



MachXO2 Soft I2C Slave with Clock Stretching

Reference Design

FPGA-RD-02092-1.2

November 2019

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. Block Diagram	5
3. Port List	6
4. Deviations from MachXO2 Hard I2C Implementation	6
4.1. Added functionality	6
4.2. Reduced functionality	7
5. Registers	7
6. Design Migration	12
6.1. EFB Identification and Parameterization	12
6.2. Soft Core installation	15
6.3. User Design Modifications	16
7. Customization	17
7.1. SDA_DEL calculation	17
7.2. I ² C Address	17
8. Special Cases	17
8.1. Concurrent EFB use	17
8.2. Multiple I ² C slaves	18
8.3. Wishbone Mux	18
9. Resource Utilization – Soft EFB I ² C	19
References	20
Technical Support Assistance	21
Revision History	22

Figures

Figure 2.1. Soft I ² C Slave	5
Figure 6.1. IPexpress Instance Example	13
Figure 6.2. HDL Instance Example	13
Figure 6.3. IPexpress I ² C Port Type Example	14
Figure 6.4. IPexpress I ² C Settings Example	14
Figure 6.5. Verilog Port and Address Example	15
Figure 6.6. VHDL Port and Address Example	15
Figure 6.7. Primary Files Added to New Implementation Example	16
Figure 8.1. WBmux21.v BLOCK DIAGRAM	18

Tables

Table 3.1. Port List	6
Table 6.1. Primary and Secondary File Sets	15

1. Introduction

The primary purpose of this Reference Design is to provide a form, fit and function equivalent, “drop-in” module to replace the Lattice MachXO2 hard I²C core in user designs which require Clock-Stretching support. This reference design has been engineered to require little or no modification to pre-existing WISHBONE bus control logic, and to provide pin-for-pin hook-up. The reference design utilizes ‘soft’ LUT resources instead of ‘hard’ logic. As this design consumes additional fabric resources, it can impact timing closure and/or design routability in certain designs. Please refer to Port List, Register Definitions, and Resource Utilization sections for more details.

I²C Clock-Stretching is an optional flow-control mechanism in the I2C specification. A slave device can use clockstretching to force a pause in an I²C transaction by holding the SCL line low (‘0’). The I²C Master is prevented from continuing the transaction until the slave releases SCL. The MachXO2 hard I2C block does not operate reliably when clock-stretching is enabled. See References section for additional information on I²C bus operation. For details regarding MachXO2 and clock-stretching support, refer to Lattice Product Change Notice PCN #10A-13 at www.latticesemi.com.

As with the original MachXO2 Embedded Function Block (EFB), the WISHBONE control port conforms to the WISHBONE Classic Bus Cycle protocol. Block cycles are not supported. Please refer to the WISHBONE specification or [Using User Flash Memory and Hardened Control Functions in MachXO2 Devices Reference Guide \(FPGA-TN-02163\)](#) for more details (see References section).

Two versions of this Reference design are provided, one containing the Primary WISHBONE registers and I²C pin names, the other containing the Secondary WISHBONE registers and I²C pin names. As the MachXO2 dedicated primary I²C port may be recovered for use as GPIO, the same device pins may be used by the Reference Design as used originally by the hard core. The same is true for the Secondary I²C port pins. Thus, no change to an existing Printed Circuit Board (PCB) is necessary.

In the rare case where multiple I²C Slaves supporting clock stretching are required, the soft core may be instantiated multiple times, and the WISHBONE ports multiplexed together externally with user circuitry (see [Special Cases](#)).

The MachXO2 is a family of ultra-low power, instant-on, non-volatile PLDs with densities from 256 to 6864 LUTs. Each device in the family contains a rich feature set, including Embedded Block RAM (EBR), Distributed RAM, User Flash Memory (UFM), Phase Locked Loops (PLLs) pre-engineered source synchronous I/O support, advanced configuration support including dual-boot capability. Hardened versions of commonly used functions such as SPI controller, I²C controller and timer/counter are contained in the Embedded Function Block (EFB). These features allow these devices to be used in low cost, high volume consumer and system applications.

2. Block Diagram

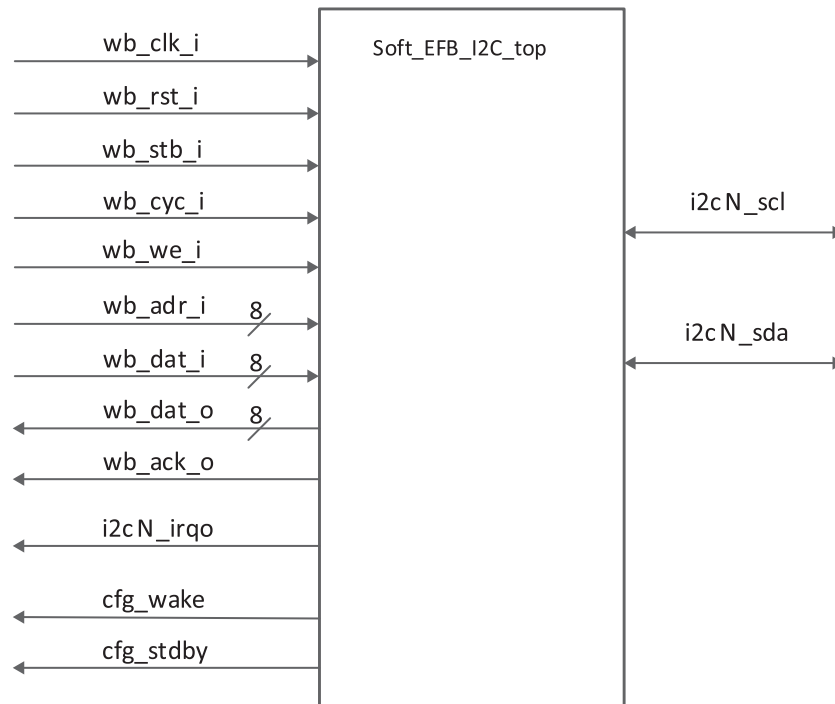


Figure 2.1. Soft I²C Slave

3. Port List

Table 3.1. Port List

Port Name	Width	Direction	BUF Type	Description
wb_clk_i	1			WISHBONE clock. All WISHBONE input signals are evaluated, and output signal updated, on the rising edge of this clock.
wb_rst_i	1	i		Module asynchronous reset. Asserted high. Both WISHBONE and I2C interfaces are reset.
wb_stb_i	1	i		WISHBONE strobe. Indicates this slave module is selected.
wb_cyc_i	1	i		WISHBONE cycle. Provided for port compatibility only. This signal is not connected internally. It may be arbitrarily driven or left floating.
wb_we_i	1	i		WISHBONE write enable. Indicates the current cycle is WRITE cycle when asserted logic '1' and is READ cycle when asserted logic '0'.
wb_adr_i	8	i		WISHBONE address. Selects target register
wb_dat_i	8	i		WISHBONE write data. Input data for write cycles.
wb_dat_o	8	o		WISHBONE read data. Output data for read cycles
wb_ack_o	1	o		WISHBONE acknowledge. Input data has been captured, or output data is valid when asserted logic '1'.
cfg_wake	1	o		Signal to wakeup for standby/sleep mode. When enabled, asserted high on I2C Slave Address match or General Call Wakeup command.
cfg_stdby	1	o		Provided for port compatibility only. This signal always drives a logic '0'.
i2c1_clk	1	i/o	open drain	Primary instance only: I2C clock
i2c1_sda	1	i/o	open drain	Primary instance only: I2C data
i2c1_irqo	1	o		Primary instance only: Interrupt, asserted high when enabled
i2c2_clk	1	i/o	open drain	Secondary instance only: I2C clock
i2c2_sda	1	i/o	open drain	Secondary instance only: I2C data
i2c2_irqo	1	o		Secondary instance only: Interrupt, asserted high when enabled

4. Deviations from MachXO2 Hard I2C Implementation

4.1. Added functionality

SDA_DEL

In the Hard IP block, SDA_DEL is set in time units, and calibrated using an internal high-speed clock reference. In this reference design, the SDA_DEL is set in WISHBONE clock periods, and calibrated using a new bit field, SDA_DEL_FACTOR. SDA_DEL_FACTOR is set in BR1 (0x43/0x4D) bits [7:4]. See [Customization](#) section for the relationships between SDA_DEL, SDA_DEL_FACTOR, and WISHBONE clock frequency.

Programmable I²C address

In the MachXO2 Hard IP block, the I2C address is set using IPexpress and is stored in the Feature Row portion of non-volatile RAM. In this reference design, the default Slave I2C address is set in the top level i2c_config.v file using a `define. In addition, a new register location (0x4A/0x54) is provided to allow dynamic R/W access to the I²C address. Lattice recommends using this feature for development or debugging purposes only.

4.2. Reduced functionality

cfg_stdby

The ability to receive and interpret the 'Enter Standby' command is reserved for the I2C Configuration block of the EFB. This Reference Design emulates the User block functionality only.

wb_cyc_i

For WISHBONE Classic bus cycles, wb_cyc_i and wb_stb_i are essentially redundant. This Reference Design utilizes only wb_stb_i.

5. Registers

Notes:

- Unless otherwise specified, all Reserved bits in writable registers shall be written '0'.
- Differences from the XO2 Hard I2C Macro are outlined in bold.

I ² C Control (Pri/Sec)								
I2C_1_CR / I2C_2_CR						0x40/0x4A		
Bit	7	6	5	4	3	2	1	0
Name	I2CEN	GCEN	WKUPEN	(Rsrv)	SDA_DEL_SEL[1:0] ¹		(Reserved)	
Default	1	0	0	0	1	0	0	0
Access	R/W	R/W	R/W	—	R/W	R/W	—	—

Note: The default value is defined by `define SDA_DEL_INIT in the file "i2c_config.v".

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

GCEN Enable bit for Generic Call Response – Enables the generic call response in slave mode.
0: Disable
1: Enable

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 the on-chip power manager to wake the device up from standby/sleep. The wake-up function is effective when the Slave Address is matched during standby/sleep mode.
0: Disable
1: Enable

SDA_DEL_SEL[1:0] SDA Output Delay (Tdel) Selection
00: 4xSDA_DEL_FACTOR + 2.5 +/- 0.5 tWB periods
01: 2xSDA_DEL_FACTOR + 2.5 +/- 0.5 tWB periods
10: 1xSDA_DEL_FACTOR + 2.5 +/- 0.5 tWB periods
11: 0.25 +/- 0.25 tWB periods

I ² C Command (Pri/Sec)								
I2C_1_CMDR / I2C_2_CMDR						0x41/0x4B		
Bit	7	6	5	4	3	2	1	0
Name	STA	STO	RD	WR	ACK	CKSDIS	(Reserved)	
Default	0	0	0	0	0	0	0	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	—	—

STA	Generate START (or Repeated START) condition.
STO	Generate STOP condition
RD	Indicate Read from slave
WR	Indicate Write to slave
ACK	Acknowledge Option – when receiving, ACK transmission selection 0: Send ACK 1: Send NACK
CKSDIS	Clock Stretching Disable. The I2C cores support “wait state” or clock stretching from slave, meaning slave can enforce a wait state if it needs time to finish the task. Bit CKSDIS disables the clock stretching if desired by the user. In that case, the overflow flag must be monitored. 0: Enabled 1: Disabled

i ² C Clock Prescale 0 (Pri/Sec)								
I2C_1_BR0 / I2C_2_BR0						0x42/0x4C		
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)							
Default	0	0	0	0	0	1	1	1
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

i ² C Clock Prescale 1 (Pri/Sec)								
I2C_1_BR1 / I2C_2_BR1						0x43/0x4D		
Bit	7	6	5	4	3	2	1	0
Name	SDA_DEL_FACTOR[3:0] ¹				(Reserved)			
Default	0	0	0	1	0	0	0	0
Access	—	—	—	—	—	—	R/W	R/W

Note: The default value is defined by `define SDA_DEL_FACTOR in the file “i2c_config.v.”

I2C_PRESCALE[9:0] I2C Clock Prescale value.

(For this Slave-only implementation, these bits are undefined and reserved)

SDA_DEL_FACTOR[3:0] SDA Delay Factor.

This 4-bit value is used in conjunction with SDA_DEL to set the SCL-to-SDA delay when XO2 is driving SDA.

I ² C Transmit Data Register (Pri/Sec)								
I2C_1_TXDR / I2C_2_TXDR						0x44/0x4E		
Bit	7	6	5	4	3	2	1	0
Name	I2C_Transmit_Data[7:0]							
Default	0	0	0	0	0	0	0	0
Access	W	W	W	W	W	W	W	W

I2C_Transmit_Data[7:0] I2C Transmit Data.

This register holds the byte that will be transmitted on the I2C bus during the Write Data phase. Bit 0 is LSB and will be transmitted last. When transmitting the slave address, Bit 0 represents the R/W bit.

I ² C Status (Pri/Sec)								
I2C_1_SR / I2C_2_SR						0x45/0x4F		
Bit	7	6	5	4	3	2	1	0
Name	TIP	BUSY	RARC	SRW	ARBL	TRRDY	TROE	HGC
Default	—	—	—	—	—	—	—	—
Access	R	R	R	R	R	R	R	R

TIP Transmit In Progress.

The current data byte is being transferred. Note that the TIP flag will suffer one-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 I2C transfer start (when BUSY is low), and it is not indicating byte in transfer, but an invalid indicator.

1: Byte transfer in progress

0: Byte transfer complete

BUSY I2C Bus busy.

The I2C bus is involved in transaction. This is set at start condition and cleared at stop. Note only when this bit is set should all other I2C SR bits be treated as valid indicators for a valid transfer.

1: I2C bus busy

0: I2C bus not busy

RARC Received Acknowledge.

An acknowledge response from the addressed slave (during master write) or from receiving master (during master read) was received.

1: No acknowledge received

0: Acknowledge received

SRW Slave R/W.

Indicates transmit or receive mode.

1: master receiving / slave transmitting

0: master transmitting / slave receiving

ARBL Arbitration Lost.

The core has lost arbitration in Master mode. This bit is capable of generating an interrupt.

1: Arbitration Lost

0: Normal

TIP Transmit In Progress.

The current data byte is being transferred. Note that the TIP flag will suffer one-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 I2C transfer start (when BUSY is low), and it is not indicating byte in transfer, but an invalid indicator.

TRRDY Transmitter or Receiver Ready.

The I2C Transmit Data register is ready to receive transmit data, or the I2C Receive Data Register contains receive data (dependent upon master/slave mode and SRW status). This bit is capable of generating an interrupt.

1: Transmitter or Receiver is ready

0: Transmitter or Receiver is not ready

TROE Transmitter/Receiver Overrun Error or NACK received.

A transmit or receive overrun error has occurred (dependent upon master/slave mode and SRW status), or a No Acknowledge was received (only when RARC also set). This bit is capable of generating an interrupt.

1: Transmitter or Receiver Overrun detected or NACK received

0: Normal

HGC Hardware General Call Received.

A hardware general call has been received in slave mode. The corresponding command byte will be available in the General Call Data Register. This bit is capable of generating an interrupt.

1: General Call Received in slave mode

0: Normal

I ² C General Call Data Register (Pri/Sec)								
I2C_1_GCDR / I2C_2_GCDR						0x46/0x50		
Bit	7	6	5	4	3	2	1	0
Name	I2C_GC_Data[7:0]							
Default	—	—	—	—	—	—	—	—
Access	R	R	R	R	R	R	R	R

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.

I ² C Receive Data Register (Pri/Sec)								
I2C_1_RXDR / I2C_2_RXDR						0x47/0x51		
Bit	7	6	5	4	3	2	1	0
Name	I2C_Receive_Data[7:0]							
Default	—	—	—	—	—	—	—	—
Access	R	R	R	R	R	R	R	R

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 was received last.

I ² C Interrupt Status (Pri/Sec)								
I2C_1_IRQ / I2C_2_IRQ						0x48/0x52		
Bit	7	6	5	4	3	2	1	0
Name	(Reserved)				IRQARBL	IRQTRRDY	IRQTROE	IRQHGC
Default	—	—	—	—	—	—	—	—
Access	—	—	—	—	R/W	R/W	R/W	R/W

IRQARBL Interrupt Status for Arbitration Lost.

When enabled, indicates ARBL was asserted. When INTCLREN='0', write a '1' to this bit to clear the interrupt. When INTCLREN='1', the interrupt is cleared automatically when this register is read.

1: Arbitration Lost Interrupt

0: No interrupt

IRQTRRDY Interrupt Status for Transmitter or Receiver Ready.

When enabled, indicates TRRDY was asserted. When INTCLREN='0', write a '1' to this bit to clear the interrupt. When INTCLREN='1', the interrupt is cleared automatically when this register is read.

1: Transmitter or Receiver Ready Interrupt

0: No interrupt

- IRQTROE** Interrupt Status for Transmitter/Receiver Overrun or NACK received.
When enabled, indicates TROE was asserted. When INTCLREN='0', write a '1' to this bit to clear the interrupt. When INTCLREN='1', the interrupt is cleared automatically when this register is read.
1: Transmitter or Receiver Overrun or NACK received Interrupt
0: No interrupt
- IRQHGC** Interrupt Status for Hardware General Call Received.
When enabled, indicates HGC was asserted. When INTCLREN='0', write a '1' to this bit to clear the interrupt. When INTCLREN='1', the interrupt is cleared automatically when this register is read.
1: General Call Received in slave mode Interrupt
0: No interrupt

I ² C Interrupt Enable (Pri/Sec)								
I2C_1_IRQEN / I2C_2_IRQEN						0x49/0x53		
Bit	7	6	5	4	3	2	1	0
Name	INTCLREN	(Reserved)			IRQARBLN	IRQTRRDYEN	IRQTROEEN	IRQHGCEN
Default	0	0	0	0	0	0	0	0
Access	—	—	—	—	R/W	R/W	R/W	R/W

- INTCLREN** Interrupt Auto Clear Enable:
1: Interrupts are cleared automatically upon a read of the I2C Interrupt Status Register
0: Interrupts are cleared manually by writing to the I2C Interrupt Status Register
- IRQARBLN** Interrupt Enable for Arbitration Lost.
1: Interrupt generation enabled
0: Interrupt generation disabled
- IRQTRRDYEN** Interrupt Enable for Transmitter or Receiver Ready
1: Interrupt generation enabled
0: Interrupt generation disabled
- IRQTROEEN** Interrupt Enable for Transmitter/Receiver Overrun or NACK received.
1: Interrupt generation enabled
0: Interrupt generation disabled
- IRQHGCEN** Interrupt Enable for Hardware General Call Received.
1: Interrupt generation enabled
0: Interrupt generation disabled

I ² C Slave Address MSB (Pri/Sec)								
I2C_1_SADR / I2C_2_SADR						0x4A/0x54		
Bit	7	6	5	4	3	2	1	0
Name	I2C_Slave_Address_MSB[7:0] ¹							
Default1	1	1	1	1	0	0	0	0
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: The default value is defined by `define IPex_I2C_slave_addr in the file "i2c_config.v". The value shown above is used when IPex_I2C_slave_addr = 0 (undefined)

I2C_Slave Address[7:0] I2C Slave Address Most Significant Bits.

I2C_Slave_Address[4:0] represent the upper 5 bits of the I2C Slave Address for 7-bit addressing.

The upper bits I2C_Slave_Address [7:5] are ignored.

I2C_Slave_Address[7:0] represent the upper 8 bits of the I2C Slave Address for 10-bit addressing.

The lower 2 LSBs of the I2C Slave Address are fixed by the Reference design: "b01" for the Primary instance and "b10" for the Secondary instance.

Note: Lattice recommends to avoid modifying this register, except for development or debugging purposes.

6. Design Migration

6.1. EFB Identification and Parameterization

It is assumed that you are familiar with your FPGA design and that you know the name and location of the EFB instance within this design. The following instructions assume the simple case where the EFB was used to instantiate a single I²C Slave core and will be replaced entirely by the Soft I²C Slave reference design. For more complicated EFB use, see the [Special Cases](#) section.

There are two common instantiations within a design implementation: IPexpress (.ipx), and HDL (.v or .vhd). See [Figure 6.1.](#) and [Figure 6.2.](#), respectively, for example instantiations (module names are user-defined and the filenames are for example only). Both instantiation types can be examined to verify:

- Primary or Secondary I²C port type
- I²C Slave address
- General Call Enable
- Wake-up Enable

These values will be needed to instantiate and customize the Soft I²C Slave reference design. Other settings, such as I²C Bus Performance and Clock Prescale Value, are not required for a Slave I²C implementation.

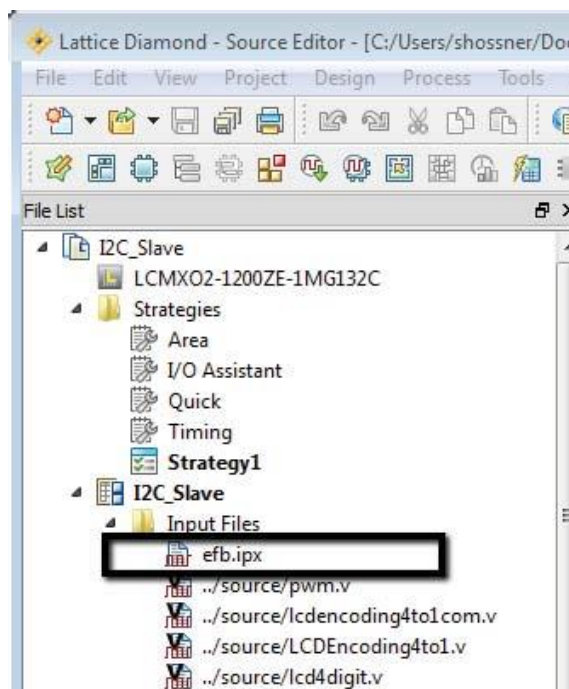


Figure 6.1. IPexpress Instance Example

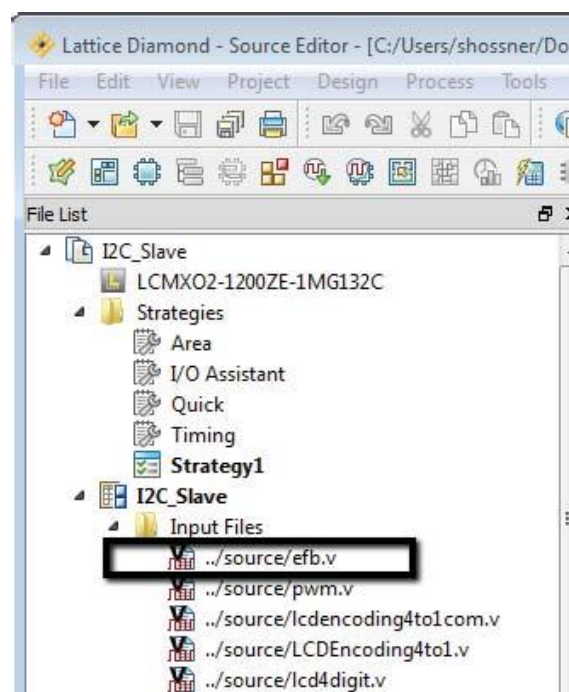


Figure 6.2. HDL Instance Example

The Primary/Secondary port type, Slave I2C address and other settings must be transferred for the Soft Core migration to be successful. These settings can be determined from either instantiation method.

For IPexpress (.ipx) instances, double-click the .ipx filename to open the IPexpress EFB configuration GUI. Primary or Secondary port type can be determined from the EFB Enables tab, while other settings are found on the I2C tab. See [Figure 6.3.](#) and [Figure 6.4.](#)

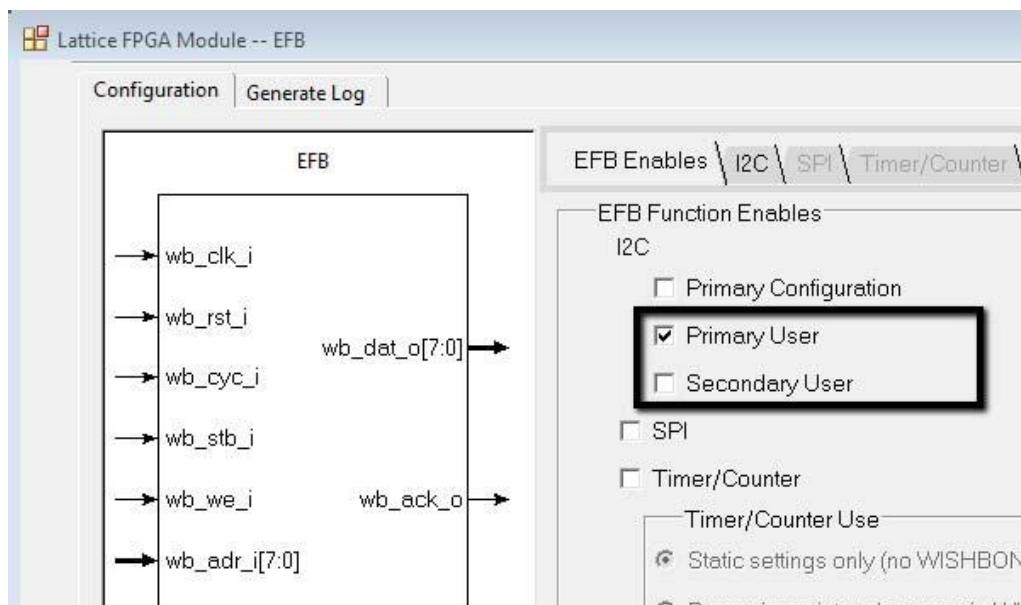


Figure 6.3. IPexpress I²C Port Type Example

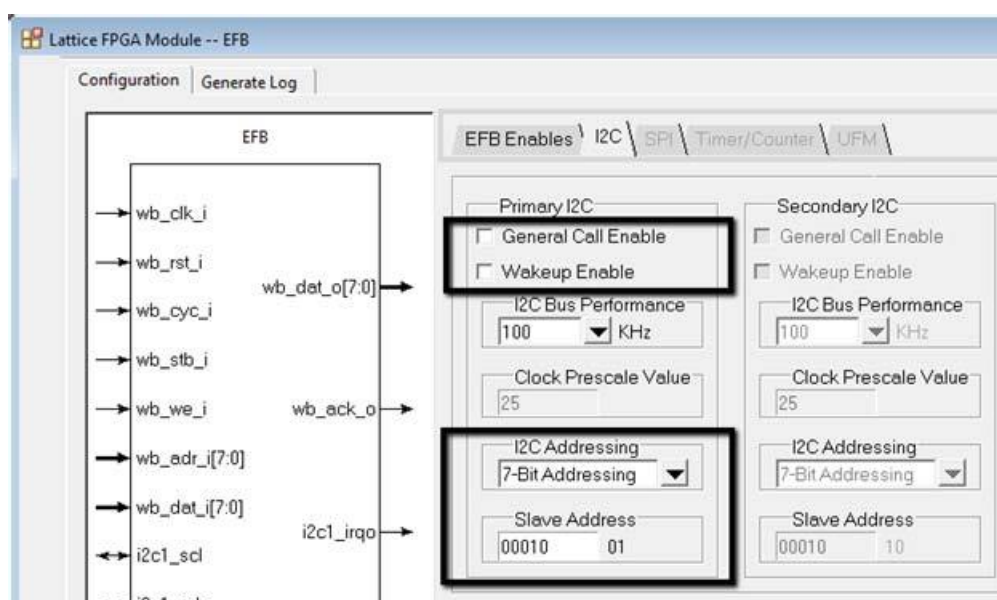


Figure 6.4. IPexpress I²C Settings Example

For Verilog (.v) instances, the active I2C port can be determined by inspecting the source file directly. Double-click the Verilog file to open the default editor and view the file. Find the defparam *.EFB_I2C1 (Primary) or *.EFB_I2C2 (Secondary) lines. In the example in Figure 6.4., the Primary port is Enabled. Also note the corresponding I2C Slave Address, Wakeup and General Call settings.

```

74 defparam EFBInst_0.EFB_SPI = "DISABLED" ;
75 defparam EFBInst_0.I2C2_WAKEUP = "DISABLED" ;
76 defparam EFBInst_0.I2C2_GEN_CALL = "DISABLED" ;
77 defparam EFBInst_0.I2C2_CLK_DIVIDER = 1 ;
78 defparam EFBInst_0.I2C2_BUS_PERF = "100kHz" ;
79 defparam EFBInst_0.I2C2_SLAVE_ADDR = "0b0011001" ;
80 defparam EFBInst_0.I2C2_ADDRESSING = "7BIT" ;
81 defparam EFBInst_0.EFB_I2C2 = "DISABLED" ;
82 defparam EFBInst_0.I2C1_WAKEUP = "DISABLED" ;
83 defparam EFBInst_0.I2C1_GEN_CALL = "DISABLED" ;
84 defparam EFBInst_0.I2C1_CLK_DIVIDER = 25 ;
85 defparam EFBInst_0.I2C1_BUS_PERF = "100kHz" ;
86 defparam EFBInst_0.I2C1_SLAVE_ADDR = "0b0001001" ;
87 defparam EFBInst_0.I2C1_ADDRESSING = "7BIT" ;
88 defparam EFBInst_0.EFB_I2C1 = "ENABLED" ;
89 defparam EFBInst_0.EFB_WB_CLK_FREQ = "10.0" ;
90 EFB EFBInst_0 (.WBCLKI(wb_clk_i), .WBRSII(wb_rst_i), .WBCYCI(wb_cyc_i),
91               WRSTRT(wb_rst_i), WRWFT(wb_we_i), WRANDT7(wb_andr_i[7]), WRANDT6

```

Figure 6.5. Verilog Port and Address Example

For VHDL (.vhd) instances, the active I2C port can be determined by inspecting the source file directly. Double-click the VHDL file to open the default editor and view the file. Find the generic assignment EFB_I2C1 (Primary) or EFB_I2C2 (Secondary) lines. In the example in Figure 6.5., the Primary port is Enabled. Also note the corresponding I2C Slave Address, Wakeup and General Call settings.

```

169 SPI_INTR_IARDI=> "DISABLED", SPI_SLAVE_SHARE=> "DISABLED",
170 SPI_PHASE_ADJ=> "DISABLED", SPI_CLK_INV=> "DISABLED",
171 SPI_LSB_FIRST=> "DISABLED", SPI_CLK_DIVIDER=> 1, SPI_MODE=> "MASTER",
172 EFB_SPI=> "DISABLED", I2C2_WAKEUP=> "DISABLED", I2C2_GEN_CALL=> "DISABLED",
173 I2C2_CLK_DIVIDER=> 1, I2C2_BUS_PERF=> "100kHz", I2C2_SLAVE_ADDR=> "0b0001001",
174 I2C2_ADDRESSING=> "7BIT", EFB_I2C2=> "DISABLED",
175 I2C1_WAKEUP=> "DISABLED",
176 I2C1_GEN_CALL=> "DISABLED", I2C1_CLK_DIVIDER=> 125,
177 I2C1_BUS_PERF=> "100kHz", I2C1_SLAVE_ADDR=> "0b0001001",
178 I2C1_ADDRESSING=> "7BIT", EFB_I2C1=> "ENABLED", EFB_WB_CLK_FREQ=> "50.0")
179 port map (WBCLKI=>wb_clk_i, WBRSII=>wb_rst_i, WBCYCI=>wb_cyc_i,
180         WRSTRT=>wb_rst_i, WRWFT=>wb_we_i, WRANDT7=>wb_andr_i[7],

```

Figure 6.6. VHDL Port and Address Example

6.2. Soft Core installation

When migrating a design from Hard EFB block to Soft I2C Slave reference design, Lattice recommends taking advantage of the multiple implementation feature available in Lattice Diamond®. Source, Strategy and Preference files can be shared, reducing migration effort and potential errors compared to creating a new design from scratch. Whether modifying an existing implementation or creating a new implementation, the following guidelines should be followed.

1. Remove the existing EFB from the implementation. Select the IPexpress (.ipx) or HDL (.v or .vhd) file, right-click and select Remove. The files remains on the hard drive, but are disassociated from the project.
2. Add the new design files to the implementation. Different files are required depending upon Primary or Secondary Port replacement. See Table 3.1 The Verilog file may be referenced from an arbitrary installation location, or copied to the Implementation 'source' directory. See Figure 6.6. The NGO file must be located in the active implementation directory.

Table 6.1. Primary and Secondary File Sets

Component	Primary	Secondary
Top	Soft_EFB_I2C_top_Pri.v	Soft_EFB_I2C_top_Sec.v
Include	i2c_config.v	i2c_config.v
Blackbox	i2c_ip_pri.v	i2c_ip_sec.v
NGO	i2c_ip_pri.ngo	i2c_ip_sec.ngo

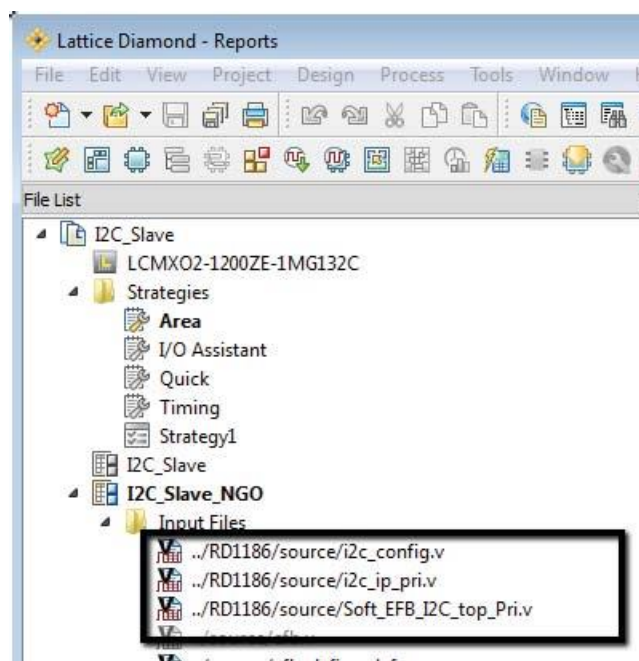


Figure 6.7. Primary Files Added to New Implementation Example

6.3. User Design Modifications

The Soft I²C Slave reference design must be modified to match the EFB settings. Additionally, the user Master Wishbone interface design may require modification to ensure new register settings are preserved during operation.

1. Modification to the Soft I2C Slave Reference Design is confined to the i2c_config.v configuration file. These settings dictate the RESET state of these register bits. Open the i2c_config.v file and modify the `defines to match the original EFB design:

`define	enable	disable
WAKEUP_INIT	1'b1	1'b0
GEN_CALL_INIT	1'b1	1'b0
IPex_I2C_slave_addr	See Customization section	
SDA_DEL_FACTOR	See Customization section	
SDA_DEL_INIT	See Customization section	

2. The instantiation of the Soft I2C Slave reference design must be modified so that module names match. In the instantiating code, change the module name to "Soft_EFB_I2C_top". For example:
 Soft_EFB_I2C_top U1 (.wb_clk_i (CLK), ...);
 It should not be necessary to modify the instantiation port list.
3. It may be necessary to modify the WISHBONE master controller, so that any write to the I2C Control register (I2C_1_CR / I2C_2_CR) or Clock Prescale 1 register (I2C_1_BR1 / I2C_2_BR1) reinforce the GCEN, WKUPEN, SDA_DEL_SEL[1:0] and SDA_DEL_FACTOR[3:0] settings included in i2c_config.v.
 It should not be necessary to modify the WISHBONE Master controller further.

7. Customization

7.1. SDA_DEL calculation

In the Soft Core implementation, the SDA Tdel value is dependent upon the WISHBONE clock period, tWB. This is a change from the hard core. Calculate SDA_DEL_FACTOR and SDA_DEL_INIT using the following information:

The initial value of SDA_DEL can be specified a combination of `define SDA\_DEL\_INIT and `define SDA_DEL_FACTOR. SDA_DEL_SEL is a 2-bit value with range 0-3 decimal. SDA_DEL_SEL set in the I2C Control register. SDA_DEL_FACTOR is a 4-bit value with range 0-15 decimal. SDA_DEL_FACTOR set in the I2C Clock Prescale 1 register. These values determine the number of tWB periods after the falling edge of SCL that SDA is updated, based on the table below:

SDA_DEL	tDEL, (tWB periods)
00	$4 \times \text{SDA_DEL_FACTOR} + 2.5 \pm 0.5 \text{ tWB}$
01	$2 \times \text{SDA_DEL_FACTOR} + 2.5 \pm 0.5 \text{ tWB}$
10	$1 \times \text{SDA_DEL_FACTOR} + 2.5 \pm 0.5 \text{ tWB}$
11	$0 \pm 0.5 \text{ tWB}$

Example 1:	fWB = 25Mhz (tWB = 40 ns): If SDA_DEL_FACTOR = 5 and SDA_DEL = "10", then tDEL = $(1 \times 5 + 2.5 \pm 0.5) \times 40\text{ns} = 300\text{ns} \pm 20\text{ns}$
Example 2:	fWB = 10Mhz (tWB = 100 ns): If SDA_DEL_FACTOR = 1 and SDA_DEL = "10", then tDEL = $(1 \times 1 + 2.5 \pm 0.5) \times 100\text{ns} = 350\text{ns} \pm 50\text{ns}$

7.2. I²C Address

Setting the I2C Address examples:

I ² C Address	IPex_I2C_slave_addr	Comment
0x41 (7-bit)	10'b111_100_0001	XO2 Hard Primary port default
0x42 (7-bit)	10'b111_100_0010	XO2 Hard Secondary port default
0x75 (7-bit)	10'b111_111_0101	Primary address example
0x179 (10-bit)	10'b010_111_1001	Primary address example
0x2C2 (10-bit)	10'b101_100_0010	Secondary address example

8. Special Cases

8.1. Concurrent EFB use

Special care is required when instantiating this reference design concurrent with the MachXO2 EFB. This would occur when other aspects of the EFB are used in the design, for example User Flash Memory, User SPI, or even User I²C (see note below). First, the WISHBONE port must be multiplexed. Second, I²C port pins must be made available.

To multiplex the WISHBONE bus, a simple 2:1 WISHBONE mux (1 Master port, 2 slave ports) is provided with this reference design and may be instantiated as needed.

To free the I²C port pins, the EFB module must be generated (or regenerated) using IPexpress to disable the Primary and/or secondary I²C port(s) as required. This allows the I²C port device pins to be recovered as GPIO and reassigned to the Reference Design I²C port(s).

Note: If the Hardened I²C port is used, be sure to set CKSDIS = '1' to disable Clock Stretching. This feature is no longer supported by the hardened logic. Due to limitations in the provided WISHBONE multiplexor, the Primary Hardened I²C block and the Primary Reference Design cannot be used together as they share the same address space. The same is true for the Secondary.

8.2. Multiple I²C slaves

This Reference Design can be instantiated multiple times as required. The WISHBONE ports must be multiplexed. For the simple case instantiating two copies of the Reference Design (a Primary and Secondary), the provided 2:1 WISHBONE mux, WBmux21.v, can be utilized. For three or more instantiations, the user must provide an appropriate multiplexor.

8.3. Wishbone Mux

This Reference Design includes a 2 port WISHBONE multiplexor, WBmux21.v.

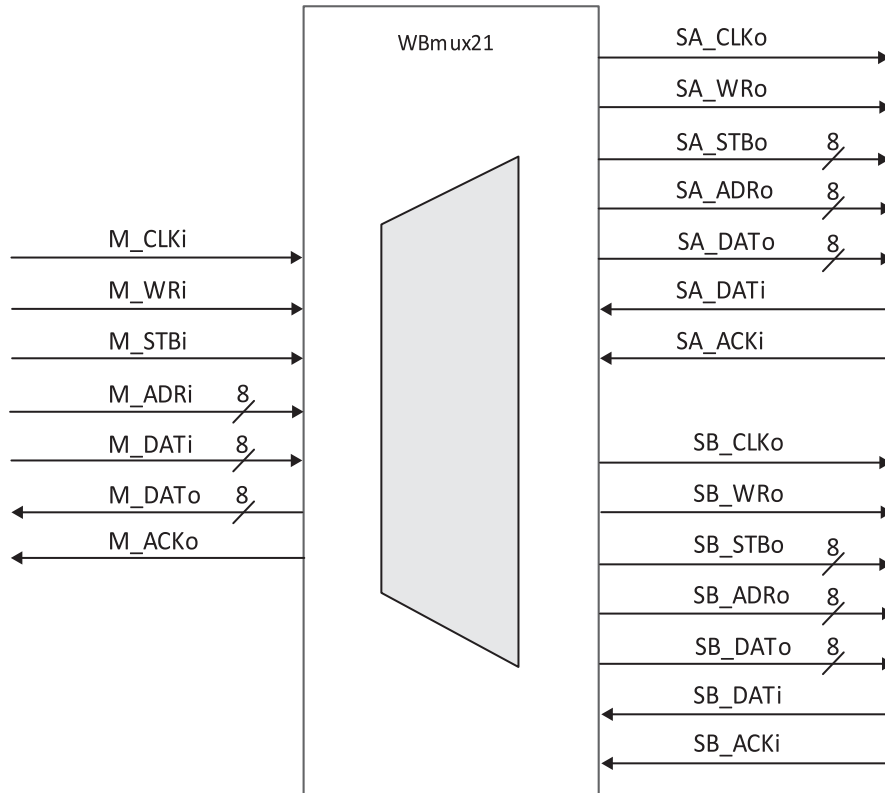


Figure 8.1. WBmux21.v BLOCK DIAGRAM

Connect the WISHBONE bus master to the master port, and the 2 slave WISHBONE ports accordingly. The multiplexor passes through the Master output signals to both Slaves. The circuit assumes only one Slave will respond. The Slave ACK signals are OR'd together and passed back to the Master. The DATo port of the ACK'ing slave is routed to the Master DATi port.

9. Resource Utilization – Soft EFB I²C

Family	Language	Speed Grade	Utilization	fmax (MHz)	I/Os
MachXO2 ¹	NGO-LSE (Lattice Synthesis Engine)	–5	421 LUTs	>50	34
	NGO-Syn (Synplify Pro®)	–5	421 LUTs	>50	34

Note: Performance and utilization characteristics are generated using LCMXO2-1200HC-5MG132C with Lattice Diamond 3.3 design software.

References

- UM10204 “I2C-bus specification and user manual” Rev. 4, 13 February 2012, NXP Semiconductor
- “THE I2C-BUS SPECIFICATION”, VERSION 2.1 JANUARY 2000, Doc. No. 9398 393 40011, Phillips Semiconductor
- [Using User Flash Memory and Hardened Control Functions in MachXO2 Devices Reference Guide \(FPGA-TN-02163\)](#)

Technical Support Assistance

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

Revision History

Revision 1.2, November 2019

Section	Change Summary
All	- Changed document number from RD1186 to FPGA-RD-02092. - Updated document template.
Disclaimers	Added this section.

Revision 1.1, November 2014

Section	Change Summary
Resource Utilization – Soft EFB I ² C	- Added support to Diamond 3.3. - Added support to Synplify Pro and LSE.

Revision 1.0, September 2013

Section	Change Summary
All	Initial release.



www.latticesemi.com