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 ........................................................................................................................................... 4
2. I2C IP Core Overview .......................................................................................................................... 4
3. Generating I2C Module with Clarity Designer ......................................................................................... 5
  3.1. Hard IP Enables ............................................................................................................................. 5
  3.2. I2C Settings .................................................................................................................................. 6
       3.2.1. General Settings ..................................................................................................................... 6
       3.2.2. Master Clock Rate ................................................................................................................ 7
       3.2.3. FIFO Mode ............................................................................................................................ 8
       3.2.4. I2C Addressing ..................................................................................................................... 8
       3.2.5. PMU Wake Options ................................................................................................................ 8
  3.3. I2C Interrupts ................................................................................................................................ 9
       3.3.1. System Bus Interrupts .......................................................................................................... 10
       3.3.2. FIFO Interrupts ..................................................................................................................... 10
4. SB_I2C Hardened IP Macro Ports and Wrapper Connections ................................................................. 12
5. I2C Usage Cases ................................................................................................................................ 13
References ................................................................................................................................................ 14
Technical Support Assistance .................................................................................................................... 15
Revision History ........................................................................................................................................ 16

Figures

Figure 3.1. I2C Module in Clarity Designer ................................................................................................. 5
Figure 3.2. Enabling Hard I2C IP Blocks ................................................................................................. 5
Figure 3.3. I2C Settings .......................................................................................................................... 6
Figure 3.4. I2C I/O Buffer ....................................................................................................................... 7
Figure 3.5. I2C I/O Connections ............................................................................................................... 7
Figure 3.6. I2C Interrupts ....................................................................................................................... 9
Figure 5.1. I2C Circuit ............................................................................................................................ 13

Tables

Table 4.1. Pins for I2C Hardened IP ........................................................................................................ 12
1. Introduction

CrossLink™ from Lattice Semiconductor is a programmable video bridging device that supports a variety of protocols and interfaces for mobile image sensors and displays. The device is based on Lattice mobile FPGA technology. It combines the extreme flexibility of an FPGA with the low power, low cost and small footprint of an ASIC.

CrossLink devices include integrated I²C blocks to interface with virtually all mobile sensors and application processors. CrossLink devices provide two I²C IP cores. The I2C0 core has dedicated I/O pins, called SCL and SDA, on the pinout spreadsheet. These pins are used for I²C programming and support the device sleep mode wakeup over I²C function. The USER_SCL and USER_SDA pins from the I2C1 core can be connected to any pin on the device.

This document provides guidance for the usage of the Lattice Semiconductor CrossLink I²C IP. It explains the steps and configuration settings required to generate the I²C module using the Lattice Diamond® Clarity Designer tool.


2. I²C IP Core Overview

The I²C hard IP provides industry standard two-pin communication interface that conforms to the I²C Bus Specification version 2.1. It may be configured in either Master or Slave mode.

In Master mode, it supports configurable data transfer rate and performs arbitration detection to allow it to operate in multi-master systems. It supports clock stretching in both Master and Slave modes with enable/disable capability.

It supports both 7 bits and 10 bits addressing in Slave mode with configurable Slave address. It supports general call address detection in both Master and Slave mode.

It provides interrupt logic for easy communicating with host. It also provides configurable digital delay at SDA output for reliably generating start/stop conditions.

The key I²C IP core features of CrossLink devices:

- Configurable Master and Slave mode
- Support for 7-bit or 10-bit configurable Slave address
- Multi-master arbitration support
- Clock stretching to ensure data setup time
- Up to 1000 kHz data transfer speed, also supports 400 kHz, 100 kHz, 50 kHz modes
- General call support
- Optional delaying input or output data, or both
- Optional FIFO mode
- Transmit FIFO size is 10 bits x 16 bytes, receive FIFO size is 10 bits x 32 bytes
- I2C0 core supports sleep mode wakeup over I²C (refer to Power Management and Calculation for CrossLink Devices (FPGA-TN-02018) for more details)
3. Generating I²C Module with Clarity Designer

The Clarity Designer tool of Lattice Diamond is used to configure the I²C hard IP functions and generate the I²C module.

3.1. Hard IP Enables

Follow these steps to enable Hard Users I²C blocks:

1. On the Lattice Diamond top menu, click Tools, and then Clarity Designer. With a CrossLink device targeted for the Diamond project, the Clarity Designer window appears and the I²C module is listed under Module > Architecture_Modules as shown in Figure 3.1.

2. Click i2c. The Lattice FPGA module – I2C window appears as shown in Figure 3.2.

3. On the Hard IP Enables tab, select the desired I2C core (I2C0 or I2C1, or both), and enter your System Bus clock frequency. The I2C Settings and I2C Interrupts tabs are enabled when at least one of the Enable Hard Users I2C block is selected.
3.2. I2C Settings

This section describes the I2C Settings tab of Lattice FPGA Module – I2C window as shown in Figure 3.3.

![I2C Settings](image)

**Figure 3.3. I2C Settings**

3.2.1. General Settings

3.2.1.1. General Call Enable

Select this checkbox to enable I2C General Call response (addresses all devices on the bus using the I2C address 0) in Slave mode. This setting can be modified dynamically by enabling the GCEN bit in the I2C Control Register I2CCR1. Refer to Advanced CrossLink I2C Hardened IP Reference Guide (FPGA-TN-02020) for register descriptions.

3.2.1.2. Include I/O Buffers

Select this checkbox to include buffers for the I2C pins. Note that I2C0 pins are hard routed through I/O buffer to the dedicated pins on the device. These buffers need to be included with I2C0.

*Figure 3.4 and Figure 3.5 show the I2C I/O buffer and connections, respectively.*
3.2.1.3. **SDA Input Delay**
Select SDA Input Delay between 0 ns and 50 ns.

3.2.1.4. **SDA Output Delay**
Select SDA Output Delay between 0 ns and 350 ns.

3.2.2. **Master Clock Rate**

3.2.2.1. **Master Clock Desired**
In the Master Clock Desired field you can specify a desired Master clock frequency (in kHz). A calculation is then made to determine a divider value to generate a clock close to this value from the input clock. The frequency of the input System Bus clock is specified on the **Hard IP Enables tab**. The divider value is rounded to the nearest integer after dividing the input System Bus clock by the value entered in this field.
3.2.2.2. Master Clock Actual
This is a read-only field. It is not always possible to divide the input System Bus clock to the exact value requested by the user, therefore the actual value (in kHz) is returned in this field. When both the desired I2C clock and System Bus clock fields contain valid data and either is updated, the Master Clock field returns the value (FREQ_SB/I2C_CLK_DIVIDER), rounded to an integer as shown in the example below. FREQ_SB is the System Bus clock frequency that has been entered on the Hard IP Enables tab. I2C_CLK_DIVIDER is a factor to be calculated as in the example below.

Master Clock calculation example:
- Divider = I2C_CLK_DIVIDER = FREQ_SB/I2C_CLK_FREQ
- Master Clock (Actual) = FREQ_SB/Divider Integer

For example, if FREQ_SB = 42.5 MHz and I2C_BUS_PERF = 400 kHz.
Diviner = 42500/400 = 106.25
Therefore, ROUND Divider Integer = 106
Actual frequency = 42500/106 = 400.9 kHz

The tool automatically calculates and displays the Actual I2C Master clock rate value based on the System Bus Frequency and Desired Master Clock rate selected.

In this example, if an I2C_CLK_FREQ of 400 kHz cannot be generated, you may use the actual value or change the System Bus clock frequency.

3.2.3. FIFO Mode

3.2.3.1. Enable FIFO Mode
Select this option to enable FIFO mode operation.

3.2.3.2. CLK Stretch Enable
Select this checkbox to enable Clock Stretching for FIFO mode. This setting may be used to ensure valid data is transmitted from the FIFO. If disabled, then overflow and underflow error flag must be monitored. The Clock Stretch for Register mode is a separate control and can be enabled or disabled through the "CKSDIS" (bit2) inside I2CCMDR register. The I2CCMDR register can be directly accessed through the System Bus. Refer to Advanced CrossLink I2C Hardened IP Reference Guide (FPGA-TN-02020) for more details.

3.2.3.3. RxFIFO Almost Full
Select the value from the list, as to when the RxFIFO’s Almost Full Flag must be triggered. The maximum FIFO size of the RxFIFO is 32 bytes.

3.2.3.4. TxFIFO Almost Empty
Select the value from the list, as to when the TxFIFO’s Almost Empty Flag must be triggered.

3.2.4. I2C Addressing
In the I2C Address box, you can choose between 7-bit or 10-bit addressing, and define the Hard I2C address.
The lower two significant bits of the I2C address are fixed for each core. You can set the upper 5 significant bits in case of 7-bit addressing, or the upper 8 significant bits in case of 10-bit addressing.

3.2.5. PMU Wake Options
This option is available for I2C0 only to enable the Wakeup port. The WKUPEN bit in the I2CCR1 can be modified dynamically allowing the Wake Up function to be enabled or disabled.

3.2.5.1. Address Match
Select this checkbox to turn on the I2C wakeup on address match.
3.2.5.2. RxFIFO Almost Full
Select this checkbox to turn on the I²C wakeup on RxFIFO’s Almost Full Flag assertion.

3.3. I²C Interrupts
This section describes the I²C Interrupts tab of Lattice FPGA Module – I²C window as shown in Figure 3.6.

![Lattice FPGA Module – I²C window showing I²C Interrupts](image)

**Figure 3.6. I²C Interrupts**
3.3.1. System Bus Interrupts
System Bus interrupts enable the interrupt port when any of the interrupts are enabled. Refer to Advanced CrossLink I2C Hardened IP Reference Guide (FPGA-TN-02020) for more details on the interrupt registers described below.

3.3.1.1. Arbitration Lost
This interrupt indicates I^2C lost arbitration. This interrupt is bit ARBL of register I2CINTSR. When enabled, it indicates that ARBL is asserted. Writing 1 to this bit clears the interrupt.
This option can be changed dynamically by modifying bit ARBLEN in register I2CINTCR.

3.3.1.2. Tx/Rx Ready
This interrupt indicates that the I^2C transmit data register (I2CTXDR) is empty or that the receive data register (I2CRXDR) is full. This interrupt is bit TRRDY of register I2CINTSR. When enabled, it indicates that TRRDY is asserted. Writing 1 to this bit clears the interrupt.
This option can be changed dynamically by modifying bit TRRDYEN in register I2CINTCR.

3.3.1.3. Overrun or NACK
This interrupt indicates that the I2CRXDR received new data before the previous data. This interrupt is bit TROE of register I2CINTSR. When enabled, it indicates that TROE is asserted. Writing 1 to this bit clears the interrupt.
This option can be changed dynamically by modifying bit I2CINTSR in register I2CINTCR.

3.3.1.4. General Call IRQ Enable
This interrupt indicates that a general call has occurred. This interrupt is bit HGC of register I2CINTSR. When enabled, it indicates that HGC is asserted. Writing 1 to this bit clears the interrupt.
This option can be changed dynamically by modifying bit HGcen in register I2CINTCR.

3.3.2. FIFO Interrupts

3.3.2.1. General Call
This interrupt indicates that a general call has occurred. This interrupt is bit HGC of register I2CFIFOINTSR. This bit can be changed by modifying bit HGcen in register I2CFIFOINTCR.

3.3.2.2. Receive NACK
This interrupt indicates that a NACK has been received. This interrupt is bit RNACK of register I2CFIFOINTSR. This bit can be changed by modifying bit RNACKEN in register I2CFIFOINTCR.

3.3.2.3. Master Read Complete
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
This is bit MRDCMPL of register I2CFIFOINTSR. This bit can be changed by modifying bit MRDCMPLen in register I2CFIFOINTCR.

3.3.2.4. Arbitration Lost
This interrupt indicates I2C lost arbitration. This interrupt is bit ARBL of register I2CFIFOINTSR. When enabled, it indicates that ARBL is asserted. Writing 1 to this bit clears the interrupt.
This option can be changed dynamically by modifying bit ARBLEN in register I2CFIFOINTCR.

3.3.2.5. TX FIFO SYNC
This interrupt indicates I2C TXFIFO synchronization error. Synchronization error happens when there are back-to-back commands in the FIFO. The previous command is overwritten. This interrupt is bit TXSERR of register I2CFIFOINTSR. This option can be changed dynamically by modifying bit TXSREN in register I2CFIFOINTCR.
3.3.2.6. TX FIFO Underflow
This interrupt is mutually exclusive with clock stretching function. This interrupt is bit TXUNDERF of register I2CFIFINTSR.
This option can be changed dynamically by modifying bit TXUNDERFEN in register I2CFIFINTCR.

3.3.2.7. RX FIFO Overflow
This interrupt is mutually exclusive with clock stretching function. This interrupt is bit RXOVERF of register I2CFIFINTSR.
This option can be changed dynamically by modifying bit RXOVERFEN in register I2CFIFINTCR.
4. SB_I2C Hardened IP Macro Ports and Wrapper Connections

When the I2C Hardened IP is enabled, the necessary signals are included in the generated module.

<table>
<thead>
<tr>
<th>Port Name</th>
<th>I2C Wrapper Name</th>
<th>Pin Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBCSI</td>
<td>i2cXcsi*</td>
<td>Input</td>
<td>Chip select signal. Activates the IP to allow the System Bus or fabric</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>interface to communicate with the IP.</td>
</tr>
<tr>
<td>SBCLKi</td>
<td>i2cXclki*</td>
<td>Input</td>
<td>System clock input.</td>
</tr>
<tr>
<td>SBWRI</td>
<td>i2cXwei*</td>
<td>Input</td>
<td>System read/write input. R=0, W=1</td>
</tr>
<tr>
<td>SBSTBI</td>
<td>i2cXstbi*</td>
<td>Input</td>
<td>System strobe signal. When asserted, indicates that the Slave component</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>is selected.</td>
</tr>
<tr>
<td>SBADRi[3:0]</td>
<td>i2cXdri[3:0]*</td>
<td>Input</td>
<td>System bus control registers address.</td>
</tr>
<tr>
<td>SBDATi[9:0]</td>
<td>i2cXdati[9:0]*</td>
<td>Input</td>
<td>System data input [7:0] for register mode; [9:0] for FIFO mode.</td>
</tr>
<tr>
<td>SBDATo[9:0]</td>
<td>i2cXdato[9:0]*</td>
<td>Output</td>
<td>System data output [7:0] for register mode; [9:0] for FIFO mode.</td>
</tr>
<tr>
<td>SBACKo</td>
<td>i2cXacko*</td>
<td>Output</td>
<td>System acknowledgement.</td>
</tr>
<tr>
<td>SBSRWo</td>
<td>i2cXsrdrw*</td>
<td>Output</td>
<td>Slave read/write signal. A “1” indicates a Slave transmitting (external</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Master receiving). A “0” means Slave receiving (external Master transmitting).</td>
</tr>
<tr>
<td>I2CIRQ</td>
<td>i2cXi2cirq*</td>
<td>Output</td>
<td>Interrupt request output signal of the I2C core – The intended use of this</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>signal is for it to be connected to a Master controller (such as a micro</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>controller or state machine) and request an interrupt when a specific</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>condition is met.</td>
</tr>
<tr>
<td>I2CPKUP</td>
<td>i2cXicwkp*</td>
<td>Output</td>
<td>Wake-up signal – The signal is enabled only if the Wakeup Enable feature</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>is set.</td>
</tr>
<tr>
<td>FIFO_RST</td>
<td>i2cXiforst*</td>
<td>Input</td>
<td>Reset for the FIFO logic.</td>
</tr>
<tr>
<td>TXFIFO_AE</td>
<td>i2cXtxfioae*</td>
<td>Output</td>
<td>TXFIFO almost empty status signal coming from TXFIFO, indicating user-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>defined almost empty threshold value is reached.</td>
</tr>
<tr>
<td>TXFIFO_E</td>
<td>i2cXtxfioe*</td>
<td>Output</td>
<td>TXFIFO empty signal coming from TXFIFO.</td>
</tr>
<tr>
<td>TXFIFO_F</td>
<td>i2cXtxfiof*</td>
<td>Output</td>
<td>TXFIFO full signal.</td>
</tr>
<tr>
<td>RXFIFO_E</td>
<td>i2cXrxfioe*</td>
<td>Output</td>
<td>RXFIFO is empty. It can be served as an active low DATA RDY signal.</td>
</tr>
<tr>
<td>RXFIFO_AF</td>
<td>i2cXrxfioaf*</td>
<td>Output</td>
<td>RXFIFO almost full signal, indicating user-defined almost full threshold</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>value is reached.</td>
</tr>
<tr>
<td>RXFIFO_F</td>
<td>i2cXrxfiof*</td>
<td>Output</td>
<td>RXFIFO full signal.</td>
</tr>
<tr>
<td>MRDCMPL</td>
<td>i2cXmrccmpl*</td>
<td>Output</td>
<td>Master Read Complete – This is only valid for Master Read Mode. A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>transaction is considered complete when: The specified number of data bytes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>from the Slave has been received in the RX FIFO, or The Master terminates</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the read transaction before the specified number of data bytes received</td>
</tr>
<tr>
<td>I2CX_SCL*</td>
<td>i2cXscl*</td>
<td>Bi-directional</td>
<td>Open drain clock line of the I2C core – The signal is an output if the I2C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>core is performing a Master operation. The signal is an input for Slave</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>operations. The “I2C0_SCL” signal (Right I2C) uses dedicated I/O pin only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The “I2C1_SCL” signal (Left I2C) can use any of the available GPIOs on the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>device.</td>
</tr>
<tr>
<td>I2CX_SDA*</td>
<td>i2cXsda*</td>
<td>Bi-directional</td>
<td>Open drain data line of the I2C core – The signal is an output when data</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>is transmitted from the I2C core. The signal is an input when data is</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>received into the I2C core.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The “I2C0_SDA” signal (Right I2C) uses dedicated I/O pin only.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The “I2C1_SDA” signal (Left I2C) can use any of the available GPIOs on the</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>device.</td>
</tr>
</tbody>
</table>

Note: X indicates the I2C: X=0 for the hard I/O pin I2C; X=1 for the selectable I/O pin I2C.
5. **I\(^2\)C Usage Cases**

The I\(^2\)C usage cases described below refer to Figure 5.1.

- CrossLink as I\(^2\)C Master accessing Slave I\(^2\)C devices.
- A System Bus Master is implemented in the CrossLink logic.
- I\(^2\)C devices 1, 2, and 3 are all Slave devices.
- The Master performs bus transactions to the I2C0 I2C controller to access external Slave I\(^2\)C Device 1 on Bus A.
- The Master performs bus transactions to the I2C1 I2C controller to access the external Slave I\(^2\)C Device 2 or Device 3 on Bus B.
- External Master I\(^2\)C device accessing Slave CrossLink I2C.
- The I\(^2\)C Devices 1, 2, and 3 are I2C Master devices.
- The external Master I\(^2\)C Device 1 on Bus A performs I\(^2\)C memory cycles to access the I2C0 I\(^2\)C controller using address yyyxxxxx01.
- The external Master I\(^2\)C Device 2 or 3 on Bus B performs I\(^2\)C memory cycles to access the I2C1 I\(^2\)C user with the address yyyxxxxx10.
- The System Bus Master in the CrossLink fabric must manage data reception and transmission. The System Bus Master can use interrupts or polling techniques to manage data transfer, and to prevent data overrun conditions.

![Figure 5.1. I²C Circuit](image-url)
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)
Technical Support Assistance
Submit a technical support case through www.latticesemi.com/techsupport.
## Revision History

### Revision 1.1, December 2020

<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>Introduction</td>
<td>Updated information on I2C0 dedicated I/O pins.</td>
</tr>
<tr>
<td>Generating I2C Module with Clarity Designer</td>
<td>Updated Figure 3.3. I2C Settings.</td>
</tr>
<tr>
<td>SB_I2C Hardened IP Macro Ports and Wrapper Connections</td>
<td>Updated descriptions of I2CX_SCL and I2CX_SDA ports in Table 4.1. Pins for I2C Hardened IP.</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>