register).

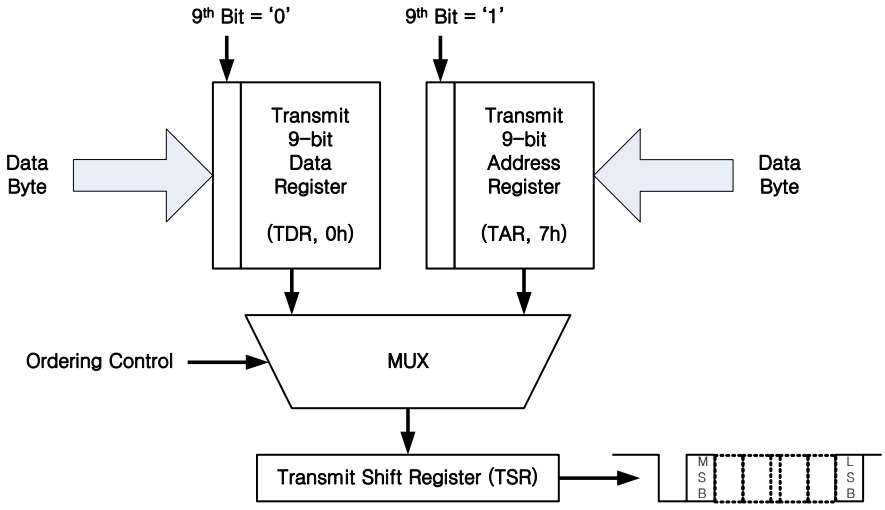


Figure 11–9: TAR & TDR of 9-bit transmit

In general system to support 9-bit communication, when serial data is sent, software command procedure is like below.

① Indicate transmitting address by setting 9th bit to '1' through THR
② Write byte data in THR
③ Indicate transmitting data by clearing 9th bit to '0' through THR
④ Write byte data in THR
⑤ End communication

But, if the way that we suggest is used, software command procedure is like below.

① Write byte data in TAR(7h) (9th bit is set to '1')
② Write byte data in TDR(0h, same to THR) (9th bit is cleared to '0')
③ End communication

If the way that we suggest is used, when 9-bit data is sent, software command procedure is more simplified and twice writing is reduced when RS422 and RS485 communication packet is transmitted. Because of this thing, the performance can be more improved and the serial communication system efficiency can be more upgraded.

11.7.2 Automatic Address Compare

In SB16C1050A UART Core, 2 interrupt sources are added by providing 9-bit communication.
First, if 9-bit serial information is sent in RBR, an interrupt is occurred to detecting 9-bit address. Second, if data of SCR (Special Character Register) is same to 9-bit data, the interrupt is occurred.

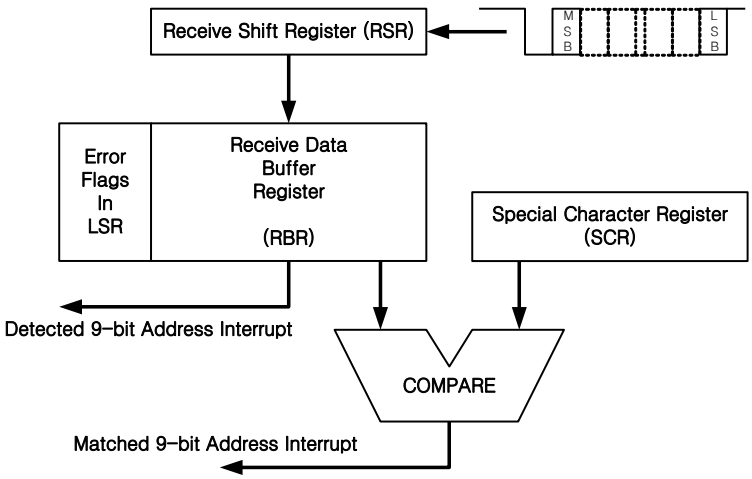


Figure 11–11: Address Auto Detection

If 9th bit is checked for '1' through 9-bit communication, it is regarded as address, it can be differentiated whether packet for right slave if it is compared to slave device. Generally, this comparison is worked in software level, but SB16C1050A UART Core provides this comparison function to operate in hardware level. If 9th bit is '1', its address has only to be stored in advance in SCR(Special Character Register), it can be compared to address in hardware. Of course, if the address designated itself is transmitted in SCR, the software can be signaled by emerging interrupt. SCR is used for Xon/Xoff that is software flow control originally and is used for space to store Xoff/Xon character, in case of Multi-Drop mode, it is used for space to store its address. With this, software and driver's overhead is reduced and the performance is improved through automatic comparison in hardware.

11.7.3 Changed Register Map

Table 11–6: UART Register Map

Addr [2:0]	Page 0 LCR[7] = 0 MCR[6] = 0	Page 1 LCR[7] = 1 LCR[7:0] ≠ BFh	Page 2 LCR[7] = 0 MCR[6] = 1	Page 3 LCR = BFh PSR[0] = 0	Page 4 LCR = BFh PSR[0] = 1
0h	THR(**TDR**)/RB	DLL	–	PSR	PSR
1h	IER	DLM	GICR	**ATR**	AFR
2h	FCR / **ISR**		GISR	**EFR**	XRCR
3h	LCR				
4h	MCR			XON1	TTR
5h	**ACR** / LSR		TCR	XON2	RTR
6h	**MDR** / MSR		RCR	XOFF1	FUR
7h	SPR / **TAR**		FSR	XOFF2	FLR

The registers with Bold Character in the upper table are new register or changed register for 9-bit data transmission. For getting detail description. You can see the detail description about the register with bold in the *Chapter 13. UART Register Descriptions.*
Follows are new or changed registers:

- TDR (Transmit 9-bit Data Register) / TAR (Transmit 9-bit Address Register)
- ISR (Interrupt Status Register)
- ACR (Auto Toggle Control Register)
- MDR (Multi Drop mode Register)
- ATR (Auto Toggling Register)
- EFR (Enhanced Feature Register)

12. UART Register Descriptions

Each UART channel in the SB16C1050A has its own set of registers selected by address lines A2, A1, and A0 with a specific channel selected. The complete register set is shown on Table 12−1 and Table 12−2.

Table 12−1: Internal Registers Map

Address A[2:0]	Page 0	Page 1	Page 2	Page 3	Page 4
	LCR[7] = 0 MCR[6] = 0	LCR[7] = 1 LCR[7:0] ≠ BFh	LCR[7] = 0 MCR[6] = 1	LCR = BFh PSR[0] = 0	LCR = BFh PSR[0] = 1
0h	THR(TDR) / RBR	DLL	—	PSR	PSR
1h	IER	DLM	—	ATR	AFR
2h	FCR / ISR		—	EFR	XRCR
3h	LCR				
4h	MCR			XON1	TTR
5h	LSR / ACR		TCR	XON2	RTR
6h	MSR / MDR		RCR	XOFF1	FUR
7h	SPR / TAR		FSR	XOFF2	FLR

Table 12−1: Internal Registers Map…*continued*

Address A[2:0]	Register	Read/Write	Comments
	Page 0 Registers		
0h	THR : Transmit Holding Register TDR : Transmit 9-bit Data Register RBR : Receive Buffer Register	Write-only (multi-drop mode) Read-only	LCR[7] = 0, MCR[6] = 0
1h	IER : Interrupt Enable Register	Read/Write	LCR[7] = 0, MCR[6] = 0
2h	FCR : FIFO Control Register ISR : Interrupt Status Register	Write-only Read-only	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh
3h	LCR : Line Control Register	Read/Write	—
4h	MCR : Modem Control Register	Read/Write	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh, LCR[7] = 0, MCR[6] = 1
5h	LSR : Line Status Register ACR : Auto Toggle Control Register	Read-only Write-only	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh
6h	MSR : Modem Status Register MDR : Multi Drop mode Register	Read-only Write-only	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh
7h	SPR : Scratch Pad Register	Read/Write	LCR[7] = 0, MCR[6] = 0,

	TAR : Transmit 9-bit Address Register		LCR[7] = 1, LCR ≠ BFh
	Page 1 Registers		
0h	DLL : Divisor Latch LSB	Read/Write	LCR[7] = 1, LCR ≠ BFh
1h	DLM : Divisor Latch MSB	Read/Write	LCR[7] = 1, LCR ≠ BFh
2h	FCR : FIFO Control Register ISR : Interrupt Status Register	Write-only Read-only	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh
3h	LCR : Line Control Register	Read/Write	—
4h	MCR : Modem Control Register	Read/Write	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh, LCR[7] = 0, MCR[6] = 1
5h	LSR : Line Status Register ACR : Auto Toggle Control Register	Read-only Write-only	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh
6h	MSR : Modem Status Register MDR : Multi Drop mode Register	Read-only Write-only	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh
7h	SPR : Scratch Pad Register TAR : Transmit 9-bit Address Register	Read/Write	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh

Table 12−1: Internal Registers Map…*continued*

Address A[2:0]	Register	Read/Write	Comments
	Page 2 Registers		
0h	None	—	—
3h	LCR : Line Control Register	Read/Write	—
4h	MCR : Modem Control Register	Read/Write	LCR[7] = 0, MCR[6] = 0, LCR[7] = 1, LCR ≠ BFh, LCR[7] = 0, MCR[6] = 1
5h	TCR : Transmit FIFO Count Register	Read-only	LCR[7] = 0, MCR[6] = 1
6h	RCR : Receive FIFO Count Register	Read-only	LCR[7] = 0, MCR[6] = 1
7h	FSR : Flow Control Status Register	Read-only	LCR[7] = 0, MCR[6] = 1
	Page 3 Registers		
0h	PSR : Page Select Register	Read/Write	LCR = BFh, PSR[0] = 0, LCR = BFh, PSR[0] = 1
1h	ATR : Auto Toggle Control Register	Read/Write	LCR = BFh, PSR[0] = 0
2h	EFR : Enhanced Feature Register	Read/Write	LCR = BFh, PSR[0] = 0
3h	LCR : Line Control Register	Read/Write	—

4h	XON1b: Xon1 Character Register	Read/Write	LCR = BFh, PSR[0] = 0
5h	XON2 : Xon2 Character Register	Read/Write	LCR = BFh, PSR[0] = 0
6h	XOFF1b: Xoff1 Character Register	Read/Write	LCR = BFh, PSR[0] = 0
7h	XOFF2 : Xoff2 Character Register	Read/Write	LCR = BFh, PSR[0] = 0
	SCR : Special Character Register		
	Page 4 Registers		
0h	PSR : Page Select Register	Read/Write	LCR = BFh, PSR[0] = 0, LCR = BFh, PSR[0] = 1
1h	AFR : Additional Feature Register	Read/Write	LCR = BFh, PSR[0] = 1
2h	XRCR : Xoff Re-transmit Count Register	Read/Write	LCR = BFh, PSR[0] = 1
3h	LCR : Line Control Register	Read/Write	—
4h	TTR : Transmit FIFO Trigger Level Register	Read/Write	LCR = BFh, PSR[0] = 1
5h	RTR : Receive FIFO Trigger Level Register	Read/Write	LCR = BFh, PSR[0] = 1
6h	FUR : Flow Control Upper Threshold Register	Read/Write	LCR = BFh, PSR[0] = 1
7h	FLR : Flow Control Lower Threshold Register	Read/Write	LCR = BFh, PSR[0] = 1

Table 12–2: Internal Registers Description

Addr. A[2:0]	Reg.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Page 0 Registers							
0h	THR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	TDR	Transmit 9-bit Data Register with logic “0” of 9th bit (Accessible when MDE is set to ‘1’)							
0h	RBR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1h	IER	0/CTS# Interrupt Enable	0/RTS# Interrupt Enable	0/Xoff Interrupt Enable	0/Sleep Mode Enable	Modem Status Interrupt Enable	Receive Line Status Interrupt Enable	THR Empty Interrupt Enable	Receive Data Available Interrupt Enable
2h	ISR	FCR[0]/ 256-TX FIFO Empty	FCR[0]/ 256-RX FIFO Full	Interrupt Priority Bit 5	Interrupt Priority Bit 4	Interrupt Priority Bit 3	Interrupt Priority Bit 2	Interrupt Priority Bit 1	Interrupt Priority Bit 0
2h	FCR	RX Trigger Level (MSB)	RX Trigger Level (LSB)	0/TX Trigger Level (MSB)	0/TX Trigger Level (LSB))	DMA Mode Select	TX FIFO Reset	RX FIFO Reset	FIFO Enable
3h	LCR	Divisor Enable	Set TX Brake	Set Parity	Parity Type Select	Parity Enable	Stop Bits	Word Length Bit 1	Word Length Bit 0
4h	MCR	Clock Select	Page 2 Select Xoff Re-Transmit Access Enable	0/Xon Any	0/Loop Back	OUT2/ INTx Enable	OUT1/ Xoff Re-Transmit Enable	RTS#	DTR#
5h	LSR	RX FIFO Data	THR & TSR	THR Empty	Receive Break	Framing Error	Parity Error	Overrun Error	Receive Data

		Error	Empty						Ready
5h	ACR	RXEN# Polarity Select	0	TXEN Polarity Select	0	0	0	0	0
6h	MSR	DCD#	RI#	DSR#	CTS#	ΔDCD#	ΔRI#	ΔDSR#	ΔCTS#
6h	MDR	0	0	0	0	Nine bit Polarity Select	0	Auto Multi Drop Enable	Multi Drop Enable
7h	SCR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
7h	TAR	Transmit 9-bit Address Register with logic “1” of 9th bit (Accessible when MDE is set to ‘1’)							
		Page 1 Registers							
0h	DLL	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1h	DLM	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		Page 2 Registers							
5h	TCR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6h	RCR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
7h	FSR	0	0	TX HW Flow Control Status	TX SW Flow Control Status	0	0	RX HW Flow Control Status	RX SW Flow Control Status

Table 12–2: Internal Registers Description…*continued*

Addr. A[2:0]	Reg.	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		Page 3 Registers							
0h	PSR	1	0	1	0	0	1	0	Page Select
1h	ATR	RXEN# Polarity Select	RXEN# Enable	TXEN Polarity Select	TXEN Enable	Auto Toggle Deassertion Mode1	Auto Toggle Deassertion Mode0	Auto Toggle Assertion Mode 1	Auto Toggle Assertion Mode 0
2h	EFR	Auto-CTS# Enable	Auto-RTS# Enable	Special Character Detect Enable	Enhanced Feature Enable	Software Flow Control Bit 3	Software Flow Control Bit 2	Software Flow Control Bit 1	Software Flow Control Bit 0
4h	XON1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
5h	XON2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6h	XOFF1	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
7h	XOFF2	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	SCR	Special Character Register in 9-bit transmission mode							
		Page 4 Registers							
1h	AFR	0	0	0	0	0	0	0	256-FIFO Enable
2h	XRCR	0	0	0	0	0	0	Bit 1	Bit 0
4h	TTR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

5h	RTR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
6h	FUR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
7h	FLR	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

12.1 Transmit Holding Register (THR, Page 0)

The transmitter section consists of the Transmit Holding Register (THR) and Transmit Shift Register (TSR). The THR is actually a 64-byte FIFO or a 256-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 terminal. If the FIFO is disabled, location zero of the FIFO is used to store the byte. Characters are lost if overflow occurs.

In the 9-bit transmission mode, this register works as TDR (Transmit 9-bit Data Register). Please refer description of TAR, *12.13 Transmit 9-bit Address Register*

12.2 Receive Buffer Register (RBR, Page 0)

The receiver section consists of the Receive Buffer Register (RBR) and Receive Shift Register (RSR). The RBR is actually a 64-byte FIFO or a 256-byte FIFO. The RSR receives serial data from external terminal. The serial data is converted to parallel data and is transferred to the RBR. This 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. If overflow occurs, characters are lost. The RBR also stores the error status bits associated with each character.

12.3 Interrupt Enable Register (IER, Page 0)

IER enables each of the seven types of Interrupt, namely receive data ready, transmit empty, line status, modem status, Xoff received, RTS# state transition from low to high, and CTS# state transition from low to high. All interrupts are disabled if bit[7:0] are cleared. Interrupt is enabled by setting appropriate bits. Table 12–3 shows IER bit settings.

Table 12–3: Interrupt Enable Register Description

Bit	Symbol	Description
7	IER[7]	**CTS# Interrupt Enable (Requires EFR[4] = 1):** 0b: Disable the CTS# interrupt (default). 1b: Enable the CTS# interrupt.
6	IER[6]	**RTS# Interrupt Enable (Requires EFR[4] = 1):** 0b: Disable the RTS# interrupt (default). 1b: Enable the RTS# interrupt.
5	IER[5]	**Xoff Interrupt Enable (Requires EFR[4] = 1):** 0b: Disable the Xoff interrupt (default). 1b: Enable the Xoff interrupt.

4	IER[4]	**Sleep Mode Enable (Requires EFR[4] = 1):** 0b: Disable sleep mode (default). 1b: Enable sleep mode.
3	IER[3]	**Modem Status Interrupt Enable:** 0b: Disable the modem status register interrupt (default). 1b: Enable the modem status register interrupt.
2	IER[2]	**Receive Line Status Interrupt Enable:** 0b: Disable the receive line status interrupt (default). 1b: Enable the receive line status interrupt.
1	IER[1]	**Transmit Holding Register Interrupt Enable:** 0b: Disable the THR interrupt (default). 1b: Enable the THR interrupt.
0	IER[0]	**Receive Buffer Register Interrupt Enable:** 0b: Disable the RBR interrupt (default). 1b: Enable the RBR interrupt.

12.4 Interrupt Status Register (ISR, Page 0)

The UART provides multiple levels of prioritized interrupts to minimize software work load. ISR provides the source of interrupt in a prioritized manner.

Table 12‑4 shows ISR[7:0] bit settings.

Table 12–4: Interrupt Status Register Description

Bit	Symbol	Description
7	ISR[7]	**FCR[0]/256 TX FIFO Empty:** When 256-byte FIFO mode is disabled (default). Mirror the content of FCR[0]. When 256-byte FIFO mode is enabled. 0b: 256-byte TX FIFO is full. 1b: 256-byte TX FIFO is not full. When TCR is '00h', there are two situations of TX FIFO full and TX FIFO empty. If 256 TX empty bit is 1b, it means TX FIFO is empty and if 0b, it means 256 bytes character is fully stored in TX FIFO.
6	ISR[6]	**FCR[0]/256 RX FIFO Full:** When 256-byte FIFO mode is disabled (default). Mirror the content of FCR[0]. When 256-byte FIFO mode is enabled. 0b: 256-byte RX FIFO is not full. 1b: 256-byte RX FIFO is full. When RCR is 00h, there are two situations of RX FIFO full and RX FIFO empty. If 256 RX empty bit is 1b, it means 256 bytes character is fully stored in RX FIFO and if 0b, it means RX FIFO is empty.

Table 12–4: Interrupt Status Register Description…*continued*

Bit	Interrupt Priority List and Reset Functions			
5:0	**Priority**	**Interrupt Type**	**Interrupt Source**	**Interrupt Reset Control**
00_0001	—	None	None	—
00_0110	1	Receiver Line Status	OE, PE, FE, BI Address Incoming Event (9th bit is '1')	Readi g the LSR.
00_0100	2	Receive Data Available	Receiver data available, reaches trigger level.	Reading the RBR or RCR falls below trigger level.
00_ 100	2	Character Timeout Indi-cation	At least one data is in RX FIFO and there are no more data in FIFO during four character time.	Reading the RBR.
00_0010	3	Transmit Holding Register Empty	When THR is empty or TCR passes above trigger level (FIFO enable).	Reading the ISR or write data on THR.
00_0000	4	Modem Status	CTS#, DSR#, DCD#, RI#	Reading the MSR.
01_0000	5	Receive Xoff or Special Character	Detection of a Xoff or special character. Detection of Special Character	Reading the ISR.
10_0000	6	RTS#, CTS# Status during Auto RTS/CTS flow control	RTS# pin or CTS# pin change state from 0b to 1b.	Reading the ISR.

12.5 FIFO Control Register (FCR, Page 0)

FCR is used for enabling the FIFOs, clearing the FIFOs, setting transmit/receive FIFO trigger level, and selecting the DMA modes. Table 12-5 shows FCR bit settings.

Table 12–5: FIFO Control Register Description

Bit	Symbol	Description
7:6	FCR[7:6]	**RX FIFO Trigger Level Select:** 00b: 8 characters (default) 01b: 16 characters 10b: 56 characters 11b: 60 characters
5:4	FCR[5:4]	**TX FIFO Trigger Level Select:** 00b: 8 characters (default) 01b: 16 characters 10b: 32 characters 11b: 56 characters FCR[5:4] can only be modified and enabled when EFR[4] is set.
3	FCR[3]	**DMA Mode Select:** 0b: Set DMA mode 0 (default) 1b: Set DMA mode 1
2	FCR[2]	**TX FIFO Reset:** 0b: No TX FIFO reset (default) 1b: Reset TX FIFO pointers and TX FIFO level counter logic.

		This bit will return to 0b after resetting FIFO.
1	FCR[1]	**RX FIFO Reset:** 0b: No RX FIFO reset (default) 1b: Reset RX FIFO pointers and RX FIFO level counter logic. This bit will return to 0b after resetting FIFO.
0	FCR[0]	**FIFO enable:** 0b: Disable the TX and RX FIFO (default). 1b: Enable the TX and RX FIFO

12.6 Line Control Register (LCR, Page 0)

LCR controls the asynchronous data communication format. The word length, the number of stop bits, and the parity type are selected by writing the appropriate bits to the LCR. Table 12−6 shows LCR bit settings.

Table 12−6: Line Control Register Description

Bit	Symbol	Description
7	LCR[7]	**Divisor Latch Enable:** 0b: Disable the divisor latch (default). 1b: Enable the divisor latch.
6	LCR[6]	**Break Enable:** 0b: No TX break condition output (default). 1b: Forces TXD output to 0b, for alerting the communication terminal to a line break condition.
5	LCR[5]	**Set Stick Parity:** LCR[5:3] = xx0b: No parity is selected. LCR[5:3] = 0x1b: Stick parity disabled. (default) LCR[5:3] = 101b: Stick parity is forced to 1b. LCR[5:3] = 111b: Stick parity is forced to 0b.
4	LCR[4]	**Parity Type Select:** LCR[5:3] =001b: Odd parity is selected. LCR[5:3] =011b: Even parity is selected.
3	LCR[3]	**Parity Enabled:** 0b: No parity (default). 1b: A parity bit is generated during the transmission and the receiver checks for receive parity.
2	LCR[2]	**Number of Stop Bits:** LCR[2:0] = 0xxb: 1 stop bit (word length = 5, 6, 7, 8). LCR[2:0] = 100b: 1.5 stop bits (word length = 5). LCR[2:0] = 11xb or 1x1b: 2 stop bits (word length = 6, 7. 8).
1:0	LCR[1:0]	**Word Length Bits:** 00b: 5 bits (default). 01b: 6 bits. 10b: 7 bits. 11b: 8 bits.

12.7 Modem Control Register (MCR, Page 0)

MCR controls the interface with the modem, data set, or peripheral device that is emulating the modem. Table 12−7 shows MCR bit settings.

Table 12−7: Modem Control Register Description

Bit	Symbol	Description
7	MCR[7]	**Clock Prescaler Select:** 0b: Divide by 1 clock input (default). 1b: Divide by 4 clock input.
6	MCR[6]	**Page 2 Select/Xoff Re-Transmit Access Enable:** 0b: Enable access to page 0 register when LCR[7] is 0b (default). 1b: Enable access to page 2 register and Xoff re-transmit bit when LCR[7] is 0b.
5	MCR[5]	**Xon Any Enable:** 0b: Disable Xon any (default). 1b: Enable Xon any.
4	MCR[4]	**Internal Loop Back Enable:** 0b: Disable loop back mode (default). 1b: Enable internal loop back mode. In this mode the MCR[3:0] signals are looped back into MSR[7:4] and TXD output is looped back to RXD input internally.
3	MCR[3]	**OUT2/Interrupt Output Enable:** 0b: INTx outputs disabled (default). During loop back mode, OUT2 output 0b and it controls MSR[7] to 1b. 1b: INTx outputs enabled. During loop back mode, OUT2 output 1b and it controls MSR[7] to 0b. OUT2 is not available as an output pin on the SB16C1050A.
2	MCR[2]	**OUT1/Xoff Re-transmit Enable:** 0b: Xoff re-transmit disable when MCR[6] is 0b. During loop back mode, OUT1 output to 0b and it controls MSR[6] to 1b. 1b: Xoff re-transmit enable when MCR[6] is 1b. During loop back mode, OUT1 output to 1b and it controls MSR[6] to 0b. OUT1 is not available as an output pin on the SB16C1050A. Xoff re-transmit is operated with XRCR, refer to XRCR.
1	MCR[1]	**RTS# Output:** 0b: Force RTS# output to 1b. During loop back mode, controls MSR[4] to 1b. 1b: Force RTS# output to 0b. During loop back mode, controls MSR[4] to 0b.
0	MCR[0]	**DTR# Output:** 0b: Force DTR# output to 1b. During loop back mode, controls MSR[5] to 1b. 1b: Force DTR# output to 0b. During loop back mode, controls MSR[5] to 0b.

12.8 Line Status Register (LSR, Page 0)

LSR provides the status of data transfers between the UART and the CPU. When LSR is read, LSR[4:2] reflect the error bits (BI, FE, PE) of the character at the top of the RX FIFO. The errors in a character are identified by reading LSR and then reading RBR. Reading LSR does not cause an increment of the RX FIFO read pointer. The RX FIFO read pointer is incremented by reading the RBR. Table 12−8 shows LSR bit settings.

Table 12−8: Line Status Register Description

Bit	Symbol	Description
7	LSR[7]	**RX FIFO data error Indicator:** 0b: No RX FIFO error (default). 1b: At least one parity error, framing error, or break indication is in the RX FIFO. This bit is cleared when there is no more error in any of characters in the RX FIFO.
6	LSR[6]	**THR and TSR Empty Indicator:** 0b: THR or TSR is not empty. 1b: THR and TSR are empty.
5	LSR[5]	**THR Empty Indicator:** 0b: THR is not empty. 1b: THR is empty. It indicates that the UART is ready to accept a new character for transmission. In addition, it uses the UART to generate an interrupt to the CPU when the THR empty interrupt enable is set to 1b.
4	LSR[4]	**Break Interrupt Indicator:** 0b: No break condition (default). 1b: The receiver received a break signal (RXD was 0b for at least one character frame time). In FIFO mode, only one character is loaded into the RX FIFO.
3	LSR[3]	**Framing Error Indicator:** 0b: No framing error (default). 1b: Framing error. It indicates that the received character did not have a valid stop bit.
2	LSR[2]	**Parity Error Indicator:** 0b: No parity error (default). 1b: Parity error. It indicates that the receive character did not have the correct even or odd parity, as selected by the LCR[4]
1	LSR[1]	**Overrun Error Indicator:** 0b: No overrun error (default). 1b: Overrun error. It indicates that the character in the RBR or RX FIFO was not read by the CPU, thereby ignored the receiving character.
0	LSR[0]	**Receive Data Ready Indicator:** 0b: No character in the RBR or RX FIFO. 1b: At least one character in the RBR or RX FIFO.

12.9 Auto Toggle Control Register (ACR, Page 0 & Page 1)

Address 5h is used for LSR (Line Status Register) in the existent UART Core with R/W permission. But we change to use address 5h with write permission to ACR (Auto Toggle Control Register).

Originally we use ATR (Auto Toggle Register) in page 3 for the setting of auto toggling. To decrease inconvenience of accessing ATR, we make other access route for the setting of Auto Toggling. That is ACR in page 0. So you can see the same functions between ATR and ACR.

Table 12–9: Auto Toggle Control Register Description

Bit	Symbol	Description
7	ACR[7]	**RXEN# Polarity Select:** It is used for controlling RXEN# output. 0b: Asserted output of RXEN# is 0b. 1b: Asserted output of RXEN# is 1b. (default) It support the function same as bit7 of ATR.
6	ACR[6]	Reserved.
5	ACR[5]	**TXEN Polarity Select:** It is used for controlling TXEN output. 0b: Asserted output of RXEN# is 0b. 1b: Asserted output of RXEN# is 1b. (default) It support the function same as bit5 of ATR.
4:0	ACR[4:0]	Reserved.

12.10 Modem Status Register (MSR, Page 0)

MSR provides the current status of control signals from modem or auxiliary devices. MSR[3:0] are set to 1b when input from modem changes and cleared to 0b as soon as CPU reads MSR. Table 12–10 shows MSR bit settings.

Table 12–10: Modem Status Register Description

Bit	Symbol	Description
7:4	MSR[7]	**DCD Input Status:** Complement of Data Carrier Detect (DCD#) input. In loop back mode this bit is equivalent to OUT2 in the MCR.
6	MSR[6]	**RI Input Status:** Complement of Ring Indicator (RI#) input. In loop back mode this bit is equivalent to OUT1 in the MCR.
5	MSR[5]	**DSR Input Status:** Complement of Data Set Ready (DSR#) input. In loop back mode this bit is equivalent to DTR in the MCR.
4	MSR[4]	**CTS Input Status:** Complement of Clear To Send (CTS#) input. In loop back mode this bit is equivalent to RTS in the MCR.

3	MSR[3]	**ΔDCD Input Status:** 0b: No change on CD# input (default). 1b: Indicates th t the DCD# input has changed state.
2	MSR[2]	**ΔRI Input Status:** 0b: No change on RI# input (default). 1b: Indicates that the RI# input has changed state from 0b to 1b.
1	MSR[ ]	**ΔDSR Input Status:** 0b: No change on DSR# input (deault). 1b: Indicates that the DSR# input has changed state.
0	MSR[0]	**ΔCTS Input Status:** 0b: No change on CTS# input (deault). 1b: Indicates that the CTS# input has changed state.

12.11 Multi Drop mode Register (SPR, Page 0 & Page 1)

Address 6h is used for MSR (Modem Status Register) in the existent UART Core with R/W permission. But we change to use address 6h with write permission to MDR (Multi Drop mode Register). This is used for setting 9-bit transmission mode of multi drop in RS422 and RS485 network.

Table 12–11: Multi Drop mode Register Description

Bit	Symbol	Description
7:4	MDR[7:4]	Reserved.
3	MDR[3]	**9th Bit Polarity Select (NPS):** 0b: when 9th bit is 0, it decide to get an address byte. when 9th bit is 1, it decide to get a data byte. 1b: when 9th bit is 1, it decide to get an address byte. (default) when 9th bit is 0, it decide to get a data byte.
2	MDR[2]	Reserved.
1	MDR[1]	**Auto Multi-drop Enable (AME):** When bit5 of EFR is set to '1', UART core compare received address byte with Special Character (Xoff2) automatically in 9-bit transmission mode. 0b: receiving byte is saved in RBR or RX FIFO irrespective of address matching. (deault). 1b: automatic compare between Xoff2 register and receiving address byte. If the receiving byte is same with Xoff2, UART core save the received byte to RBR or RX FIFO. If it is not same, the UART core discard the received byte.
0	MDR[0]	**Multi Drop Enable (MDE):** 0b: it works as normal mode (deault). 1b: Indicates that UART is working in 9th bit transmission mode. In 9th bit transmission mode, when an address (9th =1) comes to RBR or RX FIFO, Interrupt will take place.

12.12 Scratch Pad Register (SPR, Page 0)

This 8-bit Read/Write Register does not control the UART in anyway. It is intended as a scratch pad register to be used by the programmer to hold data temporarily.

12.13 Transmit 9-bit Address Register (TAR, Page 0)

Address 7h is used for SPR (Scratch Pad Register) in the existent UART Core with R/W permission. But when UART core work in 9-bit transmission mode(MDE, bit0 of MDR bit is set to '1'), this register is used for sending address bye with 9th bit, 1.

In 9-bit transmission mode, when we write a byte to TAR(7h), 9-bit Address Byte with 9th bit, 1 will send. When we write a byte to TDR(0h), 9-bit Date Byte with 9th bit, 0 will send.

12.14 Divisor Latches (DLL, DLM, Page 1)

Two 8-bit registers which store the 16-bit divisor for generation of the clock in baud rate generator. DLM stores the most significant part of the divisor, and DLL stores the least significant part of the divisor. Divisor of zero is not recommended.

Note that DLL and DLM can only be written to before sleep mode is enabled, i.e., before IER[4] is set. Chapter 12.7 describes the details of divisor latches.

12.15 Transmit FIFO Count Register (TCR, Page 2)

TCR shows the number of characters that can be stored in TX FIFO. In 64-byte FIFO mode, it consists of only TCR[6:0]. If the number of characters that can be stored in TX FIFO is 0, it is shown as 0000_0000b and if 64, it is shown as 0100_0000b. In 256-byte FIFO mode, it consists of ISR[7] + TCR[7:0]. If the number of characters that can be stored in TX FIFO is 0, it is shown as 0_0000_0000b and if 255, it is shown as 0_1111_1111b. And in case of the maximum number 256, it is shown as 1_0000_0000b.

12.16 Receive FIFO Count Register (RCR, Page 2)

RCR shows the number of characters that is stored in RX FIFO. In 64-byte FIFO mode, it consists of only RCR[6:0]. If the number of characters that is stored in RX FiFO is 0, it is shown as 0000_0000b and if 64, it is shown as 0100_0000b. In 256-byte FIFO mode, it consists of ISR[6] + RCR[7:0]. If the number of characters that is stored in RX FiFO is 0, it is shown as 0_0000_0000b and if 255, it is shown as 0_1111_1111b. And in case of the maximum number 256, it is shown as 1_0000_0000b.

12.17 Flow Control Status Register (FSR, Page 2)

FSR show the status of operation of TX Hardware Flow Control, RX Hardware Flow Control, TX Software Flow Control, and RX Software Flow Control.

Table 12−12: Flow Control Status Register Description

Bit	Symbol	Description
7:6	FSR[7:6]	Not used, always 00b.
5	FSR[5]	**TX Hardware Flow Control Status:** 0b: When FIFO or Auto-RTS flow control is disabled. If FIFO and Auto-RTS flow control is enabled, it means the number of data received in RX FIFO at the first time is less than the value of FUR, or it means the number of data in RX FIFO was more than the value of FUR and after the CPU read them, the number of data that remains unread is less than or equal to the value of FLR. That is, UART reports external device that it can receive more characters. 1b: It shows that the number of data received in RX FIFO exceeds the value of FUR and UART reports external device that it cannot receive more data. If RX FIFO has space to store more data, new data are stored in RX FIFO but after it gets full, they are lost. For more details, refer to '12.2 Hardware Flow Control'.
4	FSR[4]	**TX Software Flow Control Status:** 0b: When FIFO or Software flow control is disabled. If FIFO and Software flow control is enabled, it means the number of data received in RX FIFO at the first time is less than the value of FUR, or it means the number of data in RX FIFO was more than the value of FUR and after the CPU read them, the number of data that remains unread after the CPU read the data received in RX FIFO is less than or equal to the value of FLR. That is, UART transmits Xon character to report external device that it can receive more data. 1b: It shows that the number of data received in RX FIFO exceeds the value of FUR and transmitting Xoff character to report external device that it cannot receive more data. If RX FIFO has space to store more data, new data are stored in RX FIFO but after it gets full, they are lost. For more details, refer to '12.3 Software Flow Control'.
3:2	FSR[3:2]	Not used, always 00b.
1	FSR[1]	**RX Hardware Flow Control Status:**

		0b: When FIFO or Auto-CTS flow control is disabled. If FIFO and Auto-CTS flow control is enabled, 0b is inputted in CTS# pin and it means external device can receive more data. This time data in TX FIFO are transmitted. 1b: If FIFO and Auto-CTS flow control is enabled, 1b is inputted in CTS# pin and it means external device can not receive more data. This time data in TX FIFO are not transmitted. For more details, refer to '12.2 Hardware Flow Control'.
0	FSR[0]	**RX Software Flow Control Status:** 0b: When FIFO or RX Software flow control is disabled. If FIFO and RX Software flow control is enabled, it means Xoff character has never arrived or Xon character arrived after Xoff character had arrived(it means external device can receive more data). This time data in TX FIFO are transmitted. 1b: If FIFO and RX Software flow control is enabled, it means Xoff character has arrived and external device can not receive data any more. This time characters in TX FIFO are not transmitted. For more details, refer to '12.3 Software Flow Control'.

12.18 Page Select Register (PSR, Page 3)

If BFh is written in LCR, registers in Page3 and Page4 can be accessed. PSR is used to determine which page to use. Table 12−13 shows PSR bit settings.

Table 12−13: Page Select Register Description

Bit	Symbol	Description
7:1	PSR[7:1]	**Access Key:** When writing data on PSR to change page, Access Key must be correspondent. If the value of PSR[7:1] is 1010_010b, data is written on PSR[0] and page can be selected. If PSR[7:1] is read, it reads 0000_000b which is irrespective of Access Key.
0	PSR[0]	**Page Select:** 0b: Page 3 is selected (default). 1b: Page 4 is selected.

12.19 Auto Toggle Control Register (ATR, Page 3)

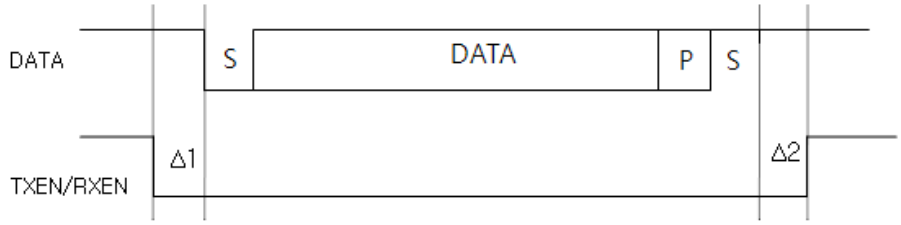
ATR controls the signals for controlling input/output signals when using Line Interface as RS422 or RS485, so eliminates additional glue logic outside. Table 12−14 shows ATR bit settings.

Table 12−14: Auto Toggle Control Register Description

Bit	Symbol	Description
7	ATR[7]	**RXEN# Polarity Select:** 0b: Asserted output of RXEN# is 0b. 1b: Asserted output of RXEN# is 1b. (default)
6	ATR[6]	**RXEN# Enable** 0b: RXEN# is outputted as same as ATR[7], irrespective of TXD signal. (default) 1b: RXEN# is outputted as same as ATR[7] when TXD signal is not transmitting. And outputted as complement of ATR[7] when TXD signal is transmitting.
5	ATR[5]	**TXEN Polarity Select:** 0b: Asserted output of TXEN is 0b. 1b: Asserted output of TXEN is 1b. (default)
4	ATR[4]	**TXEN Enable:** 0b: TXEN is outputted as same as ATR[5], irrespective of TXD signal. (default) 1b: TXEN is outputted as complement of ATR[5] when TXD signal is not transmitting, and outputted as same as ATR[5] when TXD signal is transmitting..
3:2	ATR[3:2]	**Auto Toggle De-assertion Mode:** You can select the delta between TXEN/RXEN# de-assertion and stop bit of frame. 00b: 0ns(Δ2) 01b: 75ns(Δ2) 10b: 150ns(Δ2) 11b: 300ns(Δ2)
1:0	ATR[1:0]	**Auto Toggle Assertion Mode:** You can select the delta between TXEN/RXEN# assertion and start bit of frame. When ATR[3:2] ≠ 11b, 00b: 0ns(Δ1), 01b: 75ns(Δ1), 10b: 150ns(Δ1), 11b: 300ns(Δ1) When ATR[3:2] = 11b, 00b: 0.6us(Δ1), 01b: 1us(Δ1), 10b: 1.5us(Δ1), 11b: 3us(Δ1)

Cf. After reset, TXEN and RXEN# output '0b'.

Figure 12−1: Control of Assertion & Deassertion time in Auto Toggle mode



12.20 Enhanced Feature Register (EFR, Page 3)

EFR enables or disables the enhanced features of the UART. Table 12−15 shows EFR bit settings.

Table 12−15: Enhanced Feature Register Description

Bit	Symbol	Description
7	EFR[7]	**Auto-CTS Flow Control Enable:** 0b: Auto-CTS flow control is disabled (default). 1b: Auto-CTS flow control is enabled. Transmission stops when CTS# pin is inputted 1b. Transmission resumes when CTS# pin is inputted 0b.
6	EFR[6]	**Auto-RTS Flow Control Enable:** 0b: Auto-RTS flow control is disabled (default). 1b: Auto-RTS flow control is enabled. The RTS# pin outputs 1b when data in RX FIFO fill above the FUR. RTS# pin outputs 0b when data in RX FIFO fall below the FLR.
5	EFR[5]	**Special Character Detect:** When bit5 of EFR is set to '1', UART core compare received address byte with Special Character (Xoff2) automatically in 9-bit transmission mode. This register used for selecting H/W address matching method or S/W. If bit0 of MDR is not set, this bit can't affect to special character detection. 0b: Special character detect disabled (default). 1b: Special character detect enabled. The UART compares each incoming character with data in Xoff2 register. If a match occurs, the received data is transferred to RX FIFO and ISR[4] is set to 1b to indicate that a special character has been detected.
4	EFR[4]	**Enhanced Function Bits Enable:** 0b: Disables enhanced functions and writing to IER[7:4], FCR[5:4], MCR[7:5]. 1b: Enables enhanced function IER[7:4], FCR[5:4], and MCR [7:5] can be modified, i.e., this bit is therefore a write enable.
3:0	EFR[3:0]	**Software Flow Control Select:** Single character and dual sequential characters software flow control is supported. Combinations of software flow control can be selected by programming these bits. See Table 12−1 "Software flow control options (EFR[3:0])" on page 45.

12.21 Special Character Register (SCR, Page 3)

When Bit0 of MDR is set to '1' (9-bit transmission mode), Xoff2 register is used as Special Character Register.

12.22 Additional Feature Register (AFR, Page 4)

AFR enables or disables the 256-byte FIFO mode and controls the global interrupt. Table 12–16 shows AFR bit settings.

Table 12–16: Additional Feature Register Description

Bit	Symbol	Description
7:6	AFR[7:1]	Not used, always 000_0000b.
0	AFR[0]	**256-byte FIFO Enable:** 0b: 256-byte FIFO mode is disabled and this means SB16C1050A operates as Non FIFO mode or 64-byte FIFO mode (default). 1b: 256-byte FIFO mode is enabled and ISR[7:6] operates as 256-TX FIFO Empty and 256-RX FIFO Full.

12.23 Xoff Re-transmit Count Register (XRCR, Page 4)

XRCR operates only when Software flow control is enabled by EFR[3:0] and Xoff Re-transmit function of MCR[2] is also enabled. And it determines the period of retransmission of Xoff character. Table 12–17 shows XRCR bit settings.

Table 12–17: Xoff Re-transmit Count Register Description

Bit	Symbol	Description
7:2	XRCR[7:2]	Not used, always 0000_00b.
1:0	XRCR[1:0]	**Xoff Re-transmit Count Select:** 00b: Transmits Xoff character whenever the number of received data is 1 during XOFF status. (default) 01b: Transmits Xoff character whenever the number of received data is 4 during XOFF status. 10b: Transmits Xoff character whenever the number of received data is 8 during XOFF status. 11b: Transmits Xoff character whenever the number of received data is 16 during XOFF status.

12.24 Transmit FIFO Trigger Level Register (TTR, Page 4)

TTR operates only when 256-byte FIFO mode is enabled. It sets the trigger level of 256-byte TX FIFO for generating transmit interrupt. Interrupt is generated when the number of data remained in TX FIFO after transmitting through TXD pin is less than the value of TTR. Initial value is 80h, 1000_0000b. 0000_0000b should never be written. If written, unexpected operation may occur.

12.25 Receive FIFO Trigger Level Register (RTR, Page 4)

RTR operates only when 256-byte FIFO mode is enabled. It sets the trigger level of 256-byte RX FIFO for generating receive interrupt. Interrupt is generated when the number of data remained in RX FIFO exceeds the value of RTR(this time, timeout or interrupt is valid). Initial value is 80h, 1000_0000b. 0000_0000b should never be written. If written, unexpected operation may occur.

12.26 Flow Control Upper Threshold Register (FUR, Page 4)

FUR can be written only when 256-byte FIFO mode is enabled and one of TX software flow control or Auto-RTS is enabled (In 64-byte mode, it cannot be written but can be read only, and follows the value of trigger level set in FCR[5:4]). While TX software flow control is enabled, Xoff character is transmitted when the number of data in RX FIFO exceeds the value of FUR. If Auto-RTS is enabled, 1b is outputted on RTS# pin to report that it cannot receive data any more. If both TX software flow control and Auto-RTS is enabled, Xoff character is transmitted and 1b is outputted on RTS# pin. The value of FUR must be larger than that of FLR.

12.27 Flow Control Lower Threshold Register (FLR, Page 4)

FLR can be written only when 256-byte FIFO mode is enabled and one of TX software flow control, or Auto-RTS is enabled (In 64-byte mode, it cannot be written but can be read only, and follows the value of trigger level set in FCR[7:6]). While TX software flow control is enabled, Xon character is transmitted when the number of data in RX FIFO is less than the value of FUR only if Xoff character is transmitted before. If Auto-RTS is enabled, 0b is outputted on RTS# pin to report that it can receive more data. If both TX software flow control and Auto-RTS is enabled, Xon character is transmitted only if Xoff character is transmitted before and 0b is outputted on RTS# pin. The value of FLR must be less than that of FUR.

13. Parallel Port Description

SB16C1053APCI offers one Parallel Port supporting Compatibility/Nibble/Byte/EPP/ECP modes. The Parallel Port complies with IEEE standard 1284. And it is host-based multi-function parallel port that be to transfer data between a host PC and a peripheral such as printers, scanners and external drives. The register set of the Parallel Port is compatible with Microsoft register definition.
When SB16C1053APCI is set to 1P or 2S1P mode, the Parallel Port would be activated. BAR2 of PCI Configuration Space is assigned to basic registers of Parallel Port. And BAR3 of PCI Configuration Space is assigned to extended registers of ECP mode.

Two Base Address Range, BAR2 and BAR3 are re-locatable.
When the user this parallel port in DOS mode, please set the BAR2 and BAR3 to DOS LPT1 ~ LTP3 ranges.

Table 13−1: LPT1 ~ LPT3 ranges in DOS mode

	LPT1	LPT2	LPT3
BAR2	378h	278h	3BCh
BAR3	778h	678h	7BCh

13.1 Parallel Port Signal Name Cross Reference

The name and function of the Parallel Port signals are changed for each operational mode.
Bellow is cross reference table of Parallel Port signal for 5 operational modes of Parallel Port

Table 13−2: Parallel Port Signal for 5 operational modes

SB16C1053APCI Pin Name	Pin No.	Compatible Mode	Nibble Mode	Byte Mode	EPP Mode	ECP Mode
STB#_MIOR#_OSC0	64	nSTROBE	HostClk	HostClk	nWrite	HostClk
BUSY_MIRQ2	76	BUSY	PtrBusy	PtrBusy	nWait	PeriphAck
ACK#_MIRQ1	75	nACK	PtrClk	PtrClk	Intr	nPeriphClk
SLCT_MIRQ3	71	SELECT	Xflag	Xflag	Xflag	Xflag
PE_MIRQ0	74	PERROR	AckDataReq	AckDataReq	AckDataReq	nAckReverse
ERR#_MCS1#	70	nFAULT	nDataAvail	nDataAvail	nDataAvail	nPeriphReq
INIT#_MRESET	68	nINIT	−	−	nINIT	nReverseReq
AFD#_MIOW#	65	nAUTOFD	HostBusy	HostBusy	nDStrb	HostAck
PD_MD[7:0]		PD[7:0]	−	PD[7:0]	PD[7:0]	PD[7:0]
SLIN#	69	nSLCTIN	−	−	nAStrb	ECP mode

13.2 Compatibility Mode

Compatibility mode provides an asynchronous, byte wide, forward channel (host-to-peripheral), with the data and status lines used according to original definitions, as per the original Centronics port. It is also known as SPP, industry Standard Parallel Port.

13.2.1 Pin Descriptions in the Compatibility Mode

Table 13–3: Pin Description in Compatibility Mode

Signal Name	Type	Compatibility Protocol Signal Description
STROBE#	O	STROBE: The host asserts STROBE# to latch data into the peripheral devices. This signal is controlled via the CTRL register.
BUSY	I	BUSY: BUSY is asserted by the peripheral to indicate that peripheral device is not ready to receive data from host. This signal is indicated via the STAT register.
ACK#	I	Acknowledge: The printer asserts ACK to indicate that printer has received data from host and printer is ready for new data. This signal is indicated via the STAT register.
SELECT	I	SELECT: The peripheral asserts to indicate that peripheral device is on line. This signal is indicated via the STAT register.
PERROR	I	Paper Error: The peripheral device asserts PERROR to indicate that an error has occurred. This signal is indicated via the STAT register.
FAULT#	I	FAULT: The peripheral device asserts this signal to indicate that an error occurred. This signal is indicated via the STAT register.
INIT#	O	INITIALIZE: The host asserts INIT# to reset the peripheral device. This signal is controlled via the CTRL register.
AUTOFD#	O	Auto Feed: The host asserts to make the peripheral into auto-line feed mode. This signal is controlled via the CTRL register.
PD[7:0]	O	DATA: these are 8-bit forward channel data.
SLCTIN#	O	Select Input: The host asserts SLCTIN# to select a peripheral device. This signal is controlled via the CTRL register.

13.2.1 Register Descriptions in the Compatibility Mode

Table 13–4: Basic Parallel Port Register

Address (BAR2 +)	Abbreviation	Register Name	Access
0h	DATA	Data Register	R/W
1h	STAT	Status Register	R
2h	CTRL	Control Register	R/W

Table 13–5: Status Register Description of Parallel Port

Status Register

Bit	Name	Description
7	nBUSY	If nBUSY is asserted, it indicates that the peripheral is busy.
6	nACK	If nACK is asserted, it indicates that the peripheral has received a data byte and it is ready for another data byte.
5	PE	If PE is asserted, it indicates that the peripheral is out of paper.
4	SLCT	If SLCT is asserted, it indicates that the peripheral is selected.
3	nERR	If nERR is asserted, it indicates that the peripheral has a fault.
2	-	Reserved. Always returns 0.
1	-	Reserved. Always returns 0.
0	TIMEOUT	When timeout occurs, this signal is asserted to high. (EPP only)

Table 13–6: Control Register Description of Parallel Port

Bit	Name	Description
7		Reserved. Always returns 0.
6		Reserved. Always returns 0.
5	PDIR	Direction Bit. PDIR = 0, forward direction PDIR = 1, reverse direction
4	INTEN	If it is asserted, it allows the peripheral to interrupt the CPU.
3	SLCTIN	If it is asserted, it means the host has selected the peripheral.
2	NINIT	If it is asserted, the peripheral is initialized.
1	AUTOFD	This tells the printer to advance the paper by one line each time a carriage return is received.
0	STROBE	If it is asserted, this instructs the peripheral to accept the data on the data bus.

Default value of Control Register is 00h after reset.

13.3 Nibble Mode

Nibble mode provides an asynchronous, reverse channel(peripheral-to-host) under the control of the host. Data bytes are transmitted as two sequential, four-bit nibbles using four peripheral-to-host status lines. When the host and/or peripheral do not support bi-directional use of the data lines, Nibble Mode may be used with Compatibility mode to implement a bi-directional channel. The two meds cannot be active simultaneously.

13.3.1 Pin Descriptions in the Nibble Mode

Table 13–7: Pin Description in Nibble Mode

Signal Name	Type	Nibble Protocol Signal Description
STROBE#	O	STROBE: This value should be held negated by the host. This signal is controlled via the CTRL register.
BUSY	I	PRINTER BUSY(PtrBusy): The peripheral drives this value to transfer data bits 3 and 7 sequentially. This signal is indicated via the STAT register.
ACK#	I	PRINTER CLOCK(PtrClk): The peripheral devices asserts ACK# to indicate to the host that received data is available. The signal is subsequently asserted to qualify data being sent to the host. This is indicated via the STAT register.
SELECT	I	XFLAG: The peripheral device drives XFLAG to transfer data bits 1 and 5 sequentially. This signal is indicated via the STAT register.
PERROR	I	Acknowledge Data Request (AckDataReq): The peripheral device asserts to acknowledge for HostBusy assertion. This signal is initially high. It is subsequently used to transfer data bits 2 and 6 sequentially. This signal is indicated via the STAT register.
FAULT#	I	Data Available (DataAvail): The peripheral device asserts DataAvail to indicate data availability. It is subsequently used to transfer data bits 0 and 4 sequentially. This signal is indicated via the STAT register.
INIT#	O	INITIALIZE: This signal is controlled via the CTRL register by host.
AUTOFD#	O	Host Busy(HostBusy): The host negates AUTOFDÝ (HostBusy) in response to ACKÝ being asserted. This signal is subsequently driven low to enable the peripheral to transfer data to the host. AUTOFDÝ is then driven high to acknowledge receipt of byte data. This signal is controlled via the CTRL register.
PD[7:0]	O	DATA: This 8-bit output data path to the peripheral Host data is written to the peripheral attached to the parallel port interface on these signal lines.
SLCTIN#	O	SELECT INPUT: This signal is controlled via the CTRL register.

13.4 Byte Mode

BYTE MODE provides an asynchronous, byte wide, reverse channel (peripheral-to-host) using the eight data lines of the interface for data and the control/status lines for handshaking. Byte Mode may be used to implement a bi-directional channel, with the transfer direction controlled by the host, when both host and peripheral support bi-directional use of the data lines. It is compatible with IBM PS/2 hosts.

13.4.1 Pin Descriptions in the Byte Mode

Table 13–8: Pin Description in Byte Mode

Signal Name	Type	Byte Protocol Signal Description
STROBE#	O	Host Clock (HostClk): This signal is strobed low by the host to acknowledge receipt of data
BUSY	I	Printer Busy (PtrBusy): The peripheral asserts PtrBusy to provide forward channel peripheral busy status. This is indicated via the STAT register.
ACK#	I	Printer Clock (PtrClk): The peripheral asserts PtrClk to indicate to the host that received data is available. The signal is subsequently asserted to qualify data being sent to the host. This is indicated via the STAT register.
SELECT	I	XFLAG: The peripheral asserts XFLAG to indicate that the device is on line. This signal is indicated via the STAT register.
PERROR	I	Acknowledge Data Request (AckDataReq): This singal is initially high. The peripheral drives this signal to acknowledge for HostBusy assertion. This signal is indicated via the STAT register.
FAULT#	I	Data Availability (DataAvail): The peripheral asserts DataAvail to indicate data availability. This signal is indicated via the STAT register.
INIT#	O	INITIALIZE: This value should be held negated by the host. This signal is controlled via the CTRL register.
AUTOFD#	O	Host Busy(HostBusy): The host negates HostBusy in response to nACK being asserted.The signal is subsequently driven low to enable the peripheral to transfer data to the host. nAUTOFD is then driven high to acknowledge receipt of nibble data. This signal is controlled via the CTRL register.
PD[7:0]	O	DATA: This 8-bit data bus is used for bi-directional data transfer.
SLCTIN#	O	SELECT INPUT: This signal is controlled via the CTRL register.

13.5 EPP Mode

ENHANCED PARALLEL PORT (EPP) MODE provides an asynchronous, byte wide, bi-irectional channel controlled by the host device. This mode provides separate address and data cycles over the eight data lines of the interface.

The Enhanced Parallel Port (EPP) was designed in a joint venture between Intel, Xircom & Zenith Data Systems. EPP Ports were first specified in the EPP 1.7 standard, and then later included in the IEEE 1284 standard released in 1994. EPP has two standards, EPP 1.7 and EPP 1.9. EPP has a typical transfer rate in the order of 500KB/s to 2MB/s.

13.5.1 Pin Descriptions in the EPP Mode

Table 13–9: Pin Description in EPP Mode

Signal Name	Type	EPP Protocol Signal Description
STROBE#	O	WRITE (write#): It indicates an address or data read/write to the peripheral. When the value is low, it works as write operation. When the value is high, it works as read operation.
BUSY	I	WAIT (wait#): The peripheral assert low to indicate that the peripheral device is not ready. When BUSY is low, SB16C1053APCI make internal signal, IOCHRDY to low for the longer I/O cycle. The peripheral deassert high to indicate that the transfer of data or address is completed.
ACK#	I	Interrupt Request (Intr): The peripheral asserts to generate interrupt to the host. When this signal is low and interrupt enabled, interrupt is generated to the host.
SELECT	I	SELECT: The peripheral asserts to indicate that the device is on line. This signal is indicated via STAT register.
PERROR	I	Paper Error: The peripheral asserts to indicate that there is an error in the paper path. This signal is indicated via the STAT register.
FAULT#	I	FAULT: The peripheral asserts to indicate that an error has occurred. This signal is indicated via the STAT register.
INIT#	O	INITIALIZE: The host asserts to reset the peripheral device. This signal is controlled via CTRL register.
AUTOFD#	O	Data Strobe (DStrb#): The host asserts to indicate that valid data is present on data bus(PD[7:0]). The peripheral use DStrb# as data latch point.
PD[7:0]	I/O	DATA: The 8-bit bi-directional bus provides address or data. When write cycle is working, this data bus work as output. When read cycle is working, this data
SLCTIN#	O	Address Strobe (AStrb#): The host asserts to indicate that a valid address is present on PD[7:0]. The peripheral use AStrb# as address latch point.

13.5.2 Register Descriptions in the EPP Mode

Table 13–10: Register Description in EPP Mode

Address BAR2 +	Abbreviation	Register Name	Access
0h	DATA	Data Register	R/W
1h	STAT	Status Register	R
2h	CTRL	Control Register	R/W
3h	ADDSTR	Address Strobe Register	R/W
4h ~ 7h	DATASTR	Data Strobe Register	R/W

The description of DATA register is same with Data Register of *13.2.1 Register Descriptions in the Compatibility Mode*.
The description of STAT register is same with Status Register of *13.2.1 Register Descriptions in the Compatibility Mode*.
The description of CTRL register is same with Control Register of *13.2.1 Register Descriptions in the Compatibility Mode*.
The ADDSTR register provides a peripheral address to the peripheral via PD[7:0] during a host write, and to the host via PD[7:0] during a host address read operation. An automatic address strobe is generated on the parallel port interface when data is read or written to this register.
The DATASTR registers provide data from the host to the peripheral via PD[7:0] during a write operation, and data from the peripheral to the host during a read operation. An automatic data strobe is generated on the parallel port interface when data is read or written to these registers.

13.5.2 EPP Mode Operation

If no EPP Read, Write or Address cycle is currently being executed, the Peripheral data bus may be used in either Compatibility Mode (and/or Nibble Mode) or PS/2 (Byte) Mode. In this condition, all output signals (NSTROBE, NAUTOFD and NINIT) are set by the CTRL register and direction is controlled by the PDIR bit of the CTRL register.
Before an EPP cycle is executed, the control register PDIR bit must be set to 0 (by writing 04h or 05h to the CTRL register). If PDIR is left set to 1, the SB16C1053APCI will not be able to perform a write and will appear instead to perform an EPP read on the parallel bus without any error being indicated.
If an EPP bus cycle does not terminate within 10ms, the EPP timeout flag will be set and all following EPP bus cycles will be aborted until this flag is cleared. The flag is cleared by writing to the STAT register.

13.6 ECP Mode

EXTENDED CAPABILITIES PORT (ECP) MODE provides an asynchronous, byte wide, bi-directional channel. An interlocked handshake replaces the Compatibility Mode's minimum timing requirements. A control line is provided to distinguish between command and data transfers.

13.6.1 Pin Descriptions in the ECP Mode

Table 13–11: Pin Description in ECP Mode

Signal Name	Type	ECP Protocol Signal Description
nSTROBE	O	Host Clock (HostClk): SB16C1053APCI assert HostClk to instruct the peripheral to latch the data on PD[7:0] in forward direction. The peripheral latch data on the rising edge of HostClk. HostClk handshake with PeriphAck. After the peripheral latch the data on PD[7:0], it de-assert PeripheAck. After SB16C1053APCI detects PeriphAck asserted, it de-asserts HostClk. In reverse direction, HostClk is not used.
BUSY	I	Peripheral Acknowledge (PeriphAck): The peripheral asserts this signal to acknowledge for the receipt of data in forward direction. After the peripheral detects HostClk going high for termination of transfer, it de-asserts this signal. PeriphAck handshake with HostClk. In reverse direction, PeriphAck is low. The peripheral assert this signal to indentify Run Length Encoded (RLE) data.
nACK	I	Peripheral Clock (PeriphClk): The peripheral asserts this signal to indicated data on the data bus is valid in reverse direction. After the peripheral detects HostAck going high, it de-asserts this signal. PeriphClk handshake with HostAck.
SELECT	I	XFLAG (X flag): The peripheral asserts this signal to indicate that peripheral is on-line. This signal is indicated via STAT register.
PERROR	I	Acknowledge Reverse (nAckReverse): The peripheral asserts this signal to acknowledge for reverse transfer request from host. nAckReverse handshake with nReverseRequest. This signal is indicated via STAT register.
nFAULT	I	Peripheral Request (nPeriphReqest): The peripheral asserts this signal to request a reverse transfer. This signal is indicated via STAT register.
nINIT	O	Reverse Request (nReverseRequest): The host asserts this signal to request reverse transfer direction. The host de-asserts this signal for forward transfer direction. This signal is controlled via CTRL register.
nAUTOFD	O	Host Acknowledge (HostAck): SB16C1053APCI asserts this signal to request data in reverse direction. HostAck handshake with PeriphClk. SB16C1053APCI de-asserts this signal when peripheral indicates valid state of the data bus. HostAck represents whether PD[7:0] contain address, REL or data in forward direction.
PD[7:0]	I/O	DATA: 8 bits bi-directional data bus for data, address, RLE data.
nSLCTIN	O	ECP Mode (ECPmode): The host de-asserts this signal during ECP operation.

13.6.2 Register Descriptions in the ECP Mode

Table 13–12: Basic Parallel Port Registers in ECP Mode

Address (BAR2 +)	Abbreviation	Register Name	ECR[7:5]	Access
0h	ECPAFIFO	ECP Address	011	R/W
1h	STAT	Status Register	ALL	R
2h	CTRL	Control Register	ALL	R/W

The **ECPAFIFO** register provides a channel address to the peripheral depending on the state of bit 7. This I/O address location is only used in ECP Mode (ECR bits [7:5]=011). In this mode, bytes written to this register are placed in the parallel port FIFO and transmitted via PDOUT[7:0] using the ECP protocol. Bit 7 should always be set to 1.
The description of **STAT** register is same with Status Register of *13.2.1 Register Descriptions in the Compatibility Mode*.
The description of **CTRL** register is same with Control Register of *13.2.1 Register Descriptions in the Compatibility Mode*.

Table 13–13: Extended Parallel Port Registers in ECP Mode

Address (BAR3 +)	Abbreviation	Register Name	ECR[7:5]	Access
0h	SDFIFO	Standard Parallel Port Data FIFO	010	R/W
0h	ECPDFIFO	ECP Data FIFO	011	R/W
0h	TFIFO	Test FIFO	110	R/W
0h	CFGA	ECP Configuration A	111	R/W
1h	CFGB	ECP Configuration B	111	R/W
2h	ECR	Extended Control Register	ALL	R/W

SDFIFO is used to transfer data from the host to the peripheral when the ECR register is set for compatible FIFO mode (Bits [7:5] = 010). Data bytes written from the system to this FIFO are transmitted by a hardware handshake to the peripheral using the standard Compatibility protocol. For this register, bytes are placed in the parallel port FIFO using DATA[7:0] of MIO Bus and transmitted via PD[7:0].
ECPDFIFO is used to transfer data from the host to the peripheral when the ECR register is set for ECP mode (Bits [7:5] = 011). Data bytes written from the system to this FIFO are transmitted by a hardware handshake to the peripheral using the ECP protocol.
The **Test FIFO** provides a test mechanism for the ECP Mode FIFO by allowing data to be read, written in either direction between the system and this FIFO. This Test Mode is selected by setting ECR[7:5] = 110.
The data is transferred purely through the microprocessor interface. It may appear on the parallel

port data lines, but without any hardware handshake.
The Test FIFO does not stall when overwritten or underrun. Data is simply ignored or re-read. The full and empty bits of the ECR register (bits 1 and 0) can however be used to ascertain the correct state of the FIFO.
The **CFGA** register provides information about the ECP Mode implementation. It is a Read Only register. Access to this register is enabled by programming the ECR register (ECR[7:5] = 111). At reset CFGA is set to 10h. 10h indicates an 8-bit implementation.
The **CFGB** register checks the PINTR line to determine possible conflicts. It is a Read Only register. Access to this register is enabled by programming the ECR register (ECR[7:5] = 111).

Table 13–14: Extended Parallel Port Register Description in ECP Mode

	D7	D6	D5	D4	D3	D2	D1	D0
Data	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
ECPAFIFO	1	Address						
STAT	BUSY#	ACK#	PE	SLCT	NERR	0	0	0
CTRL		0	PDIR	INTEN	SLCTIN	INIT#	AUTOFD	STROBE
SDFIFO	Parallel Port Data FIFO							
ECPDFIFO	ECP Data FIFO							
TFIFO	Test FIFO							
CFGA	0	0	0	1	0	0	0	0
CFGB	0	PINTR	0	0	0	0	0	0
ECR	MODE			INTERR	0	SERVINT	FIFOF	FIFOE

The **ECR** register selects ECP mode, enables service and error interrupts and provides interrupt status. The ECR provides FIFO empty and FIFO full status.

Table 13–15: ECP Mode Selection (ECR[7:5])

Mode	Description
000	This puts the parallel port into Compatibility Mode and resets the pointers to the FIFO (but not its contents). Setting the direction bit in the CTRL register does not affect the parallel port interface in this mode.
001	This puts the parallel port into Byte Mode and resets the pointers to the FIFO (but not its contents). The outcome is similar to above except that the direction bit selects forward or reverse transfers.
010	This puts the parallel port into ISA Compatible FIFO mode, which is the same as mode 000 except that PWords are written to the FIFO. FIFO data is automatically transmitted using the standard parallel port protocol. Note, this mode should only be used when PDIR = 0.
011	This puts the port into ECP Mode. In the forward direction, bytes written to the ECPDFIFO and ECPAFIFO locations are placed in the ECP FIFO and transmitted automatically to the peripheral using ECP protocol. In the reverse direction, bytes are transferred from PDIN [7:0] to the ECP FIFO.
100	This puts the port into EPP Mode.

101	Reserved.
110	This selects an ECP Test Mode in which the FIFO is read and written purely through the microprocessor interface.
111	This is places the interface into Configuration Mode. In this mode, the CFGA and CFGB registers are accessible at 0h & 1h of BAR3 respectively.

Table 13−16: ECR[4:0] Description

Bit	Name	Description
4	INTERR	When 0, this bit enables error interrupts to the host when a high to low transition occurs on the nERR signal.
3	-	Reserved.
2	SERVINT	If SERVINT = '1', all service interrupts are disabled. SERVINT = '0' enables one of below two interrupts. If PDIR = 0, the SERVINT bit will be set to '1' whenever there are 'WriteIntrThreshold' or more bytes free in the FIFO If PDIR = 1, the SERVINT bit will be set to '1' whenever there are 'ReadIntrThreshold' or more valid bytes to be read from the FIFO
1	FIFOF	FIFO Full Status. This bit indicates when the FIFO is full.
0	FIFOE	FIFO Empty Status. This bit indicates when the FIFO is empty.

13.6.3 ECP Mode Operation

Prior to ECP operations, the host must negotiate on the parallel port to determine if the peripheral supports the ECP protocol. This is carried out under program control in mode 000.
After negotiation, the following must be initialized:

- Set Direction = 0 (enabling drivers)
- Set Strobe = 0 (causes NSTROBE to default to de-asserted state)
- Set autoFd = 0 (causes NAUTOFD to default to de-asserted state)
- Set Mode = 011 (ECP Mode)

ECP address bytes or data bytes can be sent automatically by writing to the ECPAFIFO or ECPDFIFO respectively.
It should be noted that all FIFO data transfers are byte wide and byte aligned. Address transfers are also byte wide and are only permitted in the forward direction.
The host may switch directions by first switching to mode 001, negotiating for the forward or reverse channel, setting direction to 1 or 0 and then setting mode 011. When PDIR = 1, the hardware will handshake each ECP read data byte and will attempt to fill the FIFO. Bytes can then be read from the ECPDFIFO providing it is not empty.
ECP transfers can also be achieved (slowly) by handshaking individual bytes under program control in mode 001 or 000
Termination can only be executed while the bus is in the forward direction. So if the bus is in the reverse direction when termination is required, it must first be set back to the forward direction.
ECP Mode supports two advanced features to improve the effectiveness of the protocol for some applications. These features are implemented by allowing the transfer of normal 8-bit data or 8-

bit commands.
When in the forward direction, normal data is transferred when nAUTOFD is high and an 8-bit command is transferred when nAUTOFD is low.
When in the reverse direction, normal data is transferred when BUSY is high and an 8-bit command is transferred when BUSY is low.

13.6.4 FIFO Operation in the ECP Mode

Transfers from host to FIFO

In the forward direction, an interrupt occurs when SERVINT = 0 and there are writeIntrThreshold or more bytes free in the FIFO. If the FIFO is empty at this time, it can be filled with a single burst before the empty bit requires to be re-read. Alternatively, it may be filled with writeIntrThreshold bytes.

writeIntrThreshold = (16 - <threshold>) free bytes in FIFO

The interrupt is therefore generated when SERVINT = 0 and the number of bytes in the FIFO is less than or equal to <threshold>. For example, if the threshold = 12, the interrupt will be set whenever there are 12 or fewer bytes of data in the FIFO.
The PINTR signal can be used with interrupt-driven systems to request the transfer. If the FIFO is empty at this time, it can be completely filled in a single burst. Alternatively, a minimum of (16 - <threshold>) bytes may be written to the FIFO in a single burst. This process is repeated until the last byte is transferred into the FIFO.

Transfers from FIFO to host

In the reverse direction, an interrupt occurs when SERVINT = 0 and readIntrThreshold bytes are available in the FIFO. If the FIFO is full at this time, it can be emptied completely in a single burst. Alternatively, readIntrThreshold bytes may be read from the FIFO in a single burst.

readIntrThreshold = (16 - <threshold>) data bytes in FIFO

The interrupt is therefore generated when SERVINT = 0 and the number of bytes in the FIFO is greater than or equal to (16 - <threshold>). For example, if the threshold = 12, the interrupt will be set whenever there are 4 to 16 bytes in the FIFO.
The PINTR signal can be used with interrupt-driven systems to request the transfer. The host must respond to the request by reading data from the FIFO. This process is repeated until the last byte is transferred out of the FIFO. If the FIFO is full at this time, it can be completely emptied in a single burst. Alternatively, a minimum of (16 - <threshold>) bytes can be read from the FIFO in a single burst.

13.6.4 RLE Operation in the ECP Mode

The SB16C1053APCI is able to decompress run-length encoded (RLE) data in ECP Mode. When an RLE byte is intercepted, the following byte is repeated the number of times specified by the given run-length count.
The run-length count specifies the number of times the following byte is repeated. Thus a run-length count of 15 expands the following byte to 16 bytes, while a run-length count of zero specifies that the following byte is read once.

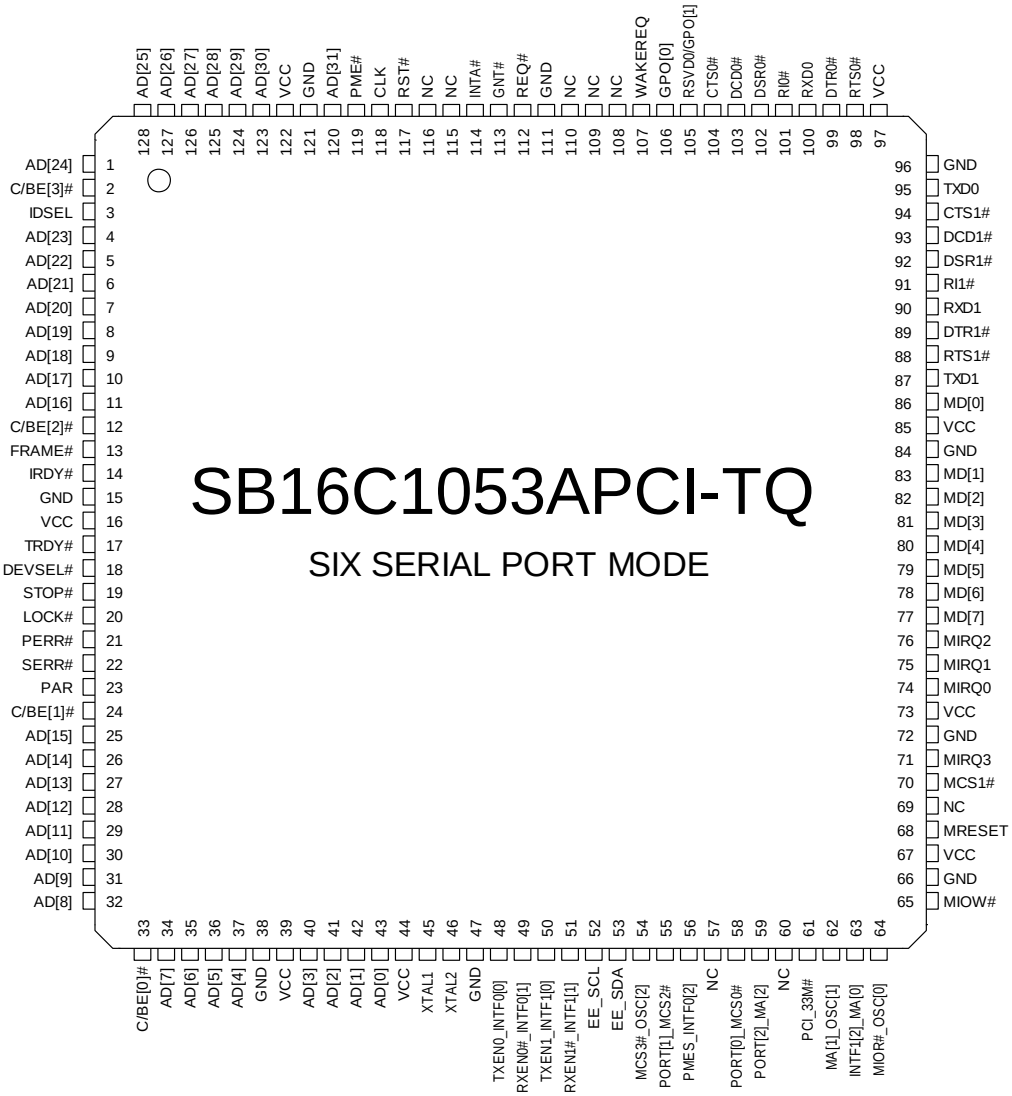
14. MIO Bus Description

SB16C1053APCI can extend to 2 serial ports or 4 serial ports additionally using external UART. Because SB16C1053APCI have multi-function pins, some pins works as MIO Bus™ signals when SB16C16C1053APCI is in 4S mode or 6S mode.

In this chapter, you can get information about the MIO Bus™ signal and the interfacing between SB16C1053APCI and external Quad-UART.

MIO Bus™ is devised by SystemBase and it is similar with ISA bus.

14.1 MIO Bus™ Signal Descriptions



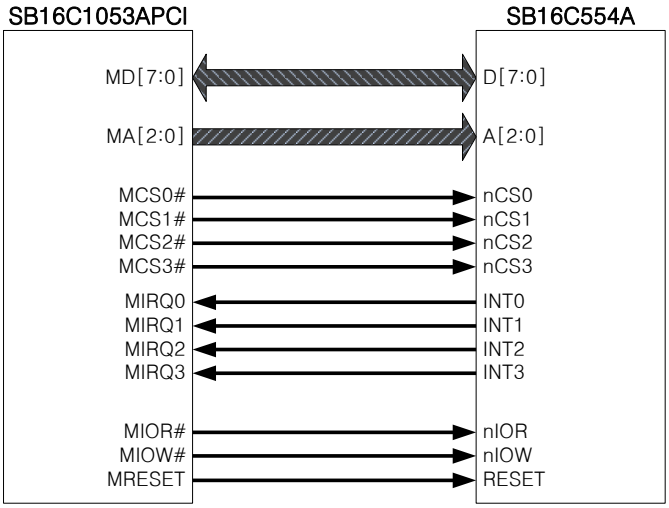
NC - No internal connection

Table 14–1: Pin Description of MIO Bus™

Signal Name	Pin No.	I/O	Signal Description
MD[0]	86	I/O	8-bit Data Bus of MIO Bus™ It used for exchange 8-bit data between UART and SB16C1053APCI.
MD[1]	83	I/O	
MD[2]	82	I/O	
MD[3]	81	I/O	
MD[4]	80	I/O	
MD[5]	79	I/O	
MD[6]	78	I/O	
MD[7]	77	I/O	
MA[0]	63	O	3-bit Address Bus of MIO Bus™ It used for selecting address of UART registers.
MA[1]	62	O	
MA[2]	59	O	
MCS[0]#	58	O	Chip Selection of MIO Bus™ for 1st external UART
MCS[1]#	70	O	Chip Selection of MIO Bus™ for 2nd external UART
MCS[2]#	55	O	Chip Selection of MIO Bus™ for 3rd external UART
MCS[3]#	54	O	Chip Selection of MIO Bus™ for 4th external UART
MIRQ[0]	74	I	Interrupt Request of MIO Bus™ from 1st external UART
MIRQ[1]	75	I	Interrupt Request of MIO Bus™ from 2nd external UART
MIRQ[2]	76	I	Interrupt Request of MIO Bus™ from 3rd external UART
MIRQ[3]	71	I	Interrupt Request of MIO Bus™ from 4th external UART
MIOR#	64	O	Read Strobe of MIO Bus™
MIOW#	65	O	Write Strobe of MIO Bus™
MRESET	68	O	Reset of MIO Bus™

14.2 Interfacing between SB16C1053APCI and Quad-UART

When you make the interfacing between SB16C1053APCI and SB16C554A, please refer below figure.



15. UART Programmer's Guide

The base set of registers that is used during high-speed data transfer has a straightforward access method. The extended function registers require special access bits to be decoded along with the address lines. The following guide will help with programming these registers. Note that the descriptions below are for individual register access. Some streamlining through interleaving can be obtained when programming all the registers.

Table 15–1: Register Programming Guide

Command	Action
Set Baud Rate to VALUE1, VALUE2	Read LCR, save in temp Set LCR to 80h Set DLL to VALUE1 Set DLM to VALUE2 Set LCR to temp
Set Xon1, Xoff1 to VALUE1, VALUE2	Read LCR, save in temp Set LCR to BFh Set Xon1 to VALUE1 Set Xoff1 to VALUE2 Set LCR to temp
Set Xon2, Xoff2 to VALUE1, VALUE2	Read LCR, save in temp Set LCR to BFh Set Xon2 to VALUE1 Set Xoff2 to VALUE2 Set LCR to temp
Set Software Flow Control Mode to VALUE	Read LCR, save in temp Set LCR to BFh Set EFR to VALUE Set LCR to temp
Set flow control threshold for 64-byte FIFO Mode	1) Set FCR to 0000_xxx1b → Set FUR to 8, set FLR to 0 2) Set FCR to 0101_xxx1b → Set FUR to 16, set FLR to 8 3) Set FCR to 1010_xxx1b → Set FUR to 56, set FLR to 16 4) Set FCR to 1111_xxx1b → Set FUR to 60, set FLR to 56
Set flow control threshold for 256-byte FIFO Mode	Set FCR to xxxx_xxx1b Read LCR, save in temp Set LCR to BFh Set PSR to A5h Set AFR to 01h

Table 15–1: Register Programming Guide…*continued*

Command	Action
	Set FUR to Upper Threshold Value
	Set FLR to Lower Threshold Value
	Set PSR to A4h
	Set LCR to temp
Set TX FIFO / RX FIFO Interrupt Trigger Level for 64-byte FIFO Mode	1) Set FCR to 0000_xxx1b → Set RTR to 8, set TTR to 8 2) Set FCR to 0101_xxx1b → Set RTR to 16, set TTR to 16 3) Set FCR to 1010_xxx1b → Set RTR to 56, set TTR to 32 4) Set FCR to 1111_xxx1b → Set RTR to 60, set TTR to 56
Set TX FIFO / RX FIFO Interrupt Trigger Level for 256-byte FIFO Mode	Set FCR to xxxx_xxx1b Read LCR, save in temp Set LCR to BFh Set PSR to A5h Set AFR to 01h Set TTR to TX FIFO Trigger Level Value Set RTR to RX FIFO Trigger Level Value Set PSR to A4h Set LCR to temp
Read Flow Control Status	Read LCR, save in temp1 Read MCR, save in temp2 Set LCR to (0111_1111b **AND** temp1) Set MCR to (0100_0000b **OR** temp2) Read FSR, save in temp3 Pass temp3 back to host Set MCR to temp2 Set LCR to temp1
Read TX FIFO / RX FIFO Count Value	Read LCR, save in temp1 Read MCR, save in temp2 Set LCR to (0111_1111b **AND** temp1) Set MCR to (0100_0000b **OR** temp2) Read TCR, save in temp3 Read RCR, save in temp4 Pass temp3 back to host Pass temp4 back to host Set MCR to temp2 Set LCR to temp1

Table 15–1: Register Programming Guide…*continued*

Command	Action
Read 256-byte TX FIFO Empty Status / RX FIFO Full Status	Set FCR to xxxx_xxx1b Read LCR, save in temp1 Set LCR to BFh Set PSR to A5h Set AFR to 01h Set PSR to A4h Set LCR to temp1 Read ISR, save in temp2 Pass temp2 back to host
Enable Xoff Re-transmit	Read LCR, save in temp1 Set LCR to not BFh Read MCR, save in temp2 Set MCR to (0100_0000b **OR** temp2) Set MCR to (0100_0100b **OR** temp2) Set MCR to (1011_1111b **AND** temp2) Set MCR to temp2 Set LCR to temp1
Disable Xoff Re-transmit	Read LCR, save in temp1 Set LCR to not BFh Read MCR, save in temp2 Set MCR to (0100_0000b **OR** temp2) Set MCR to (1011_1011b **AND** temp2) Set MCR to temp2 Set LCR to temp1

Set Prescaler Value to Divide-by-1 or 4	Read LCR, save in temp1
	Set LCR to BFh
	Read EFR, save in temp2
	Set EFR to (0001_0000b **OR** temp2)
	Set LCR to 00h
	Read MCR, save in temp3
	
	if Divide-by-1 = OK then
	Set MCR to (0111_1111b **AND** temp3)
	else
	Set MCR to (1000_0000b **OR** temp3)
	
	Set LCR to BFh
	Set EFR to temp2
	Set LCR to temp1

16. Electrical Information

16.1 Absolute Maximum Ratings

Symbol	Parameter	Min	Max	Unit
V_{DD}	DC Supply Voltage	-0.5	7.0	V
V_{IN}	Input Voltage	-0.5	V_{DD}+0.5	V
V_{OUT}	Output Voltage Range	0	V_{DD}+0.5	V
T_{STG}	Storage Temperature	-65	150	℃
T_{OP}	Operating Temperature Recommend Temp: 25℃	-40	125	℃

Absolute maximum ratings are the values beyond which damage to the device may occur. Exposure to these conditions or beyond may adversely affect device reliability. Functional operation under absolute maximum ratings is not implied.

16.2 Power Consumption

Power Consumption	Minimum	Typical	Maximum	Unit
SB16C1053APCI-TQ	-	TBD	TBD	W

16.3 DC Characteristics

16.3.1 PCI PAD 3.3V 66MHz DC Signaling Characteristics

Symbol	Parameter	Condition	Min	Max	Unit
V_{DD}	Supply Voltage		3.0	3.6	V
V_{IH}	Input High Voltage		0.5V_{DD}	V_{DD}+0.5	V
V_{IL}	Input Low Voltage		-0.5	0.3V_{DD}	V
I_{IL}	Input Leakage Current	0 < V_{IN} < V_{DD}		+/-1	uA
V_{OH}	Output High Voltage	I_{out} = -500uA	0.9V_{DD}		V
V_{OL}	Output Low Voltage	I_{out} = 1500uA		0.1V_{DD}	V

16.3.1 PCI PAD 3.3V 66MHz DC Signaling Characteristics

Symbol	Parameter	0 ℃	100 ℃	Conditions	
		Min.	Max.	VDD	
V_{IL}	Low Level Input Voltage	-0.5V	0.3V_{DD}	2.7V~3.6V	Guaranteed Input Low Voltage
V_{IH}	High Level Input Voltage	0.7V_{DD}	V_{DD}+0.5V	2.7V~3.6V	Guaranteed Input High Voltage
V_{OL}	Low Level Output Voltage		V_{SS}+0.1V	2.7V	I_{OL}=0.8mA
V_{OH}	High Level Output Voltage	V_{DD}-0.1V		2.7V	I_{OH}=0.8mA

I_I	Input Current at Max Voltage		1mA	2.7V~3.6V	Input=5.5V

17. Timing Specification

17.1 PCI BUS Timing Specifications

Symbol	Parameter	66MHz		33MHz		Units
		Min	Max	Min	Max	
T_{val}	CLK to signal valid delay - bused signals	2	6	2	11	ns
$T_{val}(ptp)$	CLK to signal valid delay - point to point signals	2	6	2	12	ns
T_{on}	float to active delay	2		2		ns
T_{off}	active to float delay		14		28	ns
T_{su}	input setup time to CLK – bused signals	3		7		ns
$T_{su}(ptp)$	input setup time to CLK - point to point signals	5		10, 12		ns
T_h	Input hold time from CLK	0		0		ns
T_{rst}	Reset active time after power stable	1		1		ms
$T_{rst-clk}$	Reset active time after CLK stable	100		100		us
$T_{rst-off}$	Reset active to output float delay		40		40	ns
T_{rhfa}	RST# high to first configuration access	2		2		clocks
T_{rhff}	RST# high to first FRAME# assertion	5		5		clocks

Table 17–1: PCI Bus Timing Specifications

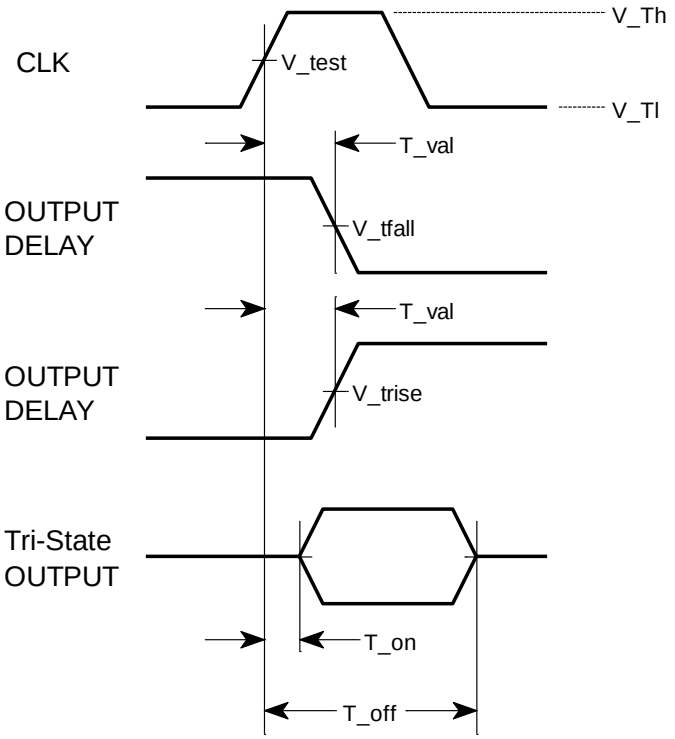


Figure 17-1: Output Timing Measurement Conditions

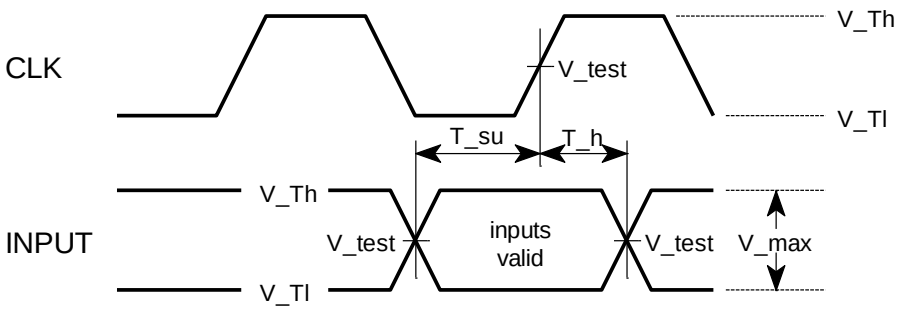


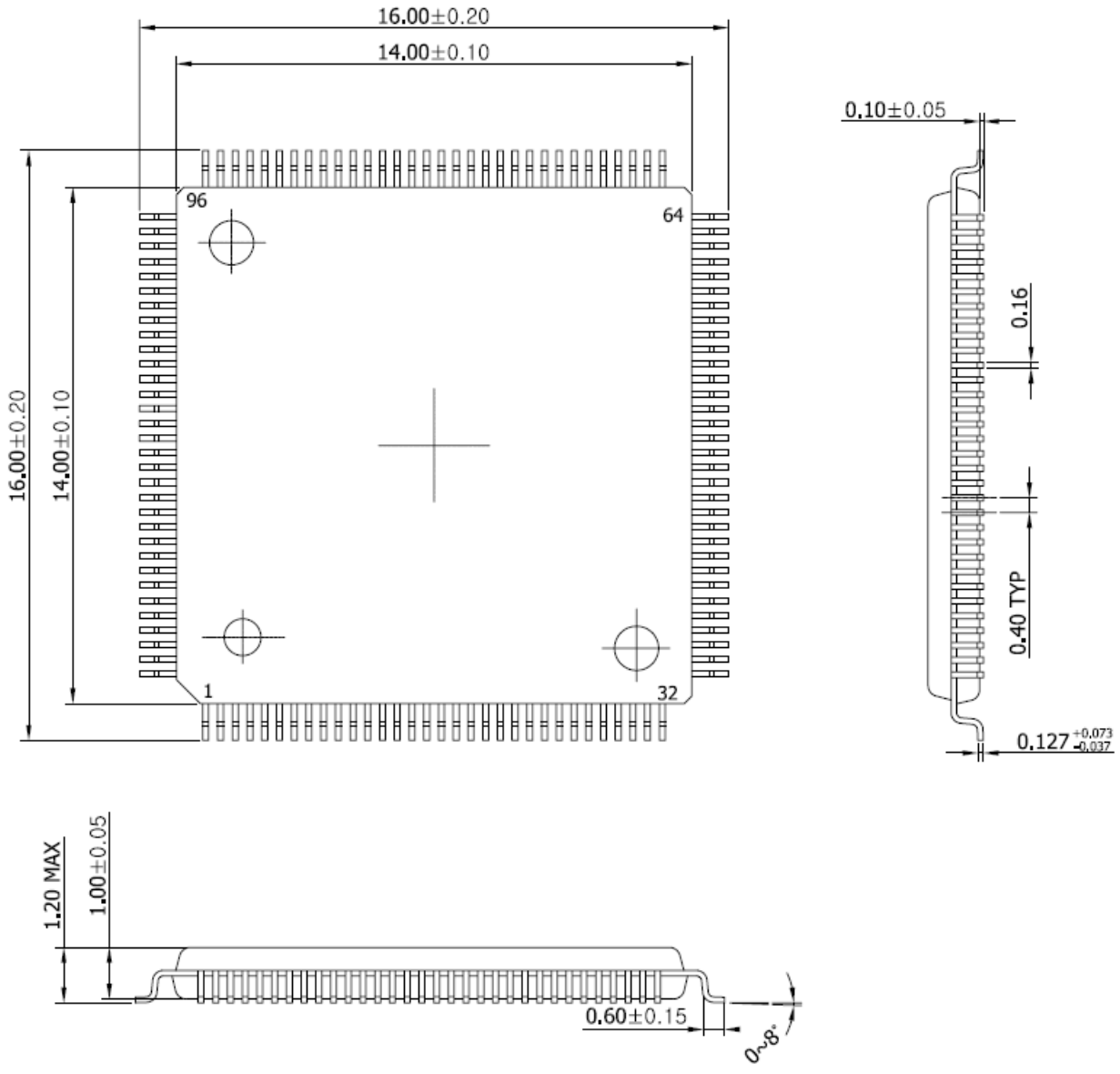
Figure 17-2: Input Timing Measurement Conditions

Symbol	3.3V Signaling	Units
V_{th}	$0.6V_{cc}$	V
V_{tl}	$0.2V_{cc}$	V
V_{test}	$0.4V_{cc}$	V
V_{trise}	$0.285V_{cc}$	V
V_{tfall}	$0.615V_{cc}$	V
V_{max}	$0.4V_{cc}$	V
Input Signal Slew Rate	1.5	V/ns

Table 17-2: PCI Bus Timing Measurement Condition Parameters

18. Package Outline

128Pin TQFP: Thin Quad Flat Package; Body 14 x 14 x 1.4 mm



Note :

All dimensions are in millimeters.