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

1. Introduction ...................................................................................................................... 5
2. System Bus Interface for CrossLink .................................................................................. 5
   2.1. System Bus Write Cycle ............................................................................................. 6
   2.2. System Bus Read Cycle ............................................................................................. 7
3. I²C Hardened IP Cores ..................................................................................................... 8
4. I²C Registers for CrossLink ............................................................................................... 9
   4.1. I²C Control Register 1 (I2CCR1) ............................................................................... 9
   4.2. I²C Command Register (I2CCMDR) ....................................................................... 10
   4.3. I²C Clock Pre-scale Register (I2CBRMSB/I2CBRLSB) ............................................ 11
   4.4. I²C Status Register (I2CSR/I2CFIFOSR) ................................................................. 11
   4.5. I²C Transmitting Data Register (I2CTXDR/I2CTXFIFO) ........................................ 13
   4.6. I²C Receiving Data Register (I2CRXDR/I2CRXFIFO) ............................................. 14
   4.7. I²C General Call Data Register ............................................................................... 15
   4.8. I²C Slave Address MSB Register (I2CSADDRI2CFIFOSADDRI2CFIFOADDRI2CSADDR) 15
   4.9. I²C Interrupt Control Register (I2CINTCR/I2CFIFOINTCR) ................................... 15
   4.10. I²C Interrupt Status Register (I2CINTSR/I2CFIFOINTSR) .................................... 16
   4.11. I²C FIFO Threshold Register (I2CFIFOTHRESHOLD) ........................................... 17
   4.12. I²C FIFO TX Byte Counter (I2CFIFOTXCNT) ......................................................... 17
   4.13. I²C FIFO RX Byte Counter (I2CFIFORXCNT) ....................................................... 17
5. I²C Read/Write Flowchart ............................................................................................... 18
6. I²C Functional Waveforms .............................................................................................. 20
References ............................................................................................................................. 21
Technical Support Assistance ............................................................................................. 22
Revision History ................................................................................................................... 23

Figures

Figure 2.1. System Bus Interface between the FPGA Core and the IP .................................... 5
Figure 2.2. System Bus Write Operation ............................................................................. 6
Figure 2.3. System Bus Read Operation ............................................................................. 7
Figure 5.1. I²C Master Read/Write Example (via System Bus) ........................................... 18
Figure 5.2. I²C Slave Read/Write Example (via System Bus) ............................................. 19
Figure 6.1. Master – I²C Write ......................................................................................... 20
Figure 6.2. Master I²C Read ........................................................................................... 20
Figure 6.3. Slave I²C Write ............................................................................................ 20
Figure 6.4. Slave I²C Read ............................................................................................. 20

Tables

Table 2.1. System Bus Slave Interface Signals of the Hardened I²C Module ....................... 6
Table 4.1. I²C Registers Summary .................................................................................... 9
Table 4.2. I²C Control Register 1 (I2CCR1) ...................................................................... 9
Table 4.3. I²C Command Register (I2CCMDR) ............................................................... 10
Table 4.4. I²C Clock Pre-scale Register (I2CBRMSB) ....................................................... 11
Table 4.5. I²C Clock Pre-scale Register (I2CBRLSB) ....................................................... 11
Table 4.6. I²C Status Register (I2CSR) ........................................................................... 11
Table 4.7. I²C FIFO Status Register (I2CFIFOSR) ............................................................. 12
Table 4.8. I2C Transmitting Data Register (I2CTXDR) ................................................................. 13
Table 4.9. I2C Transmitting FIFO Register (I2CTXFIFO) ................................................................. 13
Table 4.10. I2C Receiving Data Register (I2CRXDR) ................................................................. 14
Table 4.11. I2C Receiving FIFO Register (I2CRXFIFO) ................................................................. 14
Table 4.12. I2C General Call Data Register (I2CGCDR) ................................................................. 15
Table 4.13. I2C Slave Address MSB Register (I2CSADDR) ................................................................. 15
Table 4.14. I2C Slave Address MSB Register (I2CFIFOSADDR) ......................................................... 15
Table 4.15. I2C Interrupt Control Register (I2CINTCR) ................................................................. 15
Table 4.16. I2C Interrupt Control Register (I2CFIFOINTCR) .......................................................... 16
Table 4.17. I2C Interrupt Status Register (I2CINTSR) ................................................................. 16
Table 4.18. I2C Interrupt Status Register (I2CFIFOINTSR) ............................................................. 16
Table 4.19. I2C FIFO Threshold Register (I2CFIFOTHRESHOLD) ................................................... 17
Table 4.20. I2C FIFO TX Byte Counter (I2CFIFOTXCNT) .............................................................. 17
Table 4.21. I2C FIFO RX Byte Counter (I2CFIFORXCNT) .............................................................. 17
1. Introduction

This document provides guidance for the advanced usage of the Lattice Semiconductor CrossLink™ I2C IP, and supplements CrossLink I2C Hardened IP Usage Guide (FPGA-TN-02019).

The recommended flow for initializing the Hard IP I2C blocks is the Clarity Designer – GUI flow as described in FPGA-TN-02019.

This document includes the following:
- System Bus Protocol
- I²C Register Mapping
- I²C Timing Diagram
- Command Sequences
- Examples

2. System Bus Interface for CrossLink

The System Bus in CrossLink provides connectivity between FPGA user logic and the Hardened IP functional blocks. The user can implement a System Bus Master interface to interact with the Hardened IP System Bus Slave interface.

The block diagram in Figure 2.1 shows the supported System Bus signals between the FPGA core and the Hardened IP. Table 2.1 on the next page lists the supported signals.

![Figure 2.1. System Bus Interface between the FPGA Core and the IP](image)
Table 2.1. System Bus Slave Interface Signals of the Hardened I2C Module

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>I2C Wrapper Name*</th>
<th>I/O</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBCSI</td>
<td>i2cXcsi</td>
<td>Input</td>
<td>1</td>
<td>This chip select signal activates the IP to allow system bus to communicate with the IP.</td>
</tr>
<tr>
<td>SBCLKI</td>
<td>i2cXclkI</td>
<td>Input</td>
<td>1</td>
<td>Positive edge clock used by System Bus Interface registers and hardened functions. Supports clock speeds up to 133 MHz.</td>
</tr>
<tr>
<td>SBSTBI</td>
<td>i2cXstbi</td>
<td>Input</td>
<td>1</td>
<td>Active-high strobe, input signal, indicating the System Bus Slave is the target for the current transaction on the bus. The IP asserts an acknowledgment in response to the assertion of the strobe.</td>
</tr>
<tr>
<td>SBRWI</td>
<td>i2cXwei</td>
<td>Input</td>
<td>1</td>
<td>Level sensitive Write/Read control signal. Low indicates a Read operation, and High indicates a Write operation.</td>
</tr>
<tr>
<td>SBADRI</td>
<td>i2cXadri[3:0]</td>
<td>Input</td>
<td>4</td>
<td>4-bit wide address used to select a specific register from the register map of the IP.</td>
</tr>
<tr>
<td>SBDATI</td>
<td>i2cXdati[9:0]</td>
<td>Input</td>
<td>10</td>
<td>8-bit input data path used to write a byte of data to a specific register in the register map of the IP. 10 bits used for FIFO mode.</td>
</tr>
<tr>
<td>SBDATO</td>
<td>i2cXdato[9:0]</td>
<td>Output</td>
<td>10</td>
<td>8-bit output data path used to read a byte of data from a specific register in the register map of the IP. 10 bits used for FIFO mode.</td>
</tr>
<tr>
<td>SBACKO</td>
<td>i2cXacko</td>
<td>Output</td>
<td>1</td>
<td>Active-high, transfer acknowledge signal asserted by the IP, indicating the requested transfer is acknowledged.</td>
</tr>
</tbody>
</table>

Note: X indicates the I2C. X = 0 for the Right I2C and X = 1 for the Left I2C.

To interface with the IP, you must create a System Bus Master controller in the User Logic. In a multiple-Master configuration, the System Bus Master outputs are multiplexed through a user-defined arbiter. If two Masters request the bus in the same cycle, only the outputs of the arbitration winner reach the Slave interface.

2.1. System Bus Write Cycle

Figure 2.2 shows the waveform of a Write cycle from the perspective of the System Bus Slave interface. During a single Write cycle, only one byte of data is written to the IP block from the System Bus Master. A Write operation requires a minimum three clock cycles.

![Figure 2.2. System Bus Write Operation](image-url)
On clock Edge 0, the Master updates the address, data and asserts control signals. During this cycle the Master:

- Updates the address on the SBADRI[3:0] address lines
- Updates the data that will be written to the IP block, SBDATI[9:0] data lines
- Asserts the write enable SBRWI signal, indicating a write cycle
- Asserts the SBSTBI, selecting a specific Slave module
- On clock Edge 1, the System Bus Slave decodes the input signals presented by the Master. During this cycle:
  - The Slave decodes the address presented on the SBADRI[3:0] address lines
  - The Slave prepares to latch the data presented on the SBDATI[9:0] data lines
  - The Master waits for an active-high level on the SBACKO line and prepares to terminate the cycle on the next clock edge, if an active-high level is detected on the SBACKO line
  - The IP may insert wait states before asserting SBACKO, thereby allowing it to throttle the cycle speed. Any number of wait states may be added
  - The Slave asserts SBACKO signal

The following occurs on clock Edge 2:

- The Slave latches the data presented on the SBDATI[9:0] data lines
- The Master de-asserts the strobe signal, SBSTBI, and the write enable signal, SBRWI
- The Slave de-asserts the acknowledge signal, SBACKO, in response to the Master de-assertion of the strobe signal

2.2. System Bus Read Cycle

Figure 2.3 shows the waveform of a Read cycle from the perspective of the System Bus Slave interface. During a single Read cycle, only one byte of data is read from the IP block by the System Bus Master. A Read operation requires a minimum three clock cycles.

![System Bus Read Operation Diagram](image-url)
On clock Edge 0, the Master updates the address, and asserts control signals. The following occurs during this cycle:

1. The Master updates the address on the SBADRI[3:0] address lines
2. De-asserts the write enable SBRWI signal, indicating a Read cycle
3. Asserts the SBSTBI, selecting a specific Slave module

On clock Edge 1, the System Bus Slave decodes the input signals presented by the Master. The following occurs during this cycle:

1. The Slave decodes the address presented on the SBADRI[3:0] address lines
2. The Master prepares to latch the data presented on SBDATO[9:0] data lines from the System Bus Slave on the following clock edge
3. The Master waits for an active-high level on the SBACKO line and prepares to terminate the cycle on the next clock edge, if an active-high level is detected on the SBACKO line
4. The IP may insert wait states before asserting SBACKO, thereby allowing it to throttle the cycle speed. Any number of wait states may be added.
5. The Slave presents valid data on the SBDATO[9:0] data lines
6. The Slave asserts SBACKO signal in response to the strobe, SBSTBI signal

The following occurs on clock Edge 2:

1. The Master latches the data presented on the SBDATO[9:0] data lines
2. The Master de-asserts the strobe signal SBSTBI
3. The Slave de-asserts the acknowledge signal, SBACKO, in response to the Master de-assertion of the strobe Signal

3. **I^2C** Hardened IP Cores

I^2C is a widely used two-wire serial bus for communication between devices on the same board. Every CrossLink device contains two I^2C hardened IP cores. Either of the two cores can be operated as an I^2C Master or as an I^2C Slave. The I^2C0 core has dedicated I/O pins, called USER_SCL and USER_SDA, on the CrossLink device. This is in order to support the device sleep mode wakeup over I^2C function. The SCL and SDA pins from the I^2C1 core may be connected to any pin on the device.
4. I²C Registers for CrossLink

Both I²C cores communicate with the System Bus interface through a set of control, command, status and data registers. Table 4.1 lists the register names and their functions.

### Table 4.1. I²C Registers Summary

<table>
<thead>
<tr>
<th>Name</th>
<th>SB Address [3:0]</th>
<th>Register Function</th>
<th>Register Width</th>
<th>Support Modes</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>I2CCR1</td>
<td>0001</td>
<td>I²C Control Register 1</td>
<td>8</td>
<td>Both</td>
<td>RW</td>
</tr>
<tr>
<td>I2CBRLSB</td>
<td>0010</td>
<td>I²C Clock Presale register, LSB</td>
<td>8</td>
<td>Both</td>
<td>RW</td>
</tr>
<tr>
<td>I2CBRMSB</td>
<td>0011</td>
<td>I²C Clock Presale register, MSB</td>
<td>8</td>
<td>Both</td>
<td>RW</td>
</tr>
<tr>
<td>I2CSADDR/I2CFIFOSADDR</td>
<td>0100</td>
<td>I²C Slave address/FIFO Slave Address</td>
<td>8/10</td>
<td>Both</td>
<td>RW</td>
</tr>
<tr>
<td>I2CINTCR/I2CFIFOINTCR</td>
<td>0101</td>
<td>I²C Interrupt Control Register/FIFO interrupt Control register</td>
<td>8/10</td>
<td>Both</td>
<td>RW</td>
</tr>
<tr>
<td>I2CFIFOTHRSHOLD</td>
<td>0110</td>
<td>I²C FIFO Threshold Register</td>
<td>10</td>
<td>FIFO mode</td>
<td>RW</td>
</tr>
<tr>
<td>I2CCMDR</td>
<td>0111</td>
<td>I²C Command Register</td>
<td>8</td>
<td>Reg mode</td>
<td>RW</td>
</tr>
<tr>
<td>I2CTXDR/I2CTXFIFO</td>
<td>1000</td>
<td>I²C Transmitting Data Register/FIFO</td>
<td>8/10</td>
<td>Both</td>
<td>W</td>
</tr>
<tr>
<td>I2CRXDR/I2CRXFIFO</td>
<td>1001</td>
<td>I²C Receiving Data Register/FIFO</td>
<td>8/10</td>
<td>Both</td>
<td>R</td>
</tr>
<tr>
<td>I2CGCDR</td>
<td>1010</td>
<td>I²C General Call Information Register</td>
<td>8</td>
<td>Both</td>
<td>R</td>
</tr>
<tr>
<td>I2CSR/I2CFIFORSR</td>
<td>1011</td>
<td>I²C Status Register/FIFO Status Register</td>
<td>8/10</td>
<td>Both</td>
<td>R</td>
</tr>
<tr>
<td>I2CINTSR/I2CFIFOINTSR</td>
<td>1100</td>
<td>I²C Interrupt Status Register/FIFO Interrupt Status Register</td>
<td>8/10</td>
<td>Both</td>
<td>R</td>
</tr>
<tr>
<td>I2CFIFOSMSR</td>
<td>1101</td>
<td>I²C FIFO State Machine Status Register</td>
<td>10</td>
<td>FIFO mode</td>
<td>R</td>
</tr>
<tr>
<td>I2CFIFOTXCNTR</td>
<td>1110</td>
<td>I²C TXFIFO Byte Counter</td>
<td>10</td>
<td>FIFO mode</td>
<td>R</td>
</tr>
<tr>
<td>I2CFIFORXCNTR</td>
<td>1111</td>
<td>I²C RXFIFO Byte Counter</td>
<td>10</td>
<td>FIFO mode</td>
<td>R</td>
</tr>
</tbody>
</table>

4.1. I²C Control Register 1 (I2CCR1)

The I2CCR1 register can be read or written through System Bus. A write operation to this register, through System Bus will cause the I²C core to reset.

### Table 4.2. I²C Control Register 1 (I2CCR1)

<table>
<thead>
<tr>
<th>I2CCR1</th>
<th>Bit</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
<th>Default</th>
<th>0 to Disable</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I2CEN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00</td>
<td>Yes</td>
<td>R/W</td>
</tr>
<tr>
<td>GCEN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00</td>
<td>Yes</td>
<td>R/W</td>
</tr>
<tr>
<td>WKOPEN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00</td>
<td>Yes</td>
<td>R/W</td>
</tr>
<tr>
<td>FIFO_MODE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00</td>
<td>Yes</td>
<td>R/W</td>
</tr>
<tr>
<td>SDA_DEL_SEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00</td>
<td>Yes</td>
<td>R/W</td>
</tr>
<tr>
<td>CLKDIS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>00</td>
<td>Yes</td>
<td>R/W</td>
</tr>
</tbody>
</table>

Note: A write to I²C Control Register 1 will cause the I²C core to reset.

**I2CEN**

I²C System Enable Bit – This bit enables the I²C core functions. If I2CEN is cleared, the I²C core is disabled and forced into idle state.

0: I²C Disable
1: I²C Enable

**GCEN**

Enable bit for General Call Response – Enables the general call response in Slave mode.

0: Disable
1: Enable

The General Call address is defined as 0000000 and works with either 7-bit or 10-bit addressing.
WKUPEN  
Wake-up from Standby/Sleep (by Slave Address matching) Enable Bit – When this bit is enabled, the I²C core can send a wake-up signal to wake the device up from standby/sleep. The wake-up function is activated when the Slave Address is matched during standby/sleep mode.
0: Wakeup by Slave address matching is disabled
1: Wakeup by Slave address matching is enabled

FIFO_MODE  
Choose between using FIFO or Register modes.
0: Register mode (default)
1: FIFO mode

SDA_DEL_SEL[1:0]  
SDA Output Delay Selection. These two bits select the output delay (in Number of system bus clk cycles). The Base Delay is set by MSB of the I2CBRMSB.
00: NDelay = 4 * NBase_Delay + 3 (when NBase_Delay = 0, NDelay = 1)
01: NDelay = 2 * NBase_Delay + 3 (when NBase_Delay = 0, NDelay = 1)
10: NDelay = 1 * NBase_Delay + 3 (when NBase_Delay = 0, NDelay = 1)
11: NDelay = 0

CKSDIS  
Clock Stretching Disable Option (FIFO Mode)
Disable the clock stretching in FIFO mode if desired by user for both Master and Slave mode. Then overflow error flag must be monitored.
0: Clock Stretching is Enabled
1: Clock Stretching is Disabled

4.2. I²C Command Register (I2CCMDR)

The I2CCMDR register can be read or written through System Bus in Register mode. The RBUFDIS bit in the I2CCMDR register is always at default value (0) for FIFO mode.

Table 4.3. I²C Command Register (I2CCMDR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
<th>Reserved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>STA</td>
<td>STO</td>
<td>RD</td>
<td>WR</td>
<td>ACK</td>
<td>CKSDIS</td>
<td>RBUFDIS</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0 to Disable</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>—</td>
<td>—</td>
<td>No</td>
<td>No</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Access</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

STA  
Generate START (or Repeated START) condition (Master operation)

STO  
Generate STOP condition (Master operation)

RD  
Indicate Read from Slave (Master operation)

WR  
Indicate Write to Slave (Master operation)

ACK  
Acknowledge Option – when receiving, ACK transmission selection
0: Send ACK
1: Send NACK

CKSDIS  
Clock Stretching Disable (Register Mode) – Disables the clock stretching if desired by the user for both Master and Slave mode.
0: Enable Clock Stretching
1: Disable Clock Stretching

RBUFDIS  
Read Command with Buffer Disable – 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
4.3. I²C Clock Pre-scale Register (I2CBRMSB/I2CBRLSB)

The I2CBR register can be read or written through System Bus. Two System Bus writes or reads are required to access the I2CBR at different System Bus address. One address is for I2CBRLSB [7:0] and second address is for I2CBRMSB [7:0]. A write operation through System Bus to either I2CBRLSB or I2CBRMSB will cause the I²C core to reset.

Table 4.4. I²C Clock Pre-scale Register (I2CBRLSB)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Default</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit7</td>
<td>I2C_PRESCALE[7:0]</td>
<td>0000000</td>
<td>R/W</td>
</tr>
<tr>
<td>Bit6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bit0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

If the I2C_PRESCALE[9:0] value is ZERO, then a default value (specified inside the i2c_defines.v) will be taken to set the FSCL to 400 kHz. The default value should be set according to a default fabric clock frequency. The System Bus clock frequency is divided by \((4*(I2C_PRESCALE+1))\) to produce the Master I²C clock frequency supported by the I²C bus.

The I2CBRMSB [7:4] is utilized for trimming the Base Delay which is combined with I2CCR1[3:2] to achieve the SDA output delay to meet the I²C Specification requirement (300 ns).

4.4. I²C Status Register (I2CSR/I2CFIFOSR)

This address is shared by both Register mode and FIFO mode. However, the definition of each status bit is different for each mode.

Table 4.6. I²C Status Register (I2CSR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Default</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit7</td>
<td>TIP</td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Bit6</td>
<td>BUSY</td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Bit5</td>
<td>RARC</td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Bit4</td>
<td>SRW</td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Bit3</td>
<td>ARBL</td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Bit2</td>
<td>TRRDY</td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Bit1</td>
<td>TROE</td>
<td></td>
<td>R</td>
</tr>
<tr>
<td>Bit0</td>
<td>HGC</td>
<td></td>
<td>R</td>
</tr>
</tbody>
</table>

TIP  
Transmitting In Progress - This bit indicates that current data byte is being transferred for both Master and Slave mode. Note that the TIP flag will suffer half SCL cycle latency right after the start condition because of the signal synchronization. Note also that this bit could be high after configuration wake-up and before the first valid I²C 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

BUSY  
Bus Busy – This bit indicates the bus is involved in transaction. This will be 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.
**RARC**  
Received Acknowledge – This flag represents acknowledge response from the addressed Slave during Master write or from receiving Master during Master read.  
0: No Acknowledge received  
1: Acknowledge received

**SRW**  
Slave RW  
0: Master transmitting/Slave receiving  
1: Master receiving/Slave transmitting

**ARBL**  
Arbitration Lost – This bit goes high if Master has lost its arbitration in Master mode. It will cause an interrupt to System Bus Host if system bus interrupts are enabled.  
0: Normal  
1: Arbitration Lost

**TRRDY**  
Transmitter or Receiver Ready Bit – This flag indicate that a Transmit Register ready to receive data or Receiver Register if ready for read depend on the mode (Master or Slave) and SRW bit. It will cause an interrupt to System Bus Host if system bus interrupts are enabled.  
0: Transmitter or Receiver is not ready  
1: Transmitter or Receiver is ready

**TROE**  
Transmitter/Receiver Overrun or NACK Received Bit – This flag indicate 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 will cause an interrupt to System Bus Host if system bus interrupts are enabled.  
0: Transmitter or Receiver Normal or Acknowledge Received for Transmitting  
1: Transmitter or Receiver Overrun or No-Acknowledge Received for Transmitting

**HGC**  
Hardware General Call Received – This flag indicate that a hardware general call is received from the Slave port. It will cause an interrupt to System Bus Host if SCI setup is allowed.  
0: No Hardware General Call Received in Slave Mode  
1: Hardware General Call Received in Slave Mode

### Table 4.7. I2C FIFO Status Register (I2CFIFOSR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit9</th>
<th>Bit8</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Reserved</td>
<td>HGC</td>
<td>RNACK</td>
<td>MRDCMPL</td>
<td>ARBL</td>
<td>TXSERR</td>
<td>TXUNDERF</td>
<td>RXOVERF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Access</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
</tbody>
</table>

**HGC**  
Hardware General Call Received – This flag indicate that a hardware general call is received from the Slave port. It will cause an interrupt to System Bus Host if SCI setup is allowed.  
0: No Hardware General Call Received in Slave Mode  
1: Hardware General Call Received in Slave Mode

**RNACK**  
Received NACK – This flag represents acknowledge response from the addressed Slave during Master write.  
0: Acknowledge received  
1: No Acknowledge (NACK) is received, FIFO state machine issues a STOP and go to idle state.
MRDCMPL  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
  1) the number of bytes read equals to the expected number,
  2) Master terminates the read earlier but there is data in the RX FIFO.

ARBL  Arbitration Lost – This bit goes high if the Master has lost its arbitration in Master mode.
0: Normal
1: Arbitration Lost, FIFO state machine goes to idle state.

TXSERR  TX FIFO synchronization error. This happens when there are back-to-back commands in the FIFO.
0: No synchronization error
1: Synchronization error, the previous command is overwritten, then continues with the next data entry in
   the FIFO.

TXUNDERF  TX FIFO underflow – This indicates an error condition, mutually exclusive with clock stretching function.
0: No underflow
1: FIFO underflow, data is not valid

RXOVERF  RX FIFO overflow – This indicates an error condition, mutually exclusive with clock stretching function.
0: No overflow
1: FIFO overflow, data is not valid

4.5. I2C Transmitting Data Register (I2CTXDR/I2CTXFIFO)

This address is shared by both Register mode and FIFO mode. However, the definition of each status bit is different for
each mode.

Table 4.8. I2C Transmitting Data Register (I2CTXDR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>I2C_Transmit_Data[7:0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>00000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The I2C_Transmit_Data[7:0] is I2C Transmit Data – This register holds the byte that will be transmitted on the I2C bus during
the Write Data phase. Bit 0 is the LSB and will be transmitted last. When transmitting the Slave address, Bit 0 represents the Read/Write bit.

Table 4.9. I2C Transmitting FIFO Register (I2CTXFIFO)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit9</th>
<th>Bit8</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>CMD</td>
<td>RSTAEN/LTXBYTE</td>
<td>RXBYTE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The I2CTXFIFO is write only. However a read to this location during FIFO mode will cause the I2CTXFIFO to be reset
(reset the pointers). The 2 MSBs are the command bits, while the 8 LSBs are for data or Slave address. The 8-bit data
can be interpreted differently depending on the value of Bit9.

The CMD bit and the RSTAEN/LTXBYTE bit are used when the IP is in Master mode. When CMD=1, the range of Bits[4:0]
is 0 to 31, where a “0” indicates receiving 1 byte, a “1” receiving 2 bytes, and a “31” receiving 32 bytes,. Therefore, an
I2C Read must receive at least 1 byte.
CMD, RSTAEN
10: Bits [4:0] of this byte is the number of bytes to be received (in Master mode).
Following data transaction should be sent using a STOP then a START.
11: Bits [4:0] of this byte is the number of bytes to be received (in Master mode).
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 will either go into clock stretching (CKSDIS=0), or TXFIFO will underflow
(CKSDIS=1).
01: Bits [7:0] of this byte are data bits. If this is the last data byte in TXFIFO, this indicates the last byte to
be transferred and a STOP will be issued. If this is not the last byte in TXFIFO, then this bit is ignored.

RXBYTE[7:5] Not used when CMD=1
Data byte when CMD=0
RXBYTE[4:0] RXBYTE value when CMD=1
Data byte when CMD=0

In Master mode, if users want to abort the current transaction, they should reset the TXFIFO (by issuing a read to
TXFIFO or use FIFO_RST signal).
When the TXFIFO is reset while the state machine is in transmit mode, it will issue a STOP after the current byte is
transmitted.
When the TXFIFO is reset and the state machine is in receive mode, it will issue a NACK+STOP. This is to make sure the
I2C bus is appropriately released.

4.6. I2C Receiving Data Register (I2CRXDR/I2CRXFIFO)

This address is shared by both Register mode and FIFO mode. However, the definition of each status bit is different for
each mode.

Table 4.10. I2C Receiving Data Register (I2CRXDR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>I2C_Receive_Data[7:0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>—</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

I2C_Receive_Data[7:0] I2C Receive Data – This register holds the byte captured from the I2C bus during the Read Data phase.
Bit 0 is LSB and received last.

Table 4.11. I2C Receiving FIFO Register (I2CRXFIFO)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit9</th>
<th>Bit8</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Reserved</td>
<td>DFIRST</td>
<td>DATA[7:0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Access</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
</tbody>
</table>

The I2CRXFIFO register is read only. However a write to this location during FIFO mode will cause the I2CRXFIFO to
reset (reset the pointers).

DFIRST Last byte of data
0: Normal data
1: First byte received after a Start or a ReStart is detected

DATA[7:0] Data received
4.7. I2C General Call Data Register

Table 4.12. I2C General Call Data Register (I2CGCDR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>I2C_GC_Data[7:0]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Default</td>
<td>—</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

I2C_GC_Data[7:0] – I2C General Call Data – This register holds the second (command) byte of the General Call transaction on the I2C bus.

4.8. I2C Slave Address MSB Register (I2CSADDR/I2CFIFOSADDR)

This address is shared by both Register mode and FIFO mode. However, the definition of each status bit is different for each mode.

Table 4.13. I2C Slave Address MSB Register (I2CSADDR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Bits Addressing</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>A6</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
</tr>
<tr>
<td>10 Bits Addressing</td>
<td>A9</td>
<td>A8</td>
<td>A7</td>
<td>A6</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
</tr>
<tr>
<td>Default</td>
<td>00000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.14. I2C Slave Address MSB Register (I2CFIFOSADDR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit9</th>
<th>Bit8</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 Bits Addressing</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>A6</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
<td>A1</td>
<td>A0</td>
</tr>
<tr>
<td>10 Bits Addressing</td>
<td>A9</td>
<td>A8</td>
<td>A7</td>
<td>A6</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
<td>A1</td>
<td>A0</td>
</tr>
<tr>
<td>Default</td>
<td>00000000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td>R/W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.9. I2C Interrupt Control Register (I2CINTCR/I2CFIFOINTCR)

This address is shared by both Register mode and FIFO mode. However, the definition of each status bit is different for each mode.

Table 4.15. I2C Interrupt Control Register (I2CINTCR)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>INTCLREN</td>
<td>INTFRC</td>
<td>Reserved</td>
<td>Reserved</td>
<td>ARBLEN</td>
<td>TRRDYEN</td>
<td>TROEEN</td>
<td>HGCEN</td>
</tr>
<tr>
<td>Default</td>
<td>0</td>
<td>0</td>
<td>—</td>
<td>—</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0 to Disable</td>
<td>YES</td>
<td>YES</td>
<td>—</td>
<td>—</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Access</td>
<td>R/W</td>
<td>R/W</td>
<td>—</td>
<td>—</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
</tr>
</tbody>
</table>

INTCLREN – Auto Interrupt Clear Enable – Enable the interrupt flag auto clear when the I2CINTSR has been read.


0: Normal operation
1: Force the Interrupt Request

ARBLEN – Arbitration Lost Interrupt Enable – Enable Arbitration Lost interrupt
TRRDYEN  Transmit/Receive Register Ready Interrupt Enable – Enable TRRDY interrupt
TROEEN   Transmit/Receive Register Overrun Interrupt Enable – Enable TROE interrupt
HGCEN    General Call Interrupt Enable – Enable General Call interrupt

<table>
<thead>
<tr>
<th>Name</th>
<th>I2CFIFOINTCR (FIFO Mode)</th>
<th>Bit</th>
<th>Bit9</th>
<th>Bit8</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0 to Disable</td>
<td></td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Access</td>
<td></td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
</tr>
</tbody>
</table>

- **INTCLREN**  Auto Interrupt Clear Enable – Enable the interrupt flag auto clear when the I2CINTSR been read.
- **INTFRC**    Force Interrupt Request On – Force the Interrupt Flag set to improve testability.
  0: Normal operation
  1: Force the Interrupt Request
- **HGCEN**     General Call Interrupt Enable – Enable General Call interrupt
- **RNACKEN**   Receive NACK Interrupt Enable
- **MRDCMPLEN** Master Read Complete Enable
- **ARBLEN**    Arbitration Lost Interrupt Enable – Enable Arbitration Lost Interrupt
- **TXSERREN**  TX FIFO Synchronization Error Interrupt Enable
- **TXUNDERFEN** TXFIFO Underflow Interrupt Enable
- **RXOVERFEN** RXFIFO Overflow Interrupt Enable

### 4.10. I²C Interrupt Status Register (I2CINTSR/I2CFIFOINTSR)

This address is shared by both Register mode and FIFO mode. However, the definition of each status bit is different for each mode. A System Bus write to this register with a particular bit set will cause the corresponding interrupt request flags cleared. If Bit7 of I2CINTCR, or Bit9 of I2CFIFOINTCR is set, then a read operation on the Interrupt Status Register will clear all the interrupt status flags.

| Name        | I2CINTSR (Register Mode) | Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|-------------|--------------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Default     |                          | —   | —   | —   | —   | —   | —   | —   | —   | —   | —   |
| Access      |                          | —   | —   | —   | —   | —   | —   | R/W | R/W | R/W | R/W |

- **ARBL**     Arbitration Lost Interrupt Status Flag
- **TRRDY**    Transmit/Receive Register Ready Interrupt Status Flag
- **TROE**     Transmit/Receive Register Overrun Interrupt Status Flag
- **HGC**      General Call Interrupt Status Flag

<table>
<thead>
<tr>
<th>Name</th>
<th>I2CFIFOINTSR (FIFO Mode)</th>
<th>Bit</th>
<th>Bit9</th>
<th>Bit8</th>
<th>Bit7</th>
<th>Bit6</th>
<th>Bit5</th>
<th>Bit4</th>
<th>Bit3</th>
<th>Bit2</th>
<th>Bit1</th>
<th>Bit0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td></td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Access</td>
<td></td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
</tr>
</tbody>
</table>

All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.
HGC: General Call Interrupt Status Flag
RNACK: NACK Interrupt Status Flag
MRDCMPL: Master Read Completion Interrupt Status Flag
ARBL: Arbitration Lost Interrupt Status Flag
TXSERR: TXFIFO Synchronization Error Interrupt Status Flag
TXUNDERF: TXFIFO Underflow Interrupt Status Flag
RXOVERF: RXFIFO Overflow Interrupt Status Flag

4.11. I²C FIFO Threshold Register (I2CFIFOTHRESHOLD)
This register stores the FIFO threshold values. This is a read and write register used in FIFO mode only.

Table 4.19. I²C FIFO Threshold Register (I2CFIFOTHRESHOLD)

<table>
<thead>
<tr>
<th>I2CFIFOTHRESHOLD (FIFO mode)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Default</td>
</tr>
<tr>
<td>Access</td>
</tr>
</tbody>
</table>

RXFIFO_AF_VAL: 5-bit Almost Full value for the RX FIFO.
TXFIFO_AE_VAL: 5-bit Almost Empty value for the TX FIFO.

4.12. I²C FIFO TX Byte Counter (I2CFIFOTXCNT)
This is a read only register. It stores the current count of data bytes that have been transmitted to the I²C port. The number of bytes is accumulative until the counter is cleared. A write to this register or assertion of FIFO_RST signal will cause the counter to be cleared.

Table 4.20. I²C FIFO TX Byte Counter (I2CFIFOTXCNT)

<table>
<thead>
<tr>
<th>I2CFIFOTXCNT (FIFO mode)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Default</td>
</tr>
<tr>
<td>Access</td>
</tr>
</tbody>
</table>

TX_BYTE_CNT: The number of data bytes that have been transmitted to the I²C port.

4.13. I²C FIFO RX Byte Counter (I2CFIFORXCNT)
This is a read only register. It stores the current count of data bytes that have been received at the RXFIFO. The number of bytes is accumulative until the counter is cleared. A write to this register or assertion of FIFO_RST signal will cause the counter to be cleared.

Table 4.21. I²C FIFO RX Byte Counter (I2CFIFORXCNT)

<table>
<thead>
<tr>
<th>I2CFIFORXCNT (FIFO mode)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Default</td>
</tr>
<tr>
<td>Access</td>
</tr>
</tbody>
</table>

RX_BYTE_CNT: The number of data bytes that have been received at the RX FIFO.
5. **I^2C Read/Write Flowchart**

Figure 5.1 shows a flow diagram for controlling Master I^2C reads and writes initiated via the System Bus interface.

![Flowchart Diagram]

*Figure 5.1. I^2C Master Read/Write Example (via System Bus)*

*Real-Time Delay Requirement*

- Read only 1 byte: \( \min < \text{wait} < \max \)
- Read last of 2+ bytes: \( 0 < \text{wait} < \max \)

where:
\[
\begin{align*}
\min &= 2 \times \left( \frac{1}{f_{\text{OSC}}} \right) \\
\max &= 7 \times \left( \frac{1}{f_{\text{OSC}}} \right)
\end{align*}
\]
Figure 5.2 shows a flow diagram for reading and writing from an I²C Slave device via the System Bus interface.

Figure 5.2. I²C Slave Read/Write Example (via System Bus)
6. **I²C Functional Waveforms**

**Figure 6.1. Master – I²C Write**

**Figure 6.2. Master I²C Read**

**Figure 6.3. Slave I²C Write**

**Figure 6.4. Slave I²C Read**
References
For more information, refer to the following documents:
- CrossLink Family Data Sheet (FPGA-DS-02007)
- CrossLink High-Speed I/O Interface (FPGA-TN-02012)
- CrossLink Hardware Checklist (FPGA-TN-02013)
- CrossLink Programming and Configuration Usage Guide (FPGA-TN-02014)
- CrossLink sysCLOCK PLL/DLL Design and Usage Guide (FPGA-TN-02015)
- CrossLink sysI/O Usage Guide (FPGA-TN-02016)
- CrossLink Memory Usage Guide (FPGA-TN-02017)
- Power Management and Calculation for CrossLink Devices (FPGA-TN-02018)
- CrossLink I2C Hardened IP Usage Guide (FPGA-TN-02019)
Technical Support Assistance
Submit a technical support case through www.latticesemi.com/techsupport.
## Revision History

### Revision 1.1, December 2019

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disclaimers</td>
<td>Added this section.</td>
</tr>
<tr>
<td>Revision History</td>
<td>Updated format.</td>
</tr>
</tbody>
</table>

### Revision 1.0, August 2016

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Updated document numbers.</td>
</tr>
</tbody>
</table>

### Revision 1.0, May 2016

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>First preliminary release.</td>
</tr>
</tbody>
</table>