

## Introduction

This product specification defines the architecture, hardware (signal) interface, software (register) interface and parameterization options for the XPS IIC module. It provides a low speed, two wire, serial bus interface to a large number of popular devices. XPS IIC supports all features, except high speed mode, of the Philips I<sup>2</sup>C bus, V2.1, Release January 2000. (See the Specification Exceptions section for details.)

## Features

- Connects as a 32-bit Slave on PLB V4.6 bus of 32, 64 and 128 bits data width
- Master or slave operation
- Multi-master operation
- Software selectable acknowledge bit
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt with automatic mode switching from master to slave
- START and STOP signal generation/detection
- Repeated START signal generation
- Acknowledge bit generation/detection
- Bus busy detection
- Fast mode 400 KHz operation or standard mode 100 KHz
- 7 bit or 10 bit addressing
- General call enable or disable
- Transmit and receive FIFOs - 16 bytes deep
- Throttling
- General purpose output, 1 bit to 8 bits wide
- Dynamic Start/Stop generation
- Filtering on the SCL and SDA signals to eliminate spurious pulses

LogiCORE™ IP Facts	
Core Specifics	
Supported Device Family	Spartan®-3A/3A DSP, Spartan-3, Spartan-3E, Automotive Spartan 3/3E/3A/3A DSP, Spartan-6, Virtex®-4 /4Q/4QV, Virtex-5/5FX, Virtex-6/6CX
Resources Used	
See <a href="#">Table 19</a> , <a href="#">Table 20</a> , <a href="#">Table 21</a> , <a href="#">Table 22</a> and <a href="#">Table 23</a> .	
Provided with Core	
Documentation	Product Specification
Design File Formats	VHDL
Constraints File	N/A
Verification	N/A
Instantiation Template	N/A
Design Tool Requirements	
Xilinx Implementation Tools	ISE® 11.4 or later
Verification	ModelSim PE/SE 6.4b or later
Simulation	ModelSim PE/SE 6.4b or later
Synthesis	XST
Support	
Provided by Xilinx, Inc.	

## Functional Description

Figure 1 illustrates the top-level block diagram for the XPS IIC Bus Interface module.

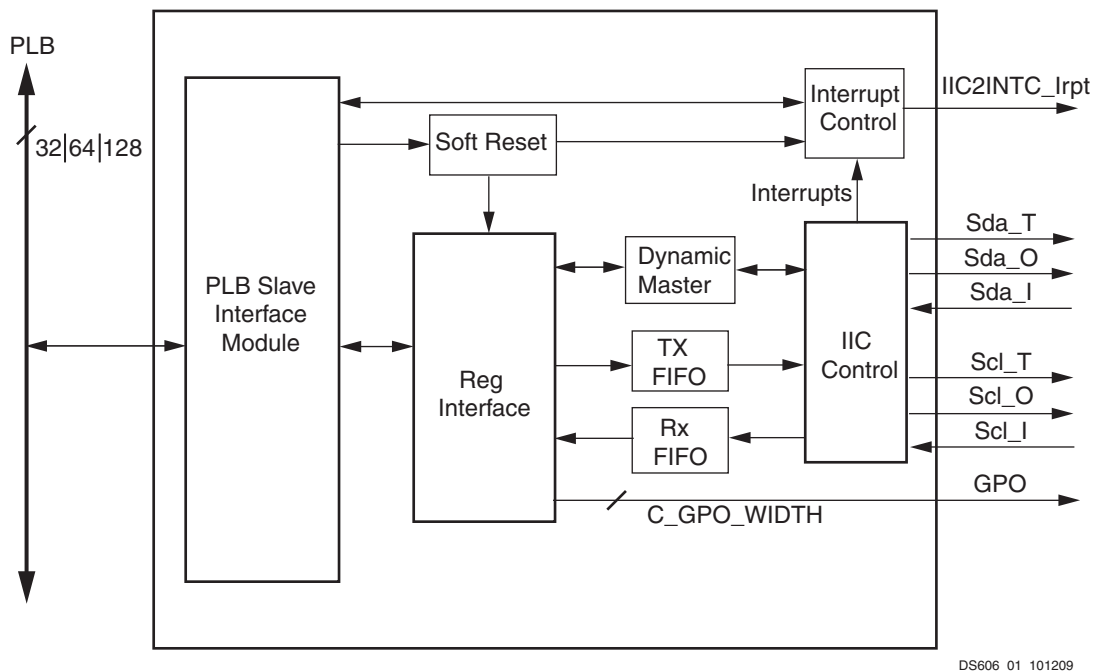


Figure 1: PLB Interconnect Diagram

The PLB Slave Interface Module provides the transaction interface to the PLB bus. A register interface block implements the address map and connectivity for the firmware to control IIC data transfer operations.

The XPS IIC does not provide explicit electrical connectivity to the IIC bus. The module expects the design to include bi-directional I/O buffers which implement open collector drivers for the SDA and SCL signals. Consequently, the user must provide external pull up devices to properly hold the bus at the logic 1 state when the connection to ground is removed.

The user must pay proper attention to the Philips specification when setting the values of these pull up devices (typically resistors) to both meet the Philips specification, the FPGA maximum ratings and the ratings of any devices on the bus itself.

## Multi Master Operation

The controller only participates in multi master arbitration when the bus is initially free when the attempt is made. After it issues the START other masters may participate in addressing and the XPS IIC will correctly relinquish the bus if another master presents a lower address than it presents. However, if the bus is not free, as indicated by SDA being low and SCL high (the START has occurred), when the request to acquire the bus is made then the XPS IIC will wait until the next bus free opportunity to arbitrate.

## Dynamic IIC Controller Logic

The dynamic controller logic provides an interface to the XPS IIC controller that is simple to use. The dynamic logic supports master mode only and 7 bit addressing only.

The dynamic logic is controlled by a start and stop bit that is located in the transmit FIFO. If neither of these bits are set, then the dynamic logic is disabled. Both of these bits are included in the FIFO and are acted upon as the Tx\_FIFO is emptied.

## Signal Filtering

The Philips Specification for I2C indicates that 0 to 50 ns of pulse rejection may be applied when operating in fast mode (>100 kHz). The user may specify the max amount allowed by the specification or more through the filtering parameters C\_SCL\_INERTIAL\_DELAY and C\_SDA\_INERTIAL\_DELAY. These parameters specify the amount of delay in *clock cycles*.

Some designs may not require any filtering and others (even those operating below <100 kHz) may require the maximum amount -- and possibly more. It depends on many factors beyond the control of the core itself. It may be necessary for the user to experiment to determine the optimum amount. In the event that more the 50 ns of pulse rejection is required it may be necessary for the user to more tightly constrain rise/fall times beyond what is required by the Philips specification to accommodate the additional delay occurring because of the filter operation.

## Design Parameters

To meet system resource or performance requirements the user may uniquely tailor the features of the XPS IIC instance via its design parameters. [Table 1](#) provides a comprehensive list of parameters and the corresponding features configurable by the user.

**Table 1: XPS IIC Design Parameters**

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
<b>System Parameter</b>					
G1	Device Family	C_FAMILY	spartan3, spartan3e, spartan3a, spartan3adsp, aspartan3, aspartan3e, aspartan3a, aspartan3adsp, spartan6, virtex4, qvirtex4, qvvirtex4, virtex5, virtex5fx, virtex6, virtex6cx	virtex5	string
<b>PLB Interface Parameters</b>					
G2	Device base address	C_BASEADDR	Valid Word Aligned Address <sup>[1]</sup>	None <sup>[2]</sup>	std_logic_vector
G3	Device maximum address	C_HIGHADDR	C_HIGHADDR - C_BASEADDR must be a power of 2 >= to C_BASEADDR+1FF F <sup>[1]</sup>	None <sup>[2]</sup>	std_logic_vector

Table 1: XPS IIC Design Parameters (Cont'd)

Generic	Feature / Description	Parameter Name	Allowable Values	Default Value	VHDL Type
G4	PLB Master ID Width	C_SPLB_MID_WIDTH	$\log_2(\text{C\_SPLB\_NUM\_MASTERS})$ with a minimum value of 1	1	integer
G5	Number of PLB Masters	C_SPLB_NUM_MASTERS	1 to 16	1	integer
G6	Width of the PLB Least Significant Address Bus	C_SPLB_AWIDTH	32	32	integer
G7	Width of the PLB Data Bus	C_SPLB_DWIDTH	32, 64, 128	32	integer
<b>XPS IIC Features</b>					
G8	Maximum frequency of the master mode generated SCL clock signal (Hz)	C_IIC_FREQ	Less than or equal to 400KHz for Fast Mode definition <sup>[3]</sup>	100E3	integer
G9	10 Bit Addressing	C_TEN_BIT_ADR	1 = The slave responds to 10 bit addresses 0 = The slave responds to 7 bit addresses	0	integer
G10	Width of GPO	C_GPO_WIDTH	1 to 8	1	integer
G11	SPLB Clock Frequency (Hz)	C_CLK_FREQ	See Allowable Parameter Combinations	25E6	integer
G12	SCL filtering	C_SCL_INERTIAL_DELAY	0-255 <sup>[4]</sup>	0	integer
G13	SDA filtering	C_SDA_INERTIAL_DELAY	0-255	0	integer

1. Address range specified by C\_BASEADDR and C\_HIGHADDR must be at least 0x200 and must be a power of 2.
2. No default value will be specified to insure that the actual value is set, i.e. if the value is not set, a compiler error will be generated.
3. The XPS IIC will only meet this frequency exactly when C\_IIC\_FREQ divides evenly into C\_CLK\_FREQ.
4. A value of 0 indicates that no filtering is applied to the given signal.

## Parameter Description

### C\_IIC\_FREQ

This parameter determines the approximate frequency of the master mode generated SCL clock signal (Hz). For C\_IIC\_FREQ ≤ 100,000 the appropriate timing specifications for standard mode operation are used. For C\_IIC\_FREQ > 100,000 the specifications for fast mode operation are used. See the *Philips I<sup>2</sup>C-bus Specification, Version 2.1*, January 2000 for details.

#### Note:

1. The actual measure SCL clock may vary from the value specified in the C\_IIC\_FREQ parameter for a number of reasons. In particular low period and high period of the clock are determined by counting off system clocks from the moment the SCL signal is sampled low or sampled high. As a result, the rise and fall time of the signals will affect the SCL clock frequency.

2. The C\_IIC\_FREQ does not equate to the data bandwidth in bps. Overhead in transmitting START, STOP and addresses reduces the effective bandwidth below the line rate.
3. IIC slaves may also control the clock rate to throttle data transfers to a manageable speed thus changing the effective SCL clock rate.
4. The actual frequency will only match the specified value when C\_CLK\_FREQ is an integer multiple of C\_IIC\_FREQ.

### **C\_TEN\_BIT\_ADR**

This parameter enables or disables the 10-bit addressing mode. Logic resource savings result when 10-bit addressing is disabled.

### **C\_GPO\_WIDTH**

This parameter sets the width of the general purpose output vector. If the user does not connect anything to this port, then logic optimization will remove any resources associated with it.

### **C\_CLK\_FREQ**

This parameter specifies (but does not set) the frequency of the PLB bus. The XPS IIC utilizes the SPLB\_CLK for its system clock and it must know the ratio of the C\_IIC\_FREQ to the C\_CLK\_FREQ to meet IIC timing specifications.

### **C\_SCL\_INERTIAL\_DELAY, C\_SDA\_INERTIAL\_DELAY**

These parameters specify the number of SPLB\_CLK cycles used to define the width of the pulse rejection. For example, a 100 Mhz clock coupled with an C\_SCL\_INERTIAL\_DELAY value of 5 gives 50 ns of pulse rejection. And, incidentally, delays the signal internally by 50 ns.

Filtering SCL without filtering SDA can have potentially undesirable effects by causing SDA to change when SCL is high resulting in false *starts* occurring.

Likewise, increasing the pulse rejection width for SDA beyond that for SCL may eliminate erroneous starts/stops by increasing the SDA hold time. Although technically the Philips specification permits 0 ns of hold time, in practice the sloppy signalling in IIC systems (caused by very large rise/fall times, due to high bus capacitance and high pull-up resistance and/or the use of level translation mosfets) can result in failures due to the high speed sampling of these analog signals.

In particular Virtex-5 I/O are so fast that the slow signal rise time plus noise exceeding the input hysteresis can result in phantom pulses internal to the circuit. The filters remove these quite effectively.

XPS IIC core provides 0 ns SDA hold time in master mode operation. If any IIC slave requires additional hold time on the SDA from the core, this can be achieved by adding delay on SCL (C\_SCL\_INERTIAL\_DELAY). For example, for 100 Mhz PLB clock, to have 300 ns hold time, C\_SCL\_INERTIAL\_DELAY parameter should be configured for a integer value of 30. The parameter C\_SDA\_INERTIAL\_DELAY can be set to 0 to 5 as per the required pulse rejection.

### **C\_BASEADDR, C\_HIGHADDR**

These two parameters determine the address range in PLB address space where the XPS IC registers reside.

The address range defined by C\_BASEADDR and C\_HIGHADDR for the XPS IIC must be a power of 2 and greater than or equal to 512 bytes (0x200). For example, if C\_BASEADDR = 0xE0000000 then C\_HIGHADDR must be 0xE0000200, 0xE0000400, . . . , etc.

### **C\_SPLB\_MID\_WIDTH**

This parameter is defined as an integer and has a minimum value of 1. It is equal to  $\log_2$  of the number of PLB Masters connected to the PLB bus or 1, whichever is greater. It is used to size the PLB\_masterID bus input from the PLB Bus to the Slave Attachment. For example, if eight PLB Masters are connected to the PLB Bus, then this parameter must be set to  $\log_2(8)$  which is equal to 3. The PLB\_masterID signal would then be sized to 3 bits wide. If only one master exists, then the parameter needs to be set to 1.

### **C\_SPLB\_NUM\_MASTERS**

This parameter is defined as an integer and is equal to the number of Masters connected to the PLB bus. This parameter is used to size the SI\_MBusy and SI\_MErr slave reply buses to the PLB. For example, if eight PLB Masters are connected to the PLB Bus, then this parameter must be set to 8. The SI\_MBusy bus and SI\_MErr bus will be sized to 8 bits wide each.

### **C\_SPLB\_AWIDTH**

This integer parameter is used by the PLB Slave to size the PLB address related components within the Slave Attachment. This value should be set 32.

### **C\_SPLB\_DWIDTH**

This integer parameter is used by the PLB Slave to size PLB data bus related components within the Slave Attachment. This value should be set to match the actual width of the PLB bus, 32, 64 or 128-Bits.

### **C\_FAMILY**

This parameter is defined as a string. It specifies the target FPGA technology for implementation of the PLB Slave. This parameter is required for proper selection of FPGA primitives. The configuration of these primitives can vary from one FPGA technology family to another.

## **Allowable Parameter Combinations**

Because of the pipelined design of the XPS IIC, the PLB bus clock frequency must be at least 25 MHz and 25 times faster than the SCL clock frequency.

## I/O Signals

The I/O signals for the XPS IIC are listed in [Table 2](#).

**Table 2: XPS IIC I/O Signal Description**

Port	Signal Name	Interface	I/O	Initial State	Description
<b>System Signals</b>					
P1	SPLB_Clk	PLB Bus	I	-	PLB bus clock
P2	SPLB_Rst	PLB Bus	I	-	PLB bus reset
P3	IIC2INTC_Irpt	System	O	0	System Interrupt output
<b>PLB Master Interface Signals</b>					
P4	PLB_ABus[0 : 31]	PLB	I	-	PLB address bus
P5	PLB_PAValiid	PLB	I	-	PLB primary address valid indicator
P6	PLB_masterID[0 : C_SPLB_MID_WIDTH - 1]	PLB	I	-	PLB current master identifier
P7	PLB_RNW	PLB	I	-	PLB read not write
P8	PLB_BE[0 : [C_SPLB_DWIDTH/8] - 1]	PLB	I	-	PLB byte enables
P9	PLB_size[0 : 3]	PLB	I	-	PLB transfer size
P10	PLB_type[0 : 2]	PLB	I	-	PLB transfer type
P11	PLB_wrDBus[0 : C_SPLB_DWIDTH - 1]	PLB	I	-	PLB write data bus
<b>Unused PLB Master Interface Signals</b>					
P13	PLB_UABus[0 : 31]	PLB	I	-	PLB upper address bits
P14	PLB_SAValiid	PLB	I	-	PLB secondary address valid
P15	PLB_rdPrim	PLB	I	-	PLB secondary to primary read request indicator
P16	PLB_wrPrim	PLB	I	-	PLB secondary to primary write request indicator
P17	PLB_abort	PLB	I	-	PLB abort bus request
P18	PLB_busLock	PLB	I	-	PLB bus lock
P19	PLB_MSize[0 : 1]	PLB	I	-	PLB data bus width indicator
P20	PLB_TAttribute[0 : 15]	PLB	I	-	PLB transfer attribute
P21	PLB_lockerr	PLB	I	-	PLB lock error
P22	PLB_wrBurst	PLB	I	-	PLB burst write transfer
P23	PLB_rdBurst	PLB	I	-	PLB burst read transfer
P24	PLB_wrPendReq	PLB	I	-	PLB pending bus write request
P25	PLB_rdPendReq	PLB	I	-	PLB pending bus read request
P26	PLB_rdPendPri[0 : 1]	PLB	I	-	PLB pending read request priority
P27	PLB_wrPendPri[0 : 1]	PLB	I	-	PLB pending write request priority
P28	PLB_reqPri[0 : 1]	PLB	I	-	PLB current request priority

Table 2: XPS IIC I/O Signal Description (Cont'd)

Port	Signal Name	Interface	I/O	Initial State	Description
<b>PLB Slave Interface Signals</b>					
P29	SI_addrAck	PLB	O	0	Slave address acknowledge
P30	SI_SSize[0 : 1]	PLB	O	0	Slave data bus size
P31	SI_wait	PLB	O	0	Slave wait indicator
P32	SI_rearbitrate	PLB	O	0	Slave rearbitrate bus indicator
P33	SI_wrDack	PLB	O	0	Slave write data acknowledge
P34	SI_wrComp	PLB	O	0	Slave write transfer complete indicator
P35	SI_rdBus[0 : C_SPLB_DWIDTH-1]	PLB	O	0	Slave read data bus
P36	SI_rdDack	PLB	O	0	Slave read data acknowledge
P37	SI_rdComp	PLB	O	0	Slave read transfer complete indicator
P38	SI_MBusy[0 : C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Slave busy indicator
P39	SI_MWrErr[0 : C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Slave write error indicator
P40	SI_MRdErr[0 : C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Slave read error indicator
<b>Unused PLB Slave Interface Signals</b>					
P41	SI_wrBTerm	PLB	O	0	Slave terminate write burst transfer
P42	SI_rdWdAddr[0 : 3]	PLB	O	0	Slave read word address
P43	SI_rdBTerm	PLB	O	0	Slave terminate read burst transfer
P44	SI_MIRQ[0 : C_SPLB_NUM_MASTERS - 1]	PLB	O	0	Master interrupt request
<b>IIC Signals</b>					
P45	Sda_I	System	I	-	IIC Serial Data Input from 3-state buffer.
P46	Sda_O	System	O	0	IIC Serial Data Output to 3-state buffer
P47	Sda_T	System	O	0	IIC Serial Data Output Enable to 3-state buffer <sup>[1]</sup>
P48	Scl_I	System	I	-	IIC Serial Clock Input from 3-state buffer
P49	Scl_O	System	O	0	IIC Serial Clock Output to 3-state buffer
P50	Scl_T	System	O	0	IIC Serial Clock Output Enable to 3-state buffer <sup>[1]</sup>
P51	Gpo(32 - C_GPO_WIDTH: 31)	System	O	0	General Purpose Outputs

1. The Sda\_T and Scl\_T signals are the 3-state enable signals that control the data direction for the Sda and Scl signal.



## Parameter - Port Dependencies

The width of some of the XPS IIC signals depends on parameters selected in the design. The dependencies between the XPS IIC design parameters and I/O signals are shown in [Table 3](#).

**Table 3: XPS IIC Parameter Port Dependencies**

Generic or Port	Name	Affects	Depends	Relationship Description
<b>Design Parameters</b>				
G3	C_HIGHADDR	-	G2	Address range pair dependency
G4	C_SPLB_MID_WIDTH	P6	G5	Affects the width of current master identifier signals and depends on $\log_2(\text{C\_SPLB\_NUM\_MASTERS})$ with a minimum value of 1
G5	C_SPLB_NUM_MASTERS	P38, P39, P40, P44	-	Affects the width of busy and error signals
G7	C_SPLB_DWIDTH	P8, P11, P35	-	Affects number of bits in slave data bus
G10	C_GPO_WIDTH	P51	-	Specifies signal vector width
<b>I/O Signals</b>				
P6	PLB_masterID	-	G4	Width varies with the size of the PLB master identifier bus
P8	PLB_BE	-	G7	Width varies with the value of C_SPLB_DWIDTH/8
P11	PLB_wrDBus	-	G7	Width varies with the value of C_SPLB_DWIDTH
P35	SI_rdDBus	-	G7	Width varies with the value of C_SPLB_DWIDTH
P38	SI_MBusy	-	G5	Width varies with the value of C_SPLB_NUM_MASTERS
P39	SI_MWrErr	-	G5	Width varies with the value of C_SPLB_NUM_MASTERS
P40	SI_MRdErr	-	G5	Width varies with the value of C_SPLB_NUM_MASTERS
P44	SI_MIRQ	-	G5	Width varies with the value of C_SPLB_NUM_MASTERS
P51	GPO	-	G10	Width varies with the value of C_GPO_WIDTH

## Protocol and Electrical Characteristics of IIC

To understand and utilize the register based software interface in the XPS IIC module it is helpful to have a basic understanding of the IIC protocol, and the electrical characteristics of the bus. For more details and timing diagrams, see the Philips I<sup>2</sup>C specification.

## Electrical Issues

An IIC bus consists of two wires named serial data (SDA) and serial clock (SCL), which carry information between the devices connected to the bus. The 400 pF maximum signal load capacitance limits the maximum number of devices connectable to the same bus.

Both SDA and SCL transport data bidirectionally between connected devices using wired-and electrical connectivity. To implement the wired-and each device utilizes an open-collector / open-drain output that only sinks current to ground to pull the signal to logic-0. Electrically that means it may not *drive* a logic-1 on to either bus signal but may only *release* or *float* the output. When no device asserts a logic-0 onto the bus, external pull-up devices (typically resistors) bring the signal state high. This method creates a source of confusion since no device may actually *set* the state of SCL or SDA to its high (logic-1) state.

The system designer must pay careful attention to the value of these pull-ups to guarantee that the implementation (consisting of the XPS IIC pcore, the Xilinx FPGA, and other devices on the bus) does not violate the IIC timing parameters. Selecting the value of pull-up resistors for a particular application is beyond the scope of the XPS IIC and this document.

The user should consider utilizing the small, additional amount of logic necessary for filtering of SCL and SDA by specifying non-zero values for the parameters C\_SCL\_INERTIAL\_DELAY and C\_SDA\_INERTIAL\_DELAY. Reliability of the system may increase substantially.

When all devices on the bus release their drivers and both SDA and SCL are high for a specified period of time the bus is considered to be in the *bus free* state.

## Protocol for Address and Data Transfer

Each device on the bus has a unique seven-bit or ten-bit address, operate both as a transmitter and receiver and additionally may be a master or slave. Master device initiate data transfer on the bus and generate the clock signal for that transfer. Slaves respond to the address clocked into them by the master and either accept ("write") data from or provide ("read") data to the master

The IIC protocol defines an arbitration procedure that insures that if more than one Master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted. The arbitration and clock synchronization procedures defined in the Philips IIC specification are supported by the XPS IIC Bus Interface module.

Data transfers on the IIC bus are initiated with a START condition and are terminated with a STOP condition. After reaching the bus free state a master may signal a START defined by a high-to-low transition on SDA while SCL is high. Likewise, the master signals a STOP by a low-to-high transition on the SDA line while SCL is high. Between the START and STOP conditions of the bus, data on the SDA signal must be stable during the high period of the SCL signal and must meet any required setup and hold times during the low period of the SCL signal.

**Figure 2** illustrates how the definitions of: (a) the bus free state, and (b) the times when SDA and SCL may change relative to each other, ensure that the START and STOP conditions are not confused as data.

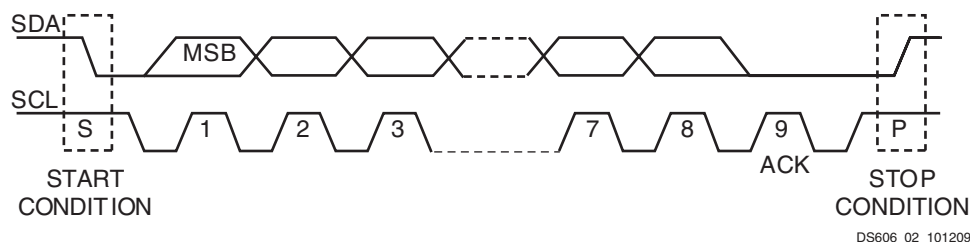


Figure 2: Data Transfer on the IIC Bus

Each transfer on the IIC bus consists of nine clock pulses on SCL to move eight bits of data and one acknowledge bit. Master and slave transmitter send data with the most significant bit first (MSB).

After providing data for the eight clock period, the (master | slave) transmitter releases the SDA line during the acknowledgement clock period to permit the receiver to transfer a 1-bit acknowledgment.

If a slave-receiver issues a not-acknowledge (by releasing the SDA signal during the acknowledgement clock period) this indicates that the slave-receiver was unable to accept the prior 8-bits transferred (consisting of address or data bits.) Note that after a byte of data is transferred the slave (receiver | transmitter) has the unique capability to throttle the transfer by keeping the SCL line in its low state by actively pulling the SCL line low for an arbitrary period of time.<sup>(1)(2)</sup> This ability allows it time to determine internally what value it should place on the SDA line for the acknowledgement.

If the *master*-receiver signals a not-acknowledge, this indicates to the *slave*-transmitter that this byte was the last byte of the transfer.

Standard communication on the bus between a Master and a Slave is composed of four parts: START, Slave address, Data transfer, and STOP. The IIC protocol defines a transfer format for both 7-bit and 10-bit addressing.

A seven bit address is initiated as follows. After the START condition, a Slave address is sent. This address is seven bits long followed by an eighth-bit which is the read/write bit. A High indicates a request for data (read) and a Low indicates a data transmission (write).

Only the Slave with the calling address that matches the address transmitted by the Master (that won arbitration) responds by sending back an acknowledge bit by pulling the SDA line Low on the ninth clock.

For 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slave that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the slave address bits 9:8, and the LSB bit (bit 8) is the R/W bit. The second byte transferred sets bits 7:0 of the slave address.

1. The wired-and nature of the bus signals and *pull-low* or *release* output capability of each device permit bi-directional data transfer.

2. This means the master and slaves in the system cooperatively determine the speed of data transfers. The masters set the maximum speed and the slaves (and/or masters) can arbitrarily slow it down as needed. It also means that because the master may only release the SCL line, that it must check to see that the SDA actually went high before proceeding with the next clock period.

Once successful Slave addressing is achieved, the data transfer proceeds byte-by-byte in the direction as specified by the read/write bit.

The Master can terminate the communication by generating a STOP signal to free the bus when the receiver signals a not-acknowledge (signaled by releasing SDA during the acknowledgement clock period.) However, the Master may generate a START signal without generating a STOP signal first. This is called a repeated START. A repeated start allows the master to change the direction of data transfer or address a different slave without giving up the bus.

## XPS IIC Register Descriptions

Table 4 specifies the name, offset and accessibility of each firmware addressable register from the three classes of registers within the XPS IIC pcore. User access to each register is from an offset to the base address set the C\_BASEADDR parameter. For example, C\_BASEADDR + 0x100 represents the address of the Control Register (CR).

Table 4 shows all of the XPS IIC registers and their addresses.

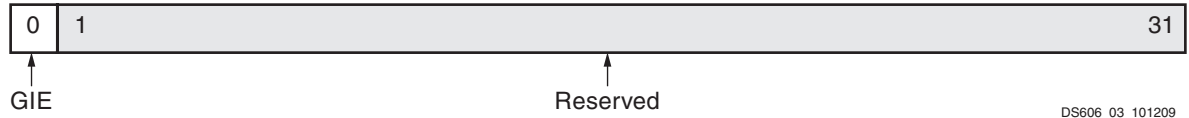
**Table 4: XPS IIC Registers**

Base Address + Offset (hex)	Register Name	Access Type	Default value (hex)
<b>Interrupt Registers</b>			
C_BASEADDR + 0x01C	Global Interrupt Enable (GIE) <sup>[1]</sup>	Read/Write	0x00
C_BASEADDR + 0x020	Interrupt Status Register (ISR) <sup>[1]</sup>	Read/Toggle <sup>[3]</sup> on Write	0xD0
C_BASEADDR + 0x028	Interrupt Enable Register (IER) <sup>[1]</sup>	Read/Write	0x00
<b>Soft Reset</b>			
C_BASEADDR + 0x040	Soft Reset Register (SOFT) <sup>[2]</sup>	Write Only	N/A
<b>IIC Configuration, Control, Data</b>			
C_BASEADDR + 0x100	Control Register (CR)	Read/Write	0x00
C_BASEADDR + 0x104	Status Register (SR)	Read	0xC0
C_BASEADDR + 0x108	Transmit FIFO (Tx_FIFO)	Read/Write	0x00
C_BASEADDR + 0x10C	Receive FIFO (Rc_FIFO)	Read	N/A
C_BASEADDR + 0x110	Slave Address Register (ADR)	Read/Write	0x00
C_BASEADDR + 0x114	Transmit FIFO Occupancy Register (Tx_FIFO_OCY)	Read	0x00
C_BASEADDR + 0x118	Receive FIFO Occupancy Register (Rc_FIFO_OCY)	Read	0x00
C_BASEADDR + 0x11C	Slave Ten Bit Address Register (TEN_ADR)	Read/Write	0x00
C_BASEADDR + 0x120	Receive FIFO Programmable Depth Interrupt Register (Rc_FIFO_PIRQ)	Read/Write	0x0
C_BASEADDR + 0x124	General Purpose Output Register (GPO)	Read/Write	0x0

1. See page 11, product data sheet DS516 Interrupt Control V2.01a, July 2, 2008
2. The soft reset functionality is implemented by the proc\_common soft\_reset module now
3. Toggle each bit position to which a '1' is written

## Global Interrupt Enable (GIE)

The Global Interrupt Enable Register, illustrated in Figure 3, and described in Table 5, has a single defined bit, in the most significant bit that is used to globally enable the final interrupt (coalesced from the ISR) out to the system.



DS606\_03\_101209

Figure 3: Global Interrupt Enable (GIE) Register

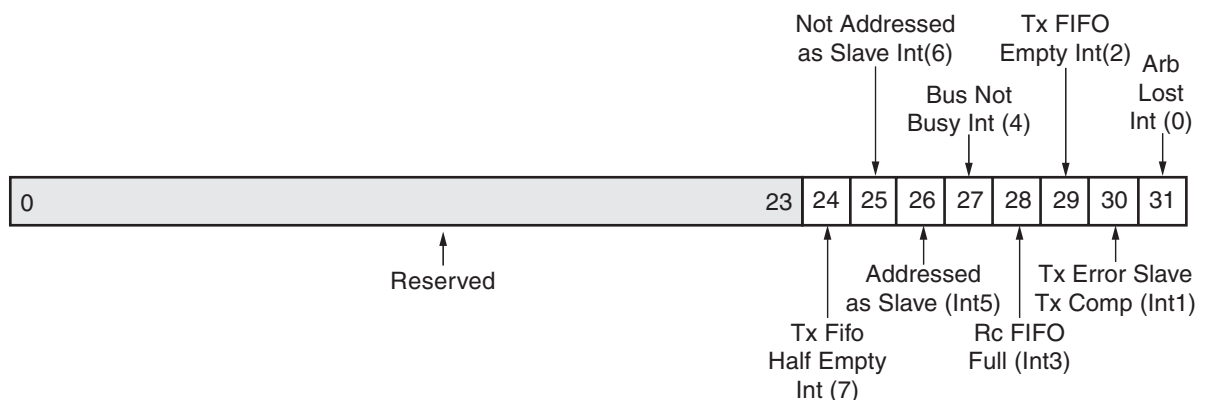
Table 5: Global Interrupt Enable (GIE) Register (C\_BASEADDR + 0x01C)

Bit(s)	Name	Core Access	Reset Value	Description
0	GIE	Read/Write	0	Global Interrupt Enable. 0 = All Interrupts disabled; no interrupt (even if unmasked in IER) possible from XPS IIC. 1 = Unmasked XPS IIC interrupts are passed to processor.
1-31	Reserved	N/A	N/A	Reserved

## Interrupt Status Register (ISR)

Firmware uses the ISR to determine which interrupt events from the XPS IIC need servicing. The register uses a toggle on write method to allow the firmware to easily clear selected interrupts by writing a '1' to the desired interrupt bit field position. This mechanism avoids the requirement on the User Interrupt Service routine to perform a Read/Modify/Write operation to clear a single bit within the register. Note that an interrupt value of '1' means the interrupt has occurred. A value of zero means that no interrupt occurred or it was cleared.

Table 6 illustrates the interrupt to bit field mappings of the IPIER (interrupt enable) and IPISR (interrupt status) registers. The number in the parenthesis is the interrupt bit number.



DS606\_04\_101209

Figure 4: Interrupt Status Register (ISR)

**Table 6: Interrupt Status Register (C\_BASEADDR + 0x020)**

Bit(s)	Name	Core Access	Reset Value	Description
0 - 25	Reserved	N/A	N/A	Reserved
24	int(7)	Read/Toggle on Write	1	Interrupt(7) -- Transmit FIFO Half Empty
25	int(6)	Read/Toggle on Write	1	Interrupt(6) -- Not Addressed As Slave
26	int(5)	Read/Toggle on Write	0	Interrupt(5) -- Addressed As Slave
27	int(4)	Read/Toggle on Write	1	Interrupt(4) -- IIC Bus is Not Busy
28	int(3)	Read/Toggle on Write	0	Interrupt(3) -- Receive FIFO Full
29	int(2)	Read/Toggle on Write	0	Interrupt(2) -- Transmit FIFO Empty
30	int(1)	Read/Toggle on Write	0	Interrupt(1) -- Transmit Error/Slave Transmit Complete
31	int(0)	Read/Toggle on Write	0	Interrupt(0) -- Arbitration Lost

### Interrupt Enable Register (IER)

The firmware uses the fields of this register to enable or disable interrupts needed to manage either the Standard Controller Logic Flow or the Dynamic Controller Logic Flow.

**Table 7: Interrupt Enable Register (C\_BASEADDR + 0x028)**

Bit(s)	Name	Core Access	Reset Value	Description
0 - 25	Reserved	N/A	N/A	Reserved
24	int(7)	Read/Write	0	Interrupt(7) -- Transmit FIFO Half Empty
25	int(6)	Read/Write	0	Interrupt(6) -- Not Addressed As Slave
26	int(5)	Read/Write	0	Interrupt(5) -- Addressed As Slave
27	int(4)	Read/Write	0	Interrupt(4) -- IIC Bus is Not Busy
28	int(3)	Read/Write	0	Interrupt(3) -- Receive FIFO Full
29	int(2)	Read/Write	0	Interrupt(2) -- Transmit FIFO Empty
30	int(1)	Read/Write	0	Interrupt(1) -- Transmit Error/Slave Transmit Complete
31	int(0)	Read/Write	0	Interrupt(0) -- Arbitration Lost

1. In any given bit position, 1=Interrupt enabled, 0=Interrupt masked

### Soft Reset Register (SOFTR)

The firmware can write to the SOFTR to initialize the XPS IIC interrupt registers (GIE, ISR, IER) to its default state. To accomplish this the firmware must write the reset key value of 0xA to the least significant nibble of the 32-bit word. After recognizing a write of 0xA the proc\_common soft\_reset module issues a pulse 4 clocks long to reset the XPS IIC. At the end of the pulse the SOFTR

acknowledges the PLB transaction. That prevents anything further from happening while the reset occurs.

Writing any value to bits 28:31 other than 0xA results in a PLB transaction acknowledge with an error status. The register is not readable.

Applying soft reset to XPS IIC core also clears the bus busy status (bit-29 of SR) which may not give the correct status of the IIC bus, if IIC bus is locked by other IIC slave. Hence user should reset external slave after the application of the soft reset to the XPS IIC core prior using the IIC bus again.

See Figure 5 for an illustration of the SOFTR bit fields and Table 8 for a description of the SOFTR bit fields.

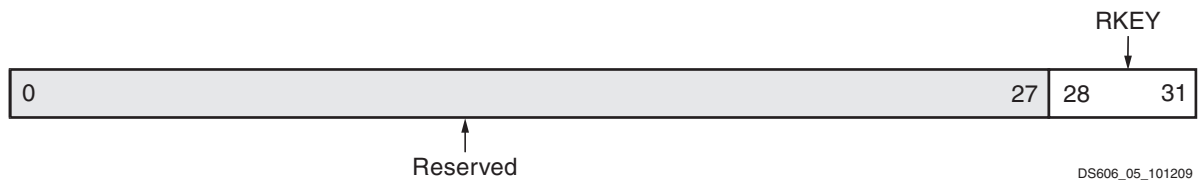


Figure 5: Soft Reset Register (SOFTR)

Table 8: Soft Reset Register (C\_BASEADDR + 0x040)

Bit(s)	Name	Core Access	Reset Value	Description
0-27	Reserved	N/A	N/A	Reserved
28-31	RKEY	Write	N/A	<b>Reset Key.</b> The firmware must write a value of 0xA to this field to cause a soft reset of the Interrupt registers of XPS IIC controller. Writing any other value results in a PLB transaction acknowledgement with error and no reset occurs.

## Control Register (CR)

Writing to the CR register configures the XPS IIC operation mode and simultaneously allows for IIC transactions to be kicked off.

Prior to setting Master Slave Mode Select (MSMS) to a 1, Tx\_FIFO should contain the address of the XPS IIC device. All the CR bits can be set at the same time as setting MSMS to a 1 to initiate a bus transaction.

When initiating a repeated start condition, the transmit FIFO must be empty. First, set the repeated start bit to a 1 and then write the address of the XPS IIC device to the transmit FIFO. The rest of the FIFO can be filled with data, if required.

The EN field provides a way for the device driver to initialize interrupts prior to enabling the device to send/receive data.

The XPS IIC Control Register is shown in Figure 6 and described in Table 9.

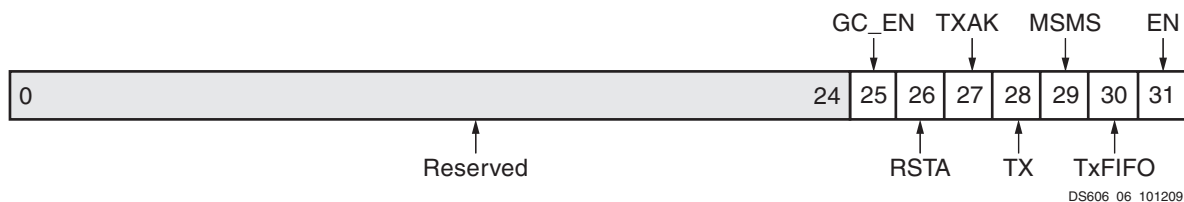


Figure 6: Control (CR) Register

Table 9: XPS IIC Control Register (C\_BASEADDR + 0x100)

Bit(s)	Name	Core Access	Reset Value	Description
0- 24	Reserved	N/A	N/A	Reserved
25	GC_EN	Read/Write	0	<b>General Call Enable.</b> Setting this bit High allows the XPS IIC to respond to a general call address. "0" - General Call Disabled. "1" - General Call Enabled.
26	RSTA	Read/Write	0	<b>Repeated Start.</b> Writing a "1" to this bit generates a repeated START condition on the bus if the XPS IIC Bus Interface is the current bus Master. Attempting a repeated START at the wrong time, if the bus is owned by another Master, results in a loss of arbitration. This bit is reset when the repeated start occurs. This bit must be set prior to writing the new address to the Tx_FIFO or DTR.
27	TXAK	Read/Write	0	<b>Transmit Acknowledge Enable.</b> This bit specifies the value driven onto the SDA line during acknowledge cycles for both Master and Slave receivers. "1" - ACK bit = "1" - not-acknowledge. "0" - ACK bit = "0" - acknowledge. Because Master receivers indicate the end of data reception by not acknowledging the last byte of the transfer, this bit is used to end a Master receiver transfer. As a slave, this bit must be set prior to receiving the byte to signal a not-acknowledge.
28	TX	Read/Write	0	<b>Transmit/Receive Mode Select.</b> This bit selects the direction of Master/Slave transfers. "1" selects an XPS IIC transmit. "0" selects an XPS IIC receive. This bit <u>does not control</u> the Read/Write bit that is sent on the bus with the address. The Read/Write bit that is sent with an address must be the <u>LSB of the address written into the Tx_FIFO</u> .

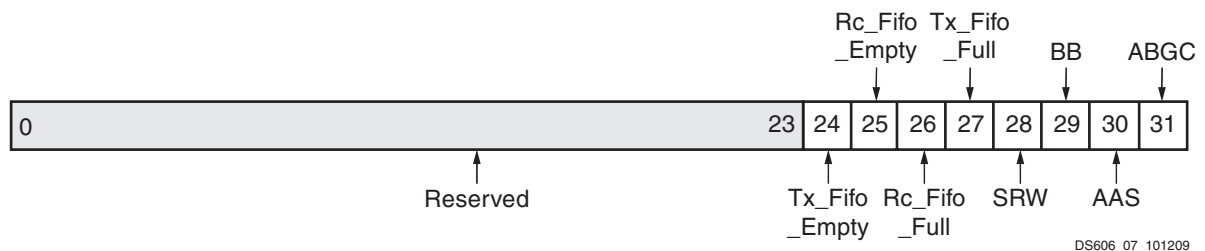


**Table 9: XPS IIC Control Register (C\_BASEADDR + 0x100) (Cont'd)**

Bit(s)	Name	Core Access	Reset Value	Description
29	MSMS	Read/Write	0	<b>Master/Slave Mode Select.</b> When this bit is changed from 0 to 1, the XPS IIC Bus Interface generates a START condition in Master mode. When this bit is cleared, a STOP condition is generated and the XPS IIC Bus Interface switches to Slave mode. When this bit is cleared by the hardware, because arbitration for the bus has been lost, a STOP condition is not generated. (See also: Arb Lost Interrupt)
30	Tx_FIFO Reset	Read/Write	0	<b>Transmit FIFO Reset.</b> This bit must be set to flush the FIFO if either (a) arbitration is lost or (b) if a transmit error occurs. "1" resets the Transmit FIFO. "0" Transmit FIFO normal operation.
31	EN	Read/Write	0	<b>XPS IIC Enable.</b> This bit must be set before any other CR bits have any effect. "1" enables the XPS IIC controller. "0" resets and disables the XPS IIC controller but not the registers or FIFOs.

## Status Register (SR)

This register contains the status of the XPS IIC Bus Interface. The read-only SR register is shown in Figure 7 and described in Table 10. All bits are cleared upon reset.


**Figure 7: Status Register (SR)**
**Table 10: Status Register (C\_BASEADDR + 0x104)**

Bit(s)	Name	Core Access	Reset Value	Description
0- 23	Reserved	N/A	N/A	Reserved
24	Tx_FIFO_Empty	Read	1	<b>Transmit FIFO empty.</b> This bit is set High when the transmit FIFO is empty.
25	Rc_FIFO_Empty	Read	1	<b>Receive FIFO empty.</b> This is set High when the receive FIFO is empty.
26	Rc_FIFO_Full	Read	0	<b>Receive FIFO full.</b> This bit is set High when the receive FIFO is full. This bit is set only when all 16 locations in the FIFO are full, regardless of the compare value field of the Rc_FIFO_PIRQ register.
27	Tx_FIFO_Full	Read	0	<b>Transmit FIFO full.</b> This bit is set High when the transmit FIFO is full.

Table 10: Status Register (C\_BASEADDR + 0x104) (Cont'd)

Bit(s)	Name	Core Access	Reset Value	Description
28	SRW	Read	0	<b>Slave Read/Write.</b> When the IIC Bus Interface has been addressed as a Slave (AAS is set), this bit indicates the value of the read/write bit sent by the Master. This bit is only valid when a complete transfer has occurred and no other transfers have been initiated. "1" indicates Master reading from Slave. "0" indicates Master writing to Slave.
29	BB	Read	0	<b>Bus Busy.</b> This bit indicates the status of the IIC bus. This bit is set when a START condition is detected and cleared when a STOP condition is detected. "1" indicates the bus is busy. "0" indicates the bus is idle.
30	AAS	Read	0	<b>Addressed as Slave.</b> When the address on the IIC bus matches the Slave address in the Address Register (ADR), the IIC Bus Interface is being addressed as a Slave and switches to Slave mode. If 10-bit addressing is selected this device will only respond to a 10-bit address or general call if enabled. This bit is cleared when a stop condition is detected or a repeated start occurs. "1" indicates being addressed as a slave. "0" indicates not being addressed as a slave.
31	ABGC	Read	0	<b>Addressed By a General Call.</b> This bit is set high when another master has issued a general call and the general call enable bit is set high, CR(25) = '1'.

## Transmit FIFO (Tx\_FIFO)

This is the keyhole address for the FIFO that contains data to be transmitted on the IIC bus. In transmit mode, data written into this FIFO is output on the IIC bus. Reading of this location will result in reading the current byte being output from the FIFO. Attempting to write to a full FIFO is not recommended and results in that data byte being lost. Firmware must clear the FIFO prior to use in anticipation of it not being empty possibly do to abnormal IIC protocol abnormal terminations or other normal controller actions such as dynamic mode reads. The Transmit FIFO (Tx\_FIFO) is shown in Figure 8 and described in Table 11.

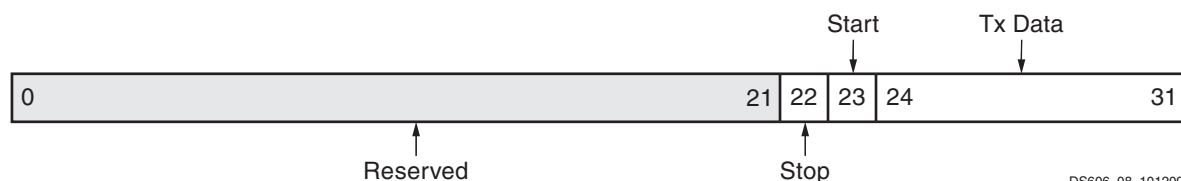


Figure 8: Transmit FIFO (Tx\_FIFO)

Table 11: XPS IIC Transmit FIFO (C\_BASEADDR + 0x108)

Bit(s)	Name	Access	Reset Value	Description
0 - 21	Reserved	N/A	N/A	Reserved
22	Stop	Read/Write	0	<b>Stop.</b> The dynamic stop bit may be used to send an IIC stop sequence on the IIC bus after the last byte has been transmitted or received. <sup>[2]</sup>

**Table 11: XPS IIC Transmit FIFO (C\_BASEADDR + 0x108)**

23	Start	Read/Write	0 <sup>[1]</sup>	<b>Start.</b> The dynamic start bit may be used to send a start or repeated start sequence on the IIC bus. A start sequence is generated if the MSMS = 0, a repeated start sequence is generated if the MSMS = 1. <sup>[2]</sup>
24 - 31	D0 - D7	Read/Write	Indeterminate <sup>[1]</sup>	XPS IIC Transmit Data. If the dynamic <b>Stop</b> bit is used and the XPS IIC is a master receiver, the value is the number of bytes to receive. <sup>[3]</sup>

1. The value that was available before the reset occurred will still appear on the FIFO outputs
2. A full description for the use of the dynamic stop and start bits is contained in the ["Dynamic Controller Logic Flow"](#) section. These bits are not readable
3. Only bits 24-31 can be read back.

## Receive FIFO (Rc\_FIFO)

This FIFO contains the data received from the IIC bus. The received IIC data is placed in this FIFO after each complete transfer. The Receive FIFO Occupancy Register (Rc\_FIFO\_OCY) must be equal to the [Receive FIFO Programmable Depth Interrupt Register \(Rc\\_FIFO\\_PIRQ\)](#) before throttling occurs. The receive FIFO is read only. Reading this FIFO when it is empty results in indeterminate data being read. The Receive FIFO is shown in [Figure 9](#) and described in [Table 12](#).

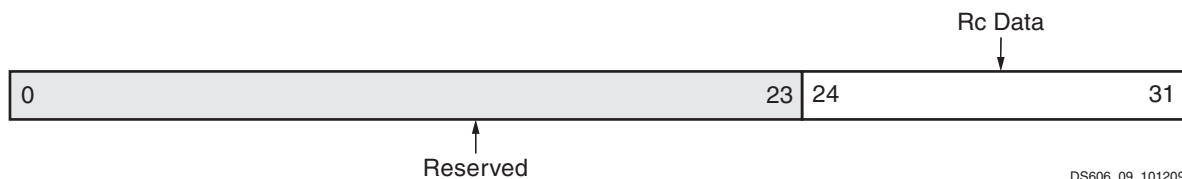


Figure 9: Receive FIFO (Rc\_FIFO)

Table 12: XPS IIC Receive FIFO (C\_BASEADDR + 0x10C)

Bit(s)	Name	Access	Reset Value	Description
0 - 23	Reserved	N/A	N/A	Reserved
24 - 31	D0 - D7	Read	Indeterminate <sup>[1]</sup>	IIC Receive Data

1. The value that was available before the reset occurred appears on the FIFO outputs

## Slave Address Register (ADR)

The user programs the ADR register (and possibly the TEN\_ADR register) to set the address at which the slave will acknowledge an address transfer operation from the bus master. The slave address field of the ADR register contains all 7-bits of the *7-bit address* or the least significant 7-bits of the *10-bit address* the XPS IIC Bus Interface recognizes when operating as a slave. Figure 10 illustrates the field layout of the register and Table 13 provides the detailed field descriptions.

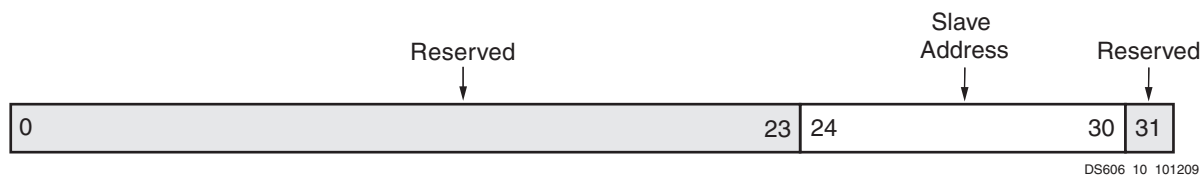


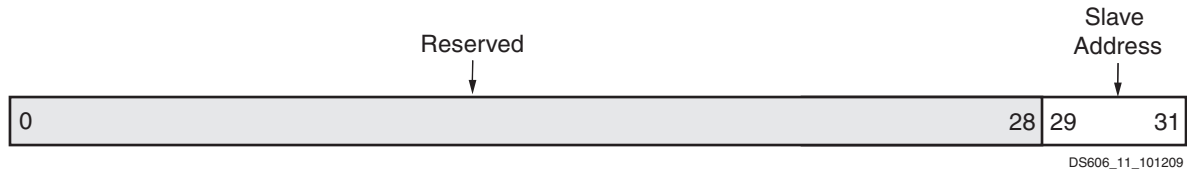
Figure 10: Address Register (ADR)

Table 13: Address Register (C\_BASEADDR + 0x110)

Bit(s)	Name	Access	Reset Value	Description
0 - 23	Reserved	N/A	N/A	Reserved
24-30	Slave Address	Read/Write	0x00	Address used by the IIC Bus Interface when in Slave mode
31	Reserved	N/A	N/A	Reserved

## Slave Ten Bit Address Register (TEN\_ADR)

The user programs the ADR register (and possibly the TEN\_ADR register) to set the address at which the slave will acknowledge an address transfer operation from the bus master. The slave address field of the TEN\_ADR register contains the most significant 3-bits of the *10-bit address* the XPS IIC Bus Interface recognizes when operating as a 10-bit addressable slave. This register exists only if the user configures the XPS IIC for 10-bit addressing by setting generic parameter C\_TEN\_BIT\_ADR to 1. The TEN\_ADR register is shown in Figure 11 and described in Table 14.



DS606\_11\_101209

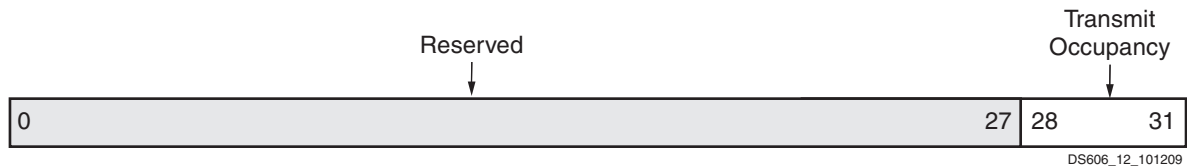
Figure 11: Ten Bit Address Register (TEN\_ADR)

Table 14: Ten Bit Address Register (C\_BASEADDR + 0x11C)

Bit(s)	Name	Access	Reset Value	Description
0 - 28	Reserved	N/A	N/A	Reserved
29 - 31	MSB of Slave Address	Read/Write	0x0	3 MSBs of the 10-bit Address used by the XPS IIC Bus Interface when in Slave mode.

## Transmit FIFO Occupancy Register (Tx\_FIFO\_OCY)

This field contains the occupancy value for the Transmit FIFO. Reading this register cannot be used to determine if the FIFO is empty. The Transmit FIFO Empty Interrupt conveys that information. The value read is the occupancy value minus one, therefore reading all zeros indicates that the first location is filled and reading all ones implies that all sixteen locations are filled. The Tx\_FIFO\_OCY register is shown in Figure 12 and described in Table 15.



DS606\_12\_101209

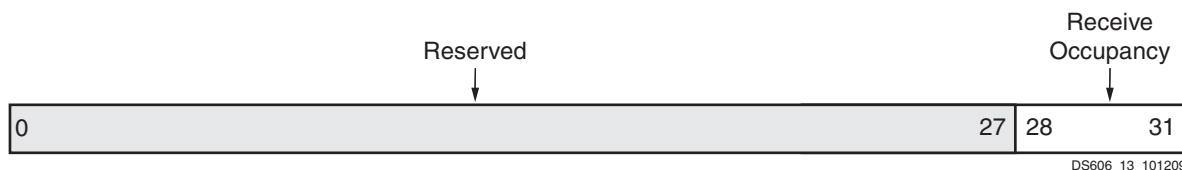
Figure 12: Transmit FIFO Occupancy Register (Tx\_FIFO\_OCY)

Table 15: Transmit FIFO Occupancy Register (C\_BASEADDR + 0x114)

Bit(s)	Name	Access	Reset Value	Description
0 - 27	Reserved	N/A	N/A	Reserved
28 -31	Occupancy Value	Read	0x0	Bit 28 is the MSB. A value of 1001 indicates that 10 locations in the FIFO are full.

## Receive FIFO Occupancy Register (Rc\_FIFO\_OCY)

This field contains the occupancy value for the Receive FIFO. This register is read only. Reading this register cannot be used to determine if the FIFO is empty. Rc\_FIFO\_Empty, bit(1) in the status register is used to convey that information. The value read is the occupancy value minus one, therefore reading all zeros implies that the first location is filled and reading all ones implies that all sixteen locations are filled. The Rc\_FIFO\_OCY register is shown in Figure 13 and described in Table 16.



DS606\_13\_101209

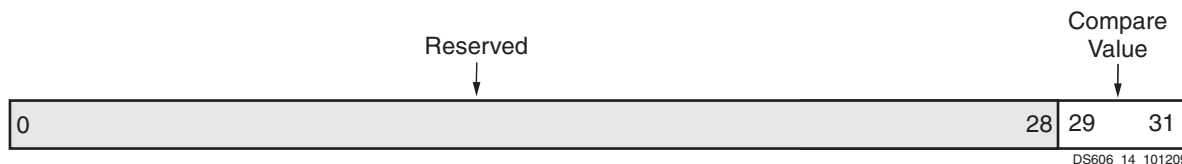
Figure 13: Receive FIFO Occupancy Register (Rc\_FIFO\_OCY)

Table 16: Receive FIFO Occupancy Register (C\_BASEADDR + 0x118)

Bit(s)	Name	Access	Reset Value	Description
0 - 27	Reserved	N/A	N/A	Reserved
28 - 31	Occupancy Value	Read	0x0	Bit 28 is the MSB. A value of 1001 implies that 10 locations in the FIFO are full.

### Receive FIFO Programmable Depth Interrupt Register (Rc\_FIFO\_PIRQ)

This field contains the value which causes the receive FIFO Interrupt to be set. When this value is equal to the Rc\_FIFO\_OCY value, the receive FIFO interrupt is set and remains set until the equality is no longer true. A read from the receive FIFO will cause the IIC Receive FIFO Interrupt to be cleared. When the Rc\_FIFO\_PIRQ is equal to the Rc\_FIFO\_OCY throttling will also occur to prevent the transmitter from transmitting. The read/write Rc\_FIFO\_PIRQ register is shown in Figure 14 and described in Table 17.



DS606\_14\_101209

Figure 14: Receive FIFO Programmable Depth Interrupt Register (Rc\_FIFO\_PIRQ)

Table 17: Receive FIFO Programmable Depth Interrupt Register (C\_BASEADDR + 0x120)

Bit(s)	Name	Access	Reset Value	Description
0 - 27	Reserved	N/A	N/A	Reserved
28 - 31	Compare Value	Read/Write	0x0	Bit 28 is the MSB. A value of 1001 implies that when ten locations in the receive FIFO are filled, the receive FIFO interrupt will be set.

### General Purpose Output Register (GPO)

The current value of the General Purpose Output field of the GPO register is reflected continuously on the GPO signal of the pcore. Thus the GPO signal of the pcore could be used to set an IIC memory device write protect for example.

If C\_GPO\_WIDTH is equal to one, the only bit populated in the register is GPO(31), the LSB. If C\_GPO\_WIDTH is equal to 8, then bits 24 through 31 in the GPO are populated. Reading unpopulated bits results in indeterminate data.

The GPO register is shown in Figure 15 and described in Table 18.

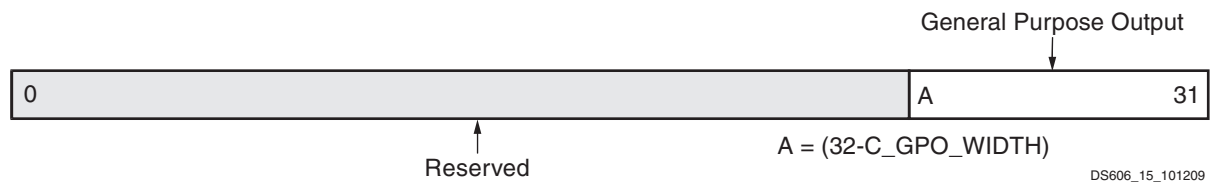


Figure 15: General Purpose Output Register (GPO)

Table 18: General Purpose Output Register (C\_BASEADDR + 0x124)

Bit(s)	Name	Access	Reset Value	Description
0 - (31-C_GPO_WIDTH)	Reserved	N/A	N/A	Reserved
(32-C_GPO_WIDTH) - 31	General Purpose Outputs	Read/Write	0x0	GPO - The LSB (bit 31) is the first bit populated.

## XPS IIC Interrupt Descriptions

The XPS IIC driver firmware has eight unique interrupts available to manage IIC data transfers. The interrupt signals generated by the XPS IIC's internal IIC control module are managed by the Interrupt Control (v2.01a) block. The registers within this block provide an interface containing many of the features commonly needed for interrupt handling.

### Interrupt(0) -- Arbitration Lost

Interrupt(0) is the Arbitration Lost interrupt. This interrupt is set when arbitration for the IIC bus is lost. The firmware must respond by first clearing the Control Register (CR) MSMS bit then second clearing this interrupt by writing a 1 to the Interrupt Status Register (ISR) INT(0) bit to toggle it. See also: Tx\_FIFO reset bit in the Control Register (CR).

### Interrupt(1) -- Transmit Error/Slave Transmit Complete

There are four possible events that cause this interrupt:

- XPS IIC operating as a Master Transmitter: Interrupt(1) implies an error. There are two possibilities: A) Either no slave was present at the transmitted address in which case the master transmitter recognizes a NOT ACKNOWLEDGE. B) The slave receiver issued a NOT ACKNOWLEDGE to signal that it is not accepting any more data. In either case the MSMS bit in the Control Register will transition from 1 to 0 causing the XPS IIC to initiate a stop condition which implies that the bus will not be busy.
- XPS IIC operating as a Master Receiver: Interrupt(1) implies a transmit complete. This interrupt is caused by setting Control Register (CR) TXAK high to indicate to the slave transmitter that the last byte has been transmitted.
- XPS IIC operating as a Slave Transmitter: Interrupt(1) implies a transmit complete. This interrupt is caused by the master device to indicate to the IIC that the last byte has been transmitted.
- XPS IIC operating as a Slave Receiver: Interrupt(1) implies an error. This interrupt is caused by the IIC (setting CR register field TXAK to 1).

The firmware must clear this interrupt by writing a 1 to the Interrupt Status Register (ISR), INT(1) bit to toggle it.

Note that this interrupt will occur before the INT(4) if INT(4) is also enabled. (The stop occurs later.)

### **Interrupt(2) -- Transmit FIFO Empty**

The controller raises (sets) the interrupt flag and keeps it raised while a transmit throttle condition exists. Once the flag has been raised and the transmit throttle condition is removed then (and only then can) the firmware lower (clear) the flag by writing a '1' to the Interrupt Status Register (ISR), INT(2) bit in order to toggle the flag state. See the [Throttle Description](#) section for information on actions that must be taken to clear a transmit throttle condition. The usual cause for a transmit throttle condition is the transmit FIFO going empty.

### **Interrupt(3) -- Receive FIFO Full**

This interrupt is set when the Receive FIFO Programmable Depth Interrupt Register (Rc\_FIFO\_PIRQ) is equal to the Receive FIFO Occupancy Register (Rc\_FIFO\_OCY). Clearing this interrupt requires that the Data Receive FIFO be read.

### **Interrupt(4) -- IIC Bus is Not Busy**

Interrupt(4) is set when the IIC bus is not busy. The condition remains set as long as the bus is not busy and cannot be cleared while the condition is true. Firmware must verify that the SR(BB) is asserted, indicating Bus Busy, before attempting to reset this interrupt bit.

A master that loses arbitration will want to get back on the bus (possibly) when it goes free. So it must immediately clear this bit in anticipation of that occurring.

If necessary, the slave should clear this bit after getting the AAS interrupt to know when the bus is not busy occurs. (A master could talk to several slaves before relinquishing the bus.)

### **Interrupt(5) -- Addressed As Slave**

This interrupt is set when the XPS IIC is being addressed as a slave.

### **Interrupt(6) -- Not Addressed As Slave**

This interrupt allows the detection of the end of receive data for a slave receiver when there has been no stop condition (repeated start). The interrupt occurs when a start condition followed by a non-matching slave address is detected. This interrupt is set when the IIC is not addressed as a slave.

### **Interrupt(7) -- Transmit FIFO Half Empty**

This interrupt is set while the MSB of the Tx\_FIFO\_OCY = 0.

## **Throttle Description**

The *Philips I<sup>2</sup>C-bus Specification* permits devices to throttle (suspend) data transmission on the bus by holding the SCL line low for an indefinite period of time. The XPS IIC controller uses this throttling mechanism to prevent either a receive overrun (Rx\_FIFO full) or a transmit underflow (Tx\_FIFO empty) by holding the SCL line low after the acknowledge bit has been sent.



Throttling is independent of Master or Slave operation and depends upon transmit or receive operation only. However initiation of stops and repeated starts may only be accomplished at times when throttling is occurring and only the XPS IIC acting as a Master may initiate such actions. When the XPS IIC is addressed as a Slave the throttling mechanism gives the firmware time to, either (A) gather data for transmission from the Tx\_FIFO, or (B) to find a place to store data received into the Rx\_FIFO.

Transmit throttling occurs when the transmit FIFO goes empty (except when a stop condition is pending.) When one or more bytes are written into the Tx\_FIFO, the transmit throttle condition is removed. The automatic throttling provides the firmware with time to handle the interrupt processing necessary for data transfer without manually having to manage the low level SCL signalling details. To prevent the throttle condition from re-appearing when the TX\_FIFO goes empty the firmware must setup the Master to issue a stop condition by resetting the CR(MSMS) bit *while the bus is throttled and prior* to writing the very last byte to be transmitted to the Tx\_FIFO.

To switch transmission to a new device while throttled a repeated start can be issued. The firmware does this by setting the Control Register (CR) RSTA bit *then* writing the device address into the Tx\_FIFO. The controller recognizes this sequence, issues the repeated start, retrieves the address byte from the Tx\_FIFO and outputs it onto the bus. If no more data was placed into the Tx\_FIFO the controller will immediately throttle again.

Receive throttling is done when the Receive FIFO Occupancy Register (Rc\_FIFO\_OCY) matches the value set in the Receive FIFO Programmable Depth Interrupt Register (Rc\_FIFO\_PIRQ). The throttle condition is removed (momentarily), when a byte from the Receive FIFO is read, thus allowing the transmitter to send the next byte. The slow rate of IIC transmissions should permit the firmware to completely empty the receive fifo prior to the receipt of the next byte but it is not necessary to do so.

When the XPS IIC is a master and a receive throttle condition exists, the XPS IIC generates a stop condition if the Control Register (CR) MSMS bit is changed from a 1 to a 0. This allows single byte reads from a slave device. The Control Register (CR) TXAK must be set equal to 1 to not-acknowledge the byte if desired.

When the XPS IIC is a master, is in a receive throttle condition and the transmit FIFO is empty, setting the repeated start in the Control Register will cause a transmit throttle condition to occur. That would raise the Interrupt(2) -- Transmit FIFO Empty flag.

## Standard Controller Logic Flow

The following is a brief discussion on setting the XPS IIC registers to initiate and complete bus transactions.

### IIC Master Transmitter, with a repeated start

1. Write the IIC device address to the Tx\_FIFO.
2. Write data to Tx\_FIFO.
3. Write to Control Register (CR) to set MSMS = 1 and TX = 1.
4. Continue writing data to Tx\_FIFO.
5. Wait for Transmit FIFO empty interrupt. This implies the IIC has throttled the bus.
6. Write to CR to set RSTA = 1.
7. Write IIC device address to Tx\_FIFO.
8. Write all data except last byte to Tx\_FIFO.

9. Wait for Transmit FIFO empty interrupt. This implies the IIC has throttled the bus.
10. Write to CR to set MSMS = 0. The IIC generates a stop condition at the end of the last byte.
11. Write last byte of data to Tx\_FIFO.

### IIC Master Receiver with a repeated start

1. Write the IIC peripheral device addresses for the first slave device to the Tx\_FIFO. Write the Receive FIFO Programmable Depth Interrupt Register (Rc\_FIFO\_PIRQ) to the total message length (call it M) minus two. It is assumed that the message is less than the maximum FIFO depth of 16 bytes.
2. Set Control Register (CR) MSMS = 1 and Control Register (CR) TX = 0.
3. Wait for the Receive FIFO interrupt indicating M-1 bytes have been received.
4. Set Control Register (CR) TXAK = 1. TXAK causes the XPS IIC controller to not-acknowledge the next byte received indicating to the slave transmitter that the master receiver will accept no further data. TXAK is set before reading data from the Rc\_FIFO, because as soon as a read from the Rc\_FIFO has occurred, the throttle condition is removed and the opportunity to set the bit is lost.
5. Read all M-1 data bytes from the Rc\_FIFO. Set the Rc\_FIFO\_PIRQ to 0 so that the last byte, soon to be received, causes the Receive FIFO full interrupt to be raised.
6. Clear the Receive FIFO full interrupt now because after a single byte is retrieved from the Rc\_FIFO the throttle condition is removed by the controller and the interrupt flag can be lowered (cleared).
7. Wait for the Receive FIFO full interrupt.
8. The controller will be throttled again with a full Rc\_FIFO. Set Control Register (CR) RSTA = 1. Write the peripheral IIC device address for a new (or same) IIC slave to the Tx\_FIFO.
9. Read the final byte of data (of the first message) from the Rc\_FIFO. This terminates the throttle condition so the Receive FIFO full interrupt can be cleared at this time. It also permits the controller to issue the IIC restart and transmit the new slave address available in the Tx\_FIFO. Also set the Rc\_FIFO\_PIRQ to be 2 less than the total 2nd message length (call it N) in anticipation of receiving the message of N-1 bytes from the second slave device.
10. Wait for the Receive FIFO full interrupt.
11. Set TXAK = 1. Write the Rc\_FIFO\_PIRQ to be 0, read the message from the Rc\_FIFO and clear the Receive FIFO full interrupt.
12. Wait for the Receive FIFO full interrupt (signalling the last byte is received).
13. Set MSMS = 0 in anticipation of giving up the bus via generation of an IIC Stop.
14. Read the final data byte of the second message from the Rc\_FIFO. This clears the throttle condition and makes way for the controller to issue the IIC Stop.

### IIC Slave Receiver

1. Set Control Register (CR) EN = 1 to enable the XPS IIC. If the IIC needs to recognize a general call then set EN = 1 and GC\_EN = 1.
2. Write the Slave address and R/ $\overline{W}$  bit to the Slave Address Register (ADR). In seven bit mode, a slave address of 0x7F should be written as 0xFE to the ADR. The 8<sup>th</sup> bit is the Read Not Write bit which is 0 in the case of a master transmit (IE Write) to slave. S0, 111 1111 0 = FE.
3. Write 0x0 to the Receive FIFO Programmable Depth Interrupt Register (Rc\_FIFO\_PIRQ) Compare Value. That causes an interrupt when 1 byte of data (not address) has been received. Because the address transmitted on the IIC bus is not stored in the receive FIFO, this interrupt will not be caused by receiving either a seven bit address or a ten bit address.
4. Wait for addressed as slave interrupt AAS.

5. Once an AAS interrupt has occurred, determine if the IIC slave is to receive or transmit data by reading Status Register bit 4
6. Clear not addressed as slave interrupt NAS.
7. If the IIC is a slave receiver, there are two basic choices for the slave receiver interrupt processing.
  - a. Set the Receive FIFO interrupt register to 0x0 and wait for either a Not Addressed as Slave NAS interrupt (no data was sent) or Rc\_FIFO\_PIRQ interrupt. In this mode, an interrupt occurs for every byte of data received plus a NAS for the end of the transmission.
  - b. Set the Receive FIFO interrupt register to 0xF and wait for either a Not Addressed as Slave NAS (some amount of data less than 16 bytes was set) or Rc\_FIFO\_PIRQ interrupt. In this mode if the Rc\_FIFO\_PIRQ interrupt occurs then 16 bytes of data exists in the FIFO to handle. Xilinx recommends that the software read the Receive FIFO occupancy register though not required. NAS may occur without Rc\_FIFO\_PIRQ interrupt. That means the Receive FIFO occupancy register should be read to indicate how many bytes of data must be handled. In this mode there is one Rc\_FIFO\_PIRQ interrupt for every 16 bytes of data plus a NAS for the end of the transmission. If less than 16 bytes of data is sent, NAS is the only interrupt.
8. In either choice above, clear the active interrupts after the data has been handled and then wait for the next interrupt.
9. Once the NAS interrupt has been received, handle the data and clear AAS.
10. Wait for the AAS interrupt.

## IIC Slave Transmitter

1. If the IIC is a slave transmitter, the following interrupt processing is available for use:
2. Ensure the Transmit Error/Slave Transmit Complete interrupt is cleared.
3. Once the IIC has been addressed as a slave transmitter the IIC will transmit the first byte of data in the Transmit FIFO. If no data exists in the transmit FIFO, the IIC will throttle the bus until data is written into the transmit FIFO.
4. If the protocol allows knowledge as to how much data the slave must transmit, fill up the FIFO and use the Transmit FIFO Empty or Transmit FIFO Half Empty interrupts to keep the transmit FIFO full. Wait for the Transmit Error/Slave Transmit Complete interrupt.
5. It is possible to write one byte of data at a time to the FIFO, then wait for a Transmit FIFO empty interrupt which means the master wants more data, or wait for the Transmit Error/Slave Transmit Complete interrupt which indicates that the master has received the required data.
6. When Transmit Error/Slave Transmit Complete has occurred, the NAS also occurs because the master has to either send a stop or send a repeated start.
7. When the NAS interrupt has been received, handle the data and clear AAS.
8. Wait for the AAS interrupt.

## Dynamic Controller Logic Flow

For initialization both the Receive FIFO (Rc\_FIFO) and Transmit FIFO (Tx\_FIFO) should be empty, and the XPS IIC should be enabled by setting Control Register (CR) EN = 1.

## Start/Repeated Start Sequence:

When sending bytes of data over the IIC bus, the Tx\_FIFO is filled first with the 7 bit device address of the IIC peripheral and the read/write bit, and any required data. In order to wake up the dynamic logic, the device address is written to the Transmit FIFO (Tx\_FIFO) as a 16 bit word (10 bits are used by the XPS IIC) with the start bit set (bit 23). Then if a read is to be performed, write the receive byte count to the Tx\_FIFO else put the data to be written to it. When the dynamic logic detects that data is available in the Tx\_FIFO and that the start bit is set, the XPS IIC will do the following:

1. Check the control register to see if MSMS is already set
  - a. If MSMS is not set, then set the MSMS bit to create a start sequence
  - b. If MSMS is already set, then set the RSTA in the control register to create a repeated start sequence
2. Transmit the 7 bit address and R/ $\overline{W}$  bit contained in the Tx\_FIFO
3. Check the least significant bit contained with the 7 bit address to determine if this is a read or write operation on the IIC bus
4. Get the next byte in the Tx\_FIFO
5. If a read access is occurring on the IIC bus, use this value as a receive byte counter. When this counter reaches zero, Control Register (CR) TXAK is forced High. This causes a not-acknowledge to be generated during reception of the last byte and will signal the IIC slave device to stop transmitting read data.
6. If a write access is occurring, the contents of the Tx\_FIFO are sent out on the SDA bus.

## Stop Sequence:

In order for the XPS IIC controller to release the IIC bus, by clearing the MSMS in the control register, bit 22 must be set in the Tx\_FIFO with the last byte to be sent for a write access. For a read access bit 22 must be set when the second word is written to the Tx\_FIFO. The least significant 8 bits of the second word contain the number of bytes to receive. If the stop bit is never set, the XPS IIC will continue to own the IIC bus.

## Pseudo Code for Dynamic IIC accesses:

It is recommended to verify that the Tx\_FIFO is not full or will not overflow with the writing of new data. For read accesses the user should reset the Rc\_FIFO or check that SR(Rc\_FIFO\_Empty)=1.

### Initialization

- Set Rc\_FIFO depth to maximum by setting Rc\_FIFO\_PIRQ=0x0F
- Set 7 bit address mode
- Reset Tx\_FIFO
- Enable XPS IIC, remove Tx\_FIFO reset, disable general call

### Read 4 bytes from an IIC device addressed as 0x34

- Check all FIFOs empty and bus not busy by reading status register
- Write 0x135 to Tx\_FIFO (set start bit, device address to 0x34, read access)
- Write 0x204 to Tx\_FIFO (set stop bit, 4 bytes to be received by the XPS IIC)
- Wait for Rc\_FIFO not empty
  - ◆ Read Rc\_FIFO byte
  - ◆ If 4<sup>th</sup> byte read, then exit otherwise continue checking Rc\_FIFO not empty

**Write 4 bytes (0x89, 0xAB, 0xCD, 0xEF) to an IIC slave EEPROM device addressed as 0x34**

Place the data at EEPROM address 0x33.

- Check all FIFOs empty and bus not busy by reading status register
- Write 0x134 to Tx\_FIFO (set start bit, device address, write access)
- Write 0x33 to Tx\_FIFO (EEPROM address for data)
- Write 0x89 to Tx\_FIFO (byte 1)
- Write 0xAB to Tx\_FIFO (byte 2)
- Write 0xCD to Tx\_FIFO (byte 3)
- Write 0xEF to Tx\_FIFO (stop bit, byte 4)

**Read 4 bytes from an IIC slave EEPROM device addressed as 0x34**

The data is at EEPROM address 0x33. First, a write access is necessary to set the EEPROM address, then a repeated start follows with the read accesses:

- Check all FIFOs empty and bus not busy by reading the status register
- Write 0x134 to Tx\_FIFO (set start bit, device address to 0x34, write access)
- Write 0x33 to Tx\_FIFO (EEPROM address for data)
- Write 0x135 to Tx\_FIFO (set start bit for repeated start, device address 0x34, read access)
- Write 0x204 to Tx\_FIFO (set stop bit, 4 bytes to be received by the XPS IIC)
- Wait for Rc\_FIFO not empty
  - ◆ Read Rc\_FIFO byte
  - ◆ If 4<sup>th</sup> byte is read, exit; otherwise, continue checking Rc\_FIFO not empty

**Timing Diagrams**

N/A

**Design Constraints**

N/A

**Design Implementation****Target Technology**

The target technology is an FPGA listed in the [Supported Device Family](#) field of the LogiCORE IP Facts Table on page 1.

## Device Utilization and Performance Benchmarks

Because the XPS IIC core will be used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When the XPS IIC core is combined with other designs in the system, the utilization of FPGA resources and timing of the XPS IIC design will vary from the results reported here.

The XPS IIC resource utilization for various parameter combinations measured with Virtex-5 as the target device are detailed in [Table 19](#).

**Table 19: Performance and Resource Utilization Benchmarks on the Virtex-5 FPGA (xc5vfx70-ff1136-1)**

Parameter Values				Device Resources			Performance
C_IIC_FREQ	C_TEN_BIT_ADR	C_GPO_WIDTH	C_SCL/SDA_INERTIAL_DELAY	Slices	Slice Flip-Flops	LUTs	F <sub>MAX</sub> (MHz)
100,000	0	1	0	142	305	406	157
400,000	0	1	0	270	308	396	154
400,000	1	1	5	151	317	403	157
400,000	0	1	5	147	311	393	155
400,000	0	8	5	265	618	400	155

The XPS IIC resource utilization for various parameter combinations measured with Virtex-4 as the target device are detailed in [Table 20](#).

**Table 20: Performance and Resource Utilization Benchmarks on the Virtex-4 FPGA (xc4vix25-ff668-10)**

Parameter Values				Device Resources			Performance
C_IIC_FREQ	C_TEN_BIT_ADR	C_GPO_WIDTH	C_SCL/SDA_INERTIAL_DELAY	Slices	Slice Flip-Flops	LUTs	F <sub>MAX</sub> (MHz)
100,000	0	1	0	476	306	540	132
400,000	0	1	5	467	309	525	140
400,000	1	1	5	475	318	558	140
400,000	0	1	5	467	309	525	141
400,000	0	8	5	409	316	543	142

The XPS IIC resource utilization for various parameter combinations measured with Spartan®--3 as the target device are detailed in Table 21.

**Table 21: Performance and Resource Utilization Benchmarks on the Spartan-3 FPGA (xc3sd1800a-fg676-4)**

Parameter Values				Device Resources			Performance
C_IIC_FREQ	C_TEN_BIT_ADR	C_GPO_WIDTH	C_SCL/SDA_INERTIAL_DELAY	Slices	Slice Flip-Flops	LUTs	F <sub>MAX</sub> (MHz)
100,000	0	1	0	390	306	471	100
400,000	0	1	5	440	309	464	100
400,000	1	1	5	484	318	485	102
400,000	0	1	5	440	309	464	100
400,000	0	8	5	443	316	468	100

The XPS IIC resource utilization for various parameter combinations measured with Virtex-6 as the target device are detailed in [Table 22](#).

**Table 22: Performance and Resource Utilization Benchmarks on the Virtex-6 FPGA (xc6vlx75t-ff784-1)**

Parameter Values				Device Resources			Performance
C_IIC_FREQ	C_TEN_BIT_ADR	C_GPO_WIDTH	C_SCL/SDA_INERTIAL_DELAY	Slices	Slice Flip-Flops	LUTs	F <sub>MAX</sub> (MHz)
100,000	0	1	0	185	305	462	160
400,000	0	1	0	186	303	439	160
400,000	1	1	5	195	312	467	160
400,000	0	1	5	186	303	439	160
400,000	0	8	5	205	310	450	160

The XPS IIC resource utilization for various parameter combinations measured with Spartan-6 as the target device are detailed in [Table 23](#).

**Table 23: Performance and Resource Utilization Benchmarks on the Spartan-6 FPGA (xc6slx16-csg324-2)**

Parameter Values				Device Resources			Performance
C_IIC_FREQ	C_TEN_BIT_ADR	C_GPO_WIDTH	C_SCL/SDA_INERTIAL_DELAY	Slices	Slice Flip-Flops	LUTs	F <sub>MAX</sub> (MHz)
100,000	0	1	0	193	305	450	100
400,000	0	1	0	182	303	429	100
400,000	1	1	5	222	312	445	100
400,000	0	1	5	182	303	429	100
400,000	0	8	5	198	310	425	100



Note: The generic parameter used in the utilization table are given below.

1. C\_CLK\_FREQ=100000000
2. C\_BASEADDR=X"00000000"
3. C\_HIGHADDR=X"000001FF"
4. C\_SPLB\_MID\_WIDTH=1
5. C\_SPLB\_NUM\_MASTERS=1
6. C\_SPLB\_AWIDTH=32
7. C\_SPLB\_DWIDTH=32

## System Performance

To measure the system performance ( $F_{MAX}$ ) of this core, this core was added to a Virtex-4 system, a Virtex-5 system, and a Spartan-3ADSP system as the Device Under Test (DUT) as shown in Figure 16, Figure 17, and Figure 18.

Because the XPS IIC core will be used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When the XPS IIC core is combined with other designs in the system, the utilization of FPGA resources and timing of the XPS IIC design will vary from the results reported here.

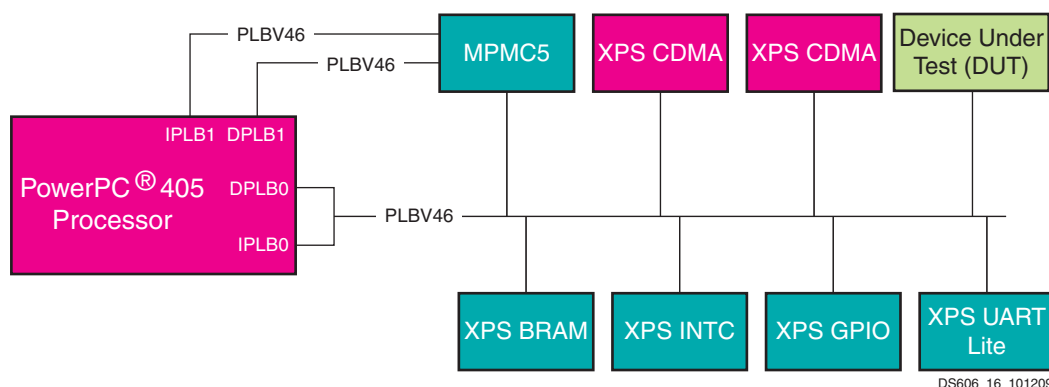


Figure 16: Virtex-4 FX FPGA System with the XPS IIC Device as the DUT

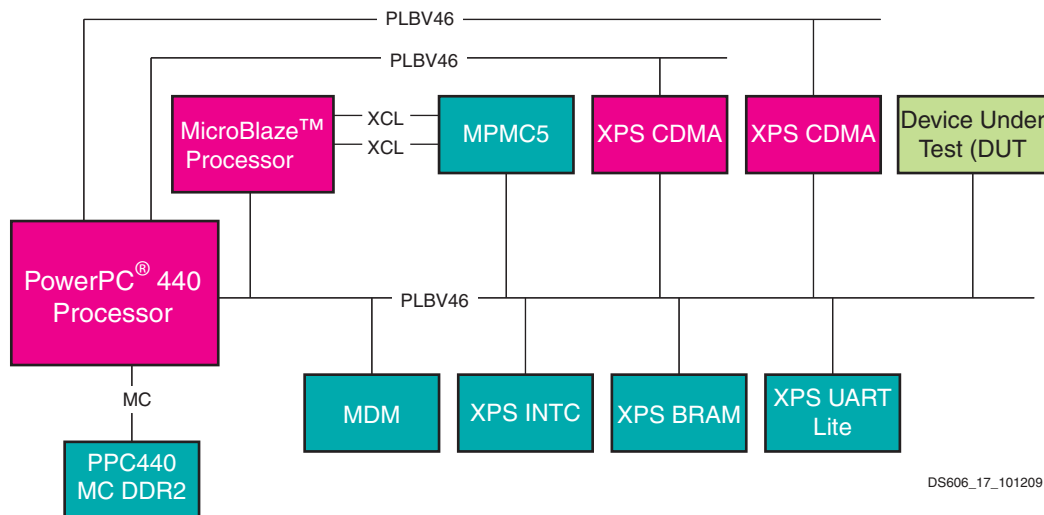


Figure 17: Virtex-5 FXT FPGA System with the XPS II Device as the DUT

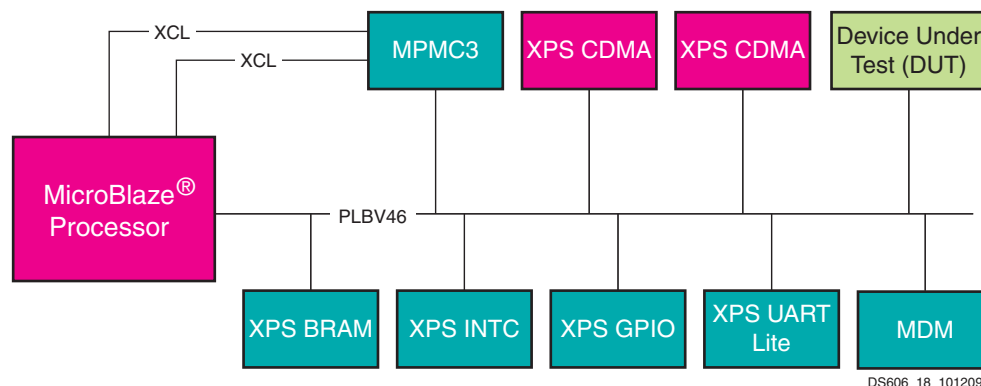


Figure 18: Spartan-3ADSP FPGA System with the XPS II Device as the DUT

The target FPGA was then filled with logic to drive the LUT and BRAM utilization to approximately 70% and the I/O utilization to approximately 80%. Using the default tool options and the slowest speed grade for the target FPGA, the resulting target  $F_{MAX}$  (frequency maximum) values are shown in Table 24.

Table 24: XPS IIC Controller Estimated System Performance

Target FPGA	Target $F_{MAX}$ (MHz)
S3ADSP3400-4	90
V4FX60-10	100
V5FXT70-1	120

The target  $F_{MAX}$  is influenced by the exact system and is provided for guidance. It is not a guaranteed value across all systems.

## XPS IIC Limitation

This core provides 0 ns SDA hold time in master mode operation as mentioned in Philips IIC bus specification. If any IIC slave requires additional hold time, this can be achieved by adding delay on SCL(C\_SCL\_INERTIAL\_DELAY). Please refer the description of this parameter.

The dynamic controller logic in XPS IIC controller supports master mode operation and 7-bit addressing only.

## Specification Exceptions

### Exceptions to the Philips IIC-bus specification version 2.1 January 2000

High-speed mode (Hs-mode) is not currently supported by the XPS IIC IP.

3-state buffers are used to perform the wired-AND function inherent in this bus structure.

The Xilinx FPGA device ratings must not be exceeded when inter-connecting the XPS IIC to other devices.

The  $t_{BUF}$  parameter ("Bus free time between a STOP and START condition", Table 5, Page 32 *Philips IIC-bus Specification*) is not met by the controller. The user must ensure the proper delay specification is met if the core is utilized with a bus device that needs the extra delay.

## Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

## Revision History

Date	Version	Description of Revisions
7/24/07	1.0	Initial Xilinx release.
10/1/2007	1.1	Added FMax Margin System Performance section.
12/3/2007	1.2	Updated to V2.00a core revision. Added information about the new C_SCL/SDA_INERTIAL_DELAY parameters used to determine the amount of pulse rejection filtering.
4/16/08	1.3	Added Automotive Spartan-3, Automotive Spartan-3E, Automotive Spartan-3A, and Automotive Spartan-3A DSP support.
7/22/08	1.4	Added QPro Virtex-4 Hi-Rel and QPro Virtex-4 Rad Tolerant FPGA support.
1/28/09	1.5	Updated to v2.01.a core revision. Removed Virtex2p support.
4/24/09	1.6	Replaced references to supported device families and tool name(s) with hyperlink to PDF file.
6/16/09	1.7	Updated performance and resource utilization table for Virtex-6 and Spartan-6 architecture.
12/2/09	1.8	Listed supported devices families in LogiCORE Table; updated images; converted to new DS template. Updated to v2.02.a core version.

## Notice of Disclaimer

Xilinx is providing this product documentation, hereinafter "Information," to you "AS IS" with no warranty of any kind, express or implied. Xilinx makes no representation that the Information, or any particular implementation thereof, is free from any claims of infringement. You are responsible for obtaining any rights you may require for any implementation based on the Information. All specifications are subject to change without notice. XILINX EXPRESSLY DISCLAIMS ANY WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE INFORMATION OR ANY IMPLEMENTATION BASED THEREON, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF INFRINGEMENT AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Except as stated herein, none of the Information may be copied, reproduced, distributed, republished, downloaded, displayed, posted, or transmitted in any form or by any means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx.