I^{2}C Hardened IP Usage Guide for Nexus Platform

Technical Note

FPGA-TN-02142-1.2

October 2020

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
FIFO	First In, First Out
FPGA	Field-Programmable Gate Array
I^{2}C	Inter-Integrated Circuit
I3C	Improved Inter-Integrated Circuit
IP	Intellectual Property
LMMI	Lattice Memory Mapped Interface
LSB	Least Significant Byte
LSR	Local Set/Reset
MSB	Most Significant Byte
NACK	No Acknowledge
Rx	Receive
SCL	Serial I^{2}C Clock Line
SDA	Serial I^{2}C Data Line
Tx	Transmit

1. Introduction

This reference guide provides guidance for the advanced usage of I^{2}C IP for devices built on the Lattice Nexus™ platform, which includes CrossLink™-NX and Certus™-NX FPGAs. The I^{2}C FIFO Hard IP provides an industry standard two-pin communication interface that conforms to Version 2.1 of the I^{2}C Bus Specification and is compliant to Version 1.0 of the MIPI I3C Specification for legacy I^{2}C devices. It can operate either as master port or as slave port.

- In master mode, it supports configurable data transfer rates and arbitration detection to allow it to operate in multi-master systems.
- Supports clock stretching with enable/disable capability.
- Supports both 7 bits and 10 bits addressing in slave mode with configurable slave address.
- Supports general call address detection in both master and slave modes.
- Provides interrupt logic for easy communication with the host.
- Provides configurable digital delay at SDA output for reliably generating start/stop condition.
- In FIFO mode, separate transmit and receive FIFOs can hold up to 32 bytes of data each to enable efficient communication and reduce the complexity of use.

This document includes the following:

- Top Level Description
- FIFO Mode Use Examples
- LMMI System Bus Protocol
- I^{2}C Timing Diagrams
- Port List
- Register Mapping

2. Top-level Description

The I^{2}C FIFO hard IP provides hardened I^{2}C functionality. It is controlled through memory-mapped registers, which interface with user logic through the Lattice Memory Mapped Interface (LMMI).

The hard IP has two main modes. Register mode requires you to control I^{2}C transactions at a low level, while FIFO mode allows you to control transactions at a higher level.

The Lattice Memory Mapped Interface (LMMI) on CrossLink-NX and Certus-NX provides connectivity between FPGA user logic and the hardened IP functional blocks. You must implement an LMMI master interface to interact with the interface in the hardened IP. The block diagram (Figure 2.1) shows the supported LMMI signals between the FPGA core and the hardened IP, as well as status signals from the hard IP. A detailed description of these signals is provided in the Port/Pin Description section.

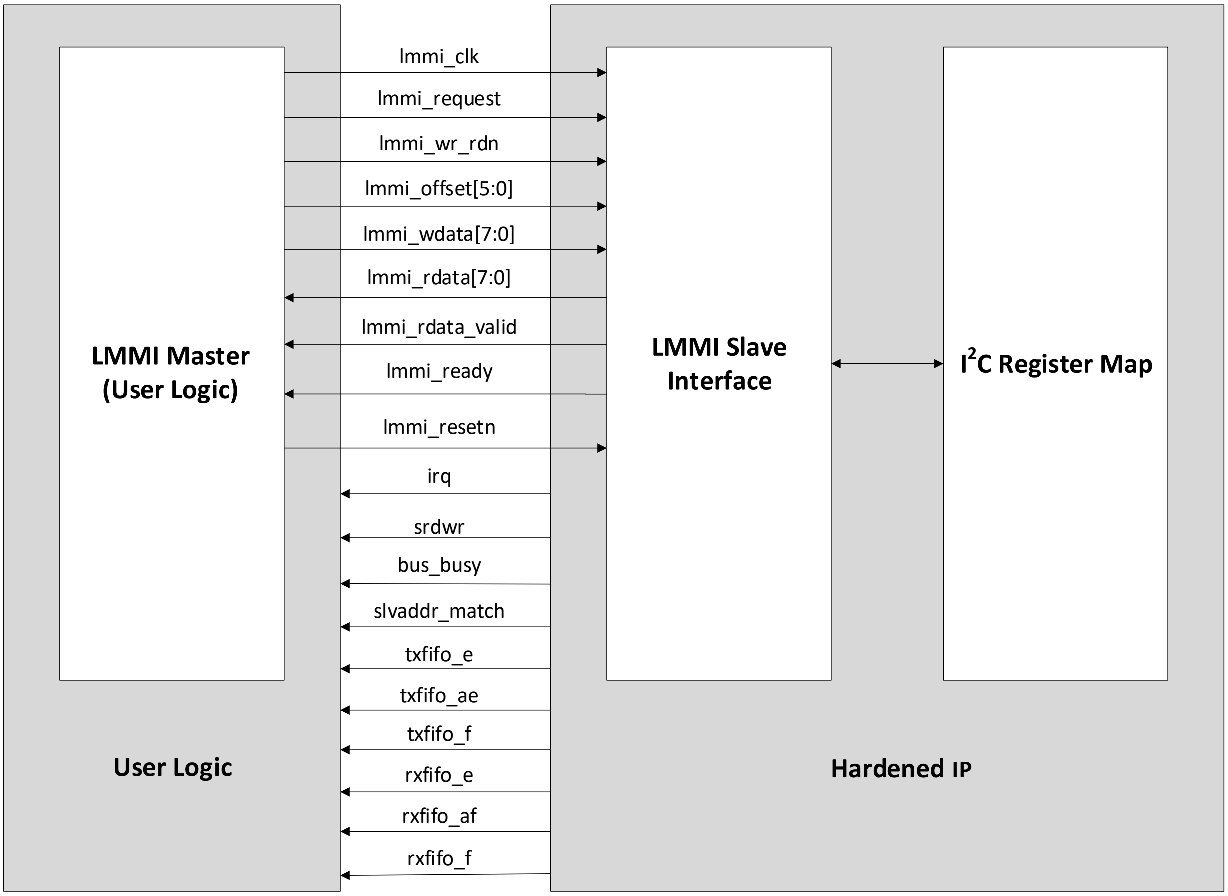
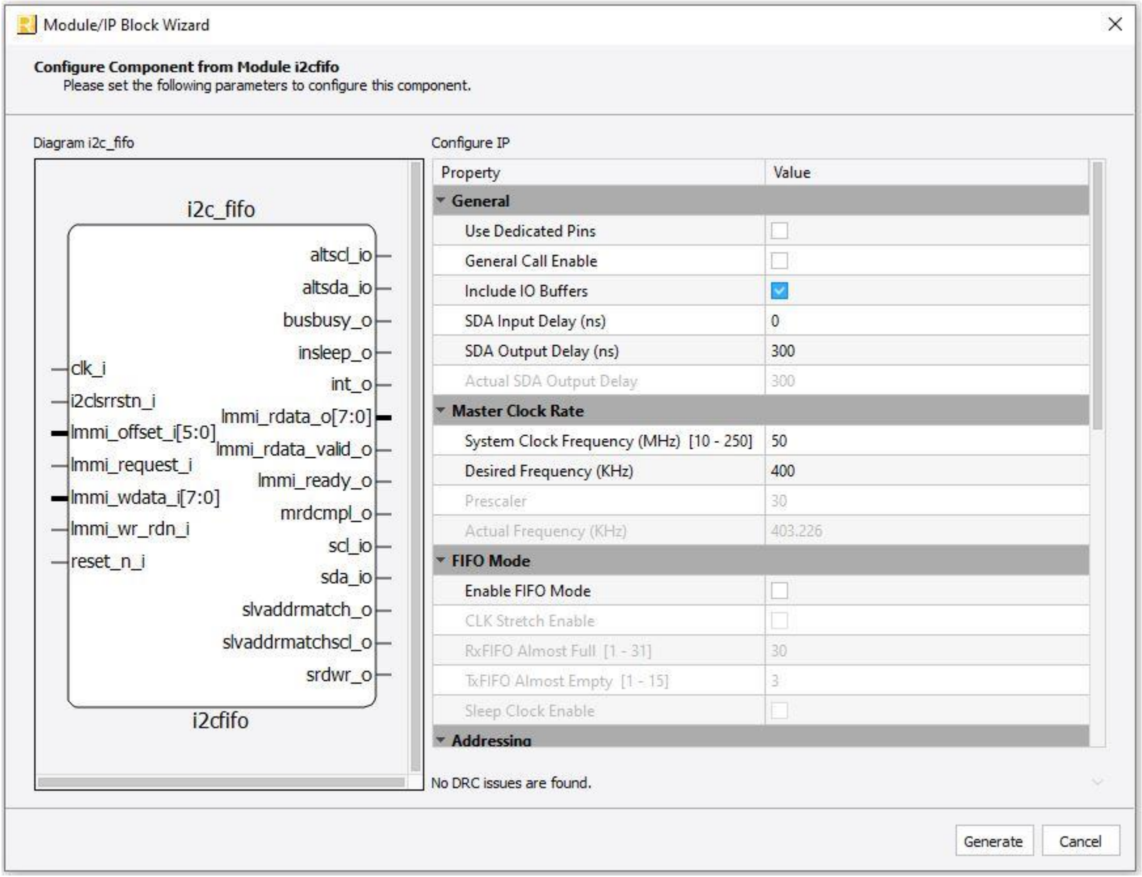


Figure 2.1. I^{2}C Hard IP Interface

3. Module Setup with IP Catalog

The IP Catalog is the recommended tool for using the I^{2}C Hard IP block. The IP Catalog flow simplifies the process of including IP in your design. It generates RTL wrapper files that correctly instantiate the IP primitives, as well as configuration and constraint files that automatically configure the module with your chosen settings.

The IP Catalog is accessed through the Lattice Radiant® software, in the main tool bar, or through the IP Catalog view of the main navigation pane. The I2CFIFO module is found in IP Catalog > IP on Local > Architecture_Modules. After clicking on the I2CFIFO module and choosing a location for the generated files to be stored and a module name, you are presented with the Module/IP Block Wizard, as shown in Figure 3.1.



Configure Component from Module i2cfifo
Please set the following parameters to configure this component.

Diagram i2c_fifo

Configure IP

Property	Value
General	
Use Dedicated Pins	□
General Call Enable	□
Include IO Buffers	☑
SDA Input Delay (ns)	0
SDA Output Delay (ns)	300
Actual SDA Output Delay	300
Master Clock Rate	
System Clock Frequency (MHz) [10 - 250]	50
Desired Frequency (KHz)	400
Prescaler	30
Actual Frequency (KHz)	403.226
FIFO Mode	
Enable FIFO Mode	□
CLK Stretch Enable	□
RxFIFO Almost Full [1 - 31]	30
TxFIFO Almost Empty [1 - 15]	3
Sleep Clock Enable	□
Addressing	

No DRC issues are found.

Figure 3.1. Module/IP Block Wizard

The setting options in the Module Wizard correspond to values in configuration registers (outlined in the Configurable I2C Registers section). The settings chosen in the Module Wizard are included in the bitstream generated during project compilation, and loaded into the appropriate registers during device configuration.

Following is an overview of the settings available in the Module Wizard.

General

- **Use Dedicated Pins** – Enables connecting the dedicated SCL and SDA pins to the hardened IP. If not selected, SCL and SDA must be assigned to general purpose I/O pins.
- **General Call Enable** – Enables the general call response when operating as a slave. See the I2C General Call Information Register (I2CGCD) section for the general call behavior.
- **Include I/O Buffers** – Includes I/O buffers in the RTL wrapper files. If this option is disabled, you must explicitly instantiate I/O buffers in fabric logic.
- **SDA Input Delay** – Enables to 50 ns SDA input delay.
- **SDA Output Delay** – Enables a range of SDA output delay options.

Master Clock Rate

- **System Clock Frequency** – Indicates the frequency of the clock provided through the LMMI system. The range of accepted input frequencies is 10 MHz – 250 MHz.
- **Desired Frequency** – The desired frequency of SCL when operating in master mode. The available frequency options are 40 KHz, 100 KHz, 400 KHz, and 1 MHz.
- **Prescaler** – Indicates the divider value used to derive the master mode SCL clock. This field is populated automatically and is not user editable.
- **Actual Frequency** – Shows the actual frequency to be used for master mode SCL. Note that it may not be possible to derive the desired frequency exactly from the specified input clock frequency. This field is populated automatically and is not user editable.

FIFO Mode

- **Enable FIFO Mode** – Enables the RX and TX FIFOs, allowing for higher level interaction with the hard IP. Enabling FIFO mode also adds several signals indicating the status of the RX and TX FIFOs.
- **CLK Stretch Enable** – Enables clock stretching. The module uses clock stretching in the following situations:
  - When there is an attempt to read from the module and the Transmitting Data Register is empty. Refer to the I2C FIFO Threshold Register (I2CFIFOTHRESHOLD) section for more details. If clock stretching is disabled, the module sends the value in the Transmitting Data Register.
  - When there is an attempt to write to the module, if FIFO mode is enabled and the Receiving Data Register is full. Refer to the I2C Command Register (I2CCMDR) section for more details. If clock stretching is disabled, the module issues a NACK, and does not store the byte.
- **RxFIFO Almost Full** – Sets the threshold at which the rxfifoaf_o signal is asserted. See the Port/Pin Description section.
- **TxFIFO Almost Empty** – Sets the threshold at which the txfifoae_o signal is asserted. See the Port/Pin Description section.

Addressing

- **I²C Addressing Mode** – Selects between 7-bit or 10-bit I²C addressing.
- **LSB Select** – Selects the source of the two least significant bits of the I²C slave address. If *Register* is selected, the address field provides the whole address. If *HW Tie* is selected, bits [1:0] of the slave address are set to *00*.
- **Address** – The user-definable portion of the I²C slave address. The range of valid values is determined by the settings chosen for I²C Addressing Mode and LSB Select.

Non-FIFO Interrupts

These settings gate interrupt signals which are relevant to operating in Register mode. These interrupts are only available in Register mode. These signals are detailed in the I2C General Call Information Register (I2CGCD) section.

FIFO Interrupts

These settings gate interrupt signals which are relevant to operating in FIFO mode. These interrupts are only available in FIFO mode. These signals are detailed in the I2C General Call Information Register (I2CGCD) section.

4. Functional Description

4.1. Functional Modes

The I²C hard IP supports two operation modes, Register mode and FIFO mode. Both register and FIFO modes implement a standard I²C interface.

In register mode, a host in the FPGA fabric controls all data transfers at a low level, issuing discrete commands to direct the transfer of each byte, and uses polling to determine the status of each transfer.

FIFO mode is an extension of the basic I²C controller. It allows the host to operate at a transactional level while the module controls the link layer activities. This *fire and forget* approach frees up the host to perform other tasks while the I²C controller is handling the data transaction.

4.2. FIFO Mode Data Transaction

In FIFO mode, the Tx and Rx FIFOs are 10 bits wide and 32 words deep. Interacting with each of the FIFOs is done through a pair of registers. The Least Significant Byte (LSB) of each 10 bit word holds data, while the Most Significant Byte (MSB) holds contextual information about the associated data byte in the LSB. MSB and LSB registers always correspond to the same 10-bit word in the FIFO.

The registers for the Tx FIFO are at offset 0x12 for the LSB and 0x13 for the MSB. Writes to the Tx FIFO should be done LSB first, then MSB. Writing to the MSB increments the FIFO's write pointer. The contents of TXFIFO_MSB[1:0] instructs the module how the contents of TXFIFO_LSB should be interpreted, as detailed in Table 4.1.

Table 4.1. TXFIFO Command Structure

TXFIFO_MSB[1:0]	Interpretation of TXFIFO_LSB
00	This is a data byte. If this is the last byte in the Tx FIFO, the module either begins clock stretching until another byte is provided, or underflows. This behavior is determined by the CKSDIS bit in the I2CC1 register.
01	This is a data byte. If this is the last byte in the Tx FIFO, this indicates the last byte to be transferred and a STOP is issued.
10	Bits [4:0] of this byte indicate the number of bytes to be received in master mode. The following data transaction is sent using a STOP then a START.
11	Bits [4:0] of this byte indicate the number of bytes to be received in master mode. The following data transaction is sent using a STOP then a RESTART.

The registers for the Rx FIFO are at offset 0x14 for the LSB and 0x15 for the MSB. Reads from the Rx FIFO should be done LSB first, then MSB. Reading from the MSB increments the FIFO's read pointer. If RXFIFO_MSB[0] is *1*, then the data in RXFIFO_LSB is the first byte received after a start or restart is detected, as detailed in Table 4.2.

Table 4.2. RXFIFO Structure

RXFIFO_MSB[0]	Interpretation of RXFIFO_LSB
0	This is a normal data byte.
1	This data byte is the first byte received after a Start or Restart is detected.

The following sections contain four examples of data transaction in FIFO mode. The status of transmitting FIFO and receiving FIFO are always available to the fabric for the host to monitor.

4.2.1. FIFO Mode I^{2}C Master Write (Master Transmitter) Example

This example demonstrates how to initiate a write on the I^{2}C bus in master mode. The host provides a starting command byte, indicating it wants to read 0 bytes, then a slave address, then data to be transmitted. In normal operation, a slave will ACK the master after receiving each byte. If a NACK is received, an interrupt is sent to the host. The host can decide to stop the transaction by clearing the FIFO, or allow the transaction to continue until all the data is transferred.

To initiate a transaction on the I^{2}C bus as a master, the host in the FPGA fabric must provide the following data, as shown in Table 4.3:

- Entry #1 — A start byte that consists of TXFIFO_MSB[1:0] = 2’b11 and TXFIFO_LSB[4:0] holding the number of bytes to be read from the slave
- Entry #2 — TXFIFO_LSB holding the slave address + W
- Entry #3 and #4 — Data bytes to be transferred
  The final byte is indicated by TXFIFO_MSB[1:0] = 2’b01.

If the number of bytes to be read is greater than zero, the host must write an additional start byte, then the slave address + R. This scenario is shown in the FIFO Mode I2C Master Read (Master Receiver) section.

Table 4.3. Master Write TXFIFO Contents

Tx FIFO Entry	TXFIFO MSB (Write Second)			TXFIFO LSB (Write First)
	Bits 7:2 Reserved	Bit 1 CMD	Bit 0 RSTA/LTXBYTE	Bits 7:0 Data
1	—	1	1	Number of bytes to be read (0 = no read)
2	—	0	0	Slave Address + W
3	—	0	0	Data 0
4	—	0	1	Data 1

Figure 4.1 shows the resulting communication on the I^{2}C bus. Data shown in white blocks is sent by the I^{2}C module. Data shown in grey blocks is sent by an external slave device.

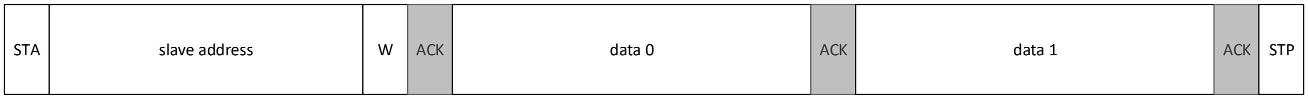


Figure 4.1. Master Mode Write – I^{2}C Bus Traffic

4.2.2. FIFO Mode I^{2}C Master Read (Master Receiver) Example

This example demonstrates how to initiate a read from a slave device. The host provides a starting command byte, indicating it wants to read 0 bytes, then a slave address + w, then a command byte to instruct the slave to provide data. The host then provides another start command, indicating that two bytes are to be read, then a slave address + r. When a restart is required, the host must provide a minimum of a command byte and a slave address byte, as shown in Table 4.4. The host can read the data as soon as the Rx FIFO is not empty. When the last byte of data is received, the host is informed by the mrdcmpl_o signal from the IP to the fabric, and the MRDCMPL bit in the I2CFIFOSR register is set. The resulting Rx FIFO contents is shown in Table 4.5. The resulting bus traffic is shown in Figure 4.2.

Table 4.4. Master Read TXFIFO Contents

Tx FIFO Entry	TXFIFO MSB (Write Second)			TXFIFO LSB (Write First)
	Bits 7:2 Reserved	Bit 1 CMD	Bit 0 RSTA/LTXBYTE	Bits 7:0 Data
1	—	1	1	Number of bytes to be read (0 = no read)
2	—	0	0	Slave Address + W
3	—	0	0	Read Command
4	—	1	1	Number of bytes to be read (2)
5	—	0	0	Slave Address + R

Table 4.5 Master Read RXFIFO Contents

Rx FIFO Entry	RXFIFO_MSB (Read Second)		RXFIFO_LSB (Read First)
	Bits 7:1 Reserved	Bit 0 DFIRST	Bits 7:0 Data
1	—	1	Data 0
2	—	0	Data 1

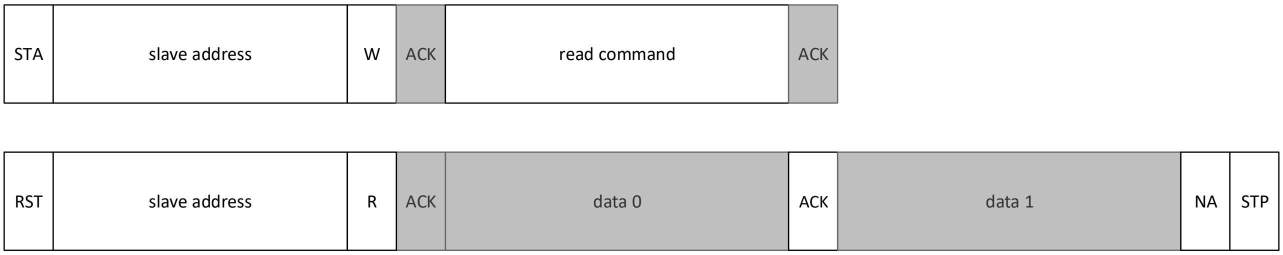


Figure 4.2. Master Mode Read – I^{2}C Bus Traffic

4.2.3. FIFO Mode I^{2}C Slave Write (Slave Receiver) Example

This example shows the result of a write from an external master. The I^{2}C module takes care of the link layer activities, and the host is informed that data is available in the Rx FIFO by the rxfifoe_o signal being de-asserted. If the Rx FIFO is full while the external master continues to send the data, the I^{2}C controller either issues a NACK or carries out clock stretching, depending on the CKSDIS bit is set or not. Table 4.6 shows the resulting data in the Rx FIFO. Figure 4.3 shows the resulting bus traffic.

Table 4.6.Slave Mode Write RXFIFO Contents

Rx FIFO Entry	RXFIFO_MSB (Read Second)		RXFIFO_LSB (Read First)
	Bits 7:1 Reserved	Bit 0 DFIRST	Bits 7:0 Data
1	—	1	Data 0
2	—	0	Data 1
3	—	0	Data 2
4	—	0	Data 3

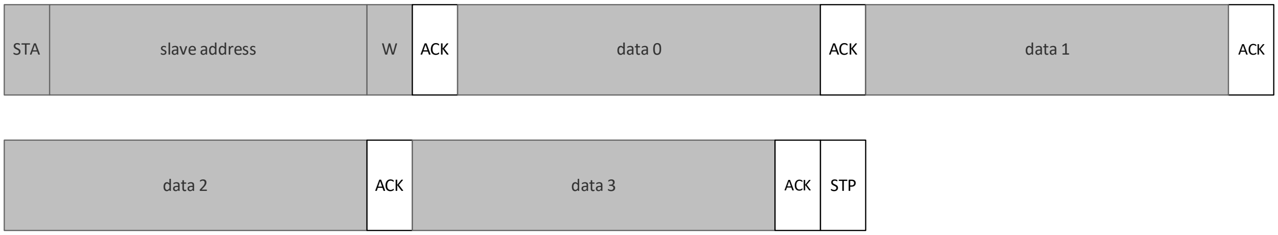


Figure 4.3. Slave Mode Write I^{2}C Data Diagram

4.2.4. FIFO Mode I2C Slave Read (Slave Transmitter)

Table 4.7 and Table 4.8 show an example of an external master reading from the hard IP operating as a slave device. The host is informed that there is a byte of data, which in this case is a read command. The host then places the requested data in the Tx FIFO. If there is no data in the TXFIFO, the I^{2}C control issues a NACK or carry out clock stretching depending on the CKSDIS bit is set or not. It is recommended the host clears the TXFIFO first before writing data into the FIFO. Note that since the external master is directing this transaction, TXFIFO_MSB[1:0] is 2'b00 for both bytes of data. Figure 4.4 shows the resulting bus traffic.

Table 4.7. Slave Mode Read RXFIFO Contents

Rx FIFO Entry	RXFIFO_MSB (Read Second)		RXFIFO_LSB (Read First)
	Bits 7:1 Reserved	Bit 0 DFIRST	Bits 7:0 Data
1	—	1	Read Command

Table 4.8. Slave Mode Read TXFIFO Contents

Tx FIFO Entry	TXFIFO MSB (Write Second)			TXFIFO LSB (Write First)
	Bits 7:2 Reserved	Bit 1 CMD	Bit 0 RSTA/LTXBYTE	Bits 7:0 Data
1	—	0	0	Data 0
2	—	0	0	Data 1

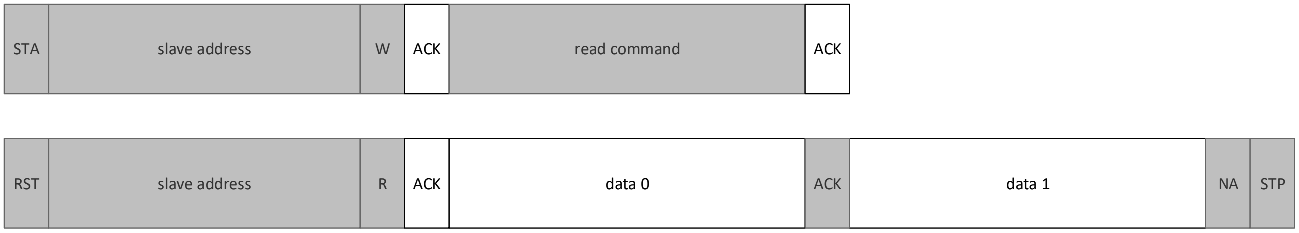


Figure 4.4. Slave Mode Read I^{2}C Data Diagram

4.3. FIFO Mode Access Flowchart

The flowcharts in this section (Figure 4.5 and Figure 4.6) are meant to demonstrate the recommended way of interacting with the hard IP in FIFO mode. It is assumed that the module has been set up through the IP Catalog tool. If IP was instantiated without the IP Catalog tool, the user logic must configure the registers that determine the I^{2}C controller behavior before it functions correctly. These registers include, but are not limited to, the control registers (I2CC1 and I2CC2) and the clock pre-scale register (I2CBR).

4.3.1. Slave Read/Write Example

This example shows the I^{2}C module operating as a slave, in a situation wherein an external master either writes or writes then reads. The general flow (Figure 4.5) is as follows:

- The host logic waits until the rxfifoe_o signal is deasserted. This indicates the presence of data in the Rx FIFO. Note that the host can wait until the slvaddrmatch_o signal goes low (this occurs after a stop or restart event), indicating that the write from the master has concluded. The host must monitor rxfifoaf_o in this scenario, to make sure the Rx FIFO doesn't overflow.
- The host should then read the data from the Rx FIFO. If the host started reading before the transaction ended, slvaddrmatch_o should be monitored to determine when the transaction has ended.
- If the data read from the Rx FIFO indicates a read by the external master, the host should reset the Tx FIFO to a known empty state, write the data to be read to the Tx FIFO, then return to the idle state.

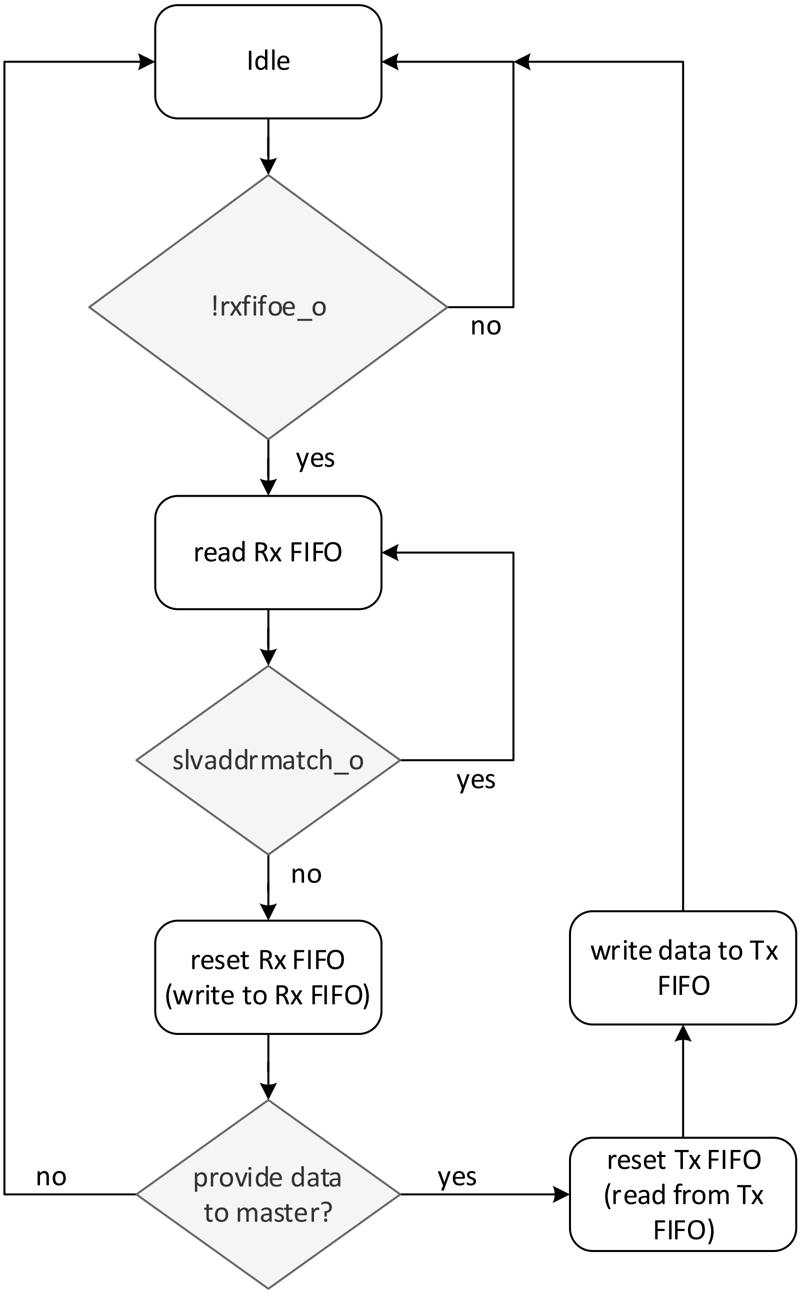


Figure 4.5. Slave Mode Read/Write Example

4.3.2. Master Read/Write Example

This example shows the module operating as a master, in a situation where it initiates all reads and writes. The general flow (Figure 4.6) is as follows:

- To initiate a write, the host logic should reset the Tx FIFO to a known good state, then write the data to be transferred to the Tx FIFO. See the FIFO Mode Data Transaction section for data format details.
- If the intended transaction is only a write, the host should return to the idle state. If the data includes a read command, the host should monitor the mrdcmpl_o signal to know when the requested number of bytes has been read into the Rx FIFO. The host can then read the data and reset the Rx FIFO to a known good state.

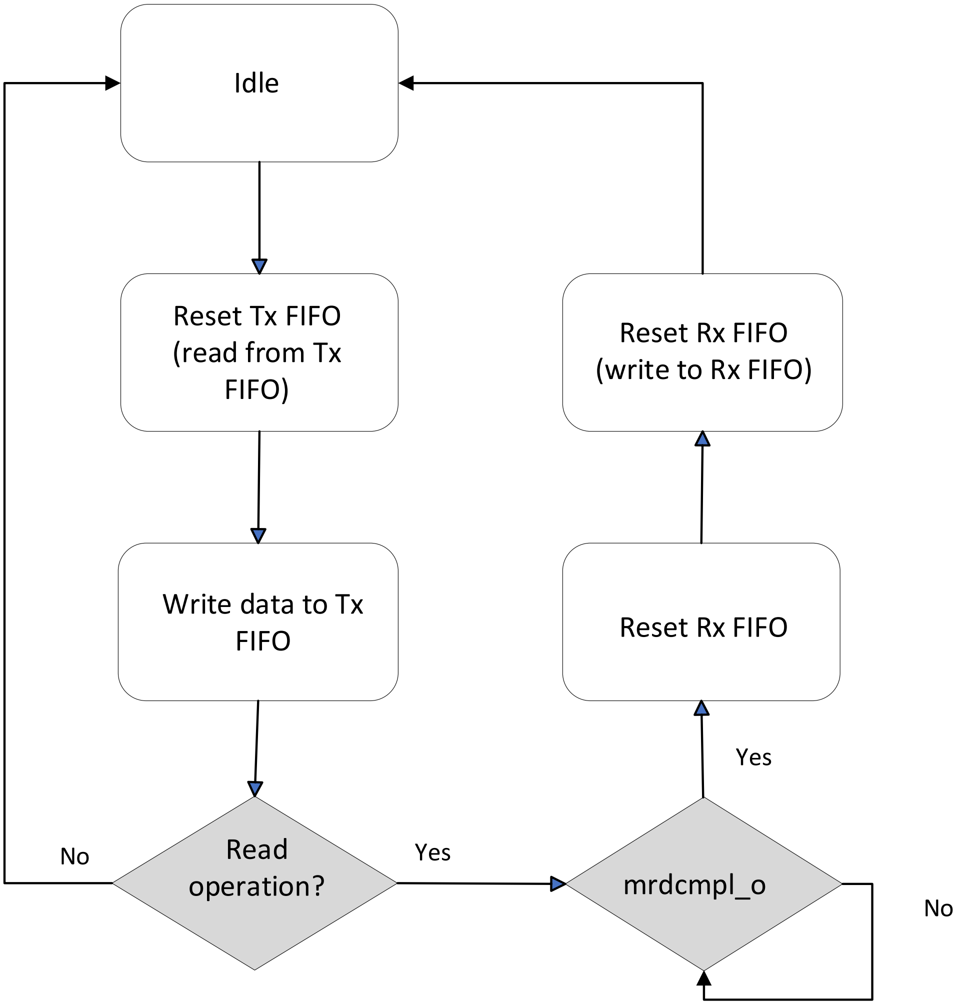


Figure 4.6. Master Mode Read/Write

4.4. Register Mode Functional Waveforms

This section contains several examples of how to operate the I^{2}C module in Register mode. To control the I^{2}C module, host logic sets bits in the I2CCMDR register. See the I2C Command Register (I2CCMDR) section for more details. Data is written to I2CTXDR and read from I2CRXDR. The I2CSR register contains several signals that indicate the state of the transaction.

4.4.1. Register Mode I^{2}C Master Write (Master Transmitter) Example

This section shows an example of the FIFO Mode I2C Master Write (Master Transmitter).

Figure 4.7 shows the timing for a write operation in master mode.

To initiate an I^{2}C write in Register mode, the host should first write the slave address + w to I2CTXDR, then set the *start* and *write* bits in I2CCMDR. The I^{2}C module sends the address, initiating the transaction. When the *trrdy* bit in the I2CSR register is set, the Tx data register (I2CTXDR) is ready to have a byte of data written to it. The host should write the data, then set the *write* bit in I2CCMDR. When the transaction is complete, the host should set the *stop* bit in I2CCMDR.

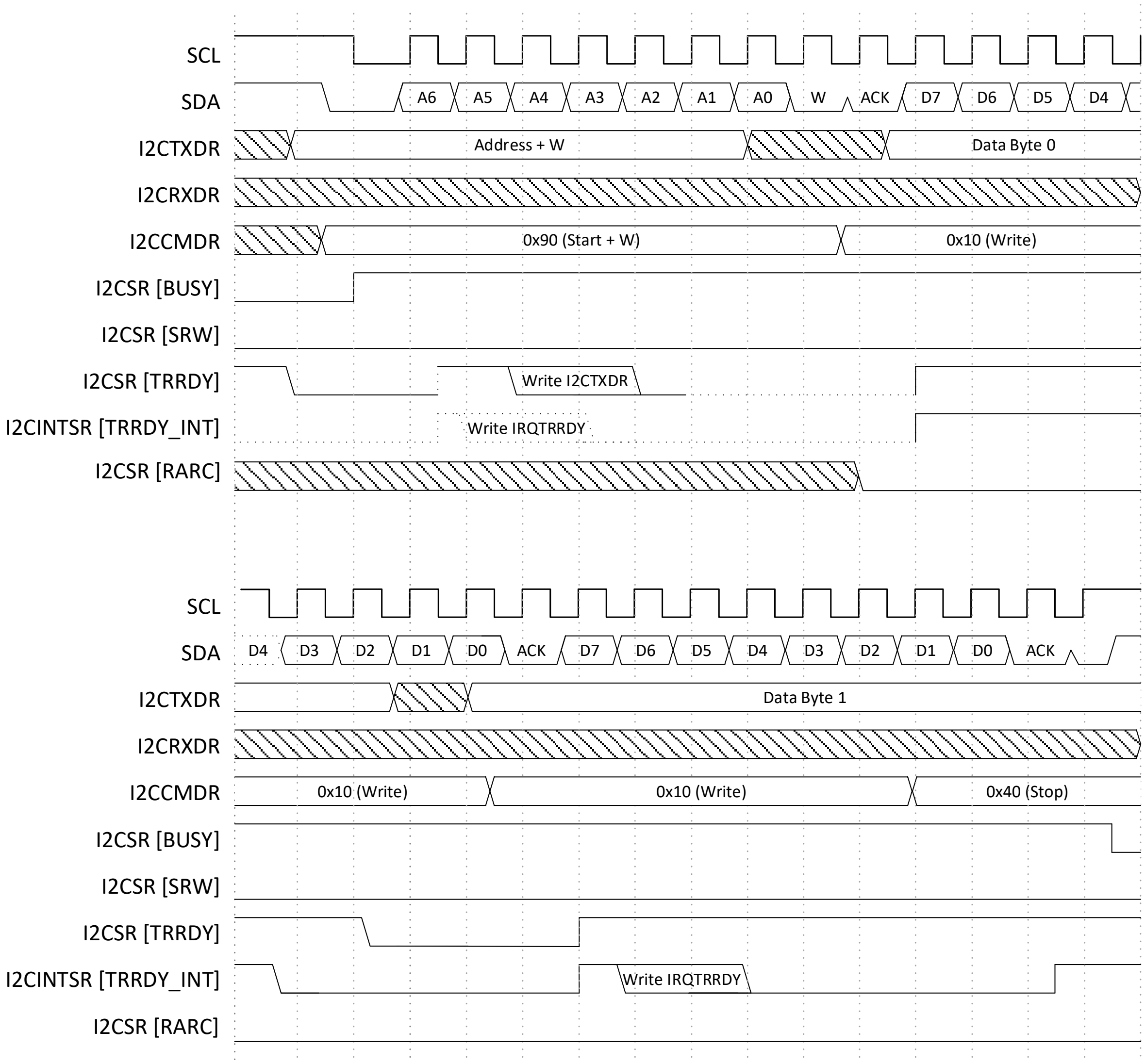


Figure 4.7. Master I^{2}C Write with Register Status Waveform

4.4.2. Register Mode I^{2}C Read Master (Master Receiver) Example

This section shows an example of the FIFO Mode I^{2}C Master Read (Master Receiver) Example section.

Figure 4.8 shows the timing for a read operation in master mode.

To initiate an I^{2}C read, the host should write the slave address + write to the I2CTXDR register, then set the *start* and *write* bits in I2CCMDR. This causes the I^{2}C controller to send the address and start reading from the external slave. When I2CSR[SRW] is high, I2CSR[TRRDY] indicated that there is data to be read in the I2CRXDR register. To stop reading, the host should set the *read*, *NACK*, and *stop* bits in I2CCMDR.

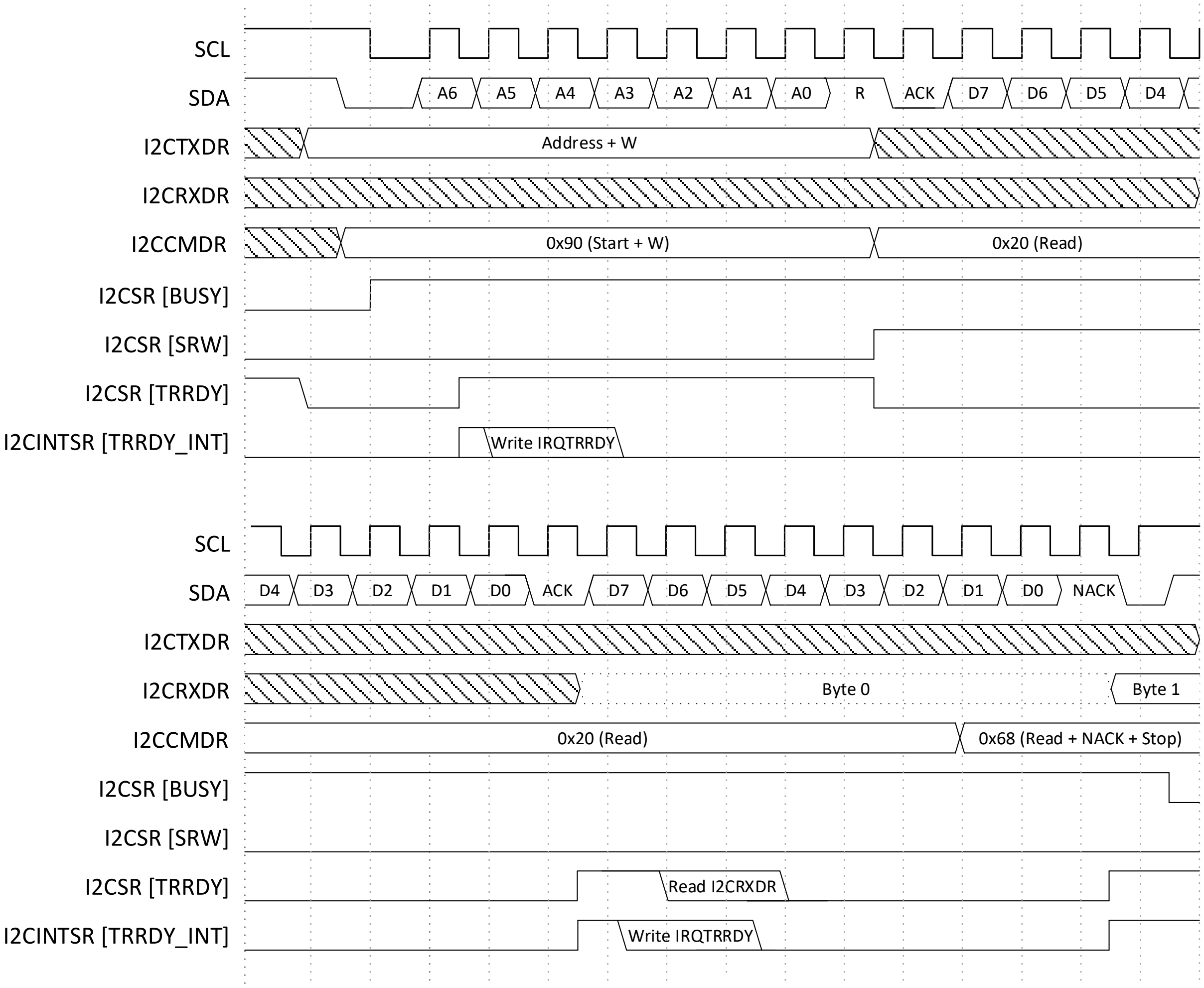


Figure 4.8. Master I^{2}C Read with Register Status Waveforms

4.4.3. Register Mode Slave I2C Write

This example (as shown in Figure 4.9) demonstrates the actions required when being written to by an external master. When the I2C module is operating in slave mode, the host logic should monitor the slvaddrmatch_o signal to be alerted to a write from an external master. Note that since the module is operating in slave mode, there is no need to write to the I2CCMDR register, unless the host wishes to NACK in response to a data byte. The I2CSR[TRRDY] signal shows when data is ready to be read in the I2CRXDR register.

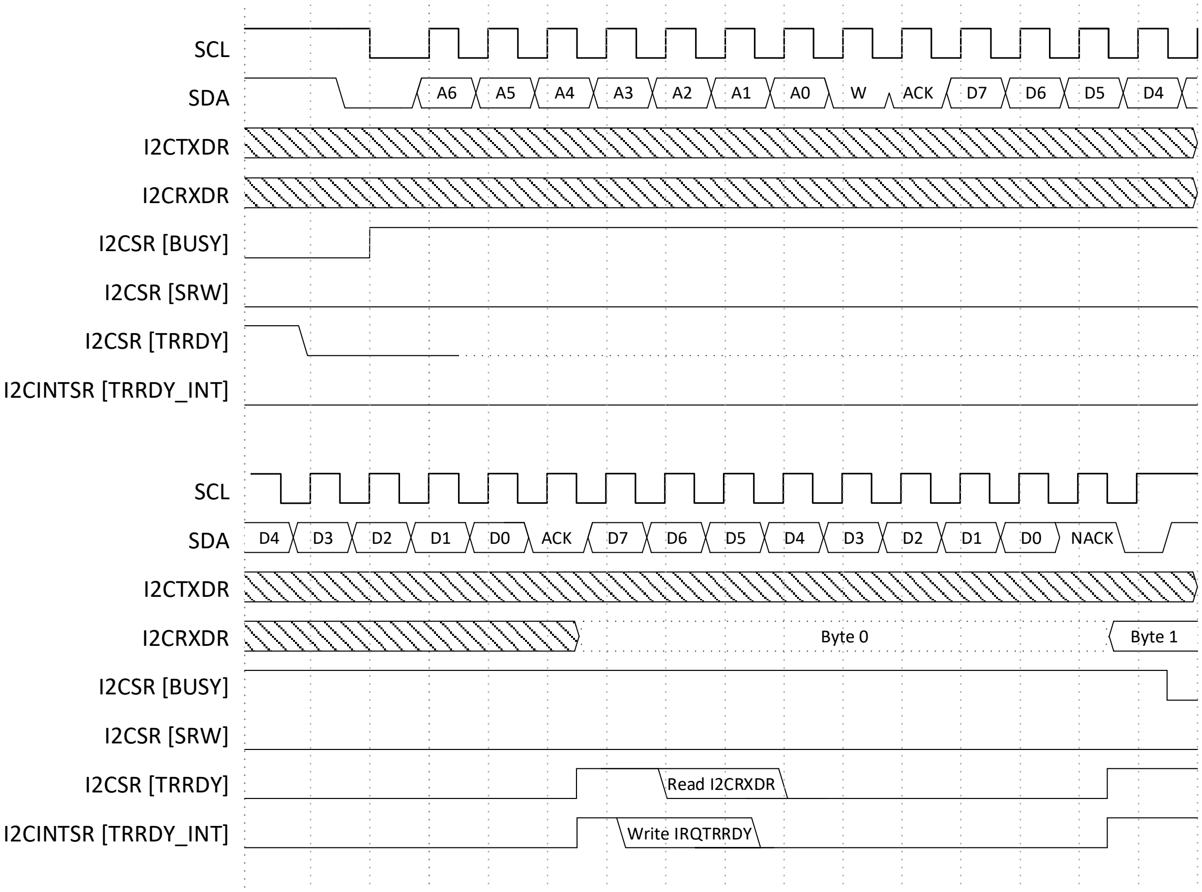


Figure 4.9. Slave I2C Write with Register Status Waveforms

4.4.4. Register Mode Slave I^{2}C Read

This example (as shown in Figure 4.10) demonstrates the action a host should take when being read from by an external master. The host must monitor slvaddrmatch_o to be alerted to being addressed by an external master. The I2CSR[SRW] signal going high while operating as a slave device indicates a read. The I2CSR[TRRDY] signal indicates when another byte of data can be written to I2CTXDR. The I2CSR[RARC] reflects the most recent ACK/NACK from the external master.

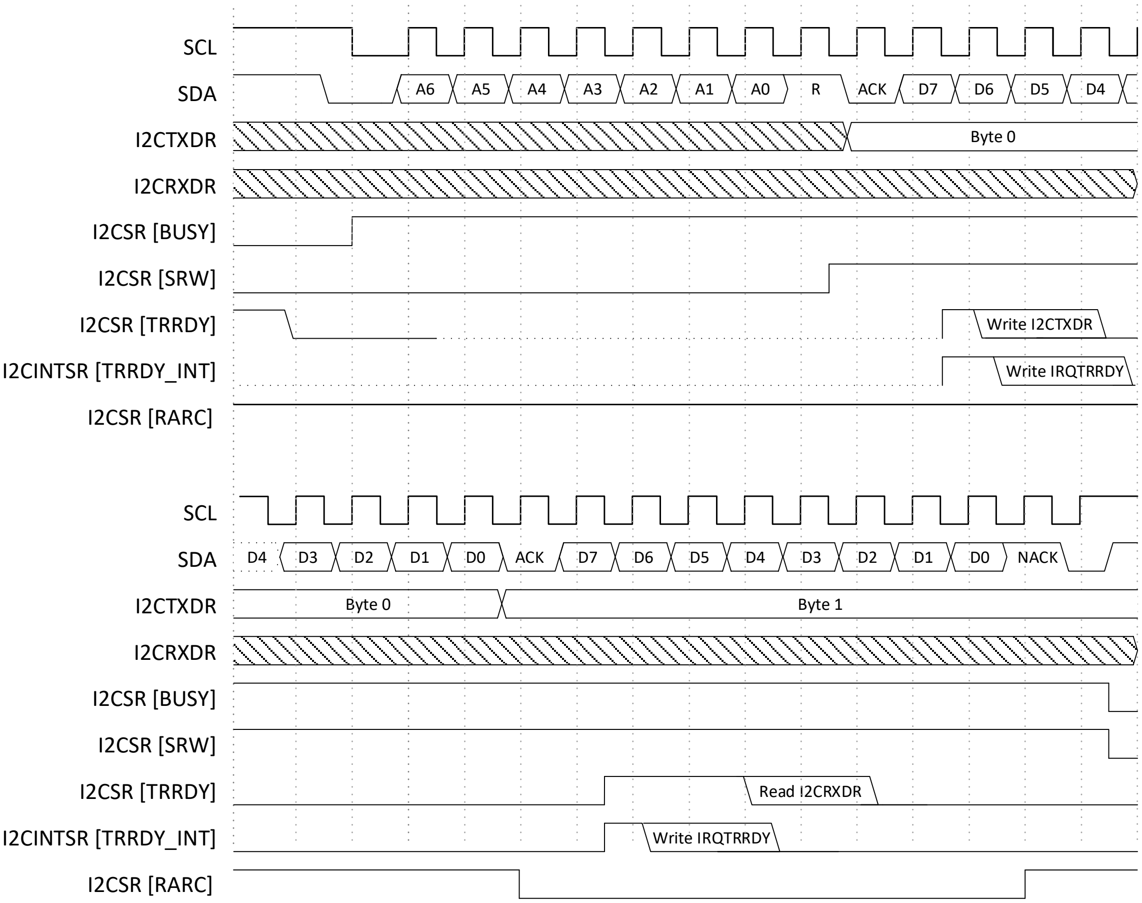


Figure 4.10. Slave I^{2}C Read with Register Status Waveform

4.5. Lattice Memory Mapped Interface (LMMI) Handshaking Protocols

The control and data interface of the I^{2}C Hard IP is done through the Lattice Memory Mapped Interface. LMMI is a simple memory-mapped address/data interface which supports both single and burst transactions with a maximum throughput of one transaction per clock cycle. LMMI supports optional wait states for slaves which need more than one clock cycle to complete a transaction.

Each hard IP presents its own set of LMMI signals. You have the flexibility to interface directly to each hard IP block, or to implement soft logic which ties multiple interfaces together into a shared bus.

4.5.1. LMMI Write Operation

Figure 4.11 demonstrates a single byte write operation:

- T0: Master decides to start a write transaction, asserts lmmi_request and lmmi_wr_rdn_i, and drives lmmi_offset_i and lmmi_wdata_i with values for the new transaction. Slave is ready to start a new a transaction in the next clock cycle and asserts lmmi_ready_o.
- T1: Master sees lmmi_ready_o high which signals that the slave has accepted the write transaction. After the appropriate hold time, Master deasserts lmmi_request and may change lmmi_wr_rdn_i, lmmi_offset_i and lmmi_wdata_i.
- T2: Slave signals that it is ready to start a new transaction by asserting lmmi_ready_o.

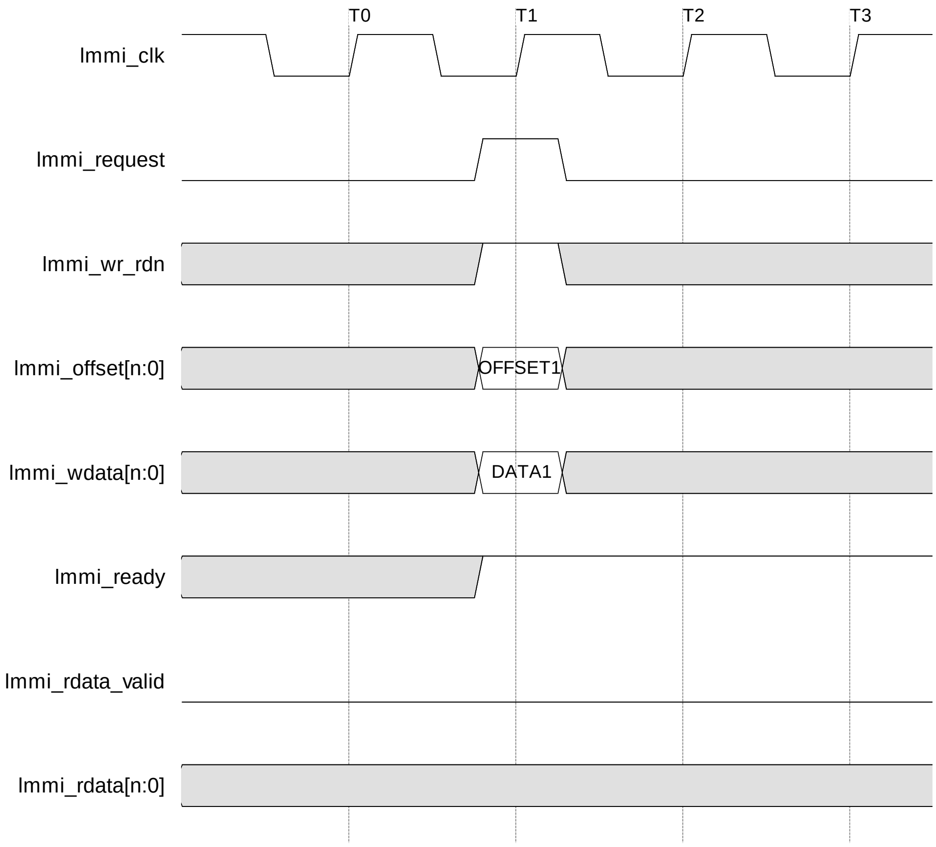


Figure 4.11. LMMI Write Operation Waveform

Figure 4.12 demonstrates a burst-write operation:

- T0: Master decides to start a write transaction, asserts lmmi_request and lmmi_wr_rdn_i, and drives lmmi_offset_i and lmmi_wdata_i with values for the new transaction. Slave is ready to start a new a transaction in the next clock cycle and asserts lmmi_ready_o.
- T1, T2: Master sees lmmi_ready_o high which signals that Slave has accepted the write transaction. After the appropriate hold time, Master changes lmmi_offset_i and lmmi_wdata_i to new values for the next transaction.
- T3: Master sees lmmi_ready_o high which signals that Slave has accepted the write transaction. After the appropriate hold time, Master deasserts lmmi_request and may change lmmi_wr_rdn_i, lmmi_offset_i and lmmi_wdata_i.
- T4: Slave signals that it is ready to start a new transaction by asserting lmmi_ready_o.

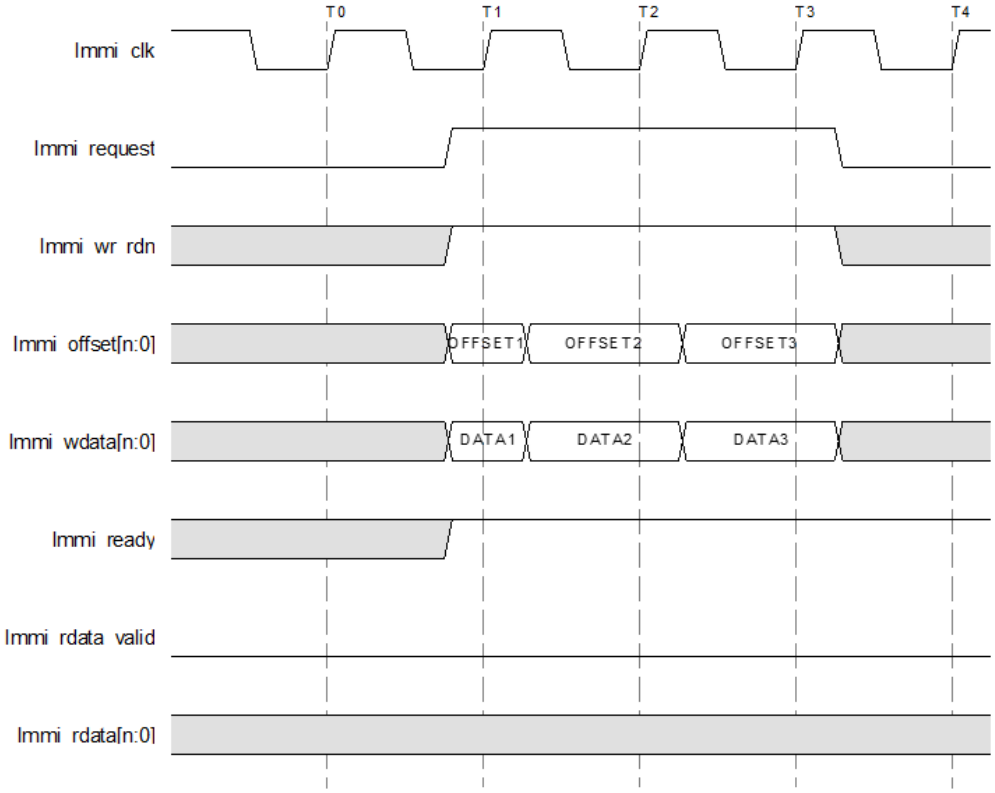


Figure 4.12. LMMI Burst Write Operation Waveform

4.5.2. LMMI Read Operation

Figure 4.13 demonstrates a single read operation:

- T0: Master decides to start a read transaction, asserts lmmi_request, deasserts lmmi_wr_rdn_i, and drives lmmi_offset_i with a value for the new transaction. Slave is ready to start a new a transaction in the next clock cycle and asserts lmmi_ready_o.
- T1: Master sees lmmi_ready_o high which signals that Slave has accepted the read transaction. After the appropriate hold time, Master deasserts lmmi_request and may change lmmi_wr_rdn_i and lmmi_offset_i.
- T2: Slave drives lmmi_rdata_o with the result of the read transaction, asserts lmmi_rdata_valid_o to signal that lmmi_rdata_o is valid, and asserts lmmi_ready_o to signal that Slave is ready to start a new transaction.

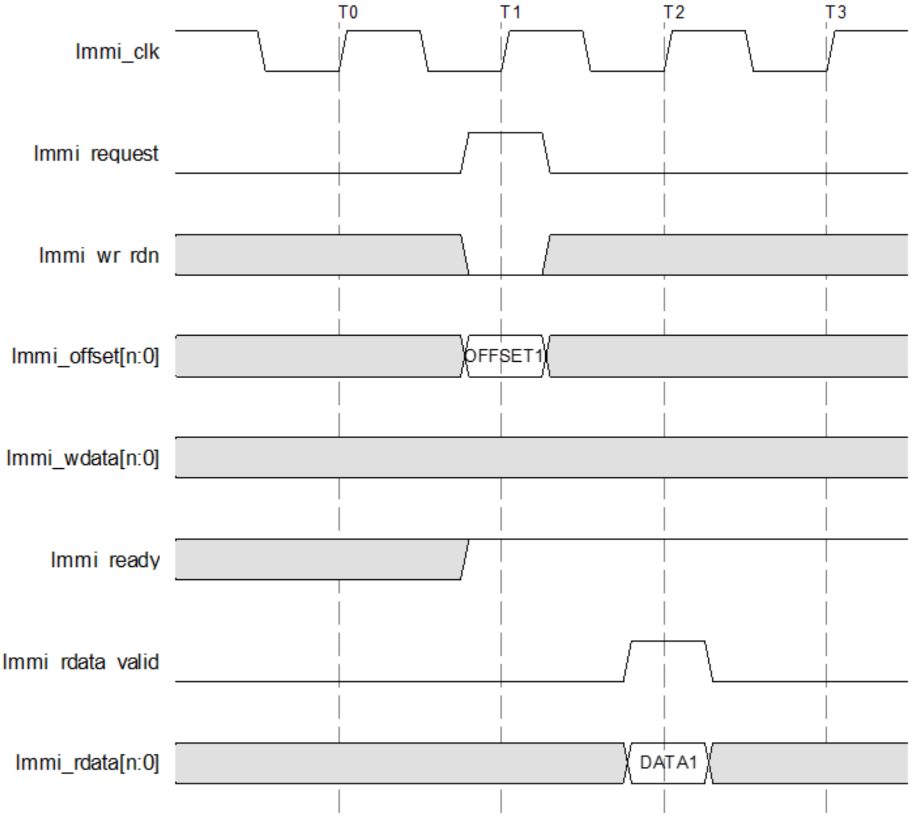


Figure 4.13. LMMI Read Operation Waveform

Figure 4.14 describes a burst read operation:

- T0: Master decides to start a read transaction, asserts lmmi_request, deasserts lmmi_wr_rdn_i, and drives lmmi_offset_i with a value for the new transaction. Slave is ready to start a new a transaction in the next clock cycle and asserts lmmi_ready_o.
- T1: Master sees lmmi_ready_o high which signals that Slave has accepted the read transaction. After the appropriate hold time, Master changes lmmi_offset_i to a new value for the next read transaction.
- T2: Slave drives lmmi_rdata_o with the result of the read transaction, asserts lmmi_rdata_valid_o to signal that lmmi_rdata_o is valid, and asserts lmmi_ready_o to signal that Slave is ready to start a new transaction. Master latches lmmi_rdata_o. After the appropriate hold time, Master deasserts lmmi_request and may change lmmi_wr_rdn_i and lmmi_offset_i.
- T3: Slave drives lmmi_rdata_o with the result of the read transaction, asserts lmmi_rdata_valid_o to signal that lmmi_rdata_o is valid, and asserts lmmi_ready_o to signal that Slave is ready to start a new transaction.

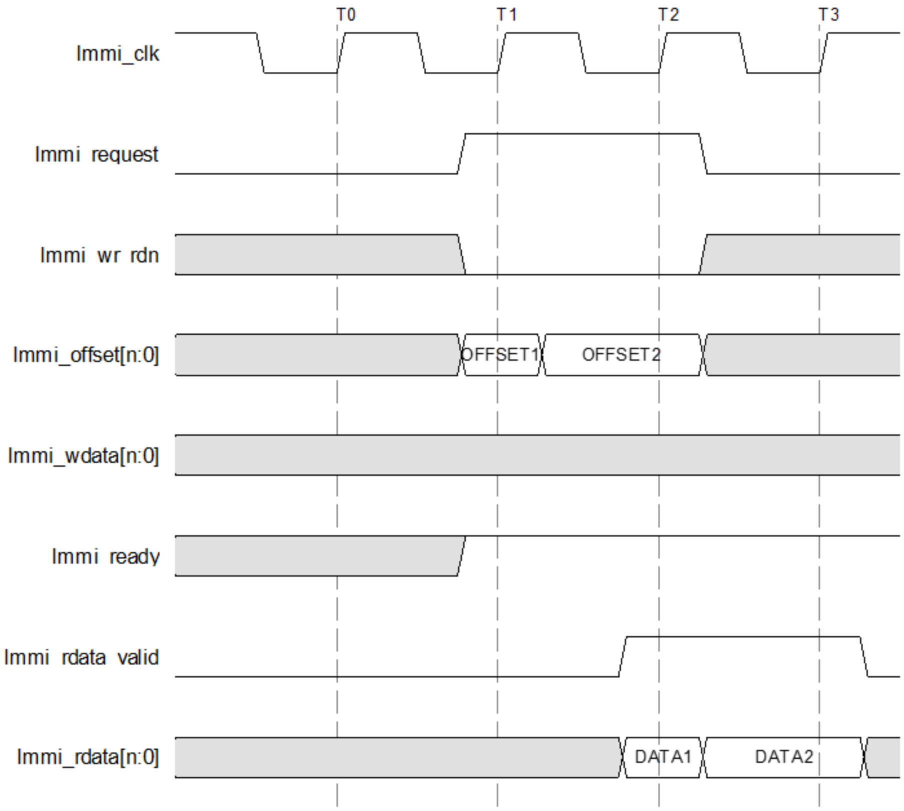


Figure 4.14. LMMI Burst Read Operation Waveform

5. Port/Pin Description

To interface with the IP, you must create an LMMI master controller in the fabric of the FPGA. Refer to the Lattice Memory Mapped Interface (LMMI) Handshaking Protocols section for more details regarding the LMMI standard.

Table 5.1. I^{2}C Hard IP Interface Signals

Signal Name	I/O	Width	Description
clk_i	Input	1	The clock input *clk_i* coordinates all activities for the internal logic within the Lattice Memory-Mapped Interface. This clock is also the derivative for Master Mode SCL and SDA output delay calculation.
lmmi_request_i	Input	1	The strobe input *lmmi_request_i* when asserted, indicates that the LMMI Slave component is selected. An LMMI Slave responds to other LMMI signals only when the *lmmi_request_i* is asserted
lmmi_wr_rdn_i	Input	1	The write enable input *wr_rdn* indicates whether or not the current LMMI cycle is a READ or WRITE cycle. The signal is negated during READ cycles, and is asserted during WRITE cycles.
lmmi_offset_i	Input	6	The data input array *lmmi_offset_i* is used to pass a binary address for LMMI Addressable Registers.
lmmi_wdata_i	Input	8	The data input array *lmmi_wdata_i* is used to write register and FIFO data and commands.
lmmi_rdata_o	Output	8	The data output array *lmmi_rdata_o* is used to read back register and FIFO binary data and status.
lmmi_ready_o	Output	1	Indicates if the IP is ready to have a transaction on a register/block.
lmmi_rdata_valid_o	Output	1	Indicates if the data on *lmmi_rdata_o* is valid.
lmmi_resetn_o	Input	1	Resets LMMI state machine to prevent lockup if wait states are implemented.
i2clsrrstn_i	Input	1	This active-low reset only resets the I^{2}C core and does not reset configuration registers. When asserted, the internal FSM in the I^{2}C core is reset to idle state, all status flag is cleared, and the I^{2}C bus is released. The assertion of this signal stops the on-going I^{2}C transaction and may cause an incomplete transaction on the I^{2}C bus.
fiforeset_i	Input	1	This active-high reset signal, when asserted, synchronously resets the FIFOs (I2CTXFIFO and I2CRXFIFO) transmit/receive byte counters (I2CFIFOTXCNT and I2CFIFORXCNT). The associated FIFO flags are cleared as well. The assertion of this signal does not affect the on-going I^{2}C transaction, which continues until the byte transaction is completed.
scl_i	Input	1	Serial clock input from fabric
scl_o	Output	1	Serial clock output to fabric
scloen_o	Output	1	Serial clock output enable to fabric
scloenn_o	Output	1	Serial clock output enable bar to fabric
alt_scl_in	Input	1	Serial clock input from hard I/O connection, if available
alt_scl_out	Output	1	Serial clock output to hard I/O connection, if available
alt_scl_oe	Output	1	Serial clock output enable to hard I/O connection, if available
alt_scl_oe_n	Output	1	Serial clock output enable bar to hard I/O connection, if available
sda_i	Input	1	Serial data input from fabric
sda_o	Output	1	Serial data output to fabric
sdaoen_o	Output	1	Serial data output enable to fabric
sdaoenn_o	Output	1	Serial data output enable bar to fabric
alt_sda_in	Input	1	Serial data input from hard I/O connections, if available
alt_sda_o	Output	1	Serial data output to hard I/O connections, if available
alt_sda_oe	Output	1	Serial data output enable to hard I/O connections, if available
alt_sdaoen_o_n	Output	1	Serial data output enable bar to hard I/O connections, if available
int_o	Output	1	The interrupt request output to fabric system host, when asserted, indicating there are pending requests coming from Interrupt register waiting to react. System host can read the I2CS/I2CFIFOS to find out the details of the interrupt request.

Signal Name	I/O	Width	Description
mrdcmpl_o	Output	1	The Master Read Complete signal. A transaction is considered complete when: • the specified number of data bytes from the slave have been received in the RX FIFO, or • the Master terminates the read transaction before the specified number of data bytes received.
srdwr_o	Output	1	Slave read/write signal. A *1* indicates a slave transmitting (external master receiving). A *0* means slave receiving (external master transmitting).
busbusy_o	Output	1	Indicates that there is communication on the I^{2}C bus. This bit is triggered even when this I^{2}C is not involved in the communication.
slvaddrmatch_o	Output	1	Indicated that the I^{2}C Slave Address is matched for this communication. Resets on Restart and Stop.
slvaddrmatchscl_o	Output	1	An active high output signal to fabric that indicates that the slave address of this I^{2}C IP has been called (matching slave address) through the I^{2}C bus by another I^{2}C master.
txfifoae_o	Output	1	TXFIFO almost empty status signal, indicating user-defined almost empty threshold value is reached.
txfifoe_o	Output	1	TXFIFO empty status signal
txfifof_o	Output	1	TXFIFO full status symbol
rxfifoaf_o	Output	1	RXFIFO almost full status signal, indicating user-defined almost full threshold.
rxfifoe_o	Output	1	RXFIFO empty status signal. Can be used as an active-low FIFO Data Ready signal.
rxfifof_o	Output	1	RXFIFO full status signal

6. Configurable I²C Registers

The I²C hard IP communicates with the Lattice Memory Mapped Interface through a set of control, command, status, and data registers.

Table 6.1 shows the register names and their functions.

Note that several registers, such as I2CSLVADDR, are split across two bytes, and accessing these registers requires two read/write operations. Two such registers, I2CTXFIFO and I2CRXFIFO, should be accessed in the order LSB then MSB, as accessing the MSB of either of the FIFOs causes a read/write pointer in that FIFO to increment.

Table 6.1. Configurable I²C Registers

Offset (Hex)	Name	Description	Support Modes (Register/ FIFO)	Read/ Write
00	—	Reserved[1]	—	—
02	I2CC1	I²C control register 1	Both	R/W
04	I2CC2	I²C control register 2	Both	R/W
06	I2CBRLSB	I²C clock prescale register, LSB	Both	R/W
08	I2CBRMSB	I²C clock prescale register, MSB	Both	R/W
0A	I2CSLVADDR_LSB	I²C slave address register, LSB	Both	R/W
0B	I2CSLVADDR_MSB	I²C slave address register, MSB	Both	R/W
0C	I2CINTENABLE/I2CFIFOINTENABLE_LSB	I²C register mode interrupt control register/ I²C FIFO mode interrupt control register, LSB	Both	R/W
0D	I2CINTENABLE/I2CFIFOINTENABLE_MSB	I²C register mode interrupt control register/ I²C FIFO mode interrupt control register, MSB	Both	R/W
0E	I2CFIFOTHRESHOLD_LSB	I²C FIFO threshold register, LSB	FIFO mode	R/W
0F	I2CFIFOTHRESHOLD_MSB	I²C FIFO threshold register, MSB	FIFO mode	R/W
10	I2CCMDR	I²C command register	Register mode	R/W
12	I2CTXDR/I2CTXFIFO_LSB	I²C transmitting data register/ I²C transmit data FIFO, LSB	Both	W[2]
13	I2CTXFIFO_MSB	I²C transmitting data register/ I²C transmit data FIFO, MSB	FIFO mode	W[2]
14	I2CRXDR/I2CRXFIFO_LSB	I²C receiving data register/ I²C receive Data FIFO, LSB	Both	R[3]
15	I2CRXFIFO_MSB	I²C receiving data register/ I²C receive data FIFO, MSB	Both	R[3]
16	I2CGCDR	I²C general call information register	Both	R
18	I2CSR/I2CFIFOSR_LSB	I²C register mode status register/ I²C FIFO mode status register, LSB	Both	R
19	I2CSR/I2CFIFOSR_MSB	I²C register mode status register/ I²C FIFO mode status register, MSB	Both	R
1A	I2CINTSR/I2CFIFOINTSR_LSB	I²C register mode interrupt status register/ I²C FIFO mode interrupt status register, LSB	Both	R
1B	I2CINTSR/I2CFIFOINTSR_MSB	I²C register mode interrupt status register/ I²C FIFO mode interrupt status register, MSB	Both	R
1C	—	Reserved[1]	—	—
1D	—	Reserved[1]	—	—
1E	I2CFIFOTXCNT_LSB	I²C TXFIFO byte counter, LSB	FIFO mode	R
1F	I2CFIFOTXCNT_MSB	I²C TXFIFO byte counter, MSB	FIFO mode	R
20	I2CFIFORXCNT_LSB	I²C RXFIFO byte counter, LSB	FIFO mode	R
21	I2CFIFORXCNT_MSB	I²C RXFIFO byte counter, MSB	FIFO mode	R
22	I2CINTFR/I2CFIFOINTSET_LSB	I²C register mode interrupt set register/ I²C FIFO mode interrupt set register, LSB	Both	R/W

Offset (Hex)	Name	Description	Support Modes (Register/ FIFO)	Read/ Write
23	I2CINTFR/I2CFIFOINTSET_MSB	I^{2}C register mode interrupt set register/ I^{2}C FIFO mode interrupt set register, MSB	Both	R/W
24	—	Reserved[1]	—	—
25	I2CSTSP_TIMER_DELAY_LSB	I^{2}C start/repeated start timer register, LSB	Both	R/W
26	I2CSTSP_TIMER_DELAY_MSB	I^{2}C start/repeated start timer register, MSB	Both	R/W
27	I2CTXFIFOBYTEAVAIL	TXFIFO bytes available to be read	FIFO mode	R

Notes:

1. Register offset marked *Reserved* should not be written to, and reads from these registers are undefined.
2. Reading from I2CTXDR/I2CTXFIFO causes that register/FIFO to reset.
3. Writing to I2CRXDR/I2CRXFIFO causes that register/FIFO to reset.

6.1. I^{2}C Control Register 1 (I2CC1)

This register (Table 6.2) contains general control options. It is recommended that the settings in this register be set by using the IP Catalog tool in Lattice Radiant software. A write to this register causes the I^{2}C core to reset.

Table 6.2. I2CC1 Register Details

0x02 – I2CC1				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	I2CEN	R/W	I^{2}C System Enable Bit – This bit enables the I^{2}C core functions. If I2CEN is cleared, I^{2}C is disabled and forced into idle state, status bits in I2CSR register are reset except for I2CSR_MSB[4] if bus is free at the time of disable. I2CFIFO must be reset prior to re-enabling to resume proper function. 0: I^{2}C disabled 1: I^{2}C enabled	0
6	GCEN	R/W	Generic call Response Enable Bit – Enable the generic call response in Slave Mode. 0: Disabled 1: Enabled	0
5	RSVD	—	Reserved	—
4	FIFO_MODE	R/W	I^{2}C Mode Select – Choose between using FIFO mode or Register mode 0: Register mode (default) 1: FIFO mode	0
3:2	SDA_DEL_SEL	R/W	SDA Output Delay Selection – These two bits select the output delay (in Number of clock cycles. The Base Delay is set by MSB of the I2CBR_MSB) 2’b00 : NDelay = 0 2’b01 : NDelay = 1 * Nbase_delay + 3 ; (when Nbase_delay = 0, NDelay = 1) 2’b10 : NDelay = 2 * Nbase_delay + 3 ; (when Nbase_delay = 0, NDelay = 1) 2’b11 : NDelay = 4 * Nbase_delay + 3 ; (when Nbase_delay = 0, NDelay = 1)	00
1	CKSDIS/RSVD	—	FIFO Mode clock stretching disable option. This bit is used in FIFO mode only (refer to I2CCMD[2] for register mode) — Disable the clock stretching if you desire for both master and slave mode. Overflow/underflow error flags must be monitored if disabled. 0: Clock stretching is enabled 1: Clock stretching is disabled	0
0	RSVD	—	Reserved	—

6.2. I²C Control Register 2 (I2CC2)

I2CC2 (Table 6.3) should be set to default value during software setup. It also can be read or written through LMMI.

Table 6.3. I2CC2 Register Details

0x04 – I2CC2				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	INTCLREN	R/W	Auto Interrupt Clear Enable – Enable the interrupt flag auto clear when the interrupt status registers (I2CINTSTATUS/I2CFIFOINTSTATUS) are read. 0: Disabled 1: Enabled	0
6	SLVADDR_HARD_TIE	R/W	Slave Address Hard Tie Disable – Disables use of '00' as the lower 2 bits of the I²C slave address and allows full register to be slave address. 0: Address using 1: Address using I2CSLVADDR value only	0
5	I2C_CORE_RSTN	R/W	I²C Core Reset – This is an active low reset that resets the I²C core.	1
4:3	RSVD	—	Reserved	—
2:0	RSVD	—	Reserved	—

6.3. I²C Clock Pre-Scale Register (I2CBR)

This pair of registers (Table 6.4 and Table 6.5) contains the clock divider value used to divide a fabric clock signal down to the desired SCL frequency in master mode. This value should be set through the IP Catalog tool in Lattice Radiant software. A write operation through LMMI to either I2CBR_LSB or I2CBR_MSB causes the I²C core reset.

The I2CBR_MSB [7:4] is utilized for trimming the base delay (NBASEDELAY) which is combined with I2CC1 [3:2] to achieve the SDA output delay to meet the I²C Specification requirement (300ns).

Table 6.4. I2CBRLSB Register Details

0x06 – I2CBR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	I2CBR[7:0]	R/W	Clock Divider FSCL = FSOURCE / (4 * (I2CBR[9:0] + 1))	00000000

Table 6.5. I2CBRMSB Register Details

0x08 – I2CBR_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:4	NBASEDELAY	R/W	Nbase_delay[3:0]	0000
3:2	RSVD	—	Reserved	—
1:0	I2CBR[9:8]	R/W	Clock Divider	00

6.4. I^{2}C Slave Address (I2CSLVADDR)

This pair of offsets (0x0A and 0x0B) is shared by both register and FIFO mode. The value in this register should be set up through the IP Catalog tool of Lattice Radiant software. Any change to this register during operation causes the I^{2}C core to reset.

Depending on the setting in I2CC2[6] (SLVADDR_HARD_TIE), the address in this register is either the full address (as shown in Table 6.8) or the upper bits of the slave address (as shown in Table 6.6 and Table 6.7). If SLVADDR_HARD_TIE is set, bits [1:0] of the slave address are set to 00.

I2CSLVADDR_MSB (offset 0x0B) is reserved except when 10-bit addressing is enabled and Hard Tied mode is disabled, as shown in Table 6.9 and Table 6.10.

If the I2CSLVADDR register in Hard Tied mode is ZERO (8'H00 for 7-bit addressing, or 10'H000 for 10-bit addressing), then a default value (8'HF8 for 7-bit addressing or 10'H3E3 for 10-bit addressing) is taken in order to avoid conflict with reserved addresses of I^{2}C specification.

Table 6.6. I2CSLVADDR_LSB Register with 7 Bit Addressing and Hard Tied LSB

0x0A – I2CSLVADDR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	RSVD	—	Reserved	—
4:0	7BITADDRMSB	R/W	7-bit I^{2}C slave address bits [6:2]	00000

Table 6.7. I2CSLVADDR_LSB Register with 10 Bit Addressing and Hard Tied LSB

0x0A – I2CSLVADDR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	10BITADDRMSB	R/W	10-bit I^{2}C slave address bits [9:2]	00000000

Table 6.8. I2CSLVADDR_LSB Register with 7 Bit Addressing and without Hard Tied LSB

0x0A – I2CSLVADDR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	RSVD	—	Reserved	—
6:0	7BITADDR	R/W	7-bit I^{2}C slave address	0000000

Table 6.9. I2CSLVADDR_LSB Register with 10 Bit Addressing and without Hard Tied LSB

0x0A – I2CSLVADDR_SLB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	10BITADDR	R/W	10-bit I^{2}C slave address bits [7:0]	00000000

Table 6.10. I2CSLVADDR_MSB Register with 10 Bit Addressing and without Hard Tied LSB

0x0B – I2CSLVADDR_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:2	RSVD	—	Reserved	—
1:0	10BITADDR	R/W	10-bit I^{2}C slave address bits [9:8]	00

6.5. I²C Interrupt Control Register (I2CINTENABLE)/I²C FIFO Interrupt Control Register (I2CFIFOINTENABLE)

This set of offsets (0x0C and 0x0D) is shared by both register and FIFO mode. Setting bits in this register enable an interrupt tied to the signal in the corresponding bit in I2CSR/I2CFIFOSR. For instance, enabling I2CINTENABLE[3] – TRRDY_EN enables an interrupt tied to I2CSR[3] – TRRDY.

Table 6.11. I2CINTENABLE_LSB Register Details (Register Mode)

0x0C – I2CINTENABLE_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:4	RSVD	—	Reserved	—
3	TRRDY_EN	R/W	Transmit/Receive register ready interrupt enable – Enable TRRDY interrupt 0: Disabled 1: Enabled	0
2	TROE_EN	R/W	Transmit/Receive Register Overrun Interrupt Enable - Enable TROE interrupt 0: Disabled 1: Enabled	0
1	ARBL_EN	R/W	Arbitration Lost Interrupt Enable - Enable arbitration Lost Interrupt 0: Disabled 1: Enabled	0
0	HGC_EN	R/W	General Call Interrupt Enable - Enable General Call interrupt 0: Disabled 1: Enabled	0

Table 6.12. I2CINTENABLE_MSB Register Details (Register Mode)

0x0D – I2CINTENABLE_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	RSVD	—	Reserved	—
4	BUS_FREE_EN	R	Enable BUS_FREE interrupt 0: Disabled 1: Enabled	0
3:0	RSVD	—	Reserved	—

Table 6.13. I2CFIFOINTENABLE_LSB Register Details (FIFO Mode)

0x0C – I2CFIFOINTENABLE_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	TXSERR_EN	R/W	TX FIFO synchronization error interrupt enable 0: Disabled 1: Enabled	0
6	RSVD	—	Reserved	—
5	RXOVERF_EN	R/W	RXFIFO overflow interrupt enable 0: Disabled 1: Enabled	0
4	RNACK_EN	R/W	Receive NACK Interrupt Enable 0: Disabled 1: Enabled	0
3	RSVD	—	Reserved	—
2	RSVD	—	Reserved	—

0x0C – I2CFIFOINTENABLE_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
1	ARBL_EN	R/W	Arbitration lost interrupt enable - Enable arbitration lost Interrupt 0: Disabled 1: Enabled	0
0	HGC_EN	R/W	General Call Interrupt Enable 0: Disabled 1: Enabled	0

Table 6.14. I2CFIFOINTENABLE_MSB Register Details (FIFO Mode)

0x0D – I2CFIFOINTENABLE_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	RSVD	—	Reserved	—
4	BUS_FREE_EN	R	Enable BUS_FREE interrupt. 0: Disabled 1: Enabled	0
3	RXFUNDERF_EN	R	RXFIFO underflow interrupt enable 0: Disabled 1: Enabled	0
2	TXOVERF_EN	R	TXFIFO overflow interrupt enable 0: Disabled 1: Enabled	0
1	RSVD	—	Reserved	—
0	MRDCMPL_EN	R/W	Master read complete enable 0: Disabled 1: Enabled	0

6.6. I²C FIFO Threshold Register (I2CFIFOTHRESHOLD)

These registers store the FIFO threshold values that control when rxfifoaf_o and txfifoae_o are asserted. The rxfifoaf_o threshold value is a concatenation of {I2CFIFOTHRESHOLD_MSB[1:0], I2CFIFOTHRESHOLD_LSB[7:5]}. The values in this register should be set up through the IP Catalog tool of Lattice Radiant software.

Table 6.15. I2CFIFOTHRESHOLD_LSB Register Details

0x0E – I2CFIFOTHRESHOLD_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	RXFIFO_AF_VAL	R/W	3 lowest bits of 5-bit Almost Full value for the RX FIFO	00000
4	RSVD	—	Reserved	—
3:0	TXFIFO_AE_VAL	R/W	4-bit Almost Empty value for the TX FIFO	00000

Table 6.16. I2CFIFOTHRESHOLD_MSB Register Details

0x0F – I2CFIFOTHRESHOLD_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:2	RSVD	—	Reserved	—
1:0	RXFIFO_AF_VAL	R/W	2 upper bits of 5-bit Almost Full value for the RX FIFO	00000

6.7. I^{2}C Command Register (I2CCMDR)

I2CCMDR offers controls for operating in register mode. See the Register Mode Functional Waveforms section for details of operating the hard IP in Register mode. RBUFDIS is always at default value (0) for FIFO mode.

Table 6.17. I2CCMDR Register Details

0x10 - I2CCMDR				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	STA	R/W	Generate start/restart condition when combined with WR bit (bit 4) for master mode. This should be set after data is written to I2CTXDR.	0
6	STO	R/W	Generate STOP Condition (Exclusive with STA bit) for master mode	0
5	RD	R/W	Read from Slave for master mode	0
4	WR	R/W	Write to Slave for master mode	0
3	ACK	R/W	Acknowledge Option – ACK transmission selection for both master and slave mode. 0: Sends ACK 1: Sends NACK	0
2	CKSDIS	R/W	Clock Stretching Disable Option – Disable clock stretching in both master and slave mode. If clock stretching is disabled, status flags must be monitored to ensure correct behavior. 0: Clock stretching is enabled 1: Clock stretching is disabled	0
1	RBUFDIS	R/W	Read Command With Buffer Disabled – Read from Slave in master mode with the double buffering disabled for easier control over single byte data communication scenario. 0: Read with buffer enabled as default 1: Read with buffer disabled	0
0	RSVD	—	Reserved	—

6.8. I^{2}C Transmitting Data Register (I2CTXD)/Transmitting FIFO (I2CTXFIFO)

This address is shared by both register and FIFO mode. During Register mode it is a 1-byte register at address 0x12. During FIFO mode, it is a 10-bit × 32 FIFO, with the LSB at 0×12 and the MSB at 0×13.

Table 6.18. I2CTXDR Register Details (Register Mode)

0x12 – I2CTXDR				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:1	TXDATA[7:1]	W	Next byte to transmit through I^{2}C	0000000
0	TXDATA[0]/R_W	W	In case of a data transfer, this bit represents the data is LSB. In case of a slave address transfer, this bit represents the RW bit: *1* = Reading from slave *0* = Writing to Slave	0

The I2CTXFIFO is write only. However, a read to this location, either to the LSB or MSB address, during FIFO mode causes the I2CTXFIFO to be reset (reset the pointers). Bits [1:0] of I2CTXFIFO_MSB are the command bits, while I2CTXFIFO_LSB is for data or slave address. The 8-bit data can be interpreted differently depending on the value of bit 9. The CMD bit and the RSTAEN/LTXBYTE bits of I2CTXFIFO_MSB are meaningful and used when the IP is in Master Mode. Writes to I2CTXFIFO are done by writing to address 0x12 then address 0x13, which increments the FIFO write pointer. Examples of using the FIFO can be found in the FIFO Mode Data Transaction section.

Table 6.19. I2CTXFIFO_MSB Register Details (FIFO Mode)

0x12 – I2CTXFIFO_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	RSVD/DATA	W	Not used when CMD =1, bit 7 of the data byte when CMD =0	0
6	RSVD/DATA	W	Not used when CMD =1, bit 6 of the data byte when CMD =0	0
5	RSVD/DATA	W	Not used when CMD =1, bit 5 of the data byte when CMD =0	0
4	RXBYTE/DATA	W	Bit 4 of RXBYTE value when CMD =1, bit 4 of the data byte when CMD = 0	0
3	RXBYTE/DATA	W	Bit 3 of RXBYTE value when CMD =1, bit 3 of the data byte when CMD = 0	0
2	RXBYTE/DATA	W	Bit 2 of RXBYTE value when CMD =1, bit 2 of the data byte when CMD = 0	0
1	RXBYTE/DATA	W	Bit 1 of RXBYTE value when CMD =1, bit 1 of the data byte when CMD = 0	0
0	RXBYTE/DATA	W	Bit 0 of RXBYTE value when CMD =1, bit 0 of the data byte when CMD = 0	0

Table 6.20. I2CTXFIFO_MSB Register Details (FIFO Mode)

0x13 – I2CTXFIFO_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:2	RSVD	—	Reserved	—
1:0	CMD / “FLAGs”	W	Two command bits to direct the I²C state machine. *CMD, RSTAEN* *10* – bits [4:0] of this byte indicate the number of bytes to be recieved. Following data transaction should be sent using a STOP then a START. *11* – bits [4:0] of this byte indicate the number of bytes to be received. Following data transaction should be sent using a START/ReSTART. The 1st data byte should always have RSTAEN bit set to 1. *CMD, LTXBYTE* *00* – bits [7:0] of this byte are data bits. If this is the last data byte in the TXFIFO, then depending on the CKSDIS bit, Master Write either goes into clock stretching (CKSDIS=0), or TXFIFO underflows (CKSDIS=1). *01* – bits [7:0] of this byte are data bytes. If this is the last data byte in TXFIFO, this indicates the last byte to be transferred and a STOP is issued. If this is not the last byte in TXFIFO, then this bit is ignored.	00

The range of bits [4:0] is 0 to 31, with a *0* indicates receiving 1 byte, a *1* receiving 2 bytes, and a *31* receiving 32 bytes, and others. Therefore, an I²C Read must receive at least 1 byte.

While operating as master, the current transaction can be aborted by resetting the TXFIFO (by issuing a read to TXFIFO), or by asserting the FIFO_RST signal. When the TXFIFO is reset while the state machine is in transmit mode, it issues a STOP after the current byte is transmitted. When the TXFIFO is reset and the state machine is in receive mode, it issues a NACK+STOP. This is to make sure the I²C bus is gracefully released.

6.9. I^{2}C Receiving Data Register (I2CRXD)/Receiving FIFO (I2CRXFIFO)

This address is shared by both register and FIFO mode. During Register mode, it is a 1-byte register 0x14. During FIFO mode, it is 10 bit x 32 word FIFO, with the LSB at address 0x14 and the MSB at address 0x15.

Table 6.21. I2CRXDR Register Details (Register Mode)

0x14 – I2CRXDR				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	RXDATA	R	Received data	00000000

The I2CRXFIFO is read only. However, a write to this location, either to the LSB or MSB address, during FIFO mode causes the I2CRXFIFO to be reset (reset the pointers). The lower byte (the LSB) of each 10 bit word is data, while the upper bits (the MSB) are information about that data. To read from I2CRXFIFO, first read from address 0x14 then from address 0x15. Reading from address 0x15 increments the FIFO read pointer. For examples of using the FIFO, refer to the FIFO Mode Data Transaction section.

Table 6.22. I2CRXFIFO_LSB Register Details (FIFO Mode)

0x14 – I2CRXFIFO_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	DATA	R	Data received	00000000

Table 6.23. I2CRXFIFO_MSB Register Details (FIFO Mode)

0x15 – I2CRXFIFO_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:1	RSVD	—	Reserved	—
0	DFIRST	R	First byte of data 0: Normal data 1: First byte received after a Start or a Restart is detected	0

6.10. I^{2}C General Call Information Register (I2CGCD)

This register contains information for the general call when I^{2}C port is addressed as a slave. If bit 0 of I2CGCD is *ZERO*, this byte is the general command (H’00 is illegal). If bit 0 is *ONE*, as a *hardware general call*, the D7:D1 is the address of the hardware master device, such as a keyboard, scanner.

Once a general call is received, if the GCEN bit inside I2CC1 is set, an interrupt is sent to the system host, and the status bits that associated with HCG are set in the status register (I2CSR) and interrupt status register (I2CINTSR).

Table 6.24. I2CGCDR Register Details

0x16 – I2CGCDR				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	GCDATA	R	Received general call data	00000000

6.11. I^{2}C Status Register (I2CSR)/I^{2}C FIFO Status Register (I2CFIFOSR)

This pair of offsets, 0×18 and 0×19, are shared by both register and FIFO modes. However, the definition of each status bit is different for each mode. In register mode, this address is labeled I2CSR. In FIFO mode, this address is referred to as I2CFIFOSR.

A *1* in I2CSR [3:0] causes an interrupt to system host, if the corresponding interrupt control registers (I2CINTENABLE [3:0]) are enabled.

A *1* in any bit of I2CFIFOSR causes an interrupt to the system host, if the corresponding interrupt control registers (I2CFIFOINTENABLE) are enabled. MRDCMPL has a dedicated interrupt signal to the fabric. The rest of the status shares a common interrupt to the fabric.

Table 6.25. I2CSR_LSB (Register Mode)

0x18 – I2CSR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	TIP	R	Transmission in Progress – This bit indicates that current data byte is being transferred for both master and slave mode. Note that the TIP flag suffers half SCL cycle latency right after the start condition because of the signal synchronization. Also note that this bit could be high after configuration wake-up and before the first valid I^{2}C transfer start (when BUSY is low), and it is not indicating byte in transfer, but an invalid indicator. 0: Byte transfer completed 1: Byte transfer in progress	0
6	BUSY	R	Bus busy – This bit indicates the bus is involved in a transaction. This is set at start condition and cleared at stop. Therefore, only when this bit is high, should all other status bits be treated as valid indicators for a valid transfer. 0: Bus is free 1: Bus is busy	0
5	SRW	R	Slave RW – This bit indicates the direction of the current transaction. This bit is valid after the falling edge of SCL after the R/W bit of the address byte. 0: Master transmitting/Slave receiving 1: Master receiving/Slave transmitting	0
4	RARC	R	Received Acknowledge – This flag represents an acknowledge response from the addressed slave during master write or from receiving master during master read. This is valid after the ACK bit, until the ACK bit of the next byte or until a stop or restart condition. 0: ACK 1: NACK	0
3	TRRDY	R	Transmitter or Receiver Ready Bit – This flag indicate that a Transmit Register ready to receive data or Receiver Register is ready for read depend on the mode (master or slave) and SRW bit. It causes an interrupt to system host if set in I2CINTENABLE. 0: Transmitter or Receiver is not ready 1: Transmitter or Receiver is ready	0
2	TROE	R	Transmitter/Receiver Overrun or NACK Received Bit – This flag indicates that a Transmit or Receive Overrun Errors happened depend on the mode (master or slave) and SRW bit, or a no-acknowledges response is received after transmitting a byte. If RARC bit is high, it is a NACK bit. Otherwise, it is overrun bit. It causes an interrupt to system host if set in I2CINTENABLE. 0: Normal 1: Overrun Error/NACK	0
1	ARBL	R	Arbitration Lost – This bit goes high if the master has lost its arbitration in Master mode. It causes an interrupt to system host if set in I2CINTENABLE. 0: Normal 1: Arbitration lost	0
0	HGC	R	Hardware General Call Received – This flag indicates that a hardware general call is received from the slave port. It causes an interrupt to system host, if set in I2CINTENABLE. 0: No hardware general call received in Slave mode 1: Hardware general call received in Slave mode	0

Table 6.26. I2CSR_MSB (Register Mode)

0x19 – I2CSR_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	RSVD	—	Reserved	—
4	BUS_FREE	R	Bus Free – This bit indicated when the bus is free. This allows the host to know if it is safe to start an I^{2}C transaction at that time. This bit is high as long as there is no transaction occurring on the bus. It causes an interrupt to system host, if set in I2CINTENABLE. 0: Bus is not free 1: Bus is free	0
3:0	RSVD	—	Reserved	—

Table 6.27. I2CFIFOSR_LSB Register Details (FIFO Mode)

0x18 – I2CFIFOSR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	TXSERR	R	TX FIFO synchronization error – This happens when there are back-to-back commands in the FIFO.	0
6	TXUNDERF	R	TX FIFO underflow – Indicates an error condition, mutually exclusive with clock stretching function. 0 : Not underflow 1 : FIFO underflow, data is not valid	0
5	RXOVERF	R	RX FIFO overflow – Indicates an error condition, mutually exclusive with clock stretching function. 0 : Not overflow 1 : FIFO overflow, data is not valid	0
4	RNACK	R	Received NACK – This flag represents acknowledge response from the addressed slave during master write.	0
3	BUSY	R	Bus busy – This bit indicates the bus is involved in a transaction. This is set at start condition and cleared at stop. Therefore, only when this bit is high, should all other status bits be treated as valid indicators for a valid transfer.	0
2	ADM	R	Slave Address Matched – Indicates that this I^{2}C slave address has matched with an address sent by an external master. This status stays active for entire transmission.	0
1	ARBL	R	Arbitration Lost – This bit goes high, if the master has lost its arbitration in Master mode.	0
0	HGC	R	Hardware General Call Received – This flag indicates that a hardware general call is received from the slave port. It causes an interrupt to system host, if set in I2CINTENABLE.	0

Table 6.28. I2CFIFOSR_MSB Register Details (FIFO Mode)

0x19 – I2CFIFOSR_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	Reserved	—	—	—
4	BUS_FREE	R	Bus Free – This bit indicated when the bus is free. This allows the host to know if it is safe to start an I^{2}C transaction at that time. This bit is high as long as there is no transaction occurring on the bus. 0: Bus is not free 1: Bus is free	0
3	RXFUNDERF	R	RX FIFO Underflow – This bit indicates an error condition. This current transaction has caused the RXFIFO to underflow. 0: No underflow 1: Underflow	0
2	TXOVERF	R	TX FIFO Overflow – This bit indicates an error condition. The current transaction has caused the TXFIFO to overflow. 0: No Overflow 1: Overflow	0
1	SRW	R	Slave RW - This bit indicates the direction of the current transaction. This bit is valid after the falling edge of SCL after the R/W bit of the address byte. 0: Master transmitting / Slave receiving 1: Master receiving / Slave transmitting	0
0	MRDCMPL	R	Master Read Complete - this is only valid for Master Read mode. 0: Transaction is not completed 1: Transaction is completed. In Master read mode, it means: • the number of bytes read equals to the expected number; • Master terminates the read earlier but there is data in the RX FIFO.	0

6.12. I²C Interrupt Status Register (I2CINTSR)/I²C FIFO Interrupt Status Register (I2CFIFOINTSR)

This set of offsets (0x1A and 0x1B) is shared by both register and FIFO mode. Writing a *1* to a bit in this register causes the corresponding interrupt request flag to be cleared. If I2CC2[7] (Auto Interrupt Clear Enable) is set, reading the register causes all IRQ flags to be cleared.

Table 6.29. I2CINTSR_LSB Register Details (Register Mode)

0x1A – I2CINTSR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:4	RSVD	—	Reserved	—
3	TRRDY_INT	R	Transmit/Receive register ready interrupt request flag 0: No interrupt request 1: Interrupt request pending	0
2	TROE_INT	R	Transmit/Receive register overrun interrupt request flag 0: No interrupt request 1: Interrupt request pending	0
1	ARBL_INT	R	Arbitration lost interrupt request flag 0: No interrupt request 1: Interrupt request pending	0
0	HGC_INT	R	General call interrupt request flag 0: No interrupt request 1: Interrupt request pending	0

Table 6.30. I2CINTSR_MSB Register Details (Register Mode)

0x1B – I2CINTSR_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	RSVD	—	Reserved	—
4	BUS_FREE_INT	R	Bus free interrupt request flag 0: Disabled 1: Enabled	0
3:0	RSVD	—	Reserved	—

Table 6.31. I2CFIFOINTSR_LSB Register Details (FIFO Mode)

0x1A – I2CFIFOINTSR_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	TXSERR_INT	R	TXFIFO synchronization error interrupt request flag 0: No interrupt request 1: Interrupt request pending	000
6	RSVD	—	Reserved	—
5	RXOVERF_INT	R	RXFIFO overflow interrupt request flag 0: No interrupt request 1: Interrupt request pending	0
4	RNACK_INT	R	NACK interrupt request flag 0: No interrupt request 1: Interrupt request pending	0
3:2	RSVD	—	Reserved	—
1	ARBL_INT	R	Arbitration lost interrupt request flag 0: No interrupt request 1: Interrupt request pending	0
0	HGC_INT	R	General call interrupt request flag 0: No interrupt request 1: Interrupt request pending	0

Table 6.32. I2CFIFOINTSR_MSB Register Details (FIFO Mode)

0x1B – I2CFIFOINTSR_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:5	RSVD	—	Reserved	—
4	BUS_FREE_INT	R	Bus free interrupt request flag 0: Disabled 1: Enabled	0
3	RXFUNDERF_INT	R	RXFIFO underflow interrupt request flag 0: Disabled 1: Enabled	0
2	TXOVERF_INT	R	TXFIFO overflow interrupt request flag 0: Disabled 1: Enabled	0
1	RSVD	—	Reserved	—
0	MRDCMPL_INT	R	Master read completion interrupt request flag 0: No interrupt request 1: Interrupt request pending	0

6.13. I^{2}C FIFO TX Byte Counter (I2CFIFOTXCNT)

This is a read-only register. It stores the current count of data bytes that have been sent through the I^{2}C port. The number of bytes is cumulative until the counter is cleared. A write to this register or assertion of FIFO_RST signal causes the counter to be cleared. This register is used in both register and FIFO mode. The total count is a concatenation of {I2CFIFOTXCNT_MSB, I2CFIFOTXCNT_LSB}.

Table 6.33. I2CFIFOTXCNT_LSB Register Details

0x1E – I2CFIFOTXCNT_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	TX_BYTE_CNT	R	Bits [7:0] of the number of data bytes that have been transmitted to the I^{2}C port	00000000

Table 6.34. I2CFIFOTXCNT_MSB Register Details

0x1F – I2CFIFOTXCNT_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	TX_BYTE_CNT	R	Bits [15:8] of the number of data bytes that have been transmitted to the I^{2}C port	00000000

6.14. I²C FIFO RX Byte Counter (I2CFIFORXCNT)

This is a read-only register. It stores the current count of data bytes that have been read into the RXFIFO. The number of bytes is accumulative until the counter is cleared. A write to this register or assertion of FIFO_RST signal causes the counter to be cleared. This register is used in both register and FIFO mode. The total count is a concatenation of {I2CFIFORXCNT_MSB, I2CFIFORXCNT_LSB}.

Table 6.35. I2CFIFORXCNT_LSB Register Details

0x20 – I2CFIFORXCNT_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	RX_BYTE_CNT	R	Bits [7:0] of the number of data bytes that have been received at the RX FIFO or RX register	00000000

Table 6.36. I2CFIFORXCNT_MSB Register Details

0x21 – I2CFIFORXCNT_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	RX_BYTE_CNT	R	Bits [15:8] of the number of data bytes that have been received at the RX FIFO or RX register	00000000

6.15. I²C Interrupt Set Register (I2CINTSET)/I²C FIFO Interrupt Set Register (I2CFIFOINTSET)

This address is shared by both register and FIFO mode. The register can be read or written through LMMI. This register is similar to the previous interrupt registers, with the exception of BUS_FREE interrupt. This interrupt cannot be forced.

Table 6.37. I2CINTSET Register Details

0x22 – I2CINTSET				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:4	RSVD	—	Reserved	—
3	TRRDY_SET	R/W	Transmit/Receive register ready force interrupt – Force TRRDY interrupt 0: Disabled 1: Enabled	0
2	TROE_SET	R/W	Transmit/Receive Register Overrun Force Interrupt – Force TROE interrupt 0: Disabled 1: Enabled	0
1	ARBL_SET	R/W	Arbitration Lost Force Interrupt – Force Arbitration Lost Interrupt 0: No Interrupt Request 1: Interrupt Request Pending	0
0	HGC_SET	R/W	General Call Force Interrupt – Force General Call interrupt 0: Disabled 1: Enabled	0

Table 6.38. I2CFIFOINTSET_LSB

0x22 – I2CFIFOINTSET_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7	TXSERR_SET	R/W	TX FIFO synchronization error force interrupt 0: Disabled 1: Enabled	0
6	RSVD	—	Reserved	—
5	RXOVERF_SET	R/W	RXFIFO overflow force interrupt 0: Disabled 1: Enabled	0
4	RNACK_SET	R/W	Receive NACK force interrupt 0: Disabled 1: Enabled	0
3:2	RSVD	—	Reserved	—
1	ARBL_SET	R/W	Arbitration Lost Force Interrupt – Force arbitration lost interrupt 0: Disabled 1: Enabled	0
0	HGC_IINTFRC	R/W	General call force interrupt 0: Disabled 1: Enabled	0

Table 6.39. I2CFIFOINTSET_MSB Register Details

0x23 – I2CFIFOINTSET_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:4	RSVD	—	Reserved	—
3	RXFUNDERF_SET	R	RXFIFO underflow force interrupt 0: Disabled 1: Enabled	0
2	TXOVERF_SET	R	TXFIFO overflow force interrupt 0: Disabled 1: Enabled	0
1	RSVD	—	Reserved	—
0	MRDCMPL_SET	R/W	Master read complete force interrupt 0: Disabled 1: Enabled	0

6.16. I^{2}C Start/Repeated Start Timer Register (I2CSTSP_TIMER_DELAY)

This pair of registers at offsets 0×25 and 0×26 holds a pre-loaded, 11-bit delay value. This 11-bit register contains a pre-loaded value (11'h497) for Start/Repeated Start timer. This delay ensures I^{2}C compliance in respect to set-up and hold times required by the protocol.

Table 6.40. I2CSTSP_TIMER_DELAY_LSB Register Details

0x25 – I2CSTSP_TIMER_DELAY_LSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:0	Timer value	R/W	Timer Value – Value that is loaded into a down counter. This value can ensure I^{2}C timing specification for Start/Repeated start signal is met.	8'h97

Table 6.41. I2CSTSP_TIMER_DELAY_MSB Register Details

0x26 – I2CSTSP_TIMER_DELAY_MSB				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
7:3	RSVD	—	Reserved	—
2:0	Timer value	R/W	Timer Value – Value that is loaded into a down counter. This value ensures I^{2}C timing specification for Start/Repeated start signal is met.	100

6.17. I^{2}C TXFIFO Bytes Available Register (I2CTXFIFOBYTEAVAIL)

This register contains the current number of bytes in the Tx FIFO. This register is only available in FIFO mode.

Table 6.42. I2CTXFIFOBYTEAVAIL Register Details

0x27 – I2CTXFIFOBYTEAVAIL				
Bit(s)	**Label**	**R/W**	**Description**	**Default**
9:5	RSVD	—	Reserved	—
4:0	TXFIFO bytes available	R	The number of bytes that are available and ready to be read in the TXFIFO.	00000

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Revision 1.2, October 2020

Section	Change Summary
Module Setup with IP Catalog	• Updated the introductory paragraph. • Updated information on how to access the I2CFIFO module. • Updated descriptions of Master Clock Rate settings. • Added reference to the I2C General Call Information Register (I2CGCD) section.
Functional Description	• Updated signal names. • Updated Slave Read/Write Example process flow. • Updated information on write operation in master mode. • Updated the introductory paragraph of the FIFO Mode Data Transaction section. • Updated the following figures: • Figure 4.3. Slave Mode Write I2C Data Diagram • Figure 4.5. Slave Mode Read/Write Example • Figure 4.6. Master Mode Read/Write • Minor editorial changes.
Port/Pin Description	In Table 5.1. I2C Hard IP Interface Signals, updated signal names and descriptions.
Configurable I²C Registers	• Spelled out LMMI in section introduction. • Updated label descriptions in: • Table 6.3. I2CC2 Register Details • Table 6.6. I2CSLVADDR_LSB Register with 7 Bit Addressing and Hard Tied LSB • Table 6.7. I2CSLVADDR_LSB Register with 10 Bit Addressing and Hard Tied LSB • Table 6.8. I2CSLVADDR_LSB Register with 7 Bit Addressing and without Hard Tied LSB • Table 6.20. I2CTXFIFO_MSB Register Details (FIFO Mode) • Updated signal names in the I2C FIFO Threshold Register (I2CFIFOTHRESHOLD) section introductory paragraph. • Updated operation description in the I2C Transmitting Data Register (I2CTXD)/Transmitting FIFO (I2CTXFIFO) section. • Added details on LSB and MSB in the I2C Receiving Data Register (I2CRXD)/Receiving FIFO (I2CRXFIFO) section. • Changed *D0* to *bit 0* in the I2C General Call Information Register (I2CGCD) section.

Revision 1.1, June 2020

Section	Change Summary
All	• Changed document title from CrossLink-NX I²C Hardened IP Usage Guide to I²C Hardened IP Usage Guide for Nexus Platform. • Added Certus-NX support.

Revision 1.0, December 2019

Section	Change Summary
All	Initial release.

LATTICE SEMICONDUCTOR®

www.latticesemi.com