Using User Flash Memory and Hardened Control Functions in MachXO2 Devices

October 2012 | Technical Note TN1205

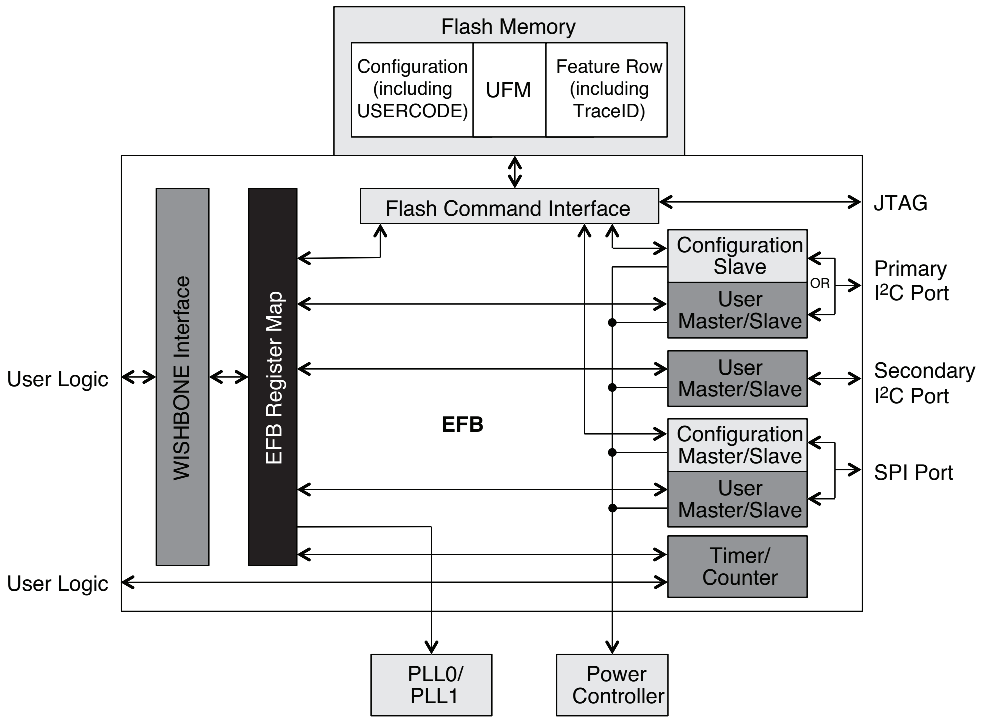
Introduction

The MachXO2™ FPGA family combines a high-performance, low power, FPGA fabric with built-in, hardened control functions and on-chip User Flash Memory (UFM). The hardened control functions ease design implementation and save general purpose resources such as LUTs, registers, clocks and routing. The hardened control functions are physically located in the Embedded Function Block (EFB). All MachXO2 devices include an EFB module. The EFB block includes the following control functions:

- Two I^{2}C cores
- One SPI core
- One 16-bit timer/counter
- Interface to Flash memory which includes:
  - User Flash Memory for MachXO2-640 and higher densities
  - Configuration logic
- Interface to Dynamic PLL configuration settings
- Interface to On-chip Power Controller through I^{2}C and SPI

Figure 9-1 shows the EFB architecture and the WISHBONE interface to the FPGA user logic.

Figure 9-1. Embedded Function Block (EFB)



The hard SPI, I^{2}C, Timer/Counter IPs contained in the EFB can save in excess of 500 LUTs when compared to implementing these same functions in FPGA logic using Lattice reference designs.

The EFB Register Map is used to access the EFB hardened functions through the Slave WISHBONE bus. Each hard IP has dedicated 8-bit Data and Control registers, with the exception of the Flash Memory (UFM/Configuration), which is accessed through the same set of registers. Ports having access to the EFB Register Map have access to all registers. As an example from the Primary I^{2}C Slave port you could access the Timer/Counter registers. The EFB Register Map is shown below:

Table 9-1. EFB Memory Map

Address Range (Hex)	8-bit Data/Control Registers Function
0x00-0x1F	PLL0 Dynamic Access[1]
0x20-0x3F	PLL1 Dynamic Access[1]
0x40-0x49	I^{2}C Primary
0x4A-0x53	I^{2}C Secondary
0x54-0x5D	SPI
0x5E-0x6F	Timer/Counter
0x70-0x75	Flash Memory (UFM/Configuration)
0x76-0x77	EFB Interrupt Source

1. There can be up to two PLLs in a MachXO2 device. PLL0 has an address range from 0x00 to 0x1F. PLL1 (if present) has an address range from 0x20 to 0x3F. Reference TN1199, MachXO2 sysCLOCK PLL Design and Usage Guide, for details on PLL configuration registers and recommended usage.

The EFB module is represented in design software as a primitive and it is described in this document. Use IPexpress™ to configure the EFB, and to generate Verilog or VHDL source code. The source code is instantiated in your design.

WISHBONE Bus Interface

The WISHBONE bus in the MachXO2 is compliant with the WISHBONE standard from OpenCores. It provides connectivity between FPGA user logic and the EFB functional blocks, as well as connectivity between the individual EFB functional blocks. The User Logic must include a WISHBONE Master interface to communicate with the WISHBONE Slave interface of the EFB. An example of a WISHBONE Master is the LatticeMico8™.

The block diagram in Figure 9-2 shows the WISHBONE bus signals between the FPGA core and the EFB. Table 9-2 provides a detailed definition of the signals.

Figure 9-2. WISHBONE Bus Interface Between the FPGA Core and the EFB Module

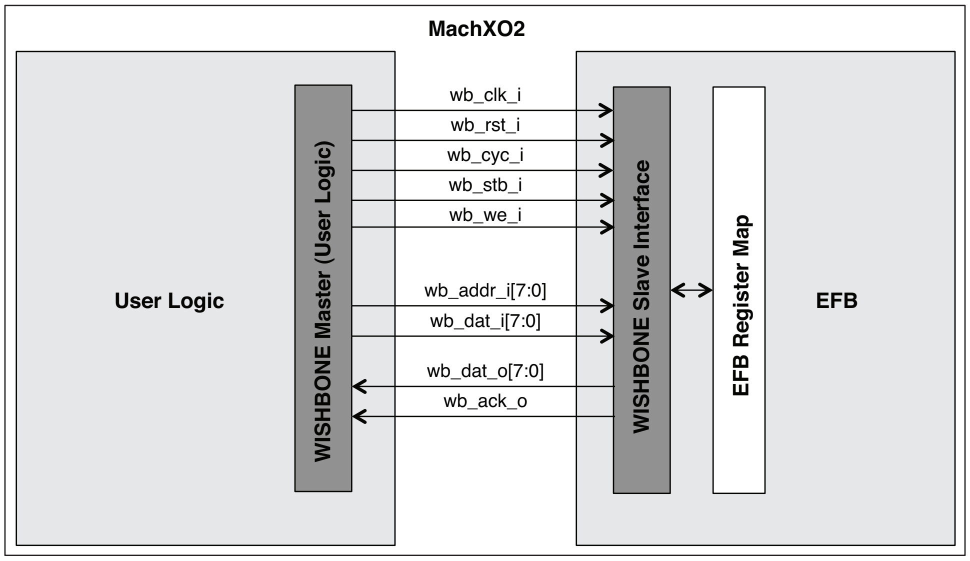


Table 9-2. WISHBONE Slave Interface Signals of the EFB Module

Signal Name	I/O	Width	Description
wb_clk_i	Input	1	Positive edge clock used by WISHBONE interface registers and hardened functions within the EFB module. Supports clock speeds up to 133 MHz.
wb_rst_i	Input	1	Synchronous reset signal that resets the WISHBONE interface logic. This signal does not affect the contents of any registers. It terminates an active bus cycle. Wait 1us after de-assertion before starting any subsequent WISHBONE transactions.
wb_cyc_i	Input	1	Asserted by the WISHBONE master, indicates a valid bus cycle is present on the bus.
wb_stb_i	Input	1	Strobe signal indicating the WISHBONE Slave is the target for the current transaction on the bus. The EFB module asserts an acknowledgment in response to the assertion of the strobe.
wb_we_i	Input	1	Level-sensitive Write/Read control signal. Low indicates a Read operation, and high indicates a Write operation.
wb_adr_i	Input	8	8-bit wide address used to selects an EFB specific register.
wb_dat_i	Input	8	A WISHBONE Master writes data to the addressed EFB register using the wb_dat_i bus during write cycles.
wb_dat_o	Output	8	A WISHBONE Mater receives data from the addressed EFB register using wb_dat_o during read memory cycles.
wb_ack_o	Output	1	Signals the WISHBONE Master the bus cycle is complete; data written to the EFB is accepted. Data read from the EFB is valid.

To interface to the EFB you must create a WISHBONE Master controller in the User Logic. In a multiple-Master configuration, the WISHBONE Master outputs are multiplexed in a user-defined arbiter. A LatticeMico8 soft processor can also be utilized along with the Mico System Builder (MSB) platform which can implement multi-Master bus configurations. If two Masters request the bus in the same cycle, only the outputs of the arbitration winner reach the Slave interface.

WISHBONE Protocol

For information on the WISHBONE protocol and command sequences read the WISHBONE section of TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.

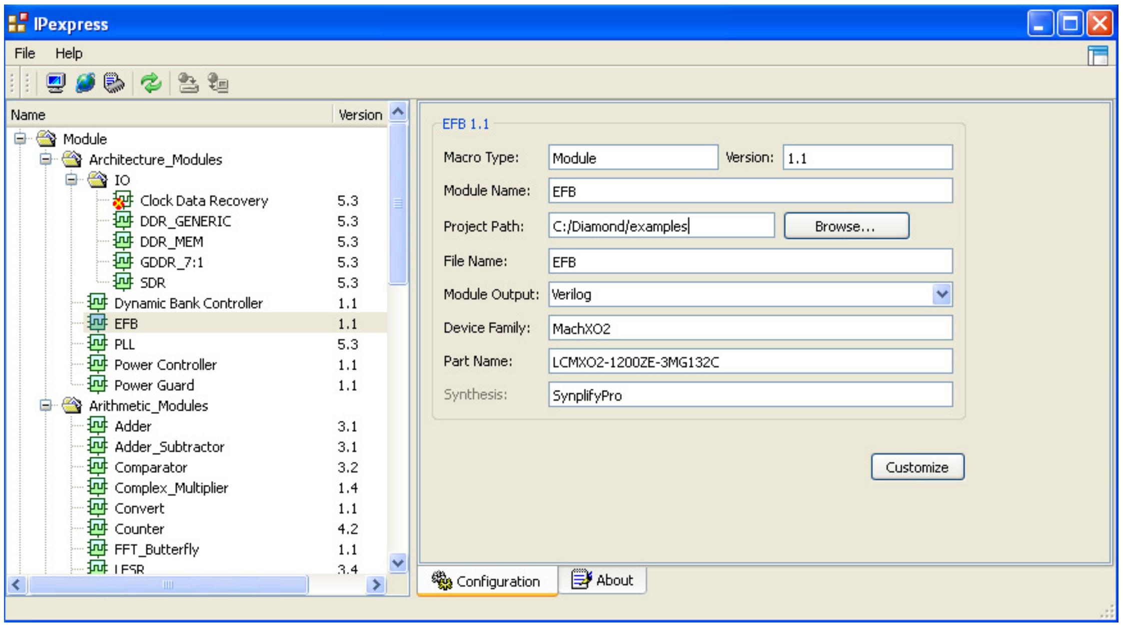
WISHBONE Design Tips

1. Take note when dynamically turning off components for power savings; the EFB requires the MachXO2 internal oscillator to be enabled even if it is not the source of the WISHBONE Clock.
2. If the EFB WISHBONE input signals are not used they should be connected to ‘0’.
3. For more information on the WISHBONE spec can be found on OpenCores website.
4. Many Lattice reference designs have a WISHBONE bus (www.latticesemi.com/products/intellectualproperty/aboutreferencedesigns.cfm)

Generating an EFB Module with IPexpress

IPexpress is used to configure the EFB hard IP functions and generate the EFB module. From the Lattice Diamond® top menu select **Tools > IPexpress**. With a MachXO2 device targeted for the Diamond project, the IPexpress window opens and the EFB module can be found under **Modules > Architecture Modules**.

Figure 9-3. EFB Module in IPexpress



EFB 1.1

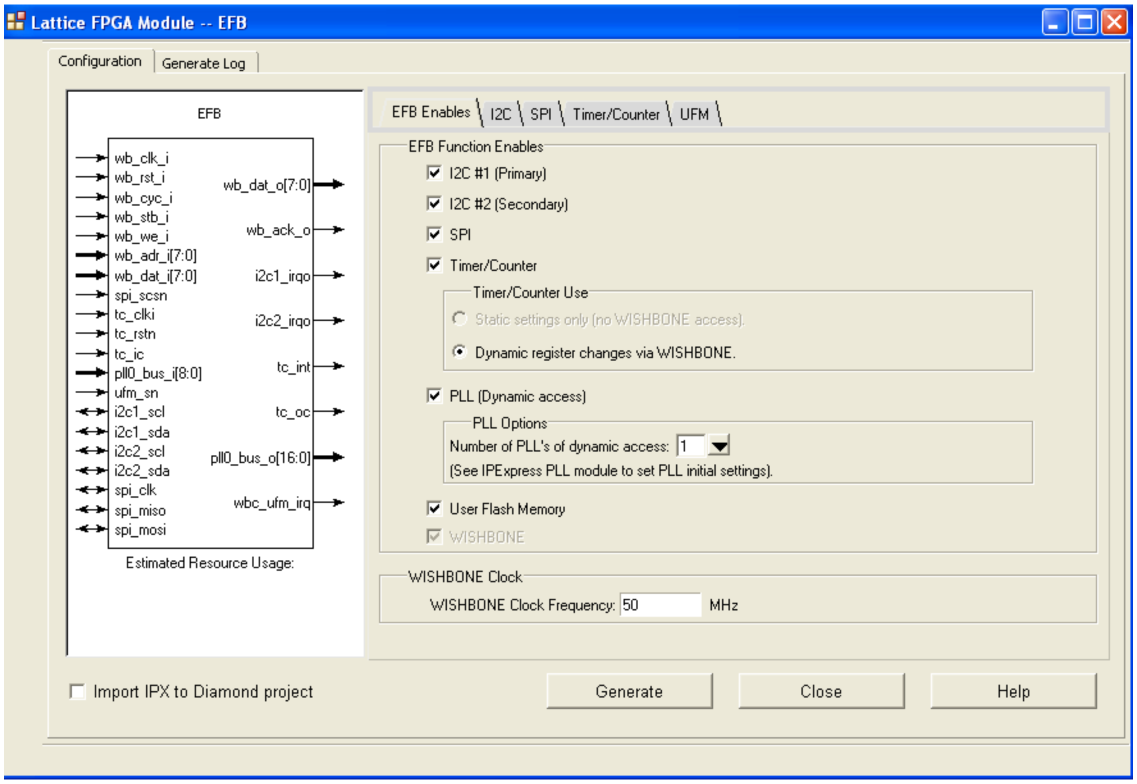
Macro Type: Module
Version: 1.1
Module Name: EFB
Project Path: C:/Diamond/examples
File Name: EFB
Module Output: Verilog
Device Family: MachXO2
Part Name: LCMXO2-1200ZE-3MG132C
Synthesis: SynplifyPro

Fill in the Project Path, File Name, and Design Entry fields, and click **Customize**.

After clicking on the Customize button, the EFB configuration dialog appears. The left side of the EFB window displays a graphical representation of the I/O associated with each IP function. The I/O pins appear and disappear as each IP is enabled or disabled. The initial tab is used to enable the hardened functions, the dynamic access to the PLL configuration settings, the User Flash Memory (UFM) and enter the WISHBONE Clock Frequency. An example EFB with all features enabled is shown in Figure 9-4.

The hardened IP functions and the UFM have individual tabs in the EFB window for individual configuration settings. These tabs will be discussed later in the document, with the technical description of the specific functions. When all functions have been configured, click on the **Generate** button and the EFB module will be generated and ready to be instantiated in your design.

Figure 9-4. Generating an EFB Module with IPexpress



EFB Function Enables
- ☑ I2C #1 (Primary)
- ☑ I2C #2 (Secondary)
- ☑ SPI
- ☑ Timer/Counter

Timer/Counter Use
- ○ Static settings only (no WISHBONE access).
- ● Dynamic register changes via WISHBONE.

- ☑ PLL (Dynamic access)

PLL Options
Number of PLL's of dynamic access: 1
(See IPExpress PLL module to set PLL initial settings).

- ☑ User Flash Memory
- ☑ WISHBONE

WISHBONE Clock
WISHBONE Clock Frequency: 50 MHz

☐ Import IPX to Diamond project

The number of available PLL modules depends on the device density and this will be reflected in the IPexpress EFB GUI. The MachXO2-256 and MachXO2-640 do not have PLL modules and so the PLL checkbox in the EFB window is not available for selection. The MachXO2-640U, MachXO2-1200, MachXO2-1200U, and MachXO2-2000 each have one PLL, and the MachXO2-2000U, MachXO2-4000 and MachXO2-7000 each have two PLLs available for dynamic access through the EFB WISHBONE Slave interface.

The default WISHBONE Clock Frequency is set to 50 MHz. Designers can enter a clock frequency up to 133 MHz. The WISHBONE clock is used by the EFB WISHBONE interface registers and also by the SPI and I^{2}C hardened IP cores. The EFB requires the MachXO2 internal oscillator to be enabled even if it is not the source of the WISHBONE Clock.

Like other modules, the EFB settings can be viewed in the Map Report as shown below:

```
Embedded Functional Block Connection Summary:
---------------------------------------------

   Desired WISHBONE clock frequency: 2.0 MHz
   Clock source:                     clk
   Reset source:                     wb_rst
   Functions mode:
      I2C #1 (Primary) Function:     ENABLED
      I2C #2 (Secondary) Function:   DISABLED
      SPI Function:                  ENABLED
      Timer/Counter Function:        DISABLED
      Timer/Counter Mode:            WB
      UFM Connection:                DISABLED
      PLL0 Connection:               DISABLED
      PLL1 Connection:               DISABLED
```

```
I2C Function Summary:
--------------------
   I2C Component:          PRIMARY
   I2C Addressing:         7BIT
   I2C Performance:        100kHz
   Slave Address:          0b0001001

   General Call:           ENABLED
   I2C Wake Up:            DISABLED
   I2C Component:          UFM/Configuration
   I2C Addressing:         7BIT
   I2C Performance:        100kHz
   Slave Address:          0b0001000
SPI Function Summary:
--------------------
   SPI Mode:               BOTH
   SPI Data Order:         LSB to MSB
   SPI Clock Inversion:    DISABLED
   SPI Phase Adjust:       DISABLED
   SPI Wakeup:             DISABLED
Timer/Counter Function Summary:
------------------------------
   None
UFM Function Summary:
--------------------
   UFM Utilization:        EBR Initialization
   Available General
   Purpose Flash Memory:   511 Pages (511*128 Bits)

   EBR Blocks with Unique
   Initialization Data:    6

   WID            EBR Instance
   ---            ------------
   0b0000000011   LCDCharMap_inst/LCDCharMap_0_0_0
   0b0000000100   STRING_TABLE_INST/EXT_ROM_INST/pmi_romXhmenusdn8101024_0_0_0
   0b0000000101
  lm8_inst/u1_isp8/u1_isp8_prom/pmi_romXhprom_initadn18112048_1_1_0
   0b0000000110
  lm8_inst/u1_isp8/u1_isp8_prom/pmi_romXhprom_initadn18112048_0_0_3
   0b0000000111
  lm8_inst/u1_isp8/u1_isp8_prom/pmi_romXhprom_initadn18112048_0_1_2
   0b0000001000
  lm8_inst/u1_isp8/u1_isp8_prom/pmi_romXhprom_initadn18112048_1_0_1
```

Hardened I^{2}C IP Cores

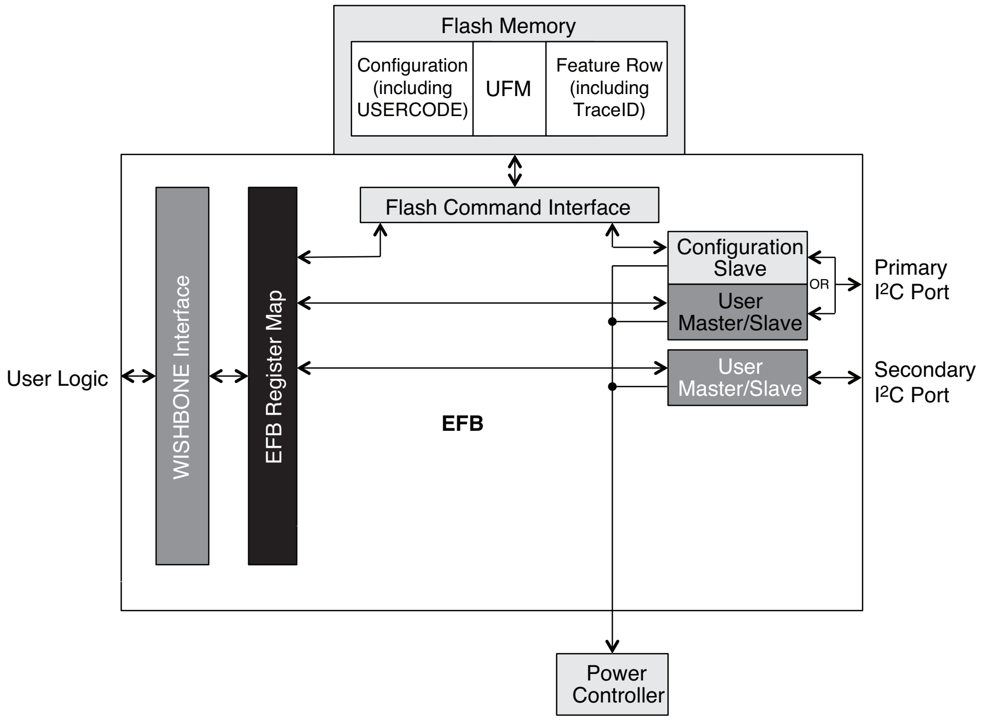
I^{2}C is a widely used two-wire serial bus for communication between devices on the same board. Every MachXO2 device contains two hardened I^{2}C IP cores designated as the “Primary” and “Secondary” I^{2}C IP cores. The two cores in the MachXO2 can operate as an I^{2}C Master or as an I^{2}C Slave. The difference between the two cores is that the Primary core has pre-assigned I/O pins while the ports of the secondary core can be assigned to any general purpose I/O. In addition, the Primary core also has access to the Flash Memory (UFM/Configuration) through the Flash Command Interface. The hardened I^{2}C IP core functionality and block diagram are shown below.

Table 9-3. Hardened I²C Functionality

	Primary I²C Configuration	Primary I²C User	Secondary I²C User
I²C Port as Master	No	Yes	Yes
I²C Port as Slave	Yes[1]	Yes[1]	Yes
Access the Flash Memory (UFM/Configuration)	Yes[1]	No	No
Access the User Logic	No	Yes	Yes
Must use dedicated I/Os	Yes	Yes	No
Wake Power Controller from Standby Mode	Yes	Yes	Yes
Enter Power Controller Standby Mode	Yes	No	No

1. Primary port can be used as Configuration/UFM port or as a User port, but not both.

Figure 9-5. I²C Block Diagram



When an EFB I²C core is a Master it can control other devices on the I²C bus through the physical interface. When an EFB I²C core is the Slave, the device can provide I/O expansion to an I²C Master. Both MachXO2 Primary and Secondary cores support the following I²C functionality:

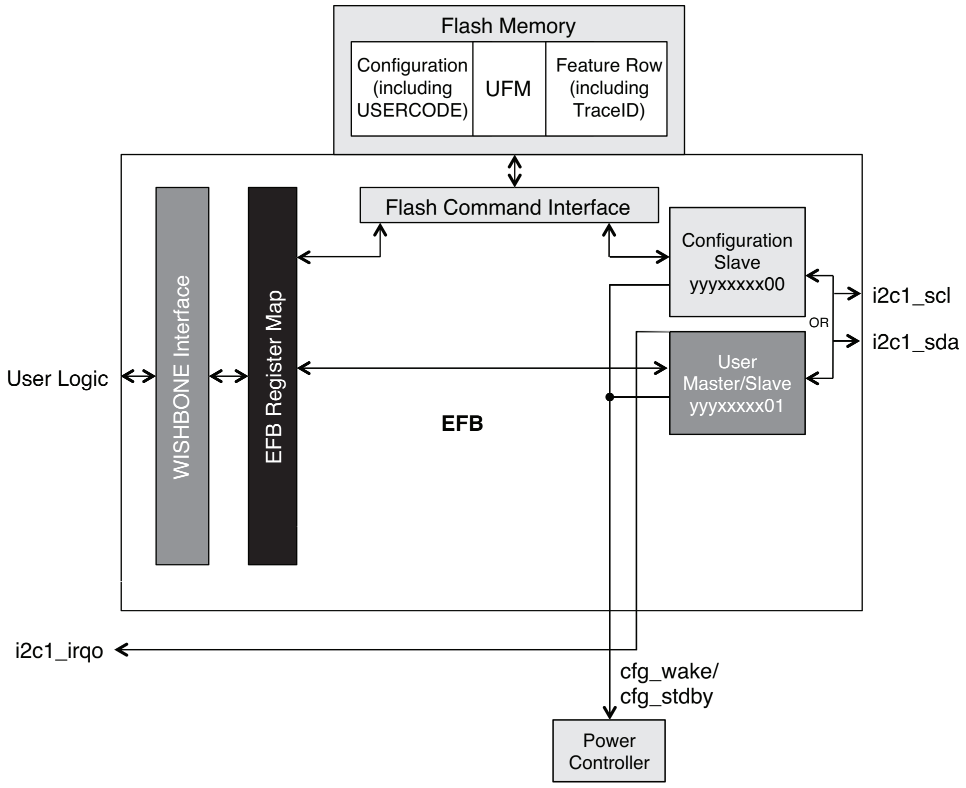
- Master/Slave mode support
- 7-bit and 10-bit addressing
- Clock stretching (I²C Slave is allowed to hold down or stretch the clock to reduce the bus speed)
- Supports 50KHz, 100KHz, and 400KHz data transfer speed
- General Call support (addresses all devices on the bus using the I²C address 0)
- Interface to User Logic through the EFB WISHBONE Slave interface

Primary I²C

The MachXO2 Primary I²C Controller is shown in Figure 9-6. The main functions of the Primary Controller are:

- Either:
  - I²C Configuration Slave provides access to the Flash Memory (UFM/Configuration); or
  - I²C User Slave provides access to the User Logic
- I²C Configuration or User Slave provides access to the MachXO2 Power Controller
- I²C User Master provides access to peripherals attached to the MachXO2

Figure 9-6. I²C Primary Block Diagram



The Primary I²C core can be used for accessing the User Flash Memory (UFM) and for programming the Configuration Flash. However, the Primary I²C port cannot be used for both UFM/Configuration access and user functions in the same design. The block diagram in Figure 9-6 shows an interface between the I²C block and the Flash Memory (UFM/Configuration). For information on Programming the MachXO2 through I²C port reference, the I²C section of TN1204, MachXO2 Programming and Configuration Usage Guide.

Slave I²C peripherals on a bus are accessed by the Master I²C calling the Slave's unique addresses. The Primary Configuration address is “yyyxxxxx00” and the Primary User address is “yyyxxxxx01” where “y” and “x” are user programmable from IPexpress.

The Primary Configuration I²C can be used to wake the Power Controller from Standby or enter Standby. The Primary User can only be used to wake the Power Controller from Standby mode. For more information on the Power Controller, reference TN1198, Power Estimation and Management for MachXO2 Devices. The I²C Power Controller features can be set up through IPexpress as documented later and the register settings are defined in TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.

Table 9-4 documents the IP signals of the Primary I²C cores.

Table 9-4. I²C Primary – IP Signals

Signal Name	Pre-Assigned Pin Name	I/O	Width	Description
i2c1_scl	SCL	Bi-directional	1	**Open drain clock line of the I²C core** – The signal is an output if the I²C core is performing a Master operation. The signal is an input for Slave operations. This signal MUST be routed directly to the corresponding Pre-Assigned Pin. Please reference the MachXO2 Family Data Sheet pin tables for detailed pad and pin locations of I²C ports in each MachXO2 device.
i2c1_sda	SDA	Bi-directional	1	**Open drain data line of the I²C core** – The signal is an output when data is transmitted from the I²C core. The signal is an input when data is received into the I²C core. This signal MUST be routed directly to the corresponding Pre-Assigned Pin. Please reference the MachXO2 Family Data Sheet pin tables for detailed pad and pin locations of I²C ports in each MachXO2 device.
i2c1_irqo	—	Output	1	**Interrupt request output signal of the I²C core** – The intended use of this signal is for it to be connected to a WISHBONE Master controller (i.e. a microcontroller or state machine) and request an interrupt when a specific condition is met. These conditions are described with the I²C section of TN1246.
cfg_wake	—	Output	1	**Wake-up signal** – Hardwired signal to be connected ONLY to the Power Controller of the MachXO2 device for functional simulation support. The signal is enabled only if the Wakeup Enable feature has been set within the EFB GUI, I²C Tab.
cfg_stdby	—	Output	1	**Stand-by signal** – Hardwired signal to be connected ONLY to the Power Controller of the MachXO2 device for functional simulation support. The signal is enabled only if the Wakeup Enable feature has been set within the EFB GUI, I²C Tab.

Secondary I²C

The Secondary I²C controller in the MachXO2 provides the same functionality as the Primary I²C controller with the exception of access to the MachXO2 Configuration Logic. The i2c2_scl and i2c2_sda ports are routed through the general purpose routing of the FPGA fabric and designers can assign them to any General Purpose I/O (GPIO).

The Secondary I²C can be used to wake the Power Controller from Standby mode. For more information on the Power Controller, reference TN1198, Power Estimation and Management for MachXO2 Devices. The I²C Power Controller features can be setup through IPexpress as documented later and the register settings are defined in TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.

Slave I²C peripherals on a bus are accessed by the User Master I²C calling the Slave's unique addresses. The Secondary User address is "yyyxxxxx10" where "y" and "x" are user-programmable from IPexpress.

Figure 9-7 shows the block diagram of the Secondary I²C core.

Figure 9-7. I²C Secondary Block Diagram

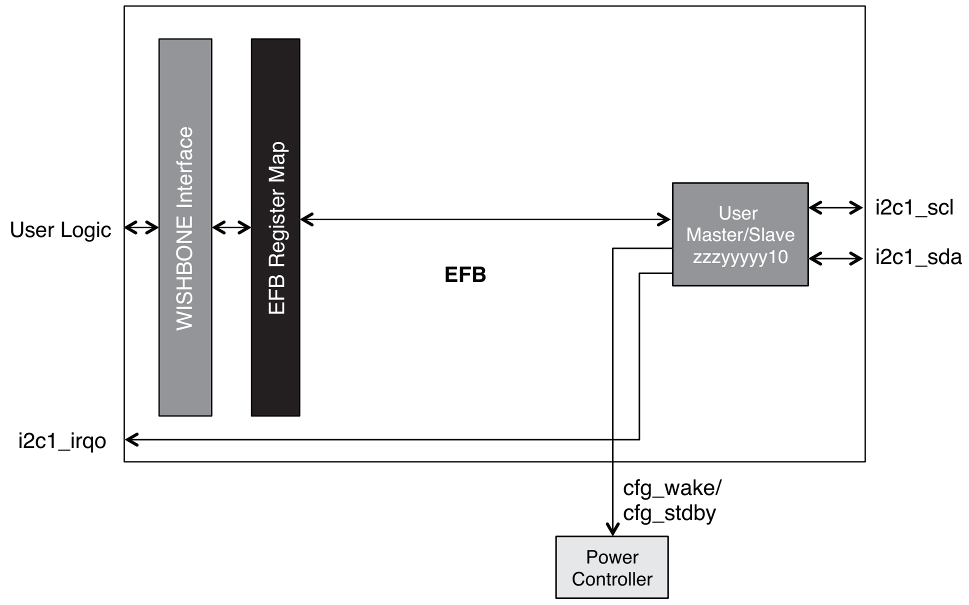


Table 9-5 documents the IP signals of the Secondary I²C cores. These signals can be routed to any GPIO of the MachXO2 devices.

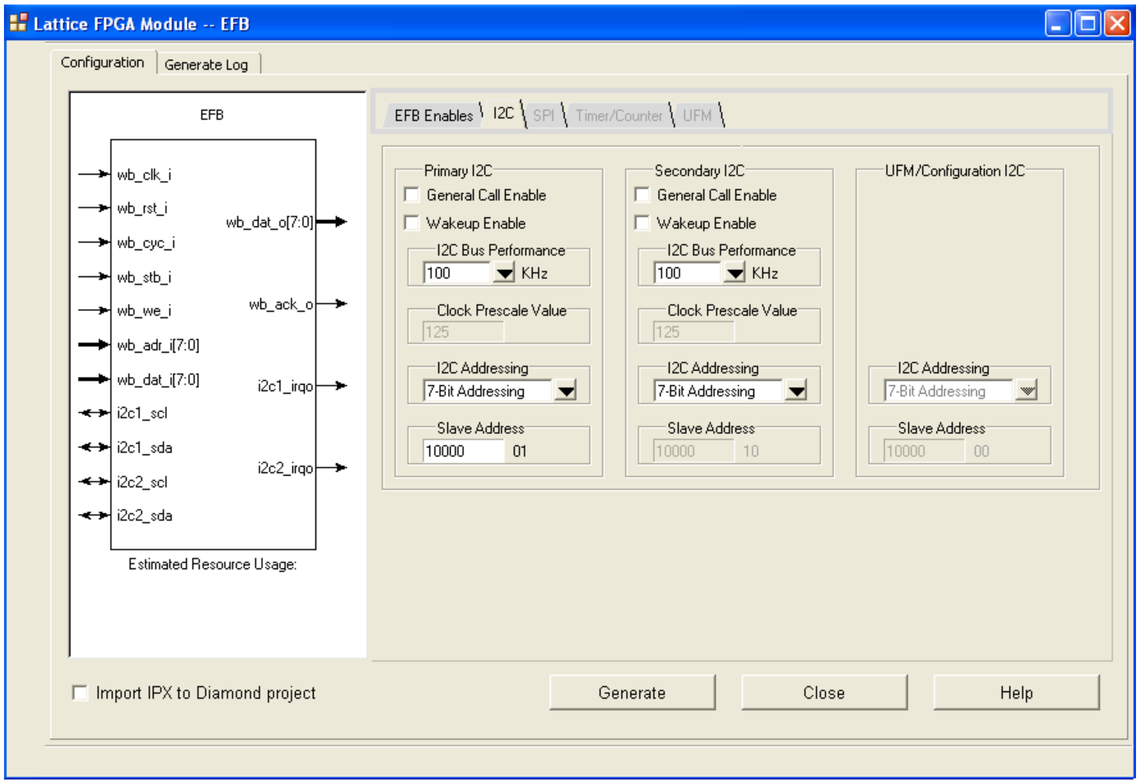
Table 9-5. I²C Secondary – IP Signals

Signal Name	Pre-assigned Pin Name	I/O	Width	Description
i2c2_scl	—	Bi-directional	1	**Open drain clock line of the I²C core**. The signal is an output if the I²C core is performing a Master operation. The signal is an input for Slave operations. The signal can be routed to any GPIO of the MachXO2.
i2c2_sda	—	Bi-directional	1	**Open drain data line of the I²C core**. The signal is an output when data is transmitted from the I²C core. The signal is an input when data is received into the I²C core. The signal can be routed to any GPIO of the MachXO2.
i2c2_irqo	—	Output	1	**Interrupt request output signal of the I²C core**. This signal is intended to be connected to a WISHBONE master controller (i.e. a microcontroller or state machine) and to request an interrupt when a specific condition is met. These conditions are described with the I²C register definitions.
cfg_wake	—	Output	1	**Wake-up signal** – Hardwired signal to be connected ONLY to the Power Controller of the MachXO2 device for functional simulation support. The signal is enabled only if the Wakeup Enable feature has been set within the EFB GUI, I²C Tab.
cfg_stdby	—	Output	1	**Stand-by signal** – Hardwired signal to be connected ONLY to the Power Controller of the MachXO2 device for functional simulation support. The signal is enabled only if the Wakeup Enable feature has been set within the EFB GUI, I²C Tab.

Configuring I²C Cores with IPexpress

Designers can configure the I²C cores and generate the EFB module with IPexpress. Selecting the I²C tab in the EFB GUI will display the configurable settings of the I²C cores. Figure 9-8 shows an example where the I²C cores are configured for an example design.

Figure 9-8. Configuring the I²C Functions of the EFB Module with IPexpress



Primary I2C
- General Call Enable: ☐
- Wakeup Enable: ☐
- I2C Bus Performance: 100 KHz
- Clock Prescale Value: 125
- I2C Addressing: 7-Bit Addressing
- Slave Address: 10000 01

Secondary I2C
- General Call Enable: ☐
- Wakeup Enable: ☐
- I2C Bus Performance: 100 KHz
- Clock Prescale Value: 125
- I2C Addressing: 7-Bit Addressing
- Slave Address: 10000 10

UFM/Configuration I2C
- I2C Addressing: 7-Bit Addressing
- Slave Address: 10000 00

☐ Import IPX to Diamond project

General Call Enable

This setting enables the 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 Primary register I2C_1_CR or the Secondary register I2C_2_CR.

Wake-up Enable

For more information on the Power Controller reference TN1198, Power Estimation and Management for MachXO2 Devices.

When the Wake-up Enable is selected an external I²C Master can cause the MachXO2 to leave the Standby Power state. There are two methods an external I²C master can use to wake the MachXO2 device:

- Primary or Secondary Slave I²C EFB address match
- Perform a General Call followed by the 0xF3 hex command opcode

The WKUPEN bit in the I2C_1_CR or the I2C_2_CR can be modified dynamically allowing the Wake Up function to be enabled or disabled.

I²C Bus Performance

Designers can select an I²C frequency of 50KHz, 100KHz, or 400KHz. This will be the frequency of the SCL clock on the I²C bus. This GUI value, together with the WISHBONE Clock Frequency attribute from the EFB Enables tab, allows the software to calculate the clock divider value for the 10-bit pre-scale registers using the equation (WB Clock)/(Clock Pre-scale Value). This pre-scale value is modified dynamically by accessing the Primary I²C Baud Rate register pair I2C_1_BR1, I2C_1_BR0 or the Secondary I²C Baud Rate register pair I2C_2_BR1, I2C_2_BR0.

I^{2}C Addressing

Designers can select between a 7-bit or 10-bit I^{2}C Slave addressing scheme. The last two bits of the 7-bit address and 10-bit address are hard-coded and select one of the I^{2}C components. The programmable bits of the I^{2}C address are shared between I^{2}C modules and defined as:

yyyxxxxxww

ww bits are hard coded with the following definition

00 = Primary Configuration Flash Memory (UFM/Configuration) I^{2}C

01 = Primary User I^{2}C

10 = Secondary User I^{2}C

11 = I^{2}C core reset

xxxxx bits are programmable using the IPexpress GUI and have the default value of 10000

yyy bits are programmable when 10-bit addressing is selected and have the default value of 000

The Primary I^{2}C address is the same as the length (seven or ten bits) as the Flash Memory (UFM/Configuration) I^{2}C address. The Primary and Secondary I^{2}C address sizes can be of differing lengths. For example the Primary I^{2}C address could be ten bits and the Secondary I^{2}C address could be seven bits.

MachXO2 I^{2}C Usage Cases

The I^{2}C usage cases described below refer to Figure 9-9.

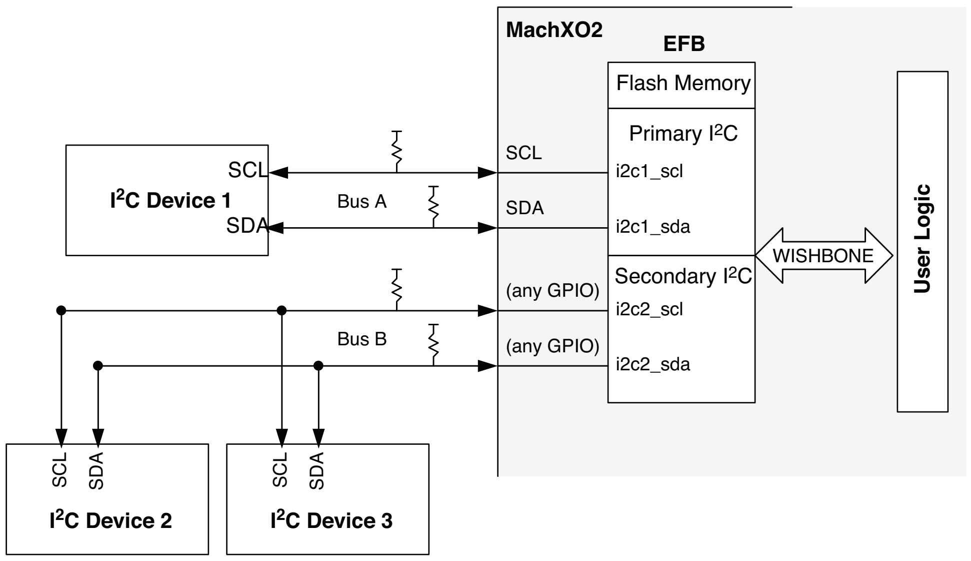
1. Master MachXO2 I^{2}C Accessing Slave External I^{2}C Devices
   a. A WISHBONE bus Master is implemented in the MachXO2 logic
   b. I^{2}C devices 1, 2, and 3 are all Slave devices
   c. The WISHBONE bus Master performs bus transactions to the Primary I^{2}C controller in the EFB to access external Slave I^{2}C Device 1 on Bus A
   d. The WISHBONE bus Mater performs bus transactions to the Secondary I^{2}C controller in the EFB to access the external Slave I^{2}C Devices number 2 or 3 on Bus B
   e. For information on the I^{2}C register definitions and command sequences reference the I^{2}C section of TN1246, Using User Flash Memory and Hardened Control Functions in MachXO2 Devices Reference Guide
2. External Master I^{2}C Device Accessing Slave MachXO2 I2C
   a. The I^{2}C devices 1, 2, and 3 are I^{2}C Master devices
   b. The external master I^{2}C Device 1 on Bus A performs I^{2}C memory cycles to access the EFB Primary I^{2}C controller using address yyyxxxxx01.
   c. The external master I^{2}C Device 2 or 3 on Bus B performs I^{2}C memory cycles to access the EFB Secondary I^{2}C User with the address yyyxxxxx10
   d. A WISHBONE bus master in the MachXO2 fabric must manage data reception and transmission. The WISHBONE master can use interrupts or polling techniques to manage data transfer, and to prevent data overrun conditions.
      i. For information on the I^{2}C register definitions and command sequences reference I^{2}C section of TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices
3. External Master I^{2}C Device Accessing the MachXO2 Flash Memory Using the Primary I^{2}C Interface
   a. The external Master I^{2}C Device 1 on Bus A performs bus transactions using address yyyxxxxx00. The external master interacts with the MachXO2 Configuration Logic using this address. The Configuration Logic provides the controls necessary for performing Flash memory operations.

b. More details on the accessing the Flash Memory (UFM/Configuration) of the MachXO2 device through I^{2}C will be found later in this document and in TN1246, MachXO2 EFB Reference Guide.

c. For information on Programming the MachXO2 through I^{2}C port reference, the I^{2}C section of TN1204, MachXO2 Programming and Configuration Usage Guide.

4. The above usage cases are not mutually exclusive. For example:
   a. External Master Device 1 on Bus A can access the MachXO2 Configuration Logic at the same time a WISHBONE Master transfers data to the I^{2}C slave devices on Bus B.
   b. A WISHBONE master can transfer data to a microprocessor on Bus A (i.e. I^{2}C Device 1), and at some future time the microprocessor can send data back to the WISHBONE Master. The microprocessor may also at some future time reprogram the MachXO2 Flash memory in order to update the MachXO2 device's functionality.

Figure 9-9. I^{2}C Circuit



I^{2}C Design Tips

1. For information on the I^{2}C register definitions and command sequences, reference the I^{2}C section of TN1246, Using User Flash Memory and Hardened Control Functions in MachXO2 Devices Reference Guide.
2. Take note when dynamically turning off components for power savings; the EFB requires the MachXO2 internal oscillator to be enabled even if it is not the source of the WISHBONE Clock.
3. I^{2}C has lower priority than JTAG Port and the Slave SPI Port when accessing the Flash Memory (UFM/Configuration). Reference "Flash Memory (UFM/Configuration) Access" on page 29 for details.
4. The Primary I^{2}C port cannot be used for both UFM/Configuration access and user functions in the same design.
5. If the secondary I^{2}C Secondary Port is enabled after issuing a Refresh command or toggling PROGRAMN it is recommended to reset the state machine with a I^{2}C STOP. I^{2}C STOP is performed with a single register write 0x40 to I2C_2_CMDR. This will cause a short low-pulse on SCK as the block signals the STOP. Normal I^{2}C activity can be commenced without additional delay.
6. There are a number of I^{2}C reference designs on the Lattice website (www.latticesemi.com/products/intellectualproperty/aboutreferencedesigns.cfm) including:
   a. RD1124: I^{2}C Slave Peripheral Using Embedded Function Block
   b. UG55: MachXO2 Hardened I^{2}C Master/Slave Demo
7. For information on the I^{2}C Protocol reference www.i2c-bus.org/

Hardened SPI IP Core

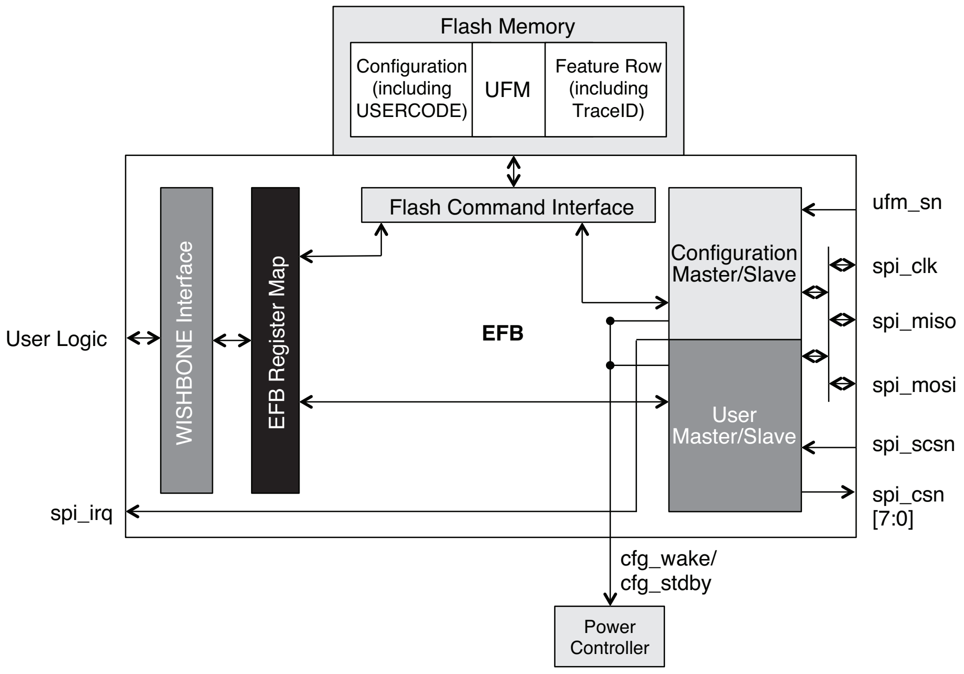
SPI is a widely used four-wire serial bus which operates in full duplex for communication between devices. The MachXO2 EFB contains a SPI Controller that can be configured as a SPI Master/Slave or a SPI Slave. When the IP core is configured as a Master/Slave it is able to control up to either other devices with Slave SPI interfaces. When the core is configured as a Slave, it is able to interface to an external SPI Master device. The SPI core interfaces with the MachXO2's Configuration Logic or the other User Logic. The hardened SPI IP core functionality and block diagram are shown below.

Table 9-6. Hardened SPI Functionality

	Configuration SPI	User SPI
Slave SPI Port	Yes	Yes
Master SPI Port	No[1]	Yes
Access the Flash Memory (UFM/Configuration)	Yes	No
Must use dedicated I/Os	Yes	Yes[2]
Wake Power Controller from Standby Mode	Yes	Yes
Enter Power Controller Standby Mode	No	Yes

1. Only for the configuring SRAM.
2. Any GPIO can be used for spi_csn[7:1] and spi_scsn.

Figure 9-10. SPI Block Diagram



The SPI IP core on MachXO2 devices supports the following functions:

- Configurable Master and Slave modes
- Mode Fault error flag with CPU interrupt capability
- Double-buffered data register for increased throughput
- Serial clock with programmable polarity and phase
- LSB First or MSB First Data Transfer
- Interface to custom logic through the EFB WISHBONE slave interface

In Master/Slave SPI mode:

- The User SPI controller has eight available Master Chip Selects (spi_csn[7:0]) ports. This allows the control of up to eight external devices with Slave SPI interface.
- The Configuration SPI upon power-up, if the SPI port has been enabled to boot the MachXO2 device from an external Slave SPI Flash memory, then the SPI port will act as a Master SPI controller and spi_csn[0] is used as a Master Chip Select for selecting a specific SPI Flash memory. For information on Programming the MachXO2 through the SPI port, reference the SPI section of TN1204, MachXO2 Programming and Configuration Usage Guide.

In Slave SPI mode:

- The User SPI core has one Slave Chip Select (spi_scsn) pin. This allows the User SPI core to be selected by an external device with a Master SPI interface. User Logic is access through the EFB WISHBONE interface by a WISHBONE Master in the FPGA logic.
- The Configuration SPI has one Slave Chip Select (ufm_sn) pin. An external SPI Master can access the MachXO2's Configuration Logic by asserting the chip select input. The external SPI Master can reprogram the MachXO2's Configuration Flash and User Flash Memory by performing bus transfers with SN asserted.

This usage guide is focused on the User SPI access. For more information on Programming the MachXO2 through SPI port reference, the SPI section of TN1204, MachXO2 Programming and Configuration Usage Guide.

The Slave Configuration SPI port can be used to wake the Power Controller from Standby or enter Standby. The Slave User SPI port can only be used to wake the Power Controller from Standby mode. For more information on the Power Controller, reference TN1198, Power Estimation and Management for MachXO2 Devices. The SPI Power Controller features can be set up through IPexpress as documented later and the register settings are defined in TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.

SPI Interface Signals

The SPI interface uses a serial transmission protocol. Data is transmitted serially (shifted out from the transmitting device) and it is received serially, shifted into the receiving device. The master device selects a specific slave device by asserting a chip select, enabling the slave device to shift in the commands/data and to respond by shifting out data.

Table 9-7 documents the signals that are generated with the IP core. Each signal has a description of the usage and how it should be connected in a design project.

Table 9-7. SPI – IP Signals

Signal Name	Pre-assigned Pin Name	I/O	Width	Description
spi_clk	MCLK/CCLK	Bi-directional	1	The signal is an output if the SPI core is in Master mode (MCLK). The signal is an input if the SPI core is in Slave mode (CCLK). This signal MUST be routed directly to the corresponding Pre-Assigned Pin. Reference the MachXO2 Family Data Sheet pin tables for detailed pad and pin locations of SPI signals in each MachXO2 device.
spi_miso	SPISO/SO	Bi-directional	1	The signal is an input if the SPI core is in Master mode (SPISO). The signal is an output if the SPI core is in Slave mode (SO). This signal MUST be routed directly to the corresponding Pre-Assigned Pin. Reference the MachXO2 Family Data Sheet pin tables for detailed pad and pin locations of SPI signals in each MachXO2 device.

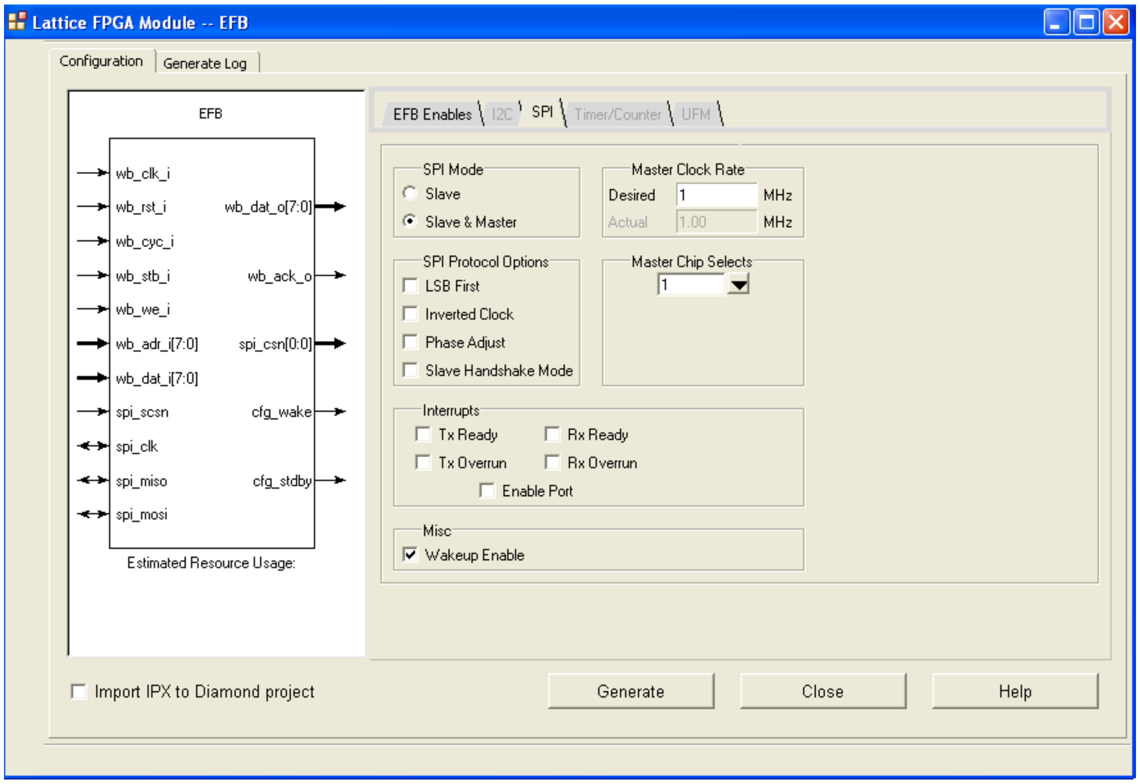
Table 9-7. SPI – IP Signals (Continued)

Signal Name	Pre-assigned Pin Name	I/O	Width	Description
spi_mosi	SISPI/SI	Bi-directional	1	The signal is an output if the SPI core is in Master mode (SISPI). The signal is an input if the SPI core is in Slave mode (SI). This signal MUST be routed directly to the corresponding Pre-Assigned Pin. Reference the MachXO2 Family Data Sheet pin tables for detailed pad and pin locations of SPI signals in each MachXO2 device.
spi_csn[7:0]	CSSPIN	Output	8	Master Chip Select (Active Low). Up to eight independent slave SPI devices can be accessed using the MachXO2 SPI Controller when it is in Master SPI mode. The signal spi_csn[0] MUST be routed directly to the corresponding Pre-Assigned Pin. Reference the MachXO2 Family Data Sheet tables for detailed pad and pin locations of SPI signals in each MachXO2 device.
spi_scsn	—	Input	1	User Slave Chip Select (Active Low). An external SPI Master controller asserts this signal to transfer data to/from the SPI Controllers Transmit Data/Receive Data registers. The signal can be routed to any GPIO of MachXO2 device.
ufm_sn	SN	Input	1	Configuration Logic Chip select (Active Low) is dedicated for selecting the Flash Memory UFM and Configuration Sectors. This signal MUST be routed directly to the corresponding Pre-Assigned Pin. Reference the MachXO2 Family Data Sheet pin tables for detailed pad and pin locations of SPI signals in each MachXO2 device. SN is an active pin whenever the SPI core is instantiated, whether 'ufm_sn' appears on the EFB primitive or not. Thus, SN cannot be recovered as user I/O. SN can be tied high externally to augment the weak internal pull-up if not connected to an external Master SPI bus. SN is also active in a blank or erased device.
spi_irq	—	Output	1	Interrupt request output signal of the SPI core. This signal is intended to be connected to a WISHBONE master controller (i.e. a microcontroller or state machine). It is asserted when specific conditions are met. These conditions controlled using the SPI register settings.
cfg_wake	—	Output	1	Wakeup signal – to be connected only to the Power Controller module of the MachXO2 device. The signal is enabled only if the “Wakeup Enable” feature has been set within the EFB GUI, SPI Tab.
cfg_stdby	—	Output	1	Stand-by signal – to be connected only to the Power Controller module of the MachXO2 device. The signal is enabled only if the “Wakeup Enable” feature has been set within the EFB GUI, SPI Tab.

Configuring the SPI Core with IPexpress

IPexpress is used to configure the SPI Controller and to generate Verilog or VHDL source code for inclusion in your design. Selecting the SPI tab, in the EFB GUI, displays the configurable settings for the SPI core. Figure 9-11 shows an example SPI Controller configuration.

Figure 9-11. Configuring the SPI Functions of the EFB Module with IPexpress



SPI Mode
- ○ Slave
- ● Slave & Master

Master Clock Rate
- Desired: 1 MHz
- Actual: 1.00 MHz

SPI Protocol Options
- ☐ LSB First
- ☐ Inverted Clock
- ☐ Phase Adjust
- ☐ Slave Handshake Mode

Master Chip Selects: 1

Interrupts
- ☐ Tx Ready
- ☐ Rx Ready
- ☐ Tx Overrun
- ☐ Rx Overrun
- ☐ Enable Port

Misc
- ☑ Wakeup Enable

☐ Import IPX to Diamond project

SPI Mode

This option allows the user to select between “Slave”, or “Slave & Master” modes for the initial mode of the SPI core. Selecting “Slave & Master” enables SPI Master settings which include Master Clock Rate and Master Chip Selects. This option can be updated dynamically by modifying the MSTR bit of the register SPICR2.

SPI Master Clock Rate

Desired Frequency: The EFB SPI Controller, when it is configured as a SPI Master, provides an output clock to the SPI Slave devices on the bus. The output frequency uses a “power of two” value to divide the WISHBONE Clock Frequency. The SPI Master uses the Master Clock Rate to time all SPI bus transactions and internal operations. The MachXO2 SPI Master interface can operate at speeds up to 45 MHz. You input the WISHBONE Clock Frequency on the EFB Enables tab of the dialog.

The divisor can be changed while the FPGA is in user mode. Updating the divider value in the SPIBR register causes the SPI Controller to reset and use a new output clock frequency.

Actual Frequency: It is not always possible to divide the input WISHBONE clock exactly to the frequency requested by the user. The actual frequency value will be returned in this read-only field. When both the desired SPI clock and WISHBONE clock fields have valid data and either is updated, this field will return the value rounded to two decimal places.

SPI Protocol Options

LSB First: This setting specifies the order of the serial shift of a byte of data. The data order (MSB or LSB first) is programmable within the SPI core. This option can be updated dynamically by modifying the LSBF bit in the register SPICR2.

Inverted Clock: The inverts the clock polarity used to sample and output data is programmable for the SPI core. When selected the edge changes from the rising to the falling clock edge. This option can be updated dynamically by accessing the CPOL bit of register SPICR2.

Phase Adjust: An alternate clock-data relationship is available for SPI devices with particular requirements. This option allows the user to specify a phase change to match the application. This option can be updated dynamically by accessing the CPHA bit in the register SPICR2.

Slave Handshake Mode: Enables Lattice proprietary extension to the SPI protocol. For use when the internal support circuit (e.g. WISHBONE host) cannot respond with initial data within the time required, and to make the Slave read out data predictably available at high SPI clock rates. This option can be updated dynamically by accessing the SDBRE bit in the register SPICR2.

Master Chip Selects
The SPI Controller provides the ability to provide up to eight individual chip select outputs for master operation. Each slave SPI device accessed by the master must have their own dedicated chip select. This option can be updated dynamically by modifying the register SPICSR.

SPI Controller Interrupts
TX Ready: An interrupt which indicates the SPI transmit data register (SPITXDR) is empty. The interrupt bit is IRQTRDY of the register SPIIRQ. When enabled, indicates TRDY was asserted. Write a ‘1’ to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQTRDYEN in the register SPICSR.

RX Ready: An interrupt which indicates the receive data register (SPIRXDR) contains valid receive data. The interrupt is bit IRQRRDY of the register SPIIRQ. When enabled, indicates RRDY was asserted. Write a ‘1’ to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQRRDYEN in the register SPICSR.

TX Overrun: An interrupt which indicates the Slave SPI chip select (SPI_SCSN) was driven low while a SPI Master. The interrupt is bit IRQMDF of the register SPIIRQ. When enabled, indicates MDF (Mode Fault) was asserted. Write a ‘1’ to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQMDFEN in the register SPICSR.

RX Overrun: An interrupt which indicates SPIRXDR received new data before the previous data. The interrupt is bit IRQROE of the register SPIIRQ. When enabled, indicates ROE was asserted. Write a ‘1’ to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQROEEN in the register SPICSR.

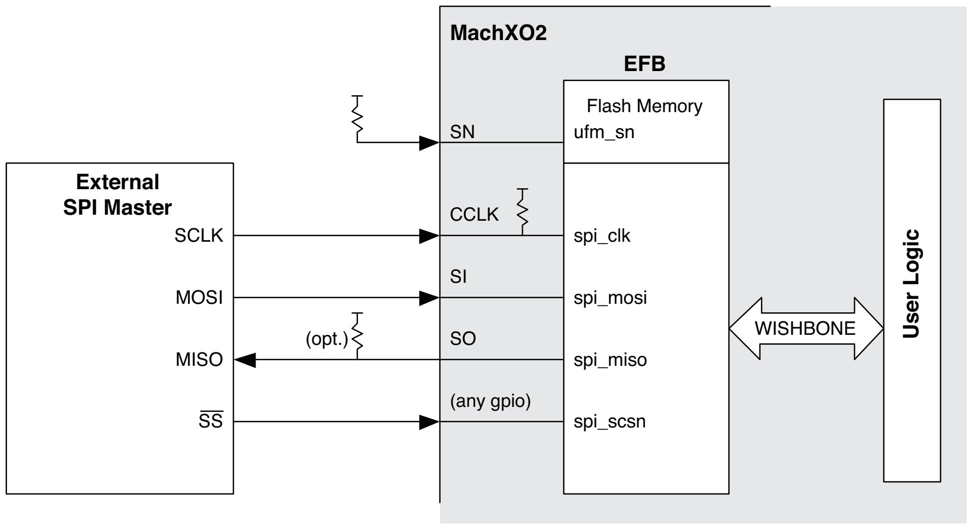
Enable Port (Interrupts): This enables the interrupt request output signal (spi_irq_ of the SPI core signal. This signal is intended to be connected to a WISHBONE master controller (i.e. a microcontroller or state machine) and to request an interrupt when a specific condition is met.

Wake-up Enable
Enables the SPI core to send a wake-up signal to the Power Controller to wake the part from standby mode when the User Slave SPI chip select (spi_csn[0]) is driven low. This option can be updated dynamically by modifying the bit WKUPEN_USER in the register SPICR1.

MachXO2 SPI Usage Cases
The SPI usage cases described below refer to the figures below:

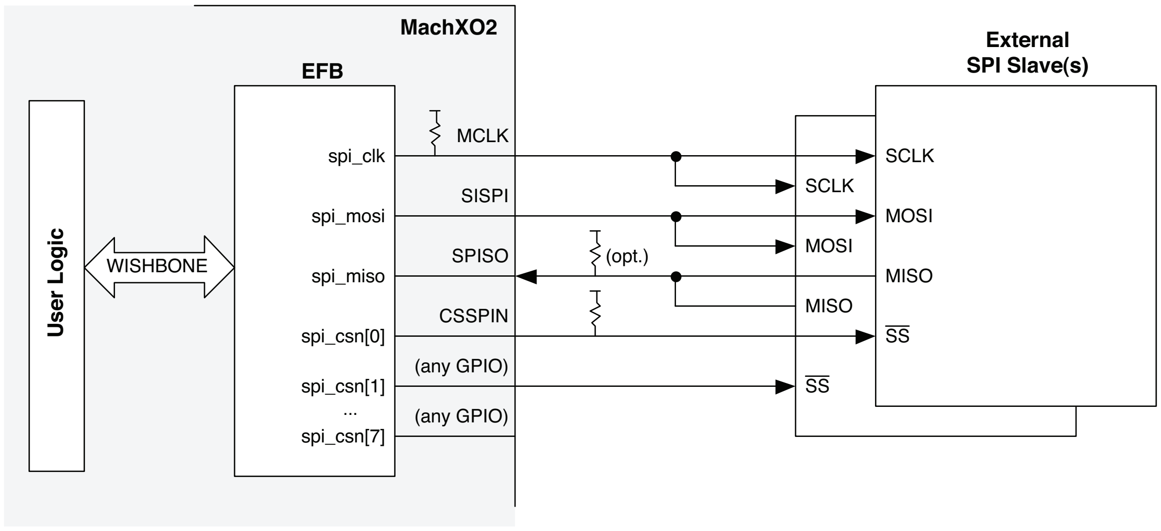
1. External Master SPI Device Accessing the Slave MachXO2 User SPI
   a. The External Master SPI is connected to the MachXO2 using the dedicated SI, SO, CCLK pins. The spi_scsn is placed on any Generic I/O. The EFB SPI Mode is set to Slave only.
   b. A WISHBONE Master controller is implemented in the MachXO2 general purpose logic array. The master controller monitors the availability to transmit or receive data by polling the SPI status registers, or by responding to interrupts generated by the SPI Controller.
      i. For information on the SPI register definitions and command sequences reference SPI section of TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.
   c. The external SPI Master does not have access to the MachXO2 Configuration Logic because the SN that selects the Configuration Logic is pulled high.

Figure 9-12. External Master SPI Device Accessing the Slave MachXO2 User SPI



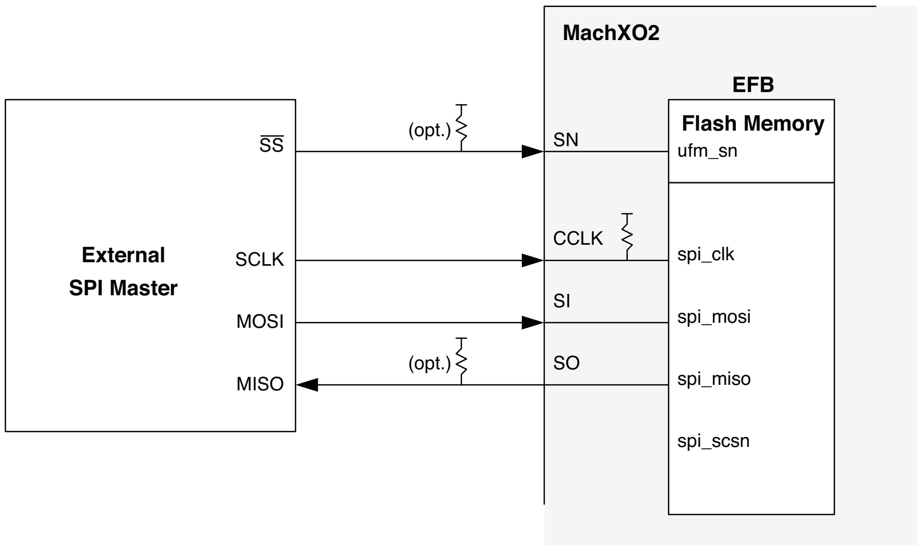
2. MachXO2 User SPI Master accessing one or multiple External Slave SPI devices
   a. The MachXO2 SPI Master is connected to External SPI Slave devices using the dedicated SPI port pins. The Chip Selects are configured as follows:
      i. The MachXO2 SPI Master Chip Select spi_scn[0] is placed on the dedicated CSSPIN and connected to the External Slave Chip Select.
      ii. The MachXO2 SPI Master Chip Select spi_scn[1] is placed on any I/O and connected to another External Slave Chip Select.
      iii. Up to eight External Slave SPIs can be connected using spi_scn[7:0]
   b. A WISHBONE Master controller is implemented in the MachXO2 general logic. It controls transfers to the slave SPI devices. It can use a polling method, or it can use SPI Controller interrupts to manage transfer and reception of data.
      i. For information on the SPI register definitions and command sequences reference SPI section of TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.

Figure 9-13. MachXO2 User SPI Master Accessing One or Multiple External Slave SPI Devices



3. External Master SPI Device accessing the MachXO2 Configuration Logic
   a. The External SPI Master is connected to the MachXO2 dedicated slave Configuration SPI port pins. The external SPI Master's chip select controls the SN input that enables the MachXO2's Configuration Logic block. The external master sends commands to the Configuration Logic block permitting it to interface to the Configuration Flash and the UFM.
      i. For information on the accessing the Flash Memory (UFM/Configuration) of the MachXO2 device through SPI can be found later in this document.
      ii. For more information on Programming the MachXO2 through SPI port reference, the SPI section of TN1204, MachXO2 Programming and Configuration Usage Guide.

Figure 9-14. External Master SPI Device Accessing the MachXO2 Configuration Logic



SPI Design Tips

1. For information on the SPI register definitions and command sequences reference SPI section of TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.
2. Take note when dynamically turning off components for power savings; the EFB requires the MachXO2 internal oscillator to be enabled even if it is not the source of the WISHBONE Clock.
3. The SPI bus is bidirectional. A byte is received for every byte transmitted. Always discard RX data to avoid Receiver Overrun Error (ROE).
4. SN is an active pin whenever the SPI core is instantiated, whether 'ufm_sn' appears on the EFB primitive or not. Thus, SN cannot be recovered as user I/O. SN should be tied high externally to augment the weak internal pull-up if not connected to an external Master SPI bus.
5. There are a number of SPI reference designs on the Lattice website (www.latticesemi.com/products/intellectualproperty/aboutreferencedesigns.cfm) including:
   a. RD1125: SPI Slave Peripheral using Embedded Function Block
   b. UG56: MachXO2 Hardened SPI Master/Slave Demo

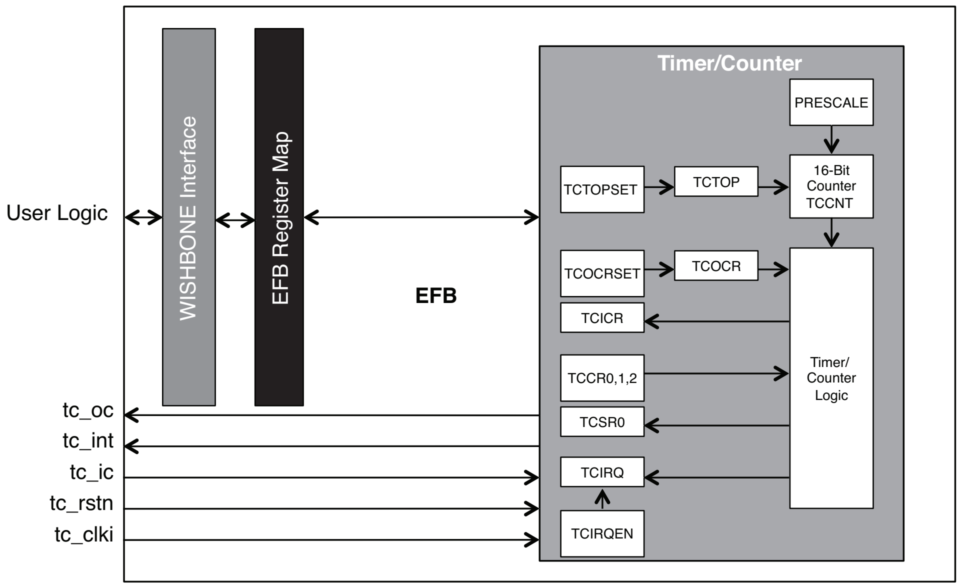
Timer/Counter

The MachXO2 EFB contains a Timer/Counter function. This Timer/Counter is a general purpose 16-bit Timer/Up Down Counter module with independent output compare units and Pulse Width Modulation (PWM) support. The Timer/Counter supports the following functions:

- Four unique modes of operation:
  – Watchdog timer

  - Clear Timer on Compare Match
  - Fast PWM
  - Phase and Frequency Correct PWM
- Programmable clock input source
- Programmable input clock pre-scale
- Interrupt output to FPGA fabric
- Three independent interrupt sources: overflow, output compare match, and input capture
- Auto reload
- Time-stamping support on the input capture unit
- Waveform generation on the output
- Glitch-free PWM waveform generation with variable PWM period
- Internal WISHBONE bus access to the control and status registers
- Stand-alone mode with preloaded control registers and direct reset input

Figure 9-15. Timer/Counter Block Diagram



The Timer/Counter communicates with the FPGA core logic through the WISHBONE interface and specific signals such as clock, reset, interrupt, timer output, and event trigger. The Timer/Counter can be included in a design with or without the WISHBONE interface.

Timer/Counter Modes of Operation

The Timer/Counter is a flexible function, which has four modes of operation. These modes are designed to meet different system needs related to sequencing of events and PWM (Pulse Width Modulation). The four Timer/Counter modes are:

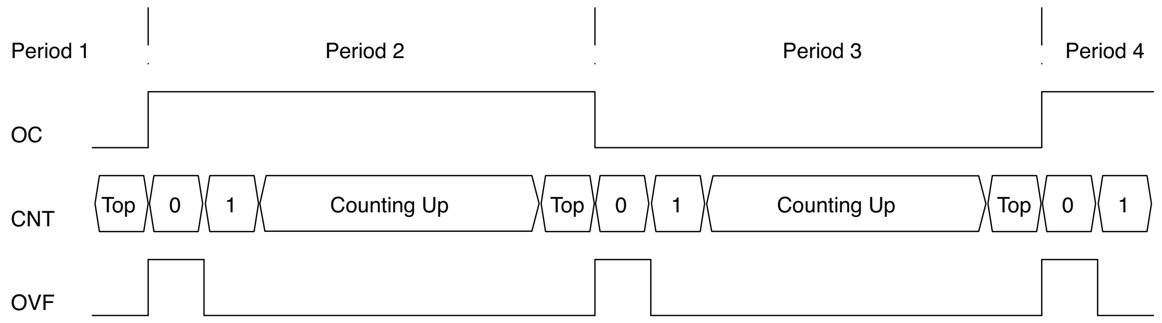
- Clear Timer on Compare Match
- Watchdog Timer
- Fast PWM
- Phase and Frequency Correct PWM

Clear Timer on Compare Match Mode

CTCM (Clear Timer on Compare Match) is a basic counter with interrupts. The counter is automatically cleared to 0x0000 when the counter value, TCCNT register, matches the value loaded in the TCTOP register. The value of the TCTOP register can by dynamically updated through the WISHBONE register, or it can hold a static value that is assigned with IPexpress at the time of IP generation. The default value of the TCTOP register is 0xFFFF.

The data loaded into the timer counter to define the top counter value is double-registered. The WISHBONE host writes the data to TCTOPSET register, which is then automatically loaded onto the TCTOP register at the moment of auto-clear. Therefore, a new top value can be written to the TCTOPSET register after the overflow flag and during the counting-up to the top value. Updating the value of the TCTOP register will change the frequency of the output signal of the Timer/Counter.

Figure 9-16. Timer/Counter Output Waveform



Watchdog Timer Mode

Watchdog timers are used to monitor a system's operating behavior and provide a reset or interrupt when the system's microcontroller or embedded state machine is no longer operational. One scenario is for a microcontroller to reset the Watchdog Timer to 0x0000 before it begins a process. The microcontroller must complete the process and reset the Watchdog Timer before the timer reaches its terminal count. In the event that the microprocessor does not clear the timer quickly enough the Watchdog Timer asserts a strobe signaling time expired. The system uses the "time exceeded" strobe to gracefully recover the system.

Another way to use the Watchdog Timer is to periodically turn OFF system modules in order to save power. It can also be used to interact with the on-chip power controller of MachXO2.

The most commonly used ports of the Timer/Counter in Watchdog Timer Mode are the clock, reset and interrupt. Optionally, the WISHBONE interface can be used to read time stamps from the TCICR register and update the top value of the counter.

Fast PWM Mode

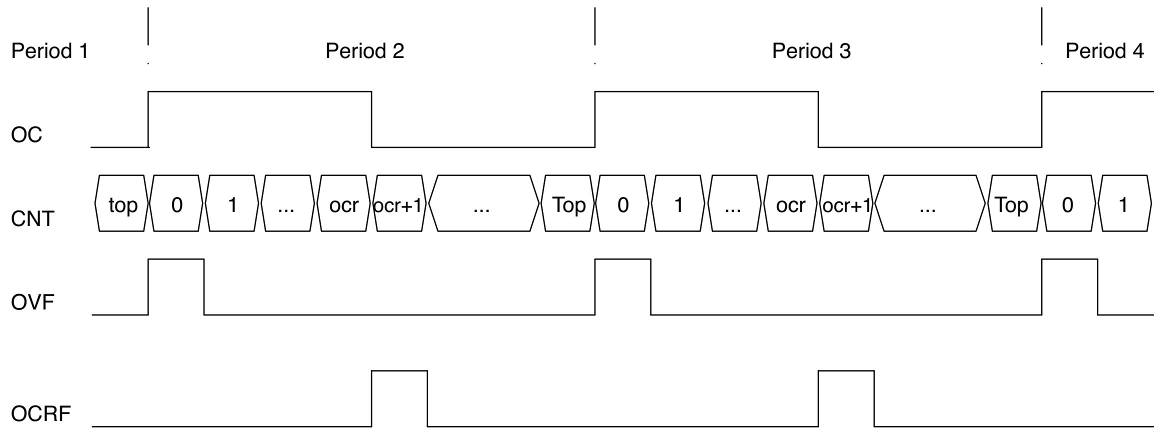
Pulse-Width Modulation (PWM) is a popular technique to digitally control analog circuits. PWM uses a rectangular pulse wave whose pulse width is modulated resulting in the variation of the average value of the waveform. Designers can vary the period and the duty cycle of the waveform by loading 16-bit digital values on the TCTOP register to define the top value of the counter, and the TCOCR register to provide a compare value for the output of the counter.

The output of the Timer/Counter is cleared when the counter value matches the top value that is loaded in the TCTOP register. The output is set when the value of the counter matches the compare value that is loaded into the TCOCR register. The clear/set functions can be inverted. This means that the output of the Timer/Counter is set when the counter value matches the top value and it is cleared when the value of the counter matches the compare value.

The interrupt line can be used for Overflow Flag (OVF) and Output Compare Flag (OCRF).

Figure 9-8 shows the PWM waveform generation. The output of the Timer/Counter is configured to be set when the counter matches the top value and clear when the counter matches the compare value.

Table 9-8. PWM Waveform Generation



Phase and Frequency Correct PWM Mode

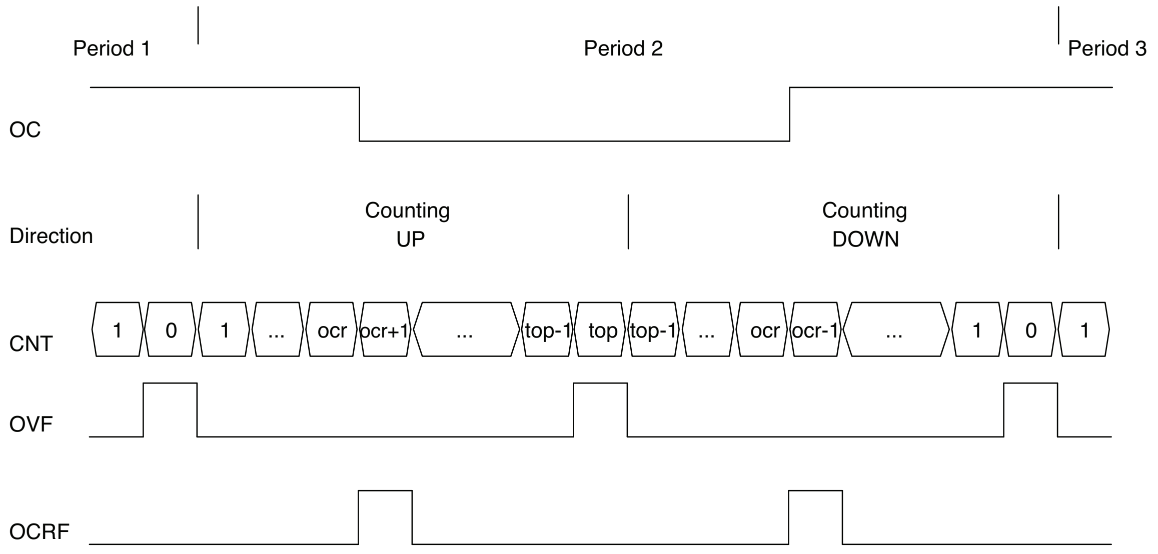
In phase and frequency correct PWM mode, the counting direction will change from up to down at the moment the counter is incrementing to the top value (top value minus 1). The moment that the counter is decrementing from 0x0001 to 0x0000, the following will occur:

1. TCTOP is updated with the value loaded in the TCTOPSET register
2. TCOCR is updated with the value loaded in the TCOCRSET register
3. Overflow TCSR[OVF] is asserted for one clock cycle

The output of the Timer/Counter is updated only when the counter value matches the compare value in TCOCR register. This occurs twice within one period. The first match occurs when the counter is counting up and the second match occurs when the counter is counting down. The Output Compare Flag TCSR[OCRF] is asserted when both matches occur. The output of the Timer/Counter is set on the first compare match and cleared on the second compare match. The order of set and clear can be inverted.

The mode allows users to adjust the frequency (based on the top value) and phase (based on the compare value) of the generated waveform.

Figure 9-17. Phase and Frequency Correct PWM Waveform Generation



Timer/Counter IP Signals

Table 9-9 documents the signals that are generated with the IP. Each signal has a description of the usage and how it should be connected in a design project.

Table 9-9. Timer/Counter – IP Signals

Signal Name	I/O	Width	Description
tc_clki	Input	1	Timer/Counter input clock signal. Can be connected to the on-chip oscillator. The clock signal is limited to 133 MHz.
tc_rstn	Input	1	This is an active-low reset signal, which resets the 16-bit counter.
tc_ic	Input	1	This is an active-high input capture trigger event, applicable for non-PWM modes with WISHBONE interface. If enabled, a rising edge of this signal will be detected and synchronized to capture the counter value (TCCNT Register) and make the value accessible to the WISHBONE interface by loading it into TCICR register. The common usage is to perform a time-stamp operation with the counter.
tc_int	Output	1	This is an interrupt signal, indicating the occurrence of a specific event such as Overflow, Output Compare Match, or Input Capture.
tc_oc	Output	1	Timer/Counter output signal

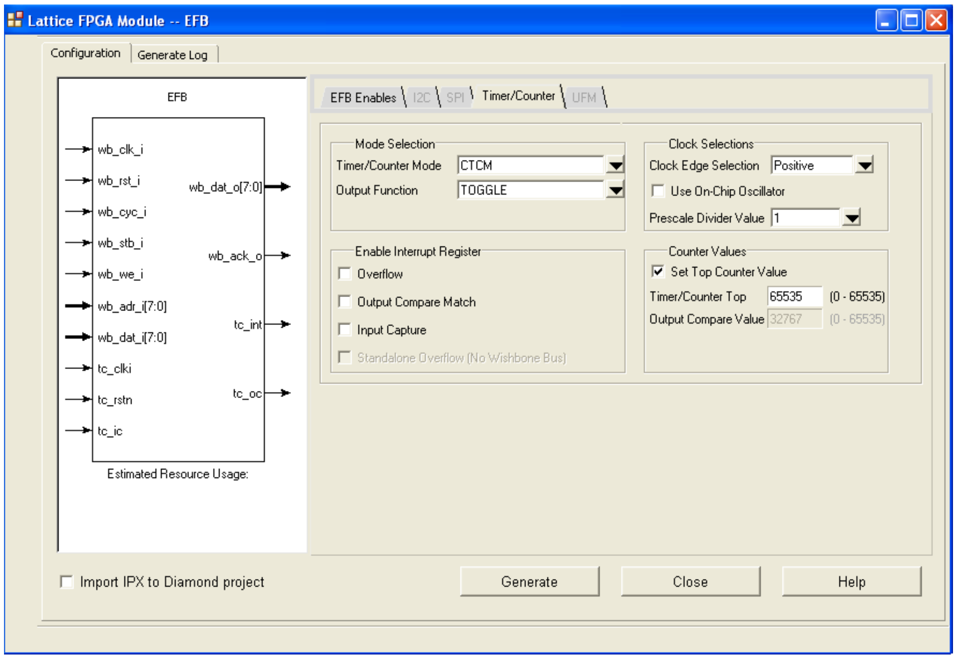
Configuring the Timer/Counter

IPexpress is used to configure the Timer/Counter. Selecting the Timer/Counter tab in the EFB GUI displays the configurable settings of the Timer/Counter core.

The Timer/Counter can be used with our without the WISHBONE interface. For usage without the WISHBONE interface, designers can select/enter values in the IPexpress EFB GUI. The values are programmed in the Timer/Counter registers with the programmable bitstream. Using the Timer/Counter with a WISHBONE interface enables users to dynamically update the register content through the WISHBONE interface. The main EFB GUI, EFB Enables tab, allows the user to make a selection between using the Timer/Counter with or without a WISHBONE screen shot.

Figure 9-18 shows an example of the Timer/Counter is configured for a specific design.

Figure 9-18. Configuring the Timer/Counter



Mode Selection
Timer/Counter Mode: CTCM
Output Function: TOGGLE

Clock Selections
Clock Edge Selection: Positive
☐ Use On-Chip Oscillator
Prescale Divider Value: 1

Enable Interrupt Register
☐ Overflow
☐ Output Compare Match
☐ Input Capture
☐ Standalone Overflow (No Wishbone Bus)

Counter Values
☑ Set Top Counter Value
Timer/Counter Top: 65535 (0 - 65535)
Output Compare Value: 32767 (0 - 65535)

☐ Import IPX to Diamond project

Timer/Counter Mode

This option allows the user to select one of the four operating modes.

- CTCM – Clear Timer on Compare Match
- WATCHDOG – Watchdog timer
- FASTPWM – Fast PWM
- PFCPWM – Phase and Frequency Correct PWM

This option can be updated dynamically by modifying the bits TCM[1:0] of the register TCCR1.

Output Function

Designers can select the function of the output signal (tc_oc) of the Timer/Counter IP. The available functions are:

- STATIC – The output of the Timer/Counter is static low
- TOGGLE – The output of the Timer/Counter toggles based on the conditions defined by the Timer/Counter Mode
- WAV_GENERATE – The waveform is generated by the Set/Clear on the conditions defined by the Timer/Counter Mode
- INV_WAV_GENERATE – The waveform is inverted

This option can be updated dynamically by modifying bits OCM[1:0] of the register TCCR1.

Clock Edge Selection

Designers can select the edge (positive or negative) of the input clock source as well as enable the usage of the on-chip oscillator. The selections are:

- PCLOCK – Positive edge of the clock from user logic
- POSC – Positive edge of the clock from the internal oscillator
- NCLOCK – Negative edge of the clock from user logic
- NOSC – Negative edge of the clock from the internal oscillator

This option can be updated dynamically by modifying the bits CLKSEL and CLKEDGE of the register TCCR0.

Pre-scale Divider Value
Pre-scale divider values (0, 1, 8, 64, 256, 1024) are provided to divide the input clock prior to reaching the 16-bit counter. This option can be updated dynamically by modifying the bits PRESCALE[2:0] of the register TCCR1.

Timer/Counter Top
Designers can select the top value of the counter. This option can be updated dynamically by modifying the bits TCTOPSET of the registers TCTOPSET1 and TCTOPSET0.

Output Compare Value
Designers can select the output compare value of the counter. This option can be updated dynamically by modifying the bits TCOCRSET the registers TCOCRSET1 and TCOCRSET0

Enable Interrupt Registers

- Overflow – An interrupt which indicates the counter matches the TCTOP0/1 register value. The interrupt is bit IRQOVF of the register TCIRQ. When enabled, indicates OVF was asserted. Write a ‘1’ to this bit to clear the interrupt. This option can be updated dynamically by modifying the bit IRQOVFEN of the register TCIRQEN.
- Output Compare Match – An interrupt which indicates when counter matches the TCOCR0/1 register value. The interrupt is bit IRQOCRF of the register TCIRQ. When enabled, indicates OCRF was asserted. Write a ‘1’ to this bit to clear the interrupt. This option can be updated dynamically by modifying the IRQOCRFEN bit of register TCIRQEN.
- Input Capture – An interrupt which indicates when the user asserts the TC_IC input signal. The interrupt is bit IRQICRF of the register TCIRQ. When enabled, indicates ICRF was asserted. Write a ‘1’ to this bit to clear the interrupt. This option can be updated dynamically by modifying IRQICRFEN bit of the register TCIRQEN.
- Standalone Overflow – Used without the WISHBONE interface and serves as the only available interrupt request.

MachXO2 Timer/Counter Usage Cases

1. Basic counter with interrupts
    a. Configure the Timer/Counter for Static or Dynamic operation. Dynamic operation enables the WISHBONE bus interface allowing a WISBONE Master to control the Timer/Counter.
    b. In the Timer Counter tab
        i. Configure the Timer/Counter Mode using the Mode Selection controls
            1. Select CTCM (Clear Timer on Compare Match) Mode
            2. Select the Output Function
                a. Hold static 0 (STATIC)
                b. Toggle (TOGGLE)
        ii. Update the Clock Selections
            1. Select the clock edge to which the Timer/Counter responds
                a. Positive (PCLOCK) or Negative (NCLOCK) edge
            2. Select the Clock Pre-scale Divider
        iii. Configure the Counter Values
            1. Enable the Top Counter Value
            2. Select a Timer/Counter Top value
        iv. Enable optional interrupts
2. Watchdog Timer
    a. Configure the Timer/Counter for Static or Dynamic operation. Dynamic operation enables the WISHBONE bus interface allowing a WISBONE Master to control the Timer/Counter.
    b. In the Timer Counter tab

        i. Configure the Timer/Counter Mode using the Mode Selection controls
            1. Select WATCHDOG mode
            2. Select Output Function
                a. Hold static 0 (STATIC)
        ii. Update the Clock Selections
            1. Select the clock edge to which the Timer/Counter responds
                a. Positive (PCLOCK) or Negative (NCLOCK) edge
            2. Select the Clock Pre-scale Divider
        iii. Configure the Counter Values
            1. Enable the Top Counter Value
            2. Select a Timer/Counter Top value
            3. Select an Output Compare Value
        iv. Enable interrupts (TC_INT)
            1. Select Output Compare Match
            2. Select Input Capture

3. PWM Output with variable duty cycle and period
    a. Configure the Timer/Counter for Static or Dynamic operation. Dynamic operation enables the WISHBONE bus interface allowing a WISBONE Master to control the Timer/Counter.
    b. In the Timer Counter tab
        i. Configure the Timer/Counter Mode using the Mode Selection controls
            1. Select the FASTPWM mode
            2. Select Output Function
                a. Hold static 0 (STATIC)
                b. PWM (WAVE_GENERATOR)
                c. Complement PWM (INV_WAVE_GENERATOR)
        ii. Update the Clock Selections
            1. Select the clock edge to which the Timer/Counter responds
                a. Positive (PCLOCK) or Negative (NCLOCK) edge
            2. Select the Clock Pre-scale Divider
        iii. Configure the PWM Values
            1. Select the PWM period (Timer Counter Top)
            2. Select the PWM duty cycle (Output Compare Value)
                a. Where the duty cycle is (Output Compare Value)/(Timer Counter Top): [(Timer Counter Top) – (Output Compare Value)]/(Timer Counter Top)
4. PWM output with 50:50 duty variable phase and period
    a. Configure the Timer/Counter for Static or Dynamic operation. Dynamic operation enables the WISHBONE bus interface allowing a WISBONE Master to control the Timer/Counter.
    b. In the Timer Counter tab
        i. Configure the Timer/Counter Mode using the Mode Selection controls
            1. Select FASTPWM
            2. Select Output Function
                a. Hold static 0 (STATIC)
                b. PWM (WAVE_GENERATOR)
                c. Complement PWM (INV_WAVE_GENERATOR)
        ii. Update the Clock Selections
            1. Select the clock edge to which the Timer/Counter responds
                a. Positive (PCLOCK) or Negative (NCLOCK) edge

2. Select the Clock Pre-scale Divider

iii. Configure the PWM Values

1. Select the PWM period (Timer Counter Top)
2. Select the Phase Adjustment (Output Compare Value)
    a. Where Phase Adjustment is (360 degrees)*(Output Compare Value)/ (Timer Counter Top)

Timer/Counter Design Tips

1. For information on the Timer/Counter register definitions and command sequences reference Timer/Counter section of TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.
2. Take note when dynamically turning off components for power savings; the EFB requires the MachXO2 internal oscillator to be enabled even if it is not the source of the WISHBONE Clock.

Flash Memory (UFM/Configuration) Access

Designers can access the Flash Memory Configuration Logic interface using the JTAG, SPI, I^{2}C, or WISHBONE interfaces. The MachXO2 Flash Memory consists of three sectors:

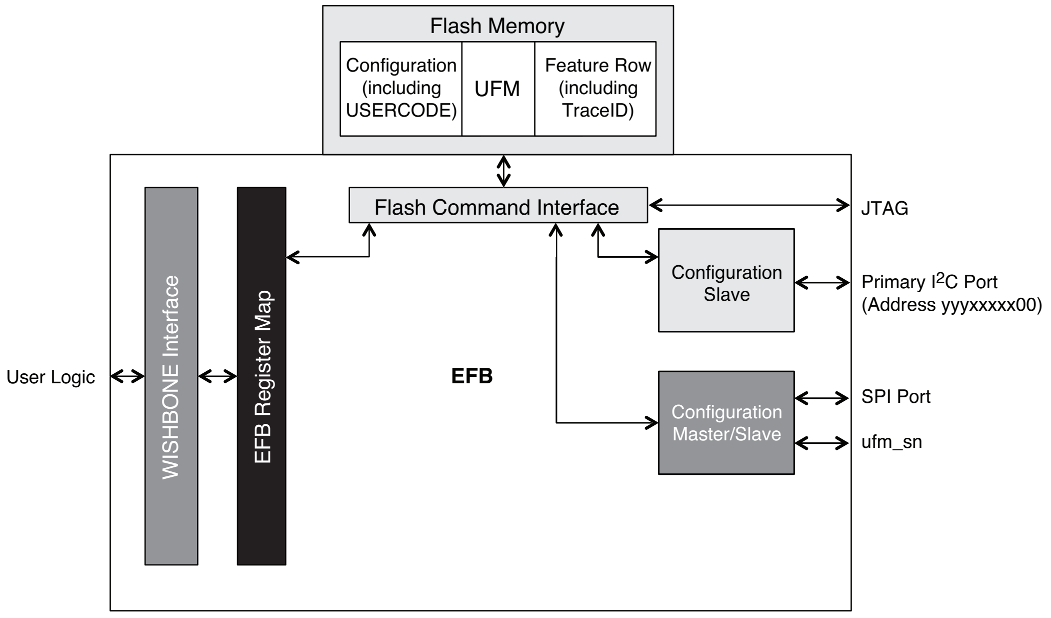
- User Flash Memory (UFM)
- Configuration
- Feature Row

The ports to access the Flash Memory and the block diagram are shown in Table 9-10.

Table 9-10. Flash Memory (UFM/Configuration) Access

	UFM	Configuration Flash
JTAG	Yes	Yes
Slave SPI Port with Chip Select (ufm_sn)	Yes	Yes
Slave SPI Port with Chip Select (spi_scsn)	No	No
Primary Slave I2C Port address (yyyxxxxx00)	Yes	Yes
Primary Slave I2C Port address (yyyxxxxx01)	No	No
Secondary Slave I2C Port address (yyyxxxxx10)	No	No
WISHBONE	Yes	Yes

Figure 9-19. Flash Memory (UFM/Configuration) Block Diagram


Flash Memory
Configuration (including USERCODE)
UFM
Feature Row (including TraceID)
Flash Command Interface
JTAG
Configuration Slave
Primary I2C Port (Address yyyxxxxx00)
User Logic
WISHBONE Interface
EFB Register Map
EFB
Configuration Master/Slave
SPI Port
ufm_sn

Flash Memory (UFM/Configuration) Access Ports

The Configuration Logic arbitrates access to the Flash Memory from the interfaces by the following priority. When higher priority ports are enabled Flash Memory access by lower priority ports is blocked.

1. JTAG Port – All MachXO2 devices have a JTAG port, which can be used to perform Read/Write operations to the Flash Memory (UFM/Configuration). The JTAG port is compliant with the IEEE 1149.1 and IEEE 1532 specifications. JTAG has the highest priority to the Flash Memory (UFM/Configuration).

2. Slave SPI Port – All MachXO2 devices have a Slave SPI port, which can be used to perform Read/Write operations to the Flash Memory (UFM/Configuration). Asserting the Flash Memory (UFM/Configuration) Slave Chip Select (ufm_sn) will enable access.
3. I^{2}C Primary Port – All MachXO2 devices have an I^{2}C port, which can be used to perform Read/Write operations to the Flash Memory (UFM/Configuration). The Primary I2C Port address is yyyxxxxx00 (Where 'y' and 'x' are user programmable).
4. WISHBONE Slave Interface – The WISHBONE Slave interface of the EFB module enables designers to access the Flash Memory (UFM/Configuration) from the FPGA user logic by creating a WISHBONE Master. In addition to the WISHBONE bus signals, an interrupt request output signal is brought to the FPGA fabric. An IRQ (wbc_ufm_irq) is provided to assist the WB Master to perform UFM/CF programming operations. It is configurable and provides information about the R/W FIFO, and arbitration errors.

   Note: To access the Flash Memory (UFM/Configuration) via WISHBONE in R1 devices, the hard SPI port or the Primary I2C port must be enabled. For more details, reference AN8086, Designing for Migration from MachXO2-1200-R1 to Standard (Non-R1) Devices.

Note: Enabling the Flash Memory (UFM/Configuration) Interface using Enable Configuration Interface command 0x74 Transparent Mode will temporarily disable certain features of the device including:

- *Power Controller*
- *GSR*
- *Hardened User SPI Port*
- *Hardened Primary User I^{2}C Port*

Functionality is restored after the Flash Memory (UFM/Configuration) Interface is disabled using Disable Configuration Interface command 0x26 followed by Bypass command 0xFF.

Interface to UFM

MachXO2-640 and higher density devices provide one sector of User Flash Memory (UFM). Designers can access the UFM sector through the Configuration Logic interface using the JTAG, Configuration SPI, Primary Configuration I^{2}C or WISHBONE interfaces. The UFM is a separate sector within the on-chip Flash Memory and is organized by pages. Each page is 128 bits (16 bytes). Table 9-11 shows the UFM resources in each device, represented in bits, bytes and pages.

Table 9-11. UFM Resources in MachXO2 Devices

	MachXO2-256	MachXO2-640	MachXO2-640U	MachXO2-1200	MachXO2-1200U	MachXO2-2000	MachXO2-2000U	MachXO2-4000	MachXO2-7000
UFM Bits	0	24,576	65,536	65,536	81,920	81,920	98,304	98,304	262,144
UFM Bytes	0	3,072	8,192	8,192	10,240	10,240	12,288	12,288	32,768
UFM Pages	0	192	512	512	640	640	768	768	2048

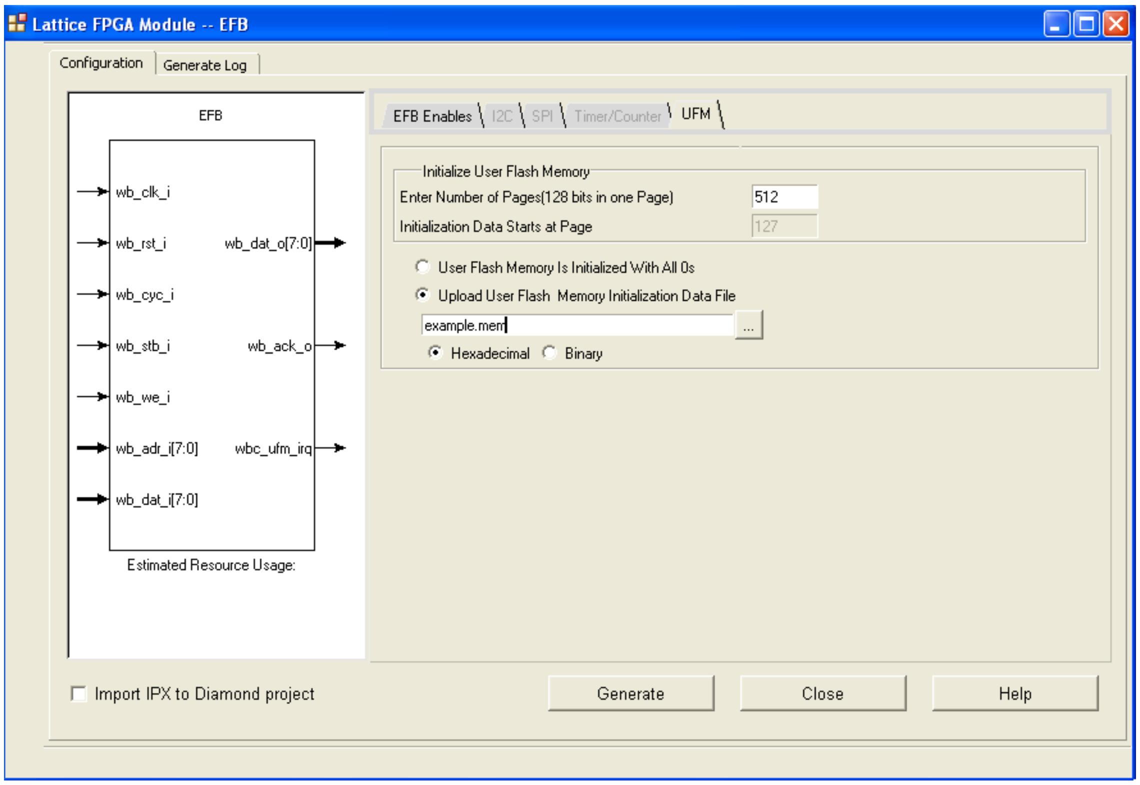
The UFM is a general purpose Flash Memory. Refer to the MachXO2 Family Data Sheet for the number of Programming/Erase Specifications. The UFM is typically used to store system-level data, Embedded Block RAM initialization data, or executable code for microprocessors and state-machines. Use the following tools to partition the UFM resource:

- IPexpress: Use IPexpress to enable the UFM and to initialize the memory
- Spreadsheet View (Diamond): The global sysConfig configuration options control the use of the UFM. Read TN1204, MachXO2 Programming and Configuration Usage Guide for a complete description of available options.

Initializing the UFM with IPexpress

Designers can initialize the UFM sector with system level non-volatile data and generate the EFB module via IPexpress. Selecting the UFM tab in the EFB GUI will display the configurable settings of the UFM.

Figure 9-20. Initializing the UFM Sector with IPexpress



Initialize User Flash Memory
Enter Number of Pages(128 bits in one Page): 512
Initialization Data Starts at Page: 127

User Flash Memory Is Initialized With All 0s [✓ Upload User Flash Memory Initialization Data File]

example.mem

[✓ Hexadecimal] Binary

Import IPX to Diamond project

Designers enter the number of pages to be pre-loaded with initialization data. Because initialization data is 'bottom justified,' the read-only field “Initialization Data Starts at Page” informs the designer with the address of the first page that is initialized. In the example used for this document, 512 pages of the UFM sector of MachXO2-2000 are to be initialized with the content of the memory file “example.mem”.

	UFM Utilization Example (MachXO2-2000) Initializing 512 Pages
Uninitialized Pages	Page 0
	Page 1
	...
First Page of Initial Data →	Page 128
	Page 129
	...
Last Page of Initial Data →	Page 639

The MachXO2-2000 has 640 pages in the UFM sector. The initialization data occupies the bottom (highest) addressable pages of the UFM, while the top (lowest) addressable pages remain uninitialized (all zeros). The format of the memory file is page-based and can be expressed in binary or hexadecimal format.

UFM Initialization Memory File

The initialization data file has the following properties and format:

Extension:	.mem
Format:	Binary, Hexadecimal
Data Width:	1 page (128 bits in one row of the file)
No. of Rows:	Less than or equal to the number of available pages

Example 1. Binary

```
1010…1010 (Placed at the starting page of the UFM initialization data, address = N)
1010…1010 (page address = N + 1)
1010…1010 (page address = N + 2)
   …
1010…1010 (Placed at the highest UFM page address)
```

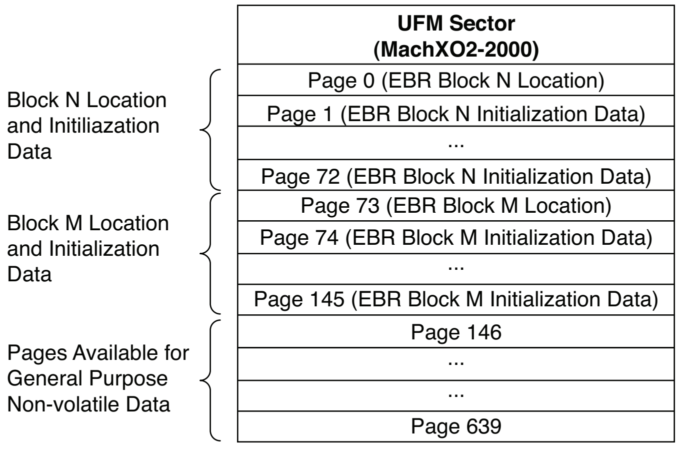
Example 2. Hexadecimal

```
A…B (Placed at the starting page of the UFM initialization data, address = N)
C…D (page address = N + 1)
E…F (page address = N + 2)
…
A…F (Placed at the highest UFM page address)
```

The most significant byte (byte 15) of the page is on the left side of the row. The least significant byte of the page (byte 0) is on the right side of the rowbit in a row is the left-most (i.e. bit 127), the least significant is right-most (bit 0)

EBR Initialization

For designers that choose to share the UFM sector with EBR initialization data, the sector map below should be referenced as an example when planning the design. Note at this time the EBR Mapping is not supported Diamond.

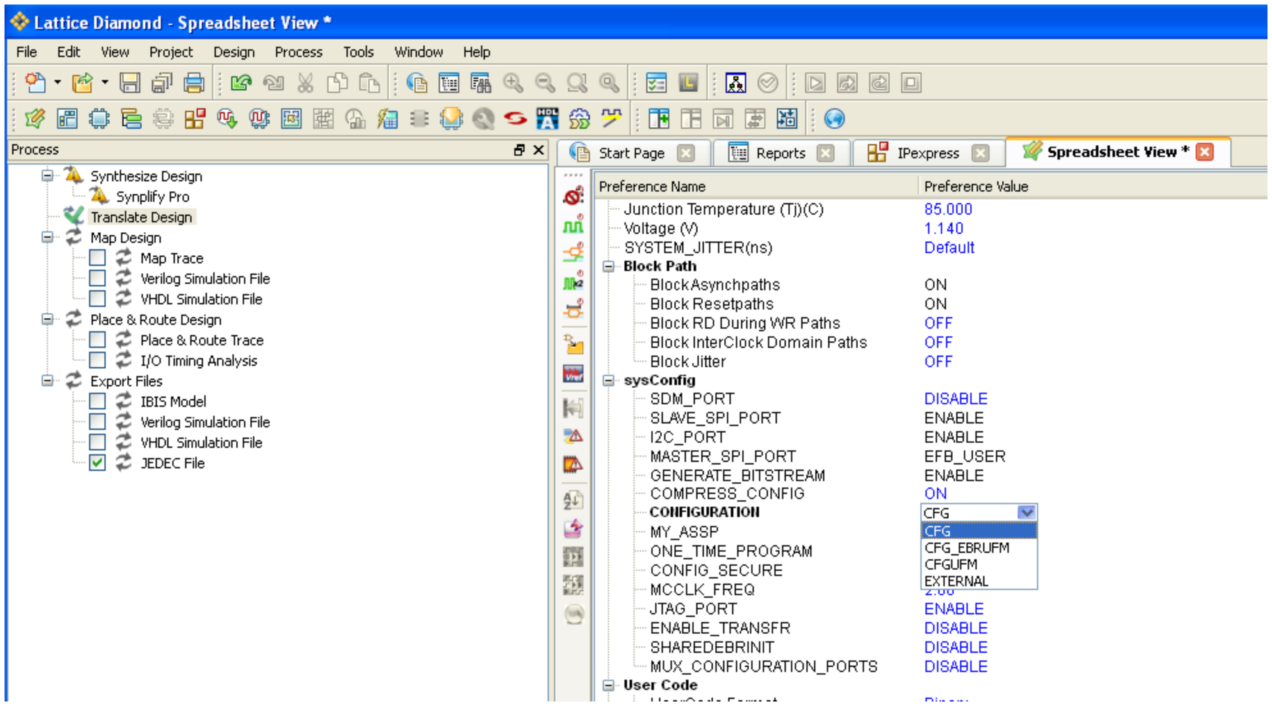

UFM Sector (MachXO2-2000)
Block N Location and Initiliazation Data
Page 0 (EBR Block N Location)
Page 1 (EBR Block N Initialization Data)
...
Page 72 (EBR Block N Initialization Data)
Block M Location and Initialization Data
Page 73 (EBR Block M Location)
Page 74 (EBR Block M Initialization Data)
...
Page 145 (EBR Block M Initialization Data)
Pages Available for General Purpose Non-volatile Data
Page 146
...
...
Page 639

Care must be taken when performing a Bulk-Erase operation to prevent the loss of EBR initialization data. Prior to erasing the UFM sector, designers should make a copy of the pages holding the EBR block location and EBR initialization data in a secondary memory resource. After the UFM sector is erased, the data can be written back into the UFM. Upon power-up or a reconfiguration command, the EBR initialization data is automatically loaded in their respective EBR blocks. The EBR Block Location data guides the configuration logic on the location of the EBR block.

UFM in Lattice Diamond Software

The UFM sector is one of the Flash Memory resources of the MachXO2. The CONFIGURATION option in the Diamond Spreadsheet view controls how UFM behaves. Each options impact to the UFM is described below.

Figure 9-21. UFM in Diamond Spreadsheet View



Preference Name	Preference Value
Junction Temperature (Tj)(C)	85.000
Voltage (V)	1.140
SYSTEM_JITTER(ns)	Default
Block Path	
BlockAsynchpaths	ON
Block Resetpaths	ON
Block RD During WR Paths	OFF
Block InterClock Domain Paths	OFF
Block Jitter	OFF
sysConfig	
SDM_PORT	DISABLE
SLAVE_SPI_PORT	ENABLE
I2C_PORT	ENABLE
MASTER_SPI_PORT	EFB_USER
GENERATE_BITSTREAM	ENABLE
COMPRESS_CONFIG	ON
CONFIGURATION	CFG
MY_ASSP	
ONE_TIME_PROGRAM	
CONFIG_SECURE	
MCCLK_FREQ	2.08
JTAG_PORT	ENABLE
ENABLE_TRANSFR	DISABLE
SHAREDEBRINIT	DISABLE
MUX_CONFIGURATION_PORTS	DISABLE
User Code	

Configuration Parameter Options

- CFG_EBRUFM – The SRAM configuration (not including EBR initialization data) is stored in the Configuration Flash. EBR initialization data, if any, is stored in the lowest page addresses (starting from Page 0) of the UFM sector. Any unoccupied UFM pages after the mapping of EBR initialization data is available as general purpose memory.
- CFG – The SRAM configuration (including EBR initialization data, if any) is stored in the Configuration Flash array and does not overflow into UFM. The full UFM sector is available as general purpose Flash memory.
- CFGUFM – The SRAM configuration (including EBR initialization data, if any) is stored in the Configuration Flash and is allowed to overflow into UFM. The UFM cannot be used as general purpose memory.
- EXTERNAL – The SRAM configuration (including EBR initialization data, if any) is stored in an external memory SPI memory. The full UFM sector is available as general purpose Flash memory.

Configuration Flash Memory

The WISHBONE interface of the EFB module allows a WISHBONE master to access the Configuration resources of MachXO2 devices. This is particularly useful for reading data from Configuration resources such as USERCODE and TraceID. A WISHBONE master can update the Configuration Flash memory using the Configuration Logic's transparent mode. The new design is active after power-up or a Configuration Refresh operation.

For more information on Programming the MachXO2, reference TN1204, MachXO2 Programming and Configuration Usage Guide and TN1246 Using User Flash Memory and Hardened Control Functions in MachXO2 Devices Reference Guide.

For more information on the TraceID, reference TN1207, Using TraceID in MachXO2 Devices.

Table 9-12. Configuration Flash Resources in MachXO2 Devices

	MachXO2-256	MachXO2-640	MachXO2-640U	MachXO2-1200	MachXO2-1200U	MachXO2-2000	MachXO2-2000U	MachXO2-4000	MachXO2-7000
CFG Bits	73,600	147,328	278,400	278,400	409,344	409,344	737,024	737,024	1,179,136
CFG Bytes	9,200	18,416	34,800	34,800	51,168	51,168	92,128	92,128	147,392
CFG Pages	575	1,151	2,175	2,175	3,198	3,198	5,758	5,758	9,212

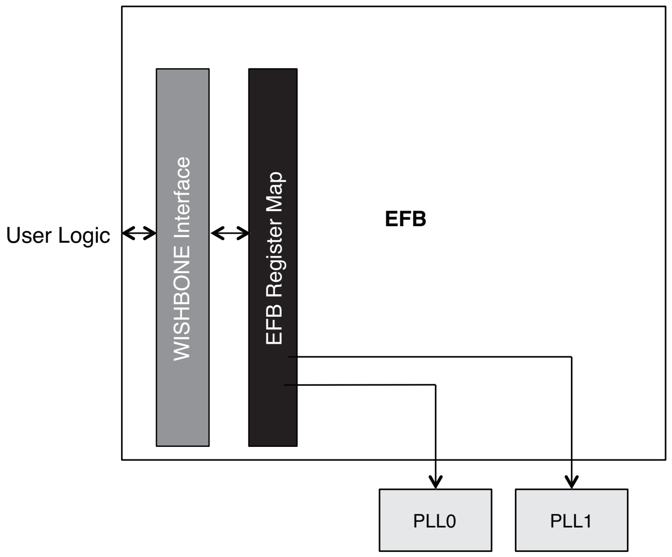
Flash Memory (UFM/Configuration) Design Tips

1. For information on the Flash Memory (UFM/Configuration) register definitions and command sequences reference Flash Memory (UFM/Configuration) section of TN1246, Reference Guide for Using User Flash Memory and Hardened Control Functions in MachXO2 Devices.
2. For more information Programming the MachXO2, reference TN1204, MachXO2 Programming and Configuration Usage Guide.
3. For more information on the TraceID, reference TN1207, Using TraceID in MachXO2 Devices.
4. Take note when dynamically turning off components for power savings; The EFB requires the MachXO2 internal oscillator to be enabled even if it is not the source of the WISHBONE Clock.
5. It is recommended when accessing the Configuration Flash the Feature Row be Read Only after initial programming as it contains the persistent settings for the Configuration ports
6. The Configuration Flash Chip Select (ufm_sn) is enabled when the UFM and SPI functions are enabled. Tie ufm_sn inactive (i.e. high) if you are not using the Slave SPI interface to access the Configuration Flash/UFM.
7. The buses used to access the Flash Memory and UFM have a priority. The bus priority is, from highest to lowest, the JTAG Port, Slave SPI Port, I^2C Primary Port, and WISHBONE Slave Interface. When higher priority ports are enabled Flash Memory access by lower priority ports is blocked. You must define a process that prevents simultaneous access to the Configuration Flash/UFM by masters on each configuration port.
8. The Primary I^2C port cannot be used for both UFM/Configuration access and user functions in the same design.
9. Enabling Flash Memory (UFM/Configuration) Interface using Enable Configuration Interface command 0x74 Transparent Mode will temporarily disable the Power Controller, GSR, Hardened User SPI port, Functionality is restored after the Flash Memory (UFM/Configuration) Interface is disabled using Disable Configuration Interface command 0x26 followed by Bypass command 0xFF.
10. In Flash memory, ‘0’ defines erased, ‘1’ defines written
11. Smallest unit for a Write operation (bits => 1) is 1 page (16 bytes)
12. Smallest unit for an Erase operation (bits => 0) is one sector (There are only two available Flash sectors: Configuration and UFM)
13. The UFM has a limited number of erase/programming cycles. The number of cycles is described in DS1035. Lattice recommends storing static, or data that varies infrequently in the UFM.
14. There are a number of Flash Memory (UFM/Configuration) Reference designs on the Lattice website (www.latticesemi.com/products/intellectualproperty/aboutreferencedesigns.cfm) including:
    a.RD1126: RAM-Type Interface for Embedded User Flash Memory
    b.UG57: MachXO2 Programming via WISHBONE Demo

Interface to Dynamic PLL Configuration Settings

The WISHBONE interface of the EFB module can be used to dynamically update configurable settings of the Phase Locked Loops (PLLs) in MachXO2 devices.

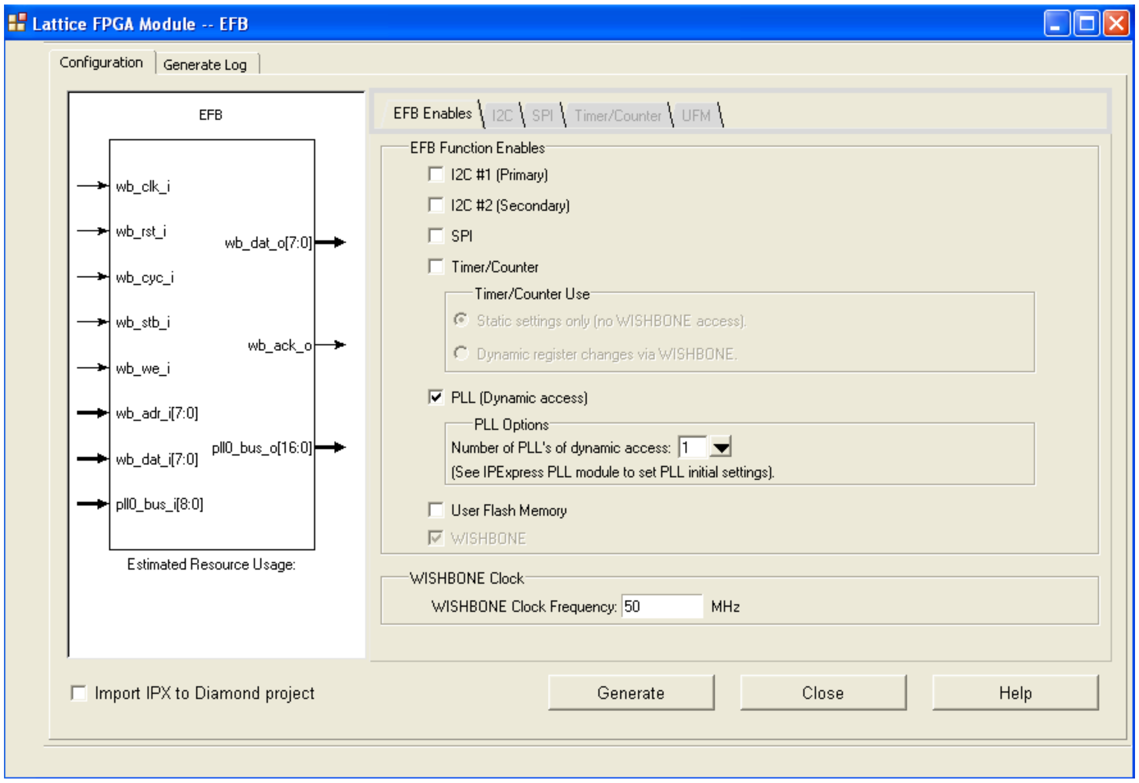
Figure 9-22. EFB Interface to Dynamic PLL



There can be up to two PLLs in a MachXO2 device. PLL0 has an address range from 0x00 to 0x1F in the EFB register map. PLL1 (if present) has an address range from 0x20 to 0x3F in the EFB register map. Reference TN1199, MachXO2 sysCLOCK PLL Design and Usage Guide, for details on PLL configuration bits and recommended usage.

Users can enable the WISHBONE interface to the PLL components in IPexpress, EFB GUI as shown in Figure 9-23.

Figure 9-23. Interface to Dynamic PLL Configuration Settings



EFB Function Enables
- [ ] I2C #1 (Primary)
- [ ] I2C #2 (Secondary)
- [ ] SPI
- [ ] Timer/Counter

Timer/Counter Use
- (●) Static settings only (no WISHBONE access).
- ( ) Dynamic register changes via WISHBONE.

- [✓] PLL (Dynamic access)

PLL Options
Number of PLL's of dynamic access: 1
(See IPExpress PLL module to set PLL initial settings).

- [ ] User Flash Memory
- [✓] WISHBONE

WISHBONE Clock
WISHBONE Clock Frequency: 50 MHz

- [ ] Import IPX to Diamond project

Enabling the interface to dynamically control PLL settings through the WISHBONE interface will generate an IP with the following ports, which are used for dedicated connections to the PLL(s).

Table 9-13 documents the signals that are generated with the IP. Each signal has a description of the usage and how it should be connected in a design project.

Table 9-13. PLL Interface – IP Signals

Signal Name	I/O	Width	Description
pll0_bus_i	Input	9	Input data and control bus. Users must connect this bus only to a PLL component that is instantiated in the design.
pll0_bus_i	Input	9	Input data and control bus. Users must connect this bus only to a PLL component that is instantiated in the design.
pll0_bus_o	Output	17	Output data and control bus. Users must connect this bus only to a PLL component that is instantiated in the design.
pll0_bus_1	Output	17	Output data and control bus. Users must connect this bus only to a PLL component that is instantiated in the design.

Technical Support Assistance

Hotline:	1-800-LATTICE (North America)
	+1-503-268-8001 (Outside North America)
e-mail:	techsupport@latticesemi.com
Internet:	www.latticesemi.com

Revision History

Date	Version	Change Summary
November 2010	01.0	Initial release.
January 2011	01.1	Updated for ultra-high I/O (“U”) devices.
June 2011	02.0	Expanded WISHBONE register definitions for I^{2}C, SPI, TC and UFM/Configuration blocks.
		Added Master I^{2}C and Master SPI WISHBONE flow diagrams.
		Expanded UFM/Configuration Command tables.
		Added UFM/Configuration Command descriptions
		Added I^{2}C and SPI timing diagrams.
		Added Configuration Flash resources table.
		Added UFM/Configuration performance table.
		Added Appendices A, B and C.
		Various minor corrections, additions and refinements to the text.
July 2011	02.1	Added diagram: I^{2}C Slave Read/Write Example (via WISHBONE).
		Added references to AN8086, Designing for Migration from MachXO2-1200-R1 to Standard (R-1) Devices, throughout the document.
August 2011	02.2	Added diagram: Basic Configuration Flash Update Example.
		Removed 'Erase_DONE' command.
		Clarified SPITXDR and WBRESET timing.
September 2011	02.3	Clarified WISHBONE timing for wait states.
		Corrected Flash Check Status checking.
		Modified I^{2}C Master Read/Write Example (Figure 9-10).
		Clarified R1 UFM/Config Access.
		Clarified Flash address auto-increment functionality.
		Added notation for non-simulation commands (Table 9-57).
October 2011	02.4	Corrected SPI Clock Prescale equation.
		Clarified user SPI port restoration command sequence following Configuration Enable.
		Added 0xC6 (Enable Offline Configuration) and 0xFF (Bypass).
December 2011	02.5	Noted operational delays which must be observed following wb_rst_i and UFM/Config Enable, Erase and Programming commands.
		Added new figure: SPI Master Read/Write Example (via WISHBONE) – Production Silicon.
		GSR and Power Controller functionality is suspended during UFM/Configuration Interface Enable (0x74)
		BYPASS command (0xFF) must now follow all Configuration Disable (0x26) commands.
		Added Feature Row capability to Erase Flash (0x0E) command.
		Corrected examples for Read Config Flash (0x73)
January 2012	02.6	Check Busy Flag (0xF0) table, updated example
		Added operational delays following Enable Configuration Interface command in Appendix B examples.
January 2012	02.7	Corrected Data Format (Binary) for Check Busy Flag (0xF0).
January 2012	02.8	Check Busy Flag (0xF0) Data Format changed from: B: bit 0: Busy Flag (1= busy) to B: bit 7: Busy Flag (1= busy)

Date	Version	Change Summary
January 2012	02.9	Updated UFM and Configuration Resource tables.
		Changed BYPASS operands to FF FF FF.
February 2012	03.0	Updated document with new corporate logo.
February 2012	03.1	Fixed Busy flag read example.
June 2012	04.0	Split into two documents: TN1205 Usage Guide and TN1246 Reference Guide.
October 2012	04.1	Added restriction: Primary port can be used as Configuration/UFM port or as a user port, but not both.
		Added restriction: Primary I^{2}C port is unavailable while in ISC_ENABLE_X (transparent) configuration access mode.