



# **MachXO4 EFB Module IP**

IP Version: v1.2.0

## **User Guide**

FPGA-IPUG-02287-1.1

December 2025

## 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, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. 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. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. 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.

## Inclusive Language

This document was created consistent with Lattice Semiconductor's inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice's inclusive language [FAQ 6878](#) for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

# Contents

|   |    |
|---|----|
| Contents .....  | 3  |
| Abbreviations in This Document.....                   | 8  |
| 1. Introduction .....                                 | 9  |
| 1.1. Overview of the IP .....                         | 9  |
| 1.2. Quick Facts .....                                | 9  |
| 1.3. IP Support Summary .....                         | 9  |
| 1.4. Features .....                                   | 10 |
| 1.5. Licensing and Ordering Information .....         | 10 |
| 1.6. Minimum Device Requirements .....                | 10 |
| 1.7. Naming Conventions .....                         | 10 |
| 1.7.1. Nomenclature.....                              | 10 |
| 1.7.2. Signal Names .....                             | 10 |
| 1.7.3. Attribute Names .....                          | 10 |
| 2. Functional Description.....                        | 11 |
| 2.1. IP Architecture Overview .....                   | 11 |
| 2.2. Clocking .....                                   | 12 |
| 2.2.1. Clocking Overview .....                        | 12 |
| 2.3. Reset .....                                      | 13 |
| 2.3.1. Reset Overview .....                           | 13 |
| 2.4. User Interfaces .....                            | 14 |
| 2.4.1. WISHBONE Bus Interface (Register Access) ..... | 14 |
| 2.4.2. I2C IP Cores.....                              | 15 |
| 2.4.3. SPI IP Core .....                              | 16 |
| 2.4.4. WISHBONE Controller (for PLL) .....            | 17 |
| 2.5. Flash Access Interface .....                     | 17 |
| 3. IP Parameter Description.....                      | 19 |
| 3.1. EFB Enables .....                                | 19 |
| 3.2. I2C .....  | 19 |
| 3.3. SPI.....   | 20 |
| 3.4. Timer/Counter.....                               | 22 |
| 3.4.1. Watchdog Timer Mode .....                      | 23 |
| 3.4.2. Clear Timer on Compare Match Mode.....         | 23 |
| 3.4.3. Fast PWM Mode.....                             | 24 |
| 3.4.4. Phase and Frequency Correct PWM Mode .....     | 24 |
| 3.5. UFM.....   | 25 |
| 4. Signal Description .....                           | 26 |
| 5. Register Description .....                         | 29 |
| 5.1. I2C Registers.....                               | 29 |
| 5.2. SPI Registers .....                              | 34 |
| 5.3. Timer/Counter Registers .....                    | 42 |
| 5.4. Flash Access Registers .....                     | 48 |
| 6. Command and Data Transfers to Flash Space .....    | 53 |
| 6.1. Command Summary by Application.....              | 53 |
| 6.2. Command Descriptions by Command Code.....        | 55 |
| 6.3. Command Framing.....                             | 65 |
| 6.3.1. I2C Framing.....                               | 65 |
| 6.3.2. SPI Framing.....                               | 66 |
| 6.3.3. WISHBONE Framing.....                          | 67 |
| 7. Designing with the IP .....                        | 69 |
| 7.1. Generating and Instantiating the IP .....        | 69 |
| 7.1.1. Generated Files and File Structure .....       | 72 |
| 7.2. Design Implementation.....                       | 72 |

|  |                                     |    |
|--|-------------------------------------|----|
| 7.3.                                   | Timing Constraints .....            | 72 |
| 7.4.                                   | Physical Constraints .....          | 72 |
| 7.5.                                   | Specifying the Strategy.....        | 72 |
| 7.6.                                   | Running Functional Simulation ..... | 73 |
| 7.6.1.                                 | Simulation Results .....            | 75 |
| 7.7.                                   | Example Designs (Simulation) .....  | 77 |
| 7.7.1.                                 | I2C Core Transaction Example.....   | 77 |
| 7.7.2.                                 | Flash Write and Read Example.....   | 80 |
| 7.7.3.                                 | SPI Core Transaction Example .....  | 83 |
| 8.                                     | Design Considerations .....         | 86 |
| Appendix A. Resource Utilization ..... |                                     | 87 |
| References.....                        |                                     | 88 |
| Technical Support Assistance .....     |                                     | 89 |
| Revision History.....                  |                                     | 90 |

## Figures

|  |    |
|--|----|
| Figure 2.1. EFB Module IP Block Diagram .....  | 11 |
| Figure 2.2. EFB Module IP Clock Overview .....                                       | 12 |
| Figure 2.3. EFB Module IP Reset Overview .....                                       | 13 |
| Figure 2.4. EFB Module WISHBONE Interface .....                                      | 14 |
| Figure 2.5. EFB Module I2C Core Interface .....                                      | 15 |
| Figure 2.6. EFB Module SPI Core Interface .....                                      | 16 |
| Figure 2.7. EFB Interface to Dynamic PLL .....                                       | 17 |
| Figure 2.8. EFB Module Flash Interface .....   | 18 |
| Figure 3.1. Timer/Counter CTCM Output Waveform .....                                 | 24 |
| Figure 3.2. Fast PWM Mode Waveform Generation .....                                  | 24 |
| Figure 3.3. Phase/Frequency Correct PWM Mode Waveform Generation .....               | 25 |
| Figure 5.1. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=0) .....              | 37 |
| Figure 5.2. SPI Control Timing (SPICR2[CPHA]=1, SPICR1[TXEDGE]=0) .....              | 37 |
| Figure 5.3. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=1) .....              | 38 |
| Figure 5.4. SPI Control Timing (SPICR2[CPHA]=1, SPICR1[TXEDGE]=1) .....              | 38 |
| Figure 5.5. Target SPI Dummy Byte Response (SPICR2[SDBRE]) Timing .....              | 38 |
| Figure 6.1. I2C Read Device ID Example .....   | 66 |
| Figure 6.2. SSPI Read Device ID Example .....  | 67 |
| Figure 6.3. WISHBONE Read Device ID Example .....                                    | 68 |
| Figure 7.1. Module/IP Block Wizard .....   | 69 |
| Figure 7.2. IP Configuration .....   | 70 |
| Figure 7.3. Check Generated Result .....   | 71 |
| Figure 7.4. Simulation Wizard .....  | 73 |
| Figure 7.5. Add and Reorder Source .....   | 74 |
| Figure 7.6. Simulation Waveform .....  | 75 |
| Figure 7.7. Sample Testbench Overview .....  | 76 |
| Figure 7.8. Sample Testbench Output .....  | 77 |
| Figure 7.9. I2C Controller Read/Write Example (through WISHBONE) .....               | 78 |
| Figure 7.10. I2C Target Read/Write Example (through WISHBONE) .....                  | 79 |
| Figure 7.11. I2C Sample Configuration (Primary and Secondary I2C Enabled) .....      | 80 |
| Figure 7.12. UFM Access Sample Configuration (through Wishbone and I2C Access) ..... | 83 |
| Figure 7.13. SPI Controller Read/Write Example (through WISHBONE) .....              | 84 |
| Figure 7.14. SPI Sample Configuration .....  | 85 |

## Tables

|   |    |
|---|----|
| Table 1.1. Summary of the EFB Module IP .....                       | 9  |
| Table 1.2. EFB Module IP Support Readiness .....                    | 9  |
| Table 2.1. User Interfaces and Supported Protocols .....            | 14 |
| Table 2.2. Hardened SPI Functionality .....                         | 16 |
| Table 3.1. EFB Enables Attributes .....                             | 19 |
| Table 3.2. I2C Attributes .....                                     | 19 |
| Table 3.3. SPI Attributes .....                                     | 20 |
| Table 3.4. Timer/Counter Attributes .....                           | 22 |
| Table 3.5. UFM Attributes .....                                     | 25 |
| Table 4.1. EFB Module IP Ports .....                                | 26 |
| Table 5.1. I2C Functions Register Map .....                         | 29 |
| Table 5.2. I2C Control (Primary/Secondary) .....                    | 29 |
| Table 5.3. I2C Command (Primary/Secondary) .....                    | 30 |
| Table 5.4. I2C Clock Prescale 0 (Primary/Secondary) .....           | 30 |
| Table 5.5. I2C Clock Prescale 1 (Primary/Secondary) .....           | 31 |
| Table 5.6. I2C Transmit Data Register (Primary/Secondary) .....     | 31 |
| Table 5.7. I2C Status (Primary/Secondary) .....                     | 31 |
| Table 5.8. I2C General Call Data Register (Primary/Secondary) ..... | 32 |
| Table 5.9. I2C Receive Data Register (Primary/Secondary) .....      | 33 |
| Table 5.10. I2C Interrupt Status (Primary/Secondary) .....          | 33 |
| Table 5.11. I2C Interrupt Enable (Primary/Secondary) .....          | 33 |
| Table 5.12. SPI Functions Register Map .....                        | 34 |
| Table 5.13. SPI Control 0 .....                                     | 34 |
| Table 5.14. SPI Control 1 .....                                     | 35 |
| Table 5.15. SPI Control 2 .....                                     | 35 |
| Table 5.16. SPI Clock Pre-scale .....                               | 39 |
| Table 5.17. SPI Controller Chip Select .....                        | 39 |
| Table 5.18. SPI Transmit Data Register .....                        | 39 |
| Table 5.19. SPI Status .....  | 40 |
| Table 5.20. SPI Receive Data Register .....                         | 40 |
| Table 5.21. SPI Interrupt Status .....                              | 40 |
| Table 5.22. SPI Interrupt Enable .....                              | 41 |
| Table 5.23. Timer/Counter Functions Register Map .....              | 42 |
| Table 5.24. Timer/Counter Control .....                             | 42 |
| Table 5.25. Timer/Counter Control 1 .....                           | 43 |
| Table 5.26. Timer/Counter Set Top Counter Value 0 .....             | 44 |
| Table 5.27. Timer/Counter Set Top Counter Value 1 .....             | 44 |
| Table 5.28. Timer/Counter Set Compare Counter Value 0 .....         | 44 |
| Table 5.29. Timer/Counter Set Compare Counter Value 1 .....         | 44 |
| Table 5.30. Timer/Counter Control 2 .....                           | 44 |
| Table 5.31. Timer/Counter Counter Value 0 .....                     | 45 |
| Table 5.32. Timer/Counter Counter Value 1 .....                     | 45 |
| Table 5.33. Timer/Counter Current Top Counter Value 0 .....         | 45 |
| Table 5.34. Timer/Counter Current Top Counter Value 1 .....         | 45 |
| Table 5.35. Timer/Counter Current Compare Counter Value 0 .....     | 46 |
| Table 5.36. Timer/Counter Current Compare Counter Value 1 .....     | 46 |
| Table 5.37. Timer/Counter Current Capture Counter Value 0 .....     | 46 |
| Table 5.38. Timer/Counter Current Capture Counter Value 1 .....     | 46 |
| Table 5.39. Timer/Counter Status Register .....                     | 47 |
| Table 5.40. Timer/Counter Interrupt Status .....                    | 47 |
| Table 5.41. Timer/Counter Interrupt Enable .....                    | 48 |
| Table 5.42. Flash Memory Functions Register Map .....               | 48 |

|   |    |
|---|----|
| Table 5.43. Flash Memory Control .....                                | 48 |
| Table 5.44. Flash Memory Transmit Data .....                          | 49 |
| Table 5.45. Flash Memory Status .....                                 | 49 |
| Table 5.46. Flash Memory Receive Data .....                           | 50 |
| Table 5.47. Flash Memory Interrupt Status .....                       | 50 |
| Table 5.48. Flash Memory Interrupt Enable .....                       | 51 |
| Table 5.49. Unused (Reserved) Register .....                          | 51 |
| Table 5.50. EFB Interrupt Source .....                                | 52 |
| Table 6.1. UFM (Sector 1) Commands .....                              | 53 |
| Table 6.2. Configuration Flash (Sector 0) Commands .....              | 53 |
| Table 6.3. Non-Volatile Register (NVR) Commands .....                 | 54 |
| Table 6.4. Erase Flash (0x0E) .....                                   | 55 |
| Table 6.5. Read TraceID Code (0x19) .....                             | 55 |
| Table 6.6. Disable Configuration Interface (0x26) .....               | 55 |
| Table 6.7. Read Status Register (0x3C) .....                          | 56 |
| Table 6.8. Reset Flash Address (0x46) .....                           | 56 |
| Table 6.9. Reset UFM Address (0x47) .....                             | 57 |
| Table 6.10. Program DONE (0x5E) .....                                 | 57 |
| Table 6.11. Program Configuration Flash (0x70) .....                  | 57 |
| Table 6.12. Read Configuration Flash (0x73) (I2C) .....               | 57 |
| Table 6.13. Read Configuration Flash (0x73) (WISHBONE) .....          | 58 |
| Table 6.14. Enable Configuration Interface (Transparent) (0x74) ..... | 58 |
| Table 6.15. Refresh (0x79) .....                                      | 59 |
| Table 6.16. STANDBY (0x7D) .....                                      | 59 |
| Table 6.17. Set Address (0xB4) .....                                  | 59 |
| Table 6.18. Read USERCODE (0xC0) .....                                | 59 |
| Table 6.19. Program USERCODE (0xC2) .....                             | 60 |
| Table 6.20. Enable Configuration Interface (Offline) (0xC6) .....     | 60 |
| Table 6.21. Program UFM (0xC9) .....                                  | 60 |
| Table 6.22. Read UFM (0xCA) (I2C) .....                               | 60 |
| Table 6.23. Read UFM (0xCA) (WISHBONE) .....                          | 61 |
| Table 6.24. Erase UFM (0xCB) .....                                    | 61 |
| Table 6.25. Program SECURITY (0xCE) .....                             | 62 |
| Table 6.26. Program SECURITY PLUS (0xCF) .....                        | 62 |
| Table 6.27. Read Device ID Code (0xE0) .....                          | 62 |
| Table 6.28. Device ID .....   | 62 |
| Table 6.29. Verify Device ID Code (0xE2) .....                        | 62 |
| Table 6.30. Program Feature (0xE4) .....                              | 63 |
| Table 6.31. Read Feature Row (0xE7) .....                             | 63 |
| Table 6.32. Check Busy Flag (0xF0) .....                              | 63 |
| Table 6.33. Program FEABITs (0xF8) .....                              | 63 |
| Table 6.34. Read FEABITs (0xFB) .....                                 | 64 |
| Table 6.35. Bypass (Null Operation) (0xFF) .....                      | 65 |
| Table 6.36. Command Framing Protocol by Interface .....               | 65 |
| Table 6.37. Command Framing Protocol by Interface .....               | 67 |
| Table 7.1. Generated File List .....                                  | 72 |
| Table 7.2. Write Two UFM Pages (WISHBONE) .....                       | 80 |
| Table 7.3. Read One UFM Page (WISHBONE) .....                         | 81 |
| Table 7.4. Read Two UFM Pages (WISHBONE) .....                        | 82 |
| Table A.1. LFMXO4-110HE-6BBG484I Device Resource Utilization .....    | 87 |

## Abbreviations in This Document

A list of abbreviations used in this document.

| Abbreviation | Definition                                     |
|--------------|--|
| ACK          | Acknowledgement                                |
| APB          | Advanced Peripheral Bus                        |
| CFG          | Configuration                                  |
| CTCM         | Clear Timer on Compare Match                   |
| DUT          | Device Under Test                              |
| EFB          | Embedded Functional Block                      |
| FIFO         | First In, First Out                            |
| FPGA         | Field Programmable Gate Array                  |
| GUI          | Graphical User Interface                       |
| GPIO         | General Purpose Input/Output                   |
| I/O          | Input/Output                                   |
| I2C          | Inter-Integrated Circuit                       |
| ID           | Identification                                 |
| IP           | Intellectual Property                          |
| IRQ          | Interrupt Request                              |
| JTAG         | Joint Test Action Group                        |
| LSE          | Lattice Synthesis Engine                       |
| LSB          | Least Significant Bit                          |
| LUT          | Look-Up Table                                  |
| MDF          | Mode Fault                                     |
| MSB          | Most Significant Bit                           |
| NACK         | No Acknowledgement                             |
| NVR          | Non-Volatile Registers                         |
| OEM          | Original Equipment Manufacturer                |
| PFCPWM       | Phase/Frequency Correct Pulse Width Modulation |
| PLL          | Phase-Locked Loop                              |
| PWM          | Pulse Width Modulation                         |
| RO           | Read-Only                                      |
| RW           | Read-Write                                     |
| SPI          | Serial Peripheral Interface                    |
| SRAM         | Static Random Access Memory                    |
| UFM          | User Flash Memory                              |
| WO           | Write-Only                                     |



# 1. Introduction

## 1.1. Overview of the IP

The embedded functional block (EFB) module provides a soft-IP implementation of the hardened control functions in the MachXO4™ FPGA devices. These hardened control functions ease design implementation and conserve general-purpose resources like LUTs, registers, clocks, and routing. The hardened control functions are physically located in the EFB.

The EFB includes the following control functions:

- Two I2C cores
- One SPI core
- One 16-bit Timer/Counter
- Interface to flash memory
- Interface to the dynamic PLL configuration settings
- Interface to the on-chip power controller through I2C and SPI

## 1.2. Quick Facts

**Table 1.1. Summary of the EFB Module IP**

|                      |                          |   |
|----------------------|--------------------------|---|
| IP Requirements      | Supported Devices        | MachXO4™  |
|                      | IP Changes               | For a list of changes to the IP, refer to the <a href="#">MachXO4 EFB Module IP Release Notes (FPGA-RN-02079)</a> . |
| Resource Utilization | Supported User Interface | WISHBONE, APB, I2C, and SPI   |
|                      | Resources                | Refer to <a href="#">Appendix A. Resource Utilization</a>   |
| Design Tool Support  | Lattice Implementation   | IP Core v1.2.0 – Lattice Radiant™ software 2025.2   |
|                      | Synthesis                | Lattice Synthesis Engine (LSE)  |
|                      |                          | Synplify Pro®   |
|                      | Simulation               | Questa Lattice OEM  |

**Note:**

1. In some instances, the IP may be updated without changes to the user guide. This user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

## 1.3. IP Support Summary

**Table 1.2. EFB Module IP Support Readiness**

| Device Family | Feature              | Radiant Timing Model | Hardware Validated |
|---------------|----------------------|----------------------|--------------------|
| MachXO4       | I2C                  | Final                | No                 |
|               | SPI                  | Final                | No                 |
|               | Timer/Counter        | Final                | No                 |
|               | PLL (dynamic config) | Final                | No                 |
|               | UFM                  | Final                | No                 |

## 1.4. Features

Key features of the EFB Module IP include:

- Provides an interface to user logic through the WISHBONE interface.
- Two I2C cores:
  - Operate as both controller and target.
  - Support configurable 7-bit and 10-bit addressing.
  - Support 50 kHz, 100 kHz, and 400 kHz data transfer speed.
  - The primary I2C core can access either flash memory or user logic.
- One SPI core:
  - Configurable as either SPI controller/target or SPI target.
  - When configured as controller/target, the IP core can control up to eight other SPI target devices.
  - Provides an interface to flash memory or user logic through the WISHBONE interface.
  - Includes a Mode Fault Error flag with CPU interrupt.
  - Features a double-buffered data register for increased throughput.
  - Supports serial clock with programmable polarity and phase.
  - Supports both LSB-first and MSB-first data transfers.
- Various Timer/Counter modes of operation:
  - Clear Timer on Compare Match (CTCM)
  - Watchdog Timer
  - Fast PWM
  - Phase and Frequency Correct PWM
- Dynamic PLL configuration.
- Interface to flash memory, which includes:
  - User flash memory
  - Configuration logic
- APB interface for bridging to the WISHBONE user interface.

## 1.5. Licensing and Ordering Information

The EFB Module IP is provided at no additional cost with the Lattice Radiant software.

## 1.6. Minimum Device Requirements

There is no limitation in device speed grade for EFB Module IP. See [Appendix A. Resource Utilization](#) for minimum required resources to instantiate this IP and maximum clock frequency supported.

## 1.7. Naming Conventions

### 1.7.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

### 1.7.2. Signal Names

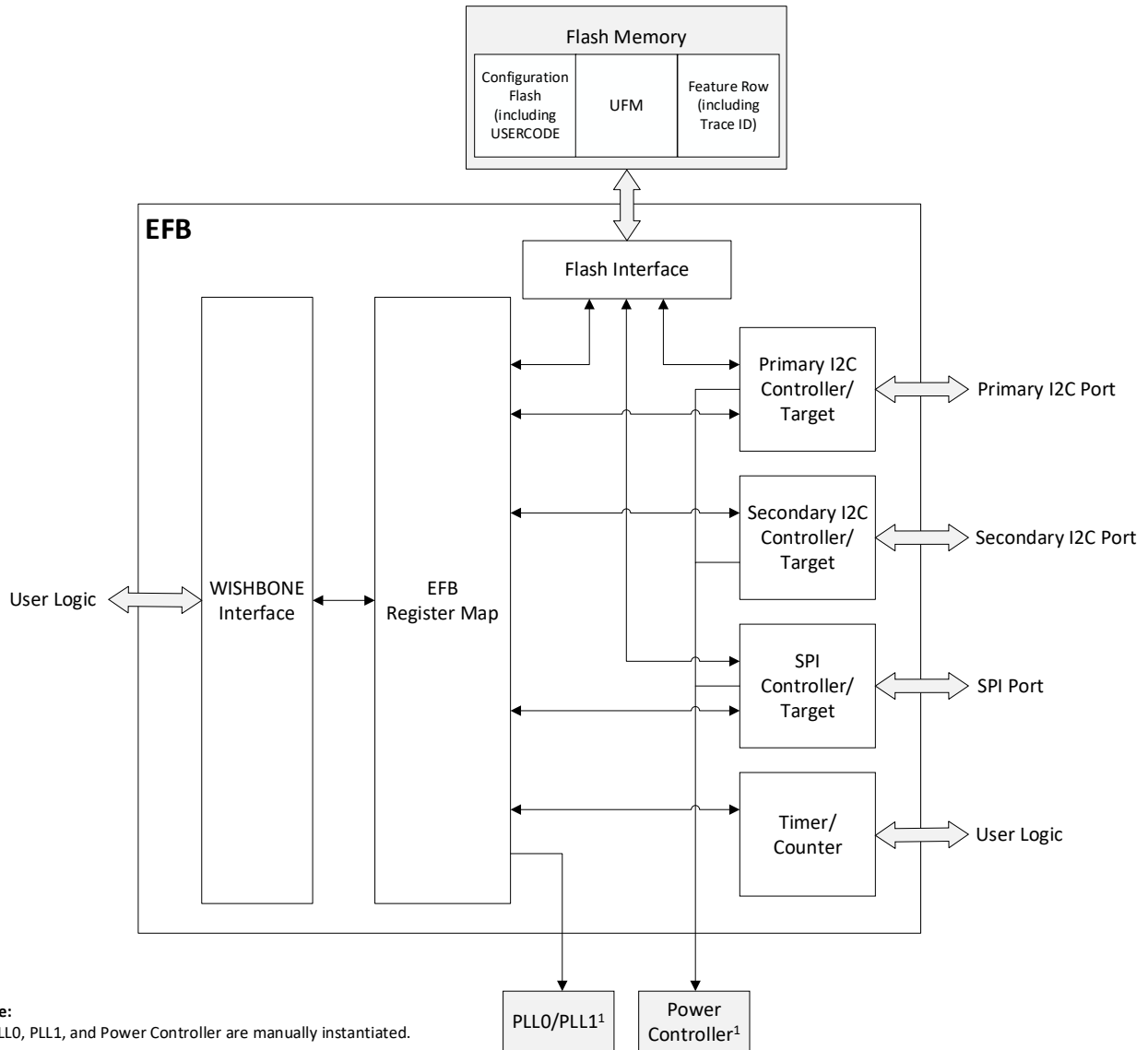
- `_n` are active low (asserted when value is logic 0)
- `_i` are input signals
- `_o` are output signals

### 1.7.3. Attribute Names

Attribute names in this document are formatted in title case and italicized (*Attribute Name*).

## 2. Functional Description

### 2.1. IP Architecture Overview

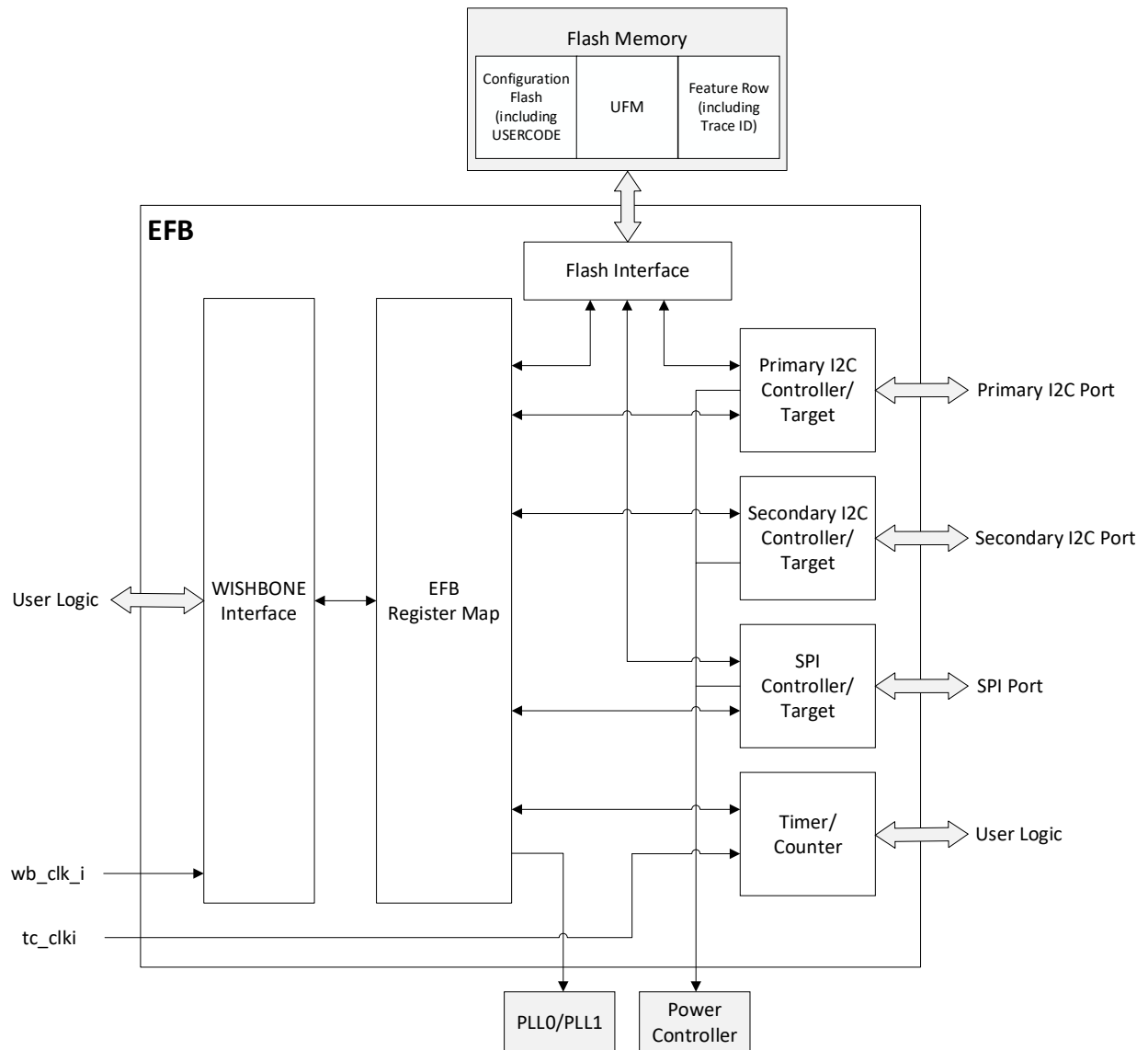


**Figure 2.1. EFB Module IP Block Diagram**

The hardened control function of MachXO4 devices is contained inside the EFB. You can interface with the logic through the WISHBONE interface to the EFB register map. Each hardened function can be controlled through these registers.

## 2.2. Clocking

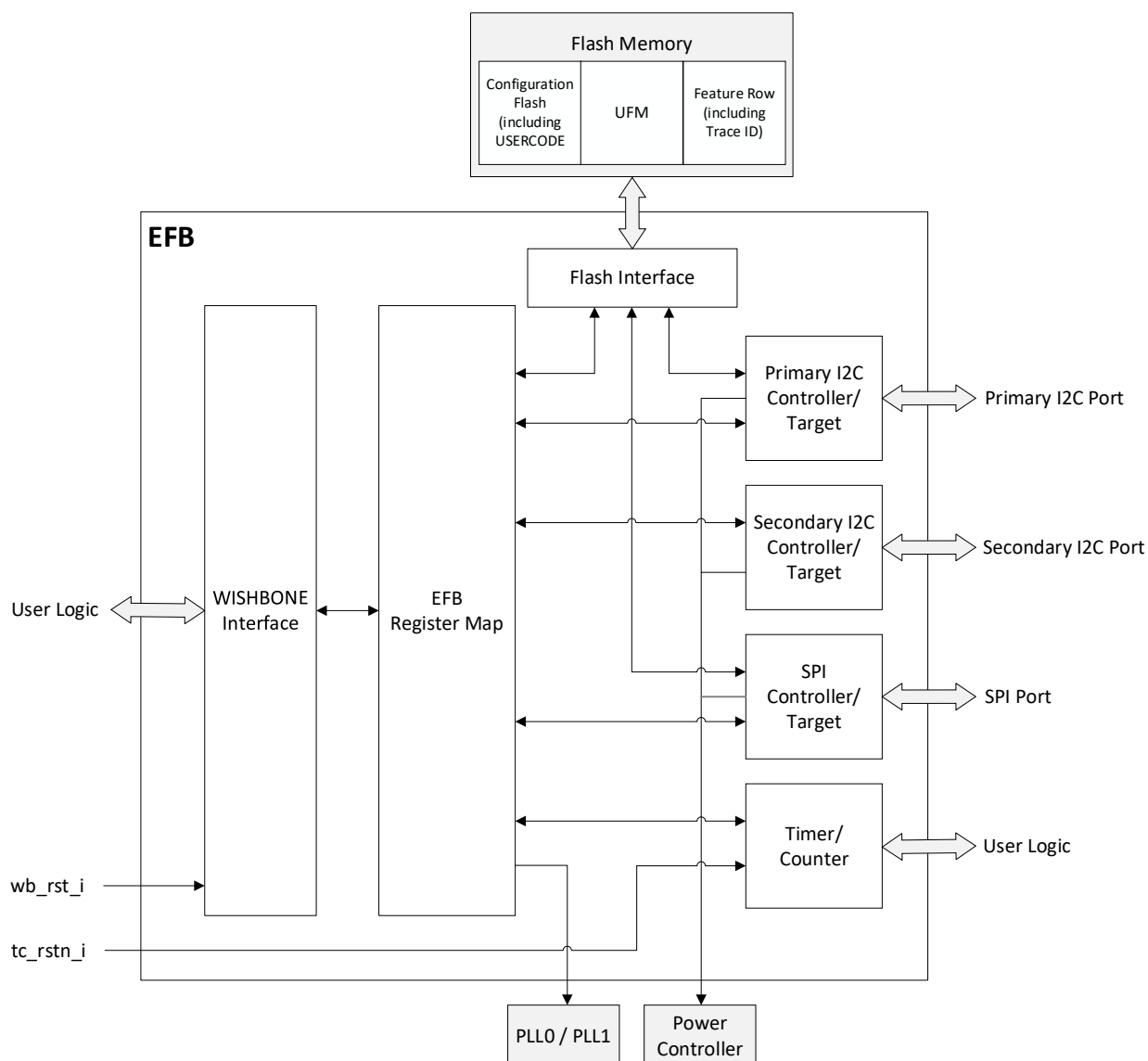
### 2.2.1. Clocking Overview



**Figure 2.2. EFB Module IP Clock Overview**

The WISHBONE interface clock is used by the WISHBONE interface logic and control registers.

### 2.3.1. Reset Overview



### Figure 2.3. EFB Module IP Reset Overview

The WISHBONE interface reset signal only resets the WISHBONE interface logic.

## 2.4. User Interfaces

**Table 2.1. User Interfaces and Supported Protocols**

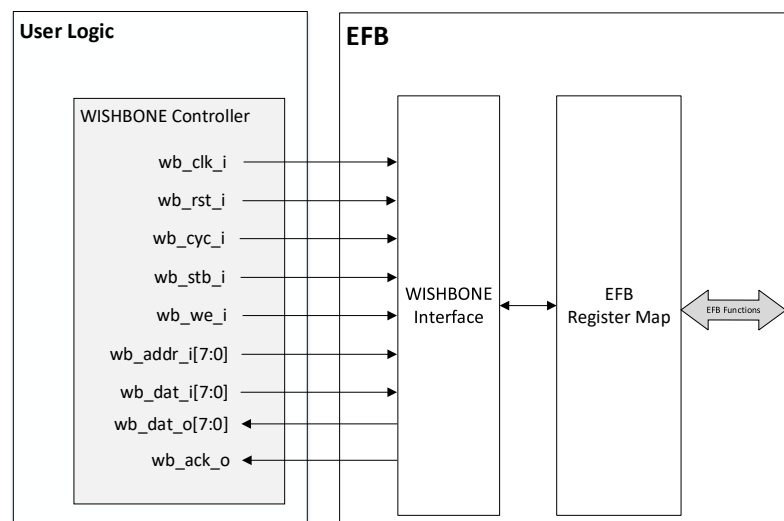
| User Interface | Supported Protocol   | Description  |
|----------------|----------------------|--|
| Data/Control   | WISHBONE (Target)    | Provides connectivity between user logic and the EFB functional blocks. The user logic must include a WISHBONE initiator interface to communicate with the WISHBONE target of the EFB. |
|                | I2C                  | Each MachXO4 device contains two hardened I2C IP cores, designated as primary and secondary. Both can operate as an I2C controller or target.  |
|                | SPI                  | Hardened SPI core, which can operate as both SPI controller and target, or solely as an SPI target device.   |
|                | APB                  | AMBA APB 3 protocol-compliant interface can be enabled to bridge transactions from a system bus to the WISHBONE target of the EFB.   |
|                | WISHBONE (Initiator) | Provides WISHBONE transactions for the dynamic reconfiguration of PLL configurations.  |

### 2.4.1. WISHBONE Bus Interface (Register Access)

The WISHBONE bus in this IP is compliant with the WISHBONE standard from OpenCores. The bus provides connectivity between the FPGA user logic and EFB functional blocks. You can implement a WISHBONE controller interface to interact with the EFB WISHBONE target interface. [Table 4.1](#) provides a detailed definition of the WISHBONE signals.

The EFB WISHBONE bus supports the classic version of the WISHBONE standard. Given that the WISHBONE bus is an open-source standard, not all features of the standard are implemented or required:

- Tags are not supported in the WISHBONE target interface of the EFB module. Given that the EFB is a hardened block, you cannot add these signals.
- The target WISHBONE bus interface of the EFB module does not require the byte select signals (sel\_i or sel\_o), since the data bus is only a single byte wide.
- The EFB WISHBONE target interface does not support the optional error and retry access termination signals. The IP does not guarantee the behavior if it receives access to an invalid register. You need to stay within the valid address range detailed in the [Register Description](#) section.
- If the EFB WISHBONE input signals are not used, they must be connected to 0.
- To ensure correct operation, wb\_cyc\_i must be asserted for the entire WISHBONE transaction. For the EFB WISHBONE interface, wb\_cyc\_i and wb\_stb\_i may be connected together.



**Figure 2.4. EFB Module WISHBONE Interface**

## 2.4.2. I2C IP Cores

The I2C is a widely used two-wire serial bus for communication between devices on the same board. Every MachXO4 device contains two hardened I2C IP cores, designated as the primary and secondary I2C IP cores. Either of the two cores can operate as an I2C controller or I2C target. 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 I2C core can be used for accessing the flash memory. However, the primary I2C port cannot be used for both flash access and user functions in the same design. When instantiating the hardened I2C IP cores for target operations, the EFB *wb\_clk\_i* input must be connected to a valid clock source of at least  $7.5 \times$  the I2C bus rate. For example,  $>3.0$  MHz when I2C rate = 400 kHz.

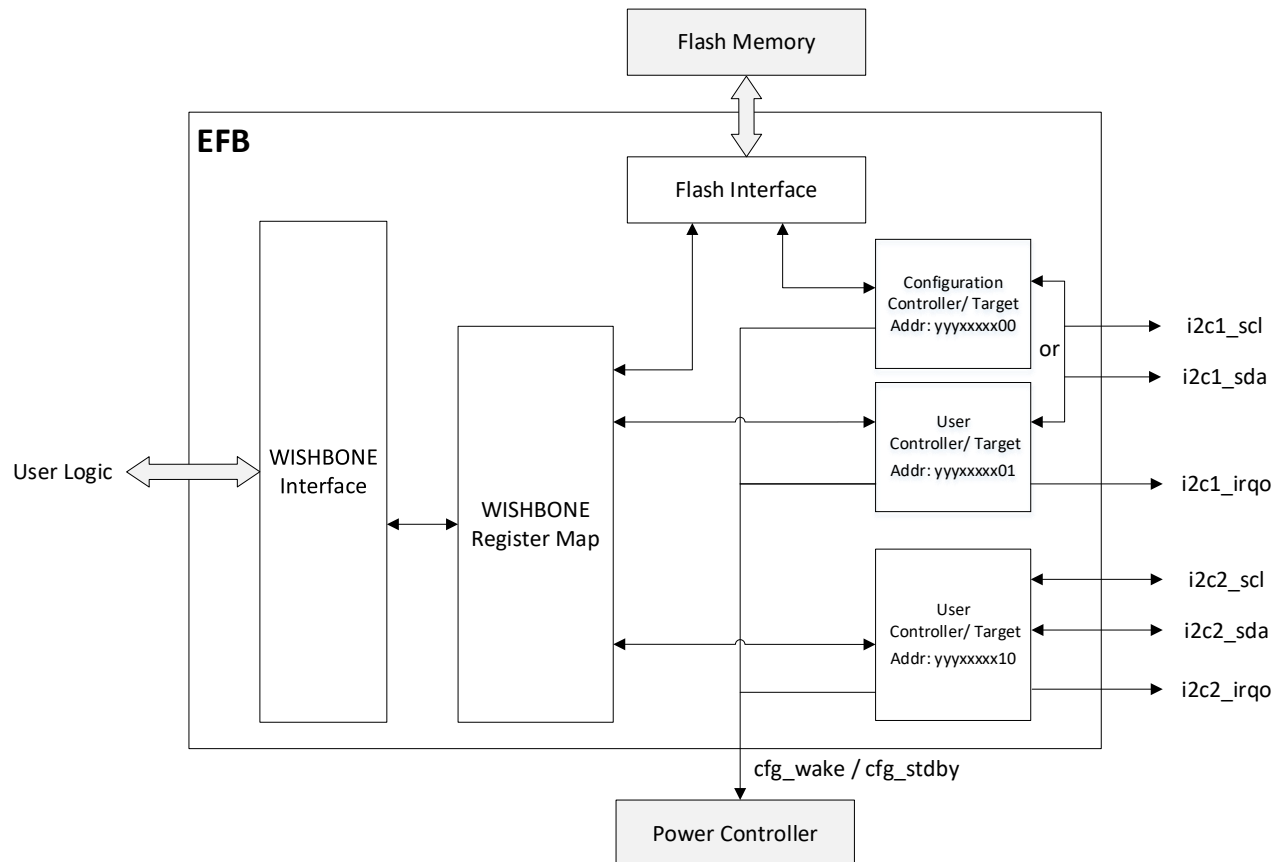


Figure 2.5. EFB Module I2C Core Interface

### 2.4.2.1. Primary I2C

The main functions of the primary controller are:

- Either:
  - I2C configuration target provides access to the flash; or
  - I2C user target provides access to the user logic.
- I2C user controller provides access to peripherals attached to the MachXO4 device.

The primary I2C core can be used for accessing the flash memory. However, the primary I2C port cannot be used for both flash access and user functions in the same design. Figure 2.5 shows an interface between the I2C block and the flash access.

Target I2C peripherals on a bus are accessed by the controller I2C calling the unique addresses of the target. The primary configuration address is yyyxxxxx00 and the primary user address is yyyxxxxx01, where y and x are user-programmable from IP generation.

### 2.4.2.2. Secondary I2C

The secondary I2C controller in a MachXO4 device provides the same functionality as the primary I2C controller with the exception of access to the flash memory. The i2c2\_scl and i2c2\_sda ports are routed through the general purpose routing of the FPGA fabric and you can assign them to any general purpose I/O (GPIO).

Target I2C peripherals on a bus are accessed by the user controller I2C calling the unique addresses of the target. The secondary user address is yyyxxxxx10, where y and x are user-programmable from IP generation.

### 2.4.3. SPI IP Core

SPI is a widely used four-wire serial bus that operates in full duplex mode for communication between devices. The MachXO4 EFB includes an SPI controller that can be configured either as both an SPI controller/target, or solely as an SPI target.

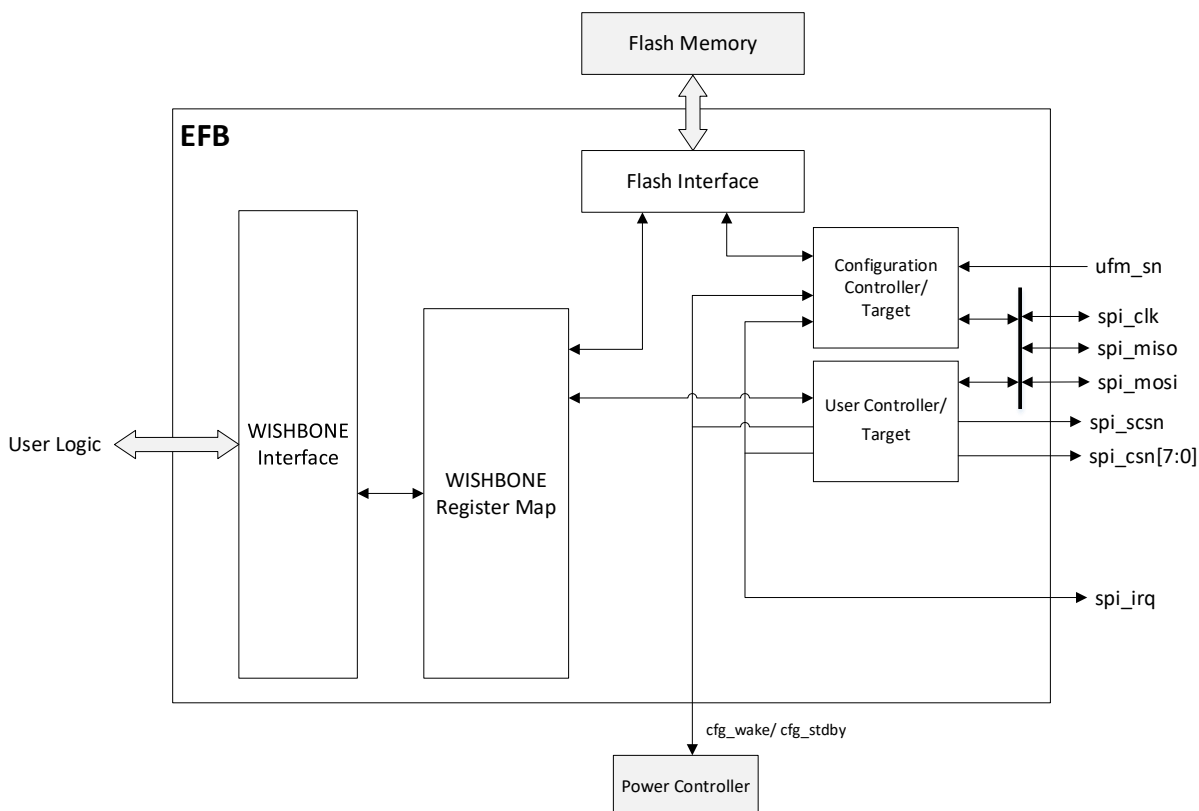
When configured as a controller/target, the IP core can control up to eight other devices with SPI target interfaces.

When configured as a target, the IP core can interface to an external SPI controller device.

The SPI core interfaces with either the configuration logic of the MachXO4 device or other user logic. The functionality and block diagram of the hardened SPI IP core are shown below.

**Table 2.2. Hardened SPI Functionality**

| Feature                                 | Configuration SPI | User SPI |
|---|-------------------|----------|
| SPI target port                         | Yes               | Yes      |
| SPI controller port                     | No                | Yes      |
| Access to Flash memory                  | Yes               | No       |
| Requires dedicated I/O                  | Yes               | Yes      |
| Wake Power Controller from Standby Mode | Yes               | Yes      |
| Enter Power Controller Standby Mode     | No                | Yes      |



**Figure 2.6. EFB Module SPI Core Interface**



#### 2.4.4. WISHBONE Controller (for PLL)

The WISHBONE interface of the EFB module can be used to dynamically update the configurable settings of the phase locked loop (PLL) IPs in MachXO4 devices. Each MachXO4 device supports up to two PLLs.

By connecting the EFB WISHBONE interface to the PLL interface, PLL0 has an address range from 0x00 to 0x1F, while PLL1 (if applicable) has an address range from 0x20 to 0x3F in the EFB register map.

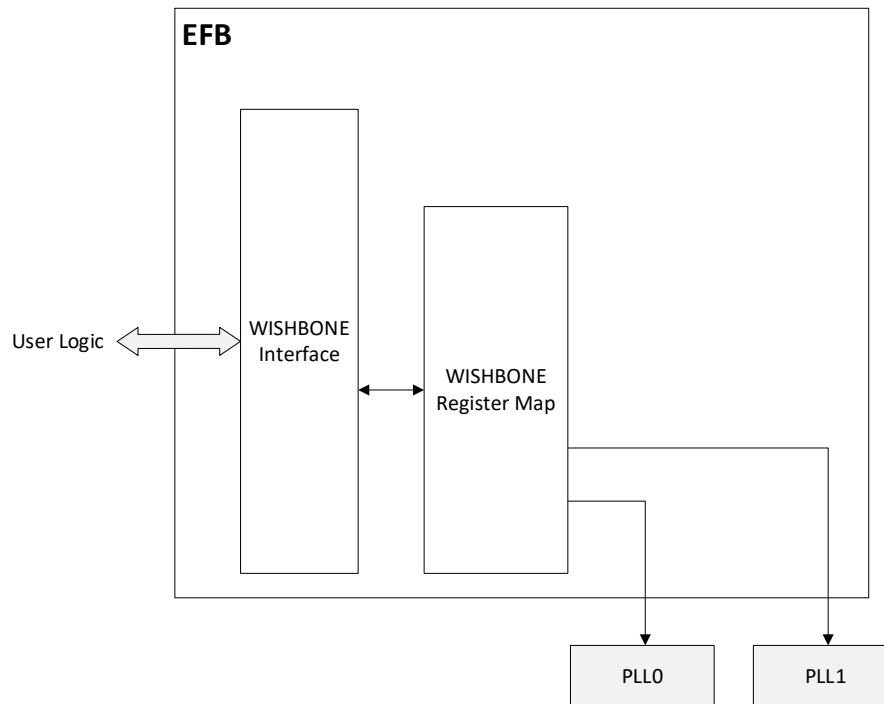


Figure 2.7. EFB Interface to Dynamic PLL

### 2.5. Flash Access Interface

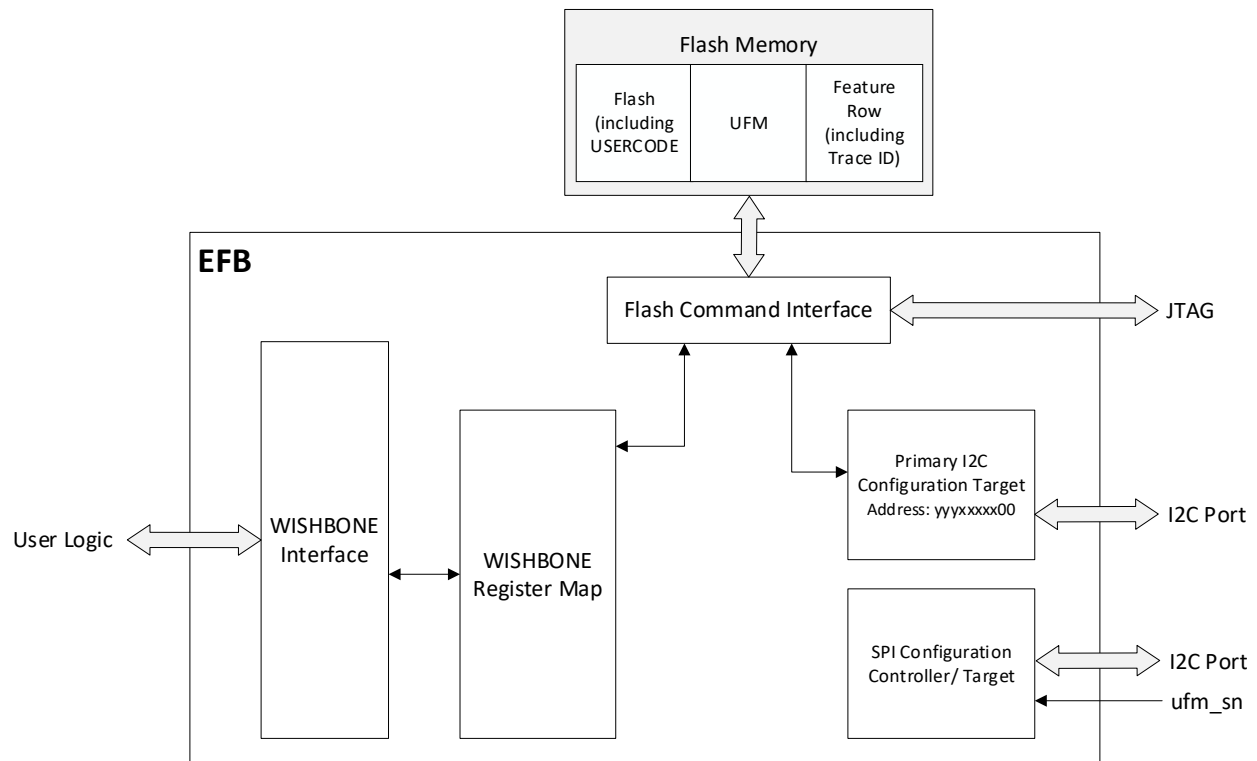
You can access the flash logic interface using the JTAG, SPI, I2C, or WISHBONE interfaces. The MachXO4 flash memory consists of three sectors:

- Configuration flash (includes USERCODE)
- UFM
- Feature Row

The flash memory is organized in pages and is not byte addressable. Each page has 128 bits (16 bytes).

The configuration logic arbitrates access from the interfaces according to the following priority. When higher priority ports are enabled, flash access by lower priority ports is blocked:

- JTAG port – All MachXO4 devices include a JTAG port that supports read/write operations to Flash. The JTAG port complies with IEEE 1149.1 and IEEE 1532 specifications. JTAG port has the highest priority for Flash operations.
- SPI target port – All MachXO4 devices include an SPI target port that supports read/write operations to Flash. Asserting the UFM/Flash target chip select (ufm\_sn) enables access.
- I2C primary port – All MachXO4 devices include an I2C port that supports read/write operations to Flash. The primary configuration I2C address is yyyxxxx00, where y and x are user-programmable bits.
- WISHBONE target interface – The WISHBONE interface of the EFB module enables designers to access the UFM/Flash from FPGA user logic by creating a WISHBONE controller.



**Figure 2.8. EFB Module Flash Interface**

The WISHBONE interface of the EFB module allows a WISHBONE host to access the configuration resources of MachXO4 devices. This is useful for reading data from configuration resources such as USERCODE and TraceID. Most importantly, this feature allows you to update the flash array of the devices while the device is in operation mode. This is a self-configuration operation. Upon power-up or a configuration refresh operation, the new content of the CFG flash is loaded into the configuration SRAM, and the device continues operation with the new configuration.

### 3. IP Parameter Description

The configurable attributes of the EFB Module IP are shown in the following tables. You can configure the IP by setting the attributes accordingly in the IP Catalog's Module/IP wizard of the Lattice Radiant software.

Wherever applicable, default values are in **bold**.

#### 3.1. EFB Enables

**Table 3.1. EFB Enables Attributes**

| Attribute   | Selectable Values                   | Description  |
|---|-------------------------------------|--|
| <b>User Interface Options</b>                       |                                     |  |
| Enable User Interface                               | <b>Checked</b> , Unchecked          | Enables the selected user interface for user logic. Some EFB features require a user interface to be enabled.  |
| User Interface                                      | WISHBONE, APB                       | Configures either the default WISHBONE interface or the AMBA APB-compliant protocol interface to bridge WISHBONE transactions.   |
| User Interface Clock Frequency                      | 1 MHz – 133 MHz, <b>50 MHz</b>      | Specifies the input value for the WISHBONE interface clock frequency. This value also affects the required clock pre-scale, depending on the selected I2C bus performance. |
| <b>I2C Options</b>                                  |                                     |  |
| Primary I2C   | <b>Checked</b> , Unchecked          | Enables access to the primary I2C (I2C1) ports.  |
| Primary I2C Access Type (if Primary I2C Enabled)    | <b>User Function</b> , Flash Access | Configures whether primary I2C (I2C1) access is assigned to user functions/logic or to UFM/configuration. This setting affects the primary I2C target address.             |
| Secondary I2C                                       | Checked, <b>Unchecked</b>           | Enables access to the secondary I2C (I2C2) ports.  |
| <b>SPI Options</b>                                  |                                     |  |
| SPI   | Checked, <b>Unchecked</b>           | Enables access to the SPI core ports.  |
| <b>Timer/Counter Options</b>                        |                                     |  |
| Timer/Counter                                       | Checked, <b>Unchecked</b>           | Enables the Timer/Counter features.  |
| Allow Dynamic Register Changes (via User Interface) | Checked, <b>Unchecked</b>           | Enables dynamic control of the Timer/Counter registers.  |
| <b>PLL Options</b>                                  |                                     |  |
| PLL (Dynamic Access)                                | Checked, <b>Unchecked</b>           | Enables the WISHBONE ports for dynamic control of the PLL registers. PLL must be included in the design, and WISHBONE ports must be enabled.                               |
| Number of PLLs for dynamic access                   | <b>1,2</b>                          | Configures the number of PLL for dynamic access. The number of available PLL depends on the specific MachXO4 device.   |
| <b>UFM Options</b>                                  |                                     |  |
| User Flash Memory                                   | Checked, <b>Unchecked</b>           | Enables the user flash memory access through WISHBONE or primary I2C flash access commands.  |

#### 3.2. I2C

**Table 3.2. I2C Attributes**

| Attribute                                  | Selectable Values         | Description  |
|--|---------------------------|--|
| <b>Primary I2C (Primary I2C = Enabled)</b> |                           |  |
| General Call Enable                        | Checked, <b>Unchecked</b> | Enables general call response for the primary I2C target. The general call address is 0000000 for both 7-bit or 10-bit addressing. |

| Attribute  | Selectable Values  | Description  |
|--|--|--|
| Wakeup Enable  | Checked, <b>Unchecked</b>  | Enables the Wake-up (cfg_wake) and Standby (cfg_stdby) signals from the primary I2C for the Power Controller.  |
| I2C Bus Performance  | 50 kHz, <b>100 kHz</b> , 400 kHz                                       | Specifies the input configuration for the primary I2C clock frequency as the controller.   |
| Clock Pre-scale Value                                      | (Display only)<br><b>125</b>   | Displays the initial configuration of the clock pre-scale based on the set user interface (WISHBONE) clock frequency and primary I2C bus performance.<br>The WISHBONE clock frequency is divided by the pre-scale value × 4 to produce the primary I2C controller clock frequency.     |
| I2C Addressing   | <b>7-bit</b> , 10-bit  | Configures whether the primary I2C uses a 7-bit or 10-bit addressing scheme on the bus.  |
| I2C Device Address   | <b>10000</b>   | The input address (in binary) for the primary I2C target.  |
| Primary I2C/ Flash Access Device Address                   | (Display only)<br><b>0b1000001</b> (User)/<br><b>0b1000000</b> (Flash) | Displays the address of the primary I2C based on the input device address and the primary I2C access type.   |
| <b>Secondary I2C (Secondary I2C = Enabled)</b>             |  |  |
| General Call Enable  | Checked, <b>Unchecked</b>  | Enables general call response for the secondary I2C target.<br>The general call address is 0000000 for both 7-bit or 10-bit addressing.  |
| Wakeup Enable  | Checked, <b>Unchecked</b>  | Enables the Wake-up (cfg_wake) and Standby (cfg_stdby) signals from the secondary I2C for the Power Controller.  |
| I2C Bus Performance  | 50 kHz, <b>100 kHz</b> , 400 kHz                                       | Specifies the input configuration for the secondary I2C clock frequency as the controller.   |
| Clock Pre-scale Value                                      | (Display only)<br><b>125</b>   | Displays the initial configuration of the clock pre-scale based on the set user interface (WISHBONE) clock frequency and secondary I2C bus performance.<br>The WISHBONE clock frequency is divided by the pre-scale value × 4 to produce the secondary I2C controller clock frequency. |
| I2C Addressing   | <b>7-bit</b> , 10-bit  | Configures whether the secondary I2C uses a 7-bit or 10-bit addressing scheme on the bus.  |
| I2C Device Address (shows only if Primary I2C is Disabled) | <b>10000</b>   | The input address (in binary) for the secondary I2C target.  |
| Secondary I2C Device Address                               | (Display only)<br><b>0b1000010</b> (User)                              | Displays the address of the secondary I2C based on the input device address.   |

### 3.3. SPI

**Table 3.3. SPI Attributes**

| Attribute                      | Selectable Values                     | Description  |
|--------------------------------|---------------------------------------|--|
| <b>General (SPI = Enabled)</b> |                                       |  |
| SPI Mode                       | <b>Target</b> , Controller and Target | Selects between target, or both controller and target modes for the initial mode of the SPI core.<br>Selecting controller and target modes enables SPI Controller settings, which include Controller Clock Rate and Controller Chip Selects.<br>This option can be updated dynamically by modifying the MSTR bit in the SPICR2 register. |

| Attribute  | Selectable Values           | Description  |
|--|-----------------------------|--|
| Wakeup Enable  | Checked, <b>Unchecked</b>   | Enables the SPI core to send wake-up signals (cfg_wake and cfg_stby) to the Power Controller, allowing the device to exit standby mode when the user target SPI chip select (spi_csn[0]) is driven low.<br><br>This option can be updated dynamically by modifying the WKUPEN_USER bit in the SPICR1 register.   |
| <b>Controller Clock Rate (SPI = Enabled AND SPI Mode = <i>Controller and Target</i>)</b>   |                             |  |
| Desired (MHz)  | <b>1-45</b>                 | Specifies the input configuration for the SPI core clock when operating as a controller.   |
| Actual (MHz)   | (Display only)<br><b>1</b>  | Displays the actual SPI core clock based on the desired input frequency.<br><br>The actual clock frequency is determined by the nearest whole pre-scale value.   |
| Clock Pre-scale Value  | (Display only)<br><b>50</b> | Displays the initial pre-scale value configuration set in the registers, based on the actual SPI core clock.   |
| <b>Controller Chip Selects (SPI = Enabled AND SPI Mode = <i>Controller and Target</i>)</b> |                             |  |
| Number of Chip Selects   | <b>1-8</b>                  | Configures the number of SPI targets than can be connected to the SPI core when operating as a controller.<br><br>This setting affects the bit width of the spi_csn signal.  |
| <b>Protocol Options (SPI = Enabled)</b>  |                             |  |
| LSB First  | Checked, <b>Unchecked</b>   | Specifies the order of the serial shift of a byte of data. The data order (MSB-first or LSB-first) is programmable within the SPI core.<br><br>This option can be updated dynamically by modifying the LSBF bit in the SPICR2 register.  |
| Inverted Clock   | Checked, <b>Unchecked</b>   | When enabled, the clock edge changes from rising to falling edge.<br><br>This option can be updated dynamically by accessing the CPOL bit of the SPICR2 register.  |
| Phase Adjust   | Checked, <b>Unchecked</b>   | Specifies the phase change to match the application.<br><br>This option can be updated dynamically by accessing the CPHA bit in the SPICR2 register.   |
| Target Handshake Mode  | Checked, <b>Unchecked</b>   | Enables Lattice proprietary extension to the SPI protocol.<br><br>This option is used when internal support circuit (such as a WISHBONE host) cannot respond with initial data within required time, and ensure SPI target read out data predictably available at high SPI clock rates.<br><br>This option can be updated dynamically by accessing the SDBRE bit in the SPICR2 register. |
| <b>SPI – Interrupts</b>  |                             |  |
| Enable Port  | Checked, <b>Unchecked</b>   | Enables the interrupt request output signal (spi_irq) from the SPI core.<br><br>This signal is intended to be connected to a WISHBONE controller and to request an interrupt when a specific condition is met.   |
| Tx Ready   | Checked, <b>Unchecked</b>   | An interrupt that indicates the SPI transmit data register (SPITXDR) is empty.<br><br>When enabled, indicates TRDY is asserted.<br><br>This option can be changed dynamically by modifying the IRQTRDYEN bit in the SPICSR register.   |
| Rx Ready   | Checked, <b>Unchecked</b>   | An interrupt that indicates the receive data register (SPIRXDR) contains valid receive data.<br><br>When enabled, indicates RRDY is asserted.<br><br>This option can be changed dynamically by modifying the IRQRRDYEN bit in the SPICSR register.   |

| Attribute  | Selectable Values         | Description  |
|------------|---------------------------|--|
| Tx Overrun | Checked, <b>Unchecked</b> | An interrupt that indicates the target SPI chip select (spi_scsn) is driven low while being transacted by the SPI Controller.<br>When enabled, indicates MDF (Mode Fault) is asserted.<br>This option can be changed dynamically by modifying the IRQMDFEN bit in the SPICSR register. |
| Rx Overrun | Checked, <b>Unchecked</b> | An interrupt that indicates SPIRXDR received new data before the previous data is captured.<br>When enabled, indicates ROE is asserted.<br>This option can be changed dynamically by modifying the IRQROEEN bit in the SPICSR register.  |

### 3.4. Timer/Counter

Table 3.4. Timer/Counter Attributes

| Attribute                        | Selectable Values  | Description   |
|----------------------------------|--|---|
| <b>Mode Selection</b>            |  |   |
| Timer/Counter Mode               | Watchdog Timer, <b>Clear Timer on Compare (CTCM)</b> , Fast PWM, Phase/Freq Correct PWM (PFCPWM) | Configures between different Timer/Counter modes of operation. Refer to the following subsections for a detailed discussion of each mode.   |
| Output Function                  | Static, <b>Toggle</b> , Wave Generator, Inverted Wave Generator                                  | Configures the function of the output signal (tc_oc) of the Timer/Counter IP.<br>The available functions are: <ul style="list-style-type: none"> <li>Static – output is static low.</li> <li>Toggle – output toggles based on conditions defined by the Timer/Counter settings.</li> <li>Wave Generator – generates a waveform based on Set/Clear conditions defined by the Timer/Counter settings.</li> <li>Inverted Wave Generator – generates an inverted waveform.</li> </ul> |
| <b>Clock Selection</b>           |  |   |
| Clock Edge Selection             | <b>Positive</b> , Negative   | Selects the edge of the input clock source.   |
| Use On-chip Oscillator           | Checked, <b>Unchecked</b>  | Enables the use of the on-chip oscillator.  |
| Prescale Divider Value           | 0, <b>1</b> , 8, 64, 256, 1024   | Configures the pre-scale value to divide the input clock prior to reaching the 16-bit counter.<br>This option can be updated dynamically by modifying the PRESCALE[2:0] bits of the TCCR1 register.   |
| <b>Enable Interrupt Register</b> |  |   |
| Overflow                         | Checked, <b>Unchecked</b>  | Enables an interrupt that indicates the counter matches the TCTOP0/1 register value.<br>When enabled, indicates OVF is asserted.<br>This option can be updated dynamically by modifying the IRQOVFEN bit of the TCIRQEN register.<br>Available if <i>Allow Dynamic Register Changes (via User Interface)</i> is enabled.  |
| Output Compare Match             | Checked, <b>Unchecked</b>  | Enables an interrupt which indicates when counter matches the TCOCR0/1 register value.<br>When enabled, indicates OCRF is asserted.<br>This option can be updated dynamically by modifying the IRQOCRFEN bit of the TCIRQEN register.<br>Available if <i>Allow Dynamic Register Changes (via User Interface)</i> is enabled.  |

| Attribute                               | Selectable Values         | Description   |
|---|---------------------------|---|
| Input Compare                           | Checked, <b>Unchecked</b> | Enables an interrupt that indicates the TC_IC input signal is asserted.<br>When enabled, indicates ICRF is asserted.<br>This option can be updated dynamically by modifying the IRQICRFEN bit of the TCIRQEN register.<br>Available if <i>Allow Dynamic Register Changes (via User Interface)</i> is enabled. |
| Standalone Overflow (no User Interface) | Checked, <b>Unchecked</b> | Enables the overflow interrupt without the user interface and serves as the only available interrupt request.<br>Available only if <i>Allow Dynamic Register Changes (via User Interface)</i> is disabled.  |
| <b>Counter Values</b>                   |                           |   |
| Set Top Counter Value                   | Checked, <b>Unchecked</b> | Enables setting a specific Timer/Counter top value, other than the maximum.   |
| Timer/Counter Top                       | 0-65535                   | Configures the top value of the Timer/Counter.  |
| Output Compare Value                    | 0-65535                   | Configures the compare value of the Timer/Counter for PWM modes.  |

### 3.4.1. Watchdog Timer Mode

Watchdog timers monitor operating behavior of a system and provide a reset or interrupt when the microcontroller of the system or embedded state machine is no longer operational.

One use case is for a microcontroller to reset the Watchdog Timer to 0x0000 before starting a process. The microcontroller must complete the process and reset the Watchdog Timer again before the timer reaches its terminal count. If the microcontroller fails to clear the timer in time, the Watchdog Timer asserts a strobe signal indicating that the time has expired. The system uses this strobe to gracefully recover the system.

Another use case is to periodically turn off system modules to save power. The Watchdog Timer can also be used to interact with the on-chip power controller of a MachXO4 device.

The most commonly used ports of the Timer/Counter in Watchdog Timer mode are the clock, reset, and interrupt.

Optionally, the user interface can be used to read time stamps from the TCICR register and update the top value of the counter.

### 3.4.2. Clear Timer on Compare Match Mode

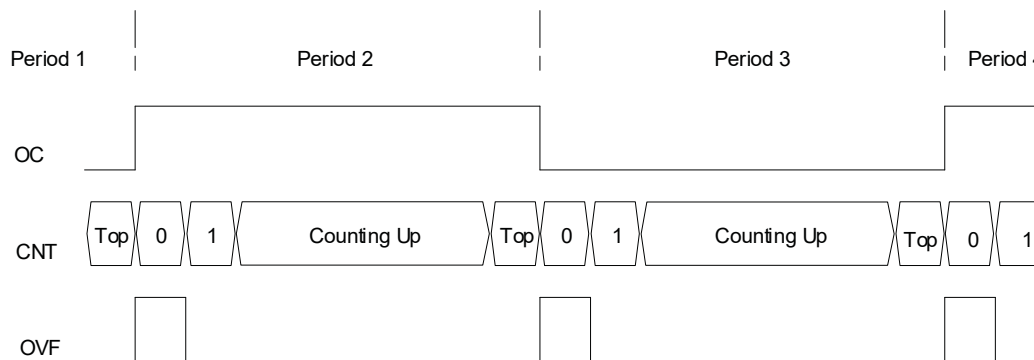
CTCM (Clear Timer on Compare Match) operates as a basic counter with interrupt capability. The counter automatically clears to 0x0000 when the counter value in the TCCNT register, matches the value in the TCTOP register.

The TCTOP register value can be dynamically updated through the WISHBONE interface or assigned as a static value during IP generation. By default, the TCTOP register holds the value 0xFFFF.

The data loaded into the timer counter to define the top counter value is double-registered. The user interface controller writes the data to the TCTOPSET register, which is then automatically loaded into 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 changes the frequency of the Timer/Counter output signal.

Refer to the following figure for the Timer/Counter CTCM output waveform.



**Figure 3.1. Timer/Counter CTCM Output Waveform**

### 3.4.3. Fast PWM Mode

Pulse-Width Modulation (PWM) is a widely used 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.

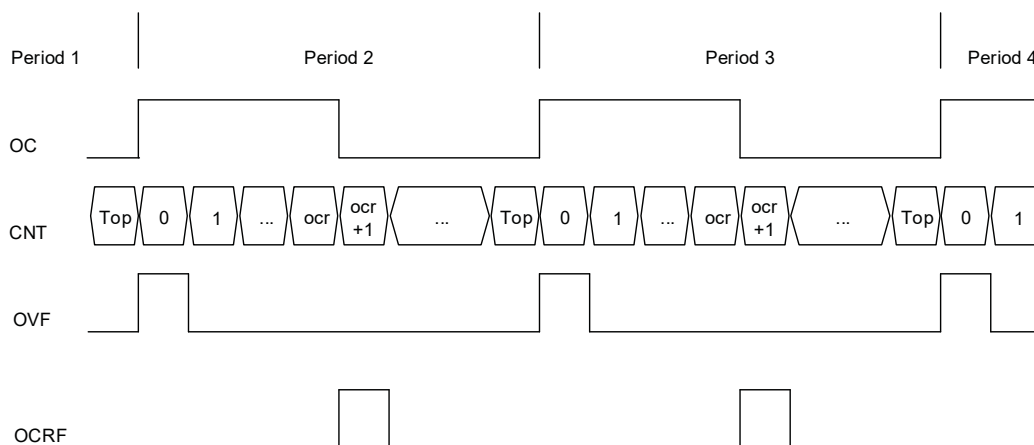
The period and the duty cycle of the waveform can be varied by loading 16-bit digital values into:

- 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 Timer/Counter output is cleared when the counter value matches the top value that is loaded into the TCTOP register. The output is set when the value of the counter matches the compare value that is loaded into the TCOCR register. These clear/set functions can be inverted, allowing the output of the Timer/Counter to be set when the counter value matches the top value and 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).

Refer to the following figure for the PWM waveform generation, where the Timer/Counter output is configured to be set when the counter matches the top value and cleared when the counter matches the compare value.



**Figure 3.2. Fast PWM Mode Waveform Generation**

### 3.4.4. Phase and Frequency Correct PWM Mode

In phase and frequency correct PWM mode, the counting direction changes from up to down when the counter is incrementing to the top value (top value minus 1). When the counter is decrementing from 0x0001 to 0x0000, the following occurs:

- The TCTOP register is updated with the value loaded in the TCTOPSET register.
- The TCOCR register is updated with the value loaded in the TCOCRSET register
- The Overflow Flag 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 the TCOCR register. This match occurs twice within one period:

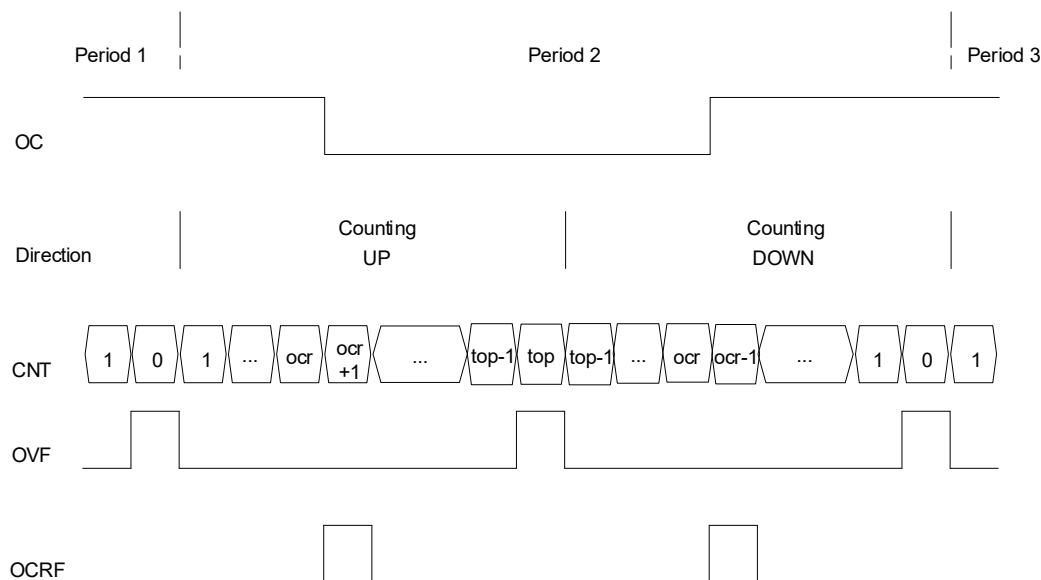
- The first match occurs when the counter is counting up.
- The second match occurs when the counter is counting down.

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.

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



**Figure 3.3. Phase/Frequency Correct PWM Mode Waveform Generation**

### 3.5. UFM

**Table 3.5. UFM Attributes**

| Attribute  | Selectable Values           | Description   |
|--|-----------------------------|---|
| <b>User Flash Memory (User Flash Memory = Enabled)</b> |                             |   |
| Number of Pages  | <b>1</b>                    | The input value to indicate the start page of the flash memory initialization.<br>The range is calculated based on the available device UFM resource. |
| Initialization Data Start Page                         | (Display only)              | Calculated based on the available device UFM resource.  |
| Initialize Flash Memory with All 0s                    | <b>Checked</b> , Unchecked  | Enables initialization of the flash memory to all zeroes.   |
| UFM Initialization Data File                           | <b>&lt;file&gt;</b>         | Specifies the input file containing initialization data for the flash memory.   |
| UFM Initialization Data File                           | <b>Hexadecimal</b> , Binary | Configures whether the input file format is hexadecimal or binary.  |

## 4. Signal Description

This section describes the EFB Module IP ports.

**Table 4.1. EFB Module IP Ports**

| Port                      | Type         | Description  |
|---------------------------|--------------|--|
| <b>Clock and Reset</b>    |              |  |
| wb_clk_i                  | Input        | The positive-edged clock used by the WISHBONE interface register and hardened functions, supporting speeds up to 133 MHz.  |
| wb_rst_i                  | Input        | The synchronous reset signal of the WISHBONE interface. This reset terminates the active bus cycle. Wait 1 $\mu$ s after de-assertion before starting a WISHBONE transaction. This reset does not reset the contents of any register.  |
| <b>Flash Memory</b>       |              |  |
| wbc_ufm_irq               | Output       | The interrupt request output of the user flash memory.   |
| <b>Wishbone Interface</b> |              |  |
| wb_cyc_i                  | Input        | Indicates a valid bus cycle on the WISHBONE interface.   |
| wb_stb_i                  | Input        | The strobe signal indicating the WISHBONE is the target for the current transaction on the bus. The wb_ack_o signal asserts an acknowledgment in response to the assertion of the strobe.  |
| wb_we_i                   | Input        | The write/read control signal. Logic low indicates a read operation, while logic high indicates a write operation.   |
| wb_adr_i                  | Input        | The address signal for the WISHBONE interface, which is used to select a specific EFB register during write cycles.  |
| wb_dat_i                  | Input        | The data signal for the WISHBONE interface, which writes data to the addressed EFB register during write cycles.   |
| wb_dat_o                  | Output       | The read data signal for the WISHBONE interface, which outputs the data from the addressed EFB register during read cycles.  |
| wb_ack_o                  | Output       | Signals that the WISHBONE cycle is complete; data written to the EFB is accepted, and data read from the EFB is valid.   |
| <b>I2C Interface</b>      |              |  |
| i2c1_scl                  | Input/Output | The open-drain serial clock of the primary I2C core. This signal is an output if the I2C core is performing a controller operation and an input during target operations. This signal must be brought to the top level of the user RTL design. The Radiant software automatically routes this signal to its pre-assigned pin (no user pin location constraint is necessary).         |
| i2c1_sda                  | Input/Output | The open-drain serial data line of the primary I2C core. The signal is an output when data is transmitted from the I2C core and an input when data is received into the I2C core. This signal must be brought to the top level of the user RTL design. The Radiant software automatically routes this signal to its pre-assigned pin (no user pin location constraint is necessary). |
| i2c1_irqo                 | Output       | The interrupt request output signal of the primary I2C core. The use of this signal is for it to be connected to a WISHBONE controller and request an interrupt when a specific condition is met. These conditions are described in the <a href="#">Flash Access Registers</a> section.  |
| i2c2_scl                  | Input/Output | The open-drain serial clock line of the secondary I2C core. This signal is an output if the I2C core is performing a controller operation and an input during target operations. This signal can be routed to any GPIO of the MachXO4 device.  |
| i2c2_sda                  | Input/Output | The open-drain data line of the secondary I2C core. This signal is an output when data is transmitted from the I2C core and an input when data is received into the I2C core. This signal can be routed to any GPIO of the MachXO4 device.   |
| i2c2_irqo                 | Output       | The interrupt request output signal of the secondary I2C core. The use of this signal is for it to be connected to a WISHBONE controller and request an interrupt when a specific condition is met. These conditions are described in the <a href="#">Flash Access Registers</a> section.  |
| <b>SPI Interface</b>      |              |  |
| spi_clk                   | Input/Output | Serial Clock of the SPI core. The signal is an output if the SPI core is a controller (MCLK) and an input if the SPI core is a target (CCLK). This signal must be brought to the top level of the user RTL design. The Radiant software automatically routes this signal to its pre-assigned pin (no user pin location constraint is necessary).                                     |

| Port                   | Type         | Description  |
|------------------------|--------------|--|
| spi_miso               | Input/Output | The signal is an output if the SPI core is a controller (SPISO) and an input if the SPI core is a target (SO). This signal must be brought to the top level of the user RTL design. The Radiant software automatically routes this signal to its pre-assigned pin (no user pin location constraint is necessary).  |
| spi_mosi               | Input/Output | The signal is an output if the SPI core is a controller (SISPI) and an input if the SPI core is a target (SI). This signal must be brought to the top level of the user RTL design. The Radiant software automatically routes this signal to its pre-assigned pin (no user pin location constraint is necessary).  |
| spi_scsn               | Input        | User Target Chip Select (Active Low). An external SPI 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 MachXO4 device.  |
| ufm_sn                 | Input        | Configuration Logic Chip Select (Active Low). It is dedicated for selecting the UFM/ Flash sectors. Radiant software automatically routes this signal to its pre-assigned pin (no user pin location constraint is necessary).  |
| spi_csn                | Output       | Controller Chip Select (Active Low). Up to eight independent Target SPI devices can be accessed using the SPI Controller when it is in controller/target SPI mode. The signal spi_csn[0] must be brought to the top level of the user RTL design. The Radiant software automatically routes this signal to its pre-assigned pin (no user pin location constraint is necessary). The other bits (spi_csn[7:1]) can be routed to any GPIO of the MachXO4 device. |
| spi_irq                | Output       | Interrupt request output signal of the SPI core. This signal is connected to a WISHBONE controller. It is asserted when specific conditions are met. These conditions controlled using the SPI register settings.  |
| <b>Timer/Counter</b>   |              |  |
| tc_clk                 | Input        | Timer/Counter input clock signal. Can be connected to the on-chip oscillator. The clock signal is limited to 133 MHz.  |
| tc_rstn                | Input        | Active-low reset signal, which resets the 16-bit counter.  |
| tc_ic                  | Input        | Active-high input capture trigger event, applicable for non-PWM modes with WISHBONE interface. If enabled, a rising edge of this signal is 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       | Interrupt signal, indicating the occurrence of a specific event such as Overflow, Output Compare Match, or Input Capture.  |
| tc_oc                  | Output       | Timer/Counter output signal.   |
| <b>PLL0 (Wishbone)</b> |              |  |
| pll0_ack_i             | Input        | Acknowledgement signal from the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |
| pll0_dati_i            | Input        | Data output bus from the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| pll0_clk_o             | Output       | Generated clock by the WISHBONE Controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| pll0_rst_o             | Output       | Generated reset output by the WISHBONE Controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |
| pll0_addr_o            | Output       | Address bus from the WISHBONE Controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| pll0_dato_o            | Output       | Data bus from the WISHBONE Controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |
| pll0_we_o              | Output       | Generated write/read control signal for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |
| pll0_stb_o             | Output       | Generated strobe signal for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |

| Port                    | Type   | Description   |
|-------------------------|--------|---|
| <b>PLL1 (Wishbone)</b>  |        |   |
| pll1_ack_i              | Input  | Acknowledgement signal from the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| pll1_dati_i             | Input  | Data output bus from the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |
| pll1_clk_o              | Output | Generated clock by the WISHBONE controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |
| pll1_rst_o              | Output | Generated reset output by the WISHBONE controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| pll1_addr_o             | Output | Address bus from the WISHBONE controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.   |
| pll1_dato_o             | Output | Data bus from the WISHBONE controller for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| pll1_we_o               | Output | Generated write/read control signal for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| pll1_stb_o              | Output | Generated strobe signal for the WISHBONE target.<br>You must connect only to a PLL component that is instantiated in the design.  |
| <b>Power Controller</b> |        |   |
| cfg_wake                | Output | Wake-up signal – Hardwired signal to be connected ONLY to the Power Controller of the MachXO4 device for functional simulation support. This signal is enabled only if the Wakeup Enable feature has been set by the I2C or SPI core settings.  |
| cfg_stdby               | Output | Stand-by signal – Hardwired signal to be connected ONLY to the Power Controller of the MachXO4 device for functional simulation support. This signal is enabled only if the Wakeup Enable feature has been set by the I2C or SPI core settings. |

## 5. Register Description

### 5.1. I2C Registers

Both I2C cores communicate with the EFB WISHBONE interface through a set of control, command, status, and data registers. The following tables show the register names and their functions.

**Table 5.1. I2C Functions Register Map**

| Offset | Register Name | Description                     | Access | Default           |
|--------|---------------|---------------------------------|--------|-------------------|
| 0x40   | I2C_1_CR      | Primary I2C Control             | RW     | 0x00              |
| 0x41   | I2C_1_CMDR    | Primary I2C Command             | RW     | 0x00              |
| 0x42   | I2C_1_BR0     | Primary I2C Clock Pre-scale 0   | RW     | 0x00 <sup>1</sup> |
| 0x43   | I2C_2_BR1     | Primary I2C Clock Pre-scale 1   | RW     | 0x00 <sup>1</sup> |
| 0x44   | I2C_1_TXDR    | Primary I2C Transmit Data       | WO     | —                 |
| 0x45   | I2C_1_SR      | Primary I2C Status              | RO     | 0x00              |
| 0x46   | I2C_1_GCDR    | Primary I2C General Call        | RO     | —                 |
| 0x47   | I2C_1_RXDR    | Primary I2C Receive Data        | RO     | —                 |
| 0x48   | I2C_1_IRQ     | Primary I2C IRQ                 | RW     | —                 |
| 0x49   | I2C_1_IRQEN   | Primary I2C IRQ Enable          | RW     | 0x00              |
| 0x4A   | I2C_2_CR      | Secondary I2C Control           | RW     | 0x00              |
| 0x4B   | I2C_2_CMDR    | Secondary I2C Command           | RW     | 0x00              |
| 0x4C   | I2C_2_BR0     | Secondary I2C Clock Pre-scale 0 | RW     | 0x00 <sup>1</sup> |
| 0x4D   | I2C_2_BR1     | Secondary I2C Clock Pre-scale 1 | RW     | 0x00 <sup>1</sup> |
| 0x4E   | I2C_2_TXDR    | Secondary I2C Transmit Data     | WO     | —                 |
| 0x4F   | I2C_2_SR      | Secondary I2C Status            | RO     | 0x00              |
| 0x50   | I2C_2_GCDR    | Secondary I2C General Call      | RO     | —                 |
| 0x51   | I2C_2_RXDR    | Secondary I2C Receive Data      | RO     | —                 |
| 0x52   | I2C_2_IRQ     | Secondary I2C IRQ               | RW     | —                 |
| 0x53   | I2C_2_IRQEN   | Secondary I2C IRQ Enable        | RW     | 0x00              |

**Note:**

- The hardware default is overridden by parameter configurations.

**Table 5.2. I2C Control (Primary/Secondary)**

| I2C_1_CR / I2C_2_CR |       |      |        |            |                  |     |            | 0x40/0x4A |
|---------------------|-------|------|--------|------------|------------------|-----|------------|-----------|
| Bit                 | 7     | 6    | 5      | 4          | 3                | 2   | 1          | 0         |
| Name                | I2CEN | GCEN | WKUPEN | (Reserved) | SDA_DEL_SEL[1:0] |     | (Reserved) |           |
| Default             | 0     | 0    | 0      | 0          | 0                | 0   | 0          | 0         |
| Access              | R/W   | R/W  | R/W    | —          | R/W              | R/W | —          | —         |

**Note:** A write to this register causes the I2C core to reset.

**I2CEN**

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

- 0: I2C function is disabled
- 1: I2C function is enabled

**GCEN**

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

- 0: Disable
- 1: Enable

The General Call address is defined as 0000000 and works with either 7-bit or 10-bit addressing.

**WKUPEN** Wake-up from Standby/Sleep (by Target 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. Currently not supported by this IP.

0: Disable

1: Enable

**SDA\_DEL\_SEL[1:0]** SDA Output Delay (Tdel) Selection.

00: 300 ns (min) 300 ns + 2000 / [wb\_clk\_i frequency in MHz] (max)

01: 150 ns (min) 150 ns + 2000 / [wb\_clk\_i frequency in MHz] (max)

10: 75 ns (min) 75 ns + 2000 / [wb\_clk\_i frequency in MHz] (max)

11: 0 ns (min) 0 ns + 2000 / [wb\_clk\_i frequency in MHz] (max)

**Table 5.3. I2C Command (Primary/Secondary)**

| 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 (Controller operation).

**STO** Generate STOP condition (Controller operation).

**RD** Indicate Read from target (Controller operation).

**WR** Indicate Write to target (Controller operation).

**ACK** Acknowledge Option – when receiving, ACK transmission selection.

0: Send ACK

1: Send NACK

**CKSDIS** Clock Stretching Disable. The I2C cores support a wait state or clock stretching from the target, meaning the target can enforce a wait state if it needs time to finish the task. If desired, the CKSDIS bit can be used to disable the clock stretching. In this case, the overflow flag must be monitored. For Controller operations, set this bit to 0. Clock stretching is used by the EFB I2C Target during both *read* and *write* operations (from the Controller perspective) when I2C Command Register bit CKSDIS = 0.

During a read operation (Target transmitting), clock stretching occurs when TXDR is empty (under-run condition). During a write operation (Target receiving) clock stretching occurs when RXDR is full (over-run condition).

Translated into I2C Status register bits, the I2C clock-stretches if TRRDY = 1. The decision to enable clock stretching is done on the 8TH SCL + 2 WISHBONE clocks.

0: Enabled

1: Disabled

**Table 5.4. I2C Clock Prescale 0 (Primary/Secondary)**

| I2C_1_BR0 / I2C_2_BR0 |                   |     |     |     |     |     | 0x42/0x4C |     |
|-----------------------|-------------------|-----|-----|-----|-----|-----|-----------|-----|
| Bit                   | 7                 | 6   | 5   | 4   | 3   | 2   | 1         | 0   |
| Name                  | I2C_PRESCALE[7:0] |     |     |     |     |     |           |     |
| Default               | 0                 | 0   | 0   | 0   | 0   | 0   | 0         | 0   |
| Access                | R/W               | R/W | R/W | R/W | R/W | R/W | R/W       | R/W |

**Note:** The hardware default value may be overridden by EFB component instantiation parameters. See the description below.

**Table 5.5. I2C Clock Prescale 1 (Primary/Secondary)**

| I2C_1_BR1 / I2C_2_BR1 |            |   |   |   |   |   |                   | 0x43/0x4D |
|-----------------------|------------|---|---|---|---|---|-------------------|-----------|
| Bit                   | 7          | 6 | 5 | 4 | 3 | 2 | 1                 | 0         |
| Name                  | (Reserved) |   |   |   |   |   | I2C_PRESCALE[9:8] |           |
| Default               | 0          | 0 | 0 | 0 | 0 | 0 | 0                 | 0         |
| Access                | —          | — | — | — | — | — | R/W               | R/W       |

**Note:** The hardware default value may be overridden by EFB component instantiation parameters. See the description below.

**I2C\_PRESCALE[9:0]** I2C Clock Prescale value. A write operation to I2C\_PRESCALE [9:8] causes an I2C core reset. The WISHBONE clock frequency is divided by (I2C\_PRESCALE × 4) to produce the Controller I2C clock frequency supported by the I2C bus (50 kHz, 100 kHz, 400 kHz).

**Notes:**

- Different from transmitting a Controller, the practical limit for Target I2C bus speed support is (WISHBONE clock) / 2048. For example, the maximum WISHBONE clock frequency to support a 50 kHz Target I2C operation is 102 MHz.
- The digital value is pre-calculated during IP Generation in the GUI. The calculation is based on the WISHBONE Clock Frequency and the I2C Frequency, both of which are user-entered values. The digital value of the divider is programmed in the MachXO4 device during device programming. After power-up or device reconfiguration, the data is loaded onto the I2C\_1\_BR1/0 and I2C\_2\_BR1/0 registers.

Registers I2C\_1\_BR1/0 and I2C\_2\_BR1/0 have Read/Write access from the WISHBONE interface. You can update these clock pre-scale registers dynamically during device operation; however, care must be taken to not violate the I2C bus frequencies.

**Table 5.6. I2C Transmit Data Register (Primary/Secondary)**

| I2C_1_TXDR / I2C_2_TXDR |                   |   |   |   |   |   |   | 0x44/0x4E |
|-------------------------|-------------------|---|---|---|---|---|---|-----------|
| Bit                     | 7                 | 6 | 5 | 4 | 3 | 2 | 1 | 0         |
| Name                    | I2C_Transmit[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 to be transmitted on the I2C bus during the Write Data phase. Bit 0 is the LSB and is transmitted last. When transmitting the target address, Bit 0 represents the Read/Write bit.

**Table 5.7. I2C Status (Primary/Secondary)**

| 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 suffers 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

|              |  |
|--------------|--|
| <b>BUSY</b>  | <p>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.</p> <p>1: I2C bus busy</p> <p>0: I2C bus not busy</p>  |
| <b>RARC</b>  | <p>Received Acknowledge. An acknowledge response was received by the acknowledge bit monitor. All ACK/NACK bits are monitored and reported, regardless of Controller/Target source or Read/Write mode.</p> <p>1: No acknowledgement received</p> <p>0: Acknowledgement received</p>  |
| <b>SRW</b>   | <p>Target Read/Write. Indicates transmit or receive mode.</p> <p>1: Controller receiving/target transmitting</p> <p>0: Controller transmitting/target receiving</p> <p><b>Note:</b> SRW is valid after TRRDY = 1 following a synchronization delay of up to four WISHBONE clock cycles. Do not test both SRW and TRRDY in the same WISHBONE transaction, but test SRW at least four WISHBONE clock cycles after TRRDY is tested true.</p>                      |
| <b>ARBL</b>  | <p>Arbitration Lost. The core has lost arbitration in Controller mode. This bit can generate an interrupt.</p> <p>1: Arbitration Lost</p> <p>0: Normal</p>   |
| <b>TRRDY</b> | <p>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 controller/target mode and SRW status). This bit is capable of generating an interrupt.</p> <p>1: Transmitter or Receiver is ready</p> <p>0: Transmitter or Receiver is not ready</p>   |
| <b>TROE</b>  | <p>Transmitter/Receiver Overrun Error. A transmit or receive overrun error has occurred (dependent upon controller /target mode and SRW status).</p> <p><b>Note:</b> When acting as a transmitter (Controller Write or Target Read) a No Acknowledge received asserts TROE indicating a possible orphan data byte exists in TXDR. This bit can generate an interrupt.</p> <p>1: Transmitter or Receiver Overrun detected or NACK received</p> <p>0: Normal</p> |
| <b>HGC</b>   | <p>Hardware General Call Received. A hardware general call has been received in target mode. The corresponding command byte will be available in the General Call Data Register. This bit can generate an interrupt.</p> <p>1: General Call Received in target mode</p> <p>0: Normal</p>   |

**Table 5.8. I2C General Call Data Register (Primary/Secondary)**

| 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.



**Table 5.9. I2C Receive Data Register (Primary/Secondary)**

| 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 the LSB and is received last.

**Table 5.10. I2C Interrupt Status (Primary/Secondary)**

| 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 is asserted. Write a 1 to this bit to clear the interrupt.

- 1: Arbitration Lost Interrupt
- 0: No interrupt

**IRQTRRDY** Interrupt Status for Transmitter or Receiver Ready. When enabled, indicates TRRDY is asserted. Write a 1 to this bit to clear the interrupt.

- 1: Transmitter or Receiver Ready Interrupt
- 0: No interrupt

**IRQTROE** Interrupt Status for Transmitter/Receiver Overrun or NACK received. When enabled, indicates TROE is asserted. Write a 1 to this bit to clear the interrupt.

- 1: Transmitter or Receiver Overrun or NACK received Interrupt
- 0: No interrupt

**IRQHGC** Interrupt Status for Hardware General Call Received. When enabled, indicates HGC is asserted. Write a 1 to this bit to clear the interrupt.

- 1: General Call Received in target mode Interrupt
- 0: No interrupt

**Table 5.11. I2C Interrupt Enable (Primary/Secondary)**

| I2C_1_IRQEN / I2C_2_IRQEN |            |   |   |   |          |            |           | 0x49/0x53 |
|---------------------------|------------|---|---|---|----------|------------|-----------|-----------|
| Bit                       | 7          | 6 | 5 | 4 | 3        | 2          | 1         | 0         |
| Name                      | (Reserved) |   |   |   | IRQARBLN | IRQTRRDYEN | IRQTROEEN | IRQHGCEN  |
| Default                   | 0          | 0 | 0 | 0 | 0        | 0          | 0         | 0         |
| Access                    | —          | — | — | — | R/W      | R/W        | R/W       | R/W       |

**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

**IRQHCEN** Interrupt Enable for Hardware General Call Received.

- 1: Interrupt generation enabled
- 0: Interrupt generation disabled

## 5.2. SPI Registers

**Table 5.12. SPI Functions Register Map**

| Offset | Register Name | Description              | Access | Default           |
|--------|---------------|--------------------------|--------|-------------------|
| 0x54   | SPICR0        | Control Register 0       | RW     | 0x00              |
| 0x55   | SPICR1        | Control Register 1       | RW     | 0x00 <sup>1</sup> |
| 0x56   | SPICR2        | Control Register 2       | RW     | 0x00 <sup>1</sup> |
| 0x57   | SPIBR         | Clock Pre-scale          | RW     | 0x00 <sup>1</sup> |
| 0x58   | SPICSR        | Controller Chip Select   | RW     | 0x00              |
| 0x59   | SPITXDR       | Transmit Data            | WO     | —                 |
| 0x5A   | SPISR         | Status                   | RO     | —                 |
| 0x5B   | SPIRXDR       | Receive Data             | RO     | —                 |
| 0x5C   | SPIIRQ        | Interrupt Request        | RW     | —                 |
| 0x5D   | SPIIRQEN      | Interrupt Request Enable | RW     | 0x00              |

**Note:**

- The hardware default is overridden by parameter configurations.

**Table 5.13. SPI Control 0**

| SPICR0  |                 |     |                  |     |     |                 |     | 0x54 |
|---------|-----------------|-----|------------------|-----|-----|-----------------|-----|------|
| Bit     | 7               | 6   | 5                | 4   | 3   | 2               | 1   | 0    |
| Name    | TIdle_XCNT[1:0] |     | TTrail_XCNT[2:0] |     |     | TLead_XCNT[2:0] |     |      |
| Default | 0               | 0   | 0                | 0   | 0   | 0               | 0   | 0    |
| Access  | R/W             | R/W | R/W              | R/W | R/W | R/W             | R/W | R/W  |

**Note:** A write to this register causes the SPI core to reset.

**TIdle\_XCNT[1:0]** Idle Delay Count. Specifies the minimum interval prior to the Controller Chip Select low assertion (controller mode only), in SCK periods.

- 00: ½
- 01: 1
- 10: 1.5
- 11: 2

**TTrail\_XCNT[2:0]** Trail Delay Count. Specifies the minimum interval between the last edge of SCK and the high deassertion of Controller Chip Select (controller mode only), in SCK periods.

- 000: ½
- 001: 1
- 010: 1.5
- ...
- 111: 4

**TLead\_XCNT[2:0]** Lead Delay Count. Specifies the minimum interval between the Controller Chip Select low assertion and the first edge of SCK (controller mode only), in SCK periods.

000: ½  
001: 1  
010: 1.5  
...  
111: 4

**Table 5.14. SPI Control 1**

| SPICR1  |     |             |            |        |            |   |   | 0x55 |
|---------|-----|-------------|------------|--------|------------|---|---|------|
| Bit     | 7   | 6           | 5          | 4      | 3          | 2 | 1 | 0    |
| Name    | SPE | WKUPEN_USER | WKUPEN_CFG | TXEDGE | (Reserved) |   |   |      |
| Default | 0   | 0           | 0          | 0      | 0          | 0 | 0 | 0    |
| Access  | R/W | R/W         | R/W        | R/W    | —          | — | — | —    |

**Note:** A write to this register causes the SPI core to reset.

**SPE** This bit enables the SPI core functions. If SPE is cleared, SPI is disabled and forced into idle state.

0: SPI disabled  
1: SPI enabled, port pins are dedicated to SPI functions

**WKUPEN\_USER** Wake-up Enable through User. Enables the SPI core to send a wake-up signal to the on-chip Power Controller to wake the part from Standby mode when the target SPI chip select (spi\_scsn) is driven low.

0: Wakeup disabled  
1: Wakeup enabled

**WKUPEN\_CFG** Wake-up Enable Configuration. Enables the SPI core to send a wake-up signal to the on-chip power controller to wake the part from standby mode when the Configuration target SPI chip select (ufm\_sn) is driven low.

0: Wakeup disabled  
1: Wakeup enabled

**TXEDGE** Data Transmit Edge. Enables Lattice proprietary extension to the SPI protocol. Selects which clock edge to transmit SPI data.

0: Transmit data on the MCLK/CCLK edge defined by SPICR2[CPOL] and SPICR2[CPHA].  
1: Transmit data ½ MCLK/CCLK earlier than defined by SPICR2[CPOL] and SPICR2[CPHA].

**Table 5.15. SPI Control 2**

| SPICR2  |      |      |       |            |            |      |      | 0x56 |
|---------|------|------|-------|------------|------------|------|------|------|
| Bit     | 7    | 6    | 5     | 4          | 3          | 2    | 1    | 0    |
| Name    | MSTR | MCSH | SDBRE | (Reserved) | (Reserved) | CPOL | CPHA | LSBF |
| Default | 0    | 0    | 0     | 0          | 0          | 0    | 0    | 0    |
| Access  | R/W  | R/W  | R/W   | —          | —          | —    | —    | —    |

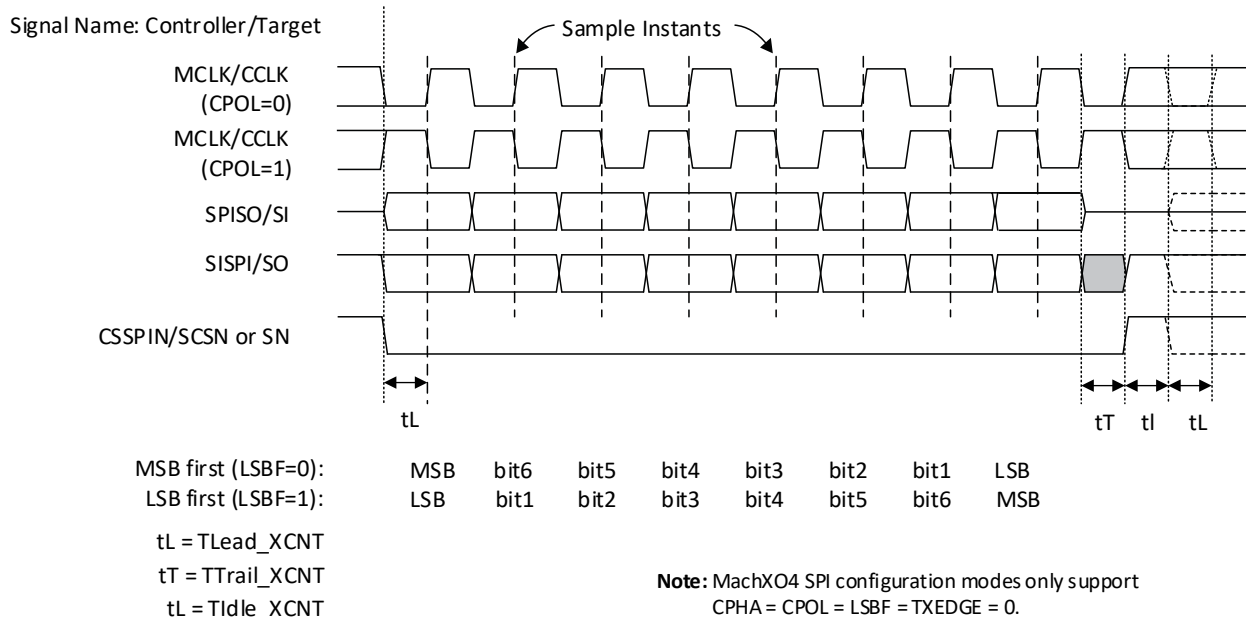
**Note:** A write to this register causes the SPI core to reset.

**MSTR** SPI Controller/Target Mode. Selects the controller/target operation mode of the SPI core. Changing this bit forces the SPI system into idle state.

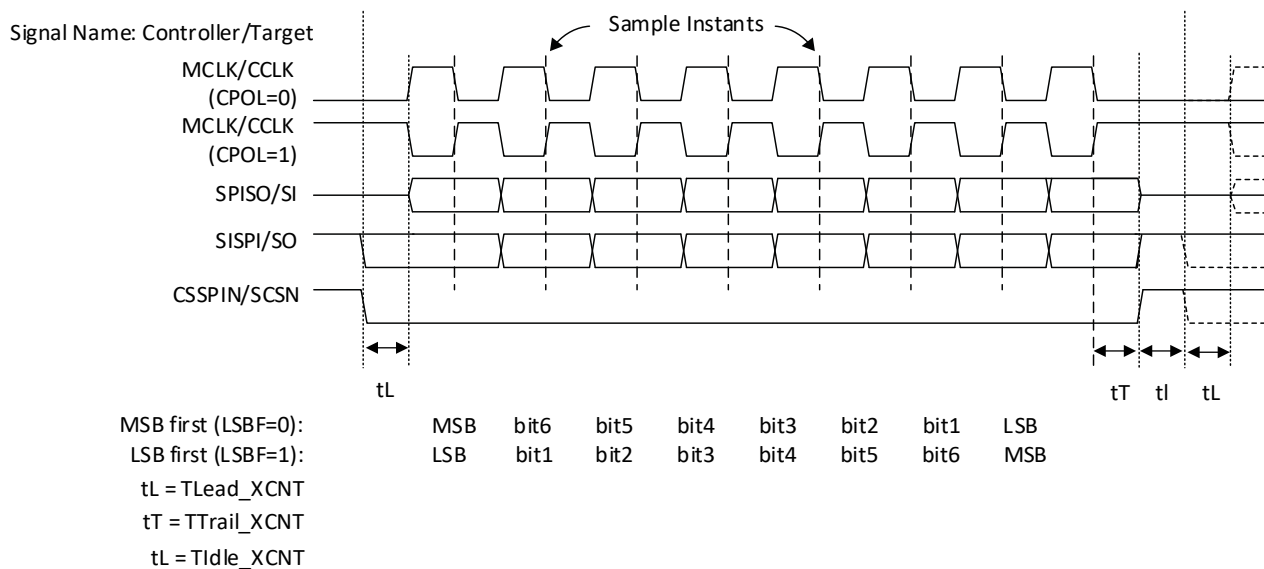
0: SPI is in target mode  
1: SPI is in controller mode

|              |   |
|--------------|---|
| <b>MCSH</b>  | <p>SPI Controller CSSPIN Hold. Holds the Controller chip select active when the host is busy, to halt the data transmission without de-asserting chip select.</p> <p><b>Note:</b> This mode must be used only when the WISHBONE clock has been divided by a value greater than four (4).</p> <p>0: Controller running as normal</p> <p>1: Controller holds chip select low even if there is no data to be transmitted</p>   |
| <b>SDBRE</b> | <p>Target Dummy Byte Response Enable. Enables Lattice proprietary extension to the SPI protocol. For use when the internal support circuit (for example, WISHBONE host) cannot respond with initial data within the required time, and ensure the target read out data predictably available at high SPI clock rates.</p> <p>When enabled, dummy 0xFF bytes are transmitted in response to a SPI target read (while SPISR[TRDY]=1) until an initial write to SPITXDR. Once a byte is written into SPITXDR by the WISHBONE host, a single byte of 0x00 is transmitted then followed immediately by the data in SPITXDR. In this mode, the external SPI controller should scan for the initial 0x00 byte when reading the SPI target to indicate the beginning of actual data.</p> <p>0: Normal target SPI operation.</p> <p>1: Lattice proprietary target dummy byte response enabled</p> <p><b>Note:</b> This mechanism only applies to the initial data delay period. Once the initial data is available, subsequent data must be supplied to SPITXDR at the required SPI bus data rate.</p> |
| <b>CPOL</b>  | <p>SPI Clock Polarity. Selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical SPICR2[CPOL] values. In controller mode, a change of this bit aborts a transmission in progress and forces the SPI system into idle state.</p> <p>0: Active-high clocks selected</p> <p>1: Active-low clocks selected</p>   |
| <b>CPHA</b>  | <p>SPI Clock Phase. Selects the SPI clock format. In controller mode, a change of this bit aborts a transmission in progress and forces the SPI system into idle state.</p> <p>0: Data is captured on a leading (first) clock edge, and propagated on the opposite clock edge.</p> <p>1: Data is captured on a trailing (second) clock edge, and propagated on the opposite clock edge.</p> <p><b>Note:</b> When CPHA=1, you must explicitly place a pull-up or pull-down on SCK pad corresponding to the value of CPOL (for example, when CPHA=1 and CPOL=0 place a pull-down on SCK). When CPHA=0, the pull direction may be set arbitrarily.</p> <p>Target SPI Configuration mode supports default setting only for CPOL and CPHA.</p>   |
| <b>LSBF</b>  | <p>LSB-First. LSB appears first on the SPI interface. In controller mode, a change of this bit aborts a transmission in progress and forced the SPI system into idle state.</p> <p><b>Note:</b> This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in bit 7.</p> <p>0: Data is transferred, most significant bit (MSB) first.</p> <p>1: Data is transferred, least significant bit (LSB) first.</p>   |

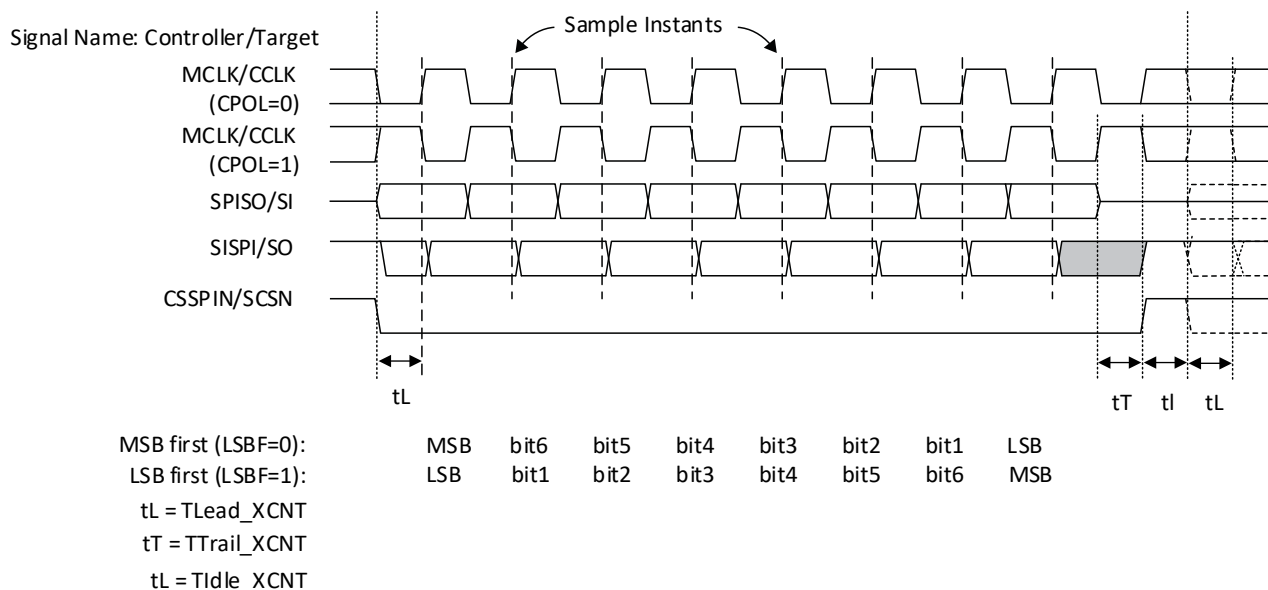
Refer to the following timing diagrams for the behaviors of the SPI Control registers.



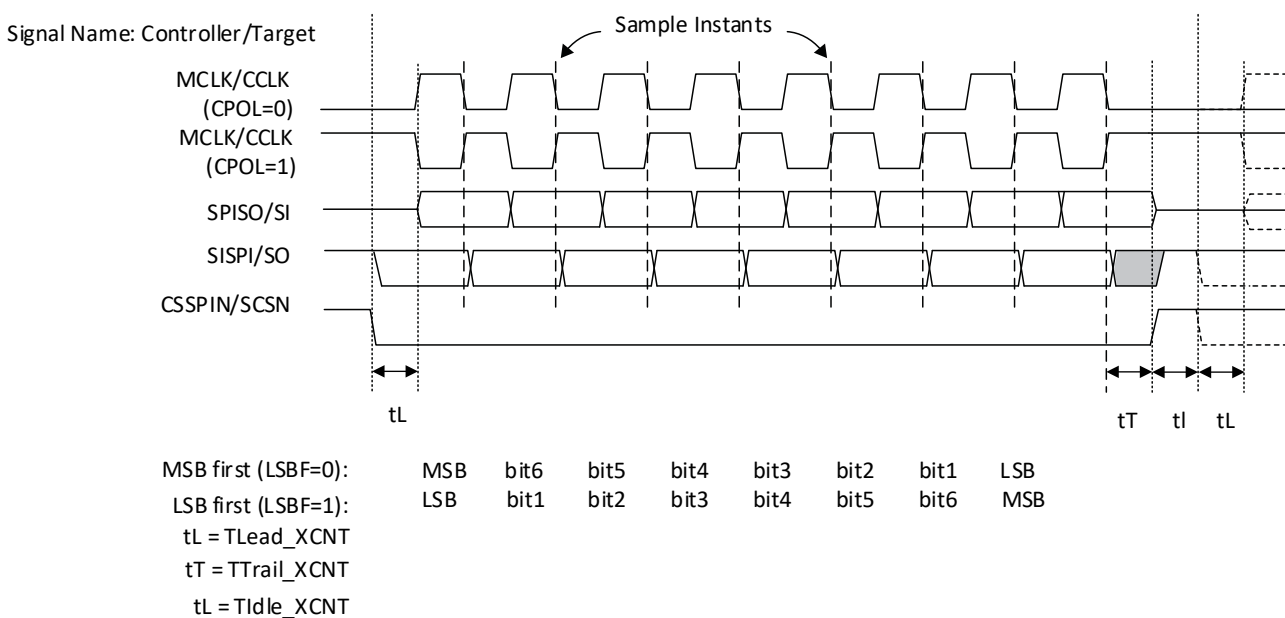
**Figure 5.1. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=0)**



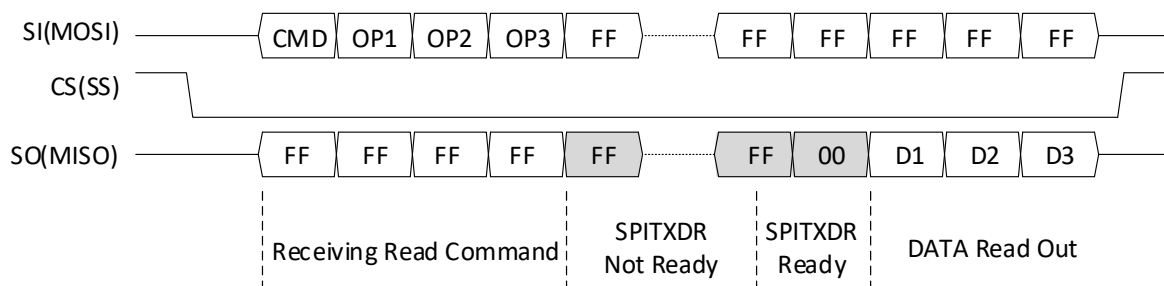
**Figure 5.2. SPI Control Timing (SPICR2[CPHA]=1, SPICR1[TXEDGE]=0)**



**Figure 5.3. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=1)**



**Figure 5.4. SPI Control Timing (SPICR2[CPHA]=1, SPICR1[TXEDGE]=1)**



**Figure 5.5. Target SPI Dummy Byte Response (SPICR2[SDBRE]) Timing**

**Table 5.16. SPI Clock Pre-scale**

| SPIBR                |            |   |              |     |     |     |     | 0x57 |
|----------------------|------------|---|--------------|-----|-----|-----|-----|------|
| Bit                  | 7          | 6 | 5            | 4   | 3   | 2   | 1   | 0    |
| Name                 | (Reserved) |   | DIVIDER[5:0] |     |     |     |     |      |
| Default <sup>1</sup> | 0          | 0 | 0            | 0   | 0   | 0   | 0   | 0    |
| Access               | —          | — | R/W          | R/W | R/W | R/W | R/W | R/W  |

**Note:**

1. Hardware default value may be overridden by EFB component instantiation parameters. See the discussion below.

**DIVIDER[5:0]**

SPI Clock Pre-scale value. The WISHBONE clock frequency is divided by (DIVIDER[5:0] + 1) to produce the desired SPI clock frequency. A write operation to this register causes an SPI core reset. The DIVIDER value must be greater than 1.

**Note:** The digital value is calculated by IP Catalog when the SPI core is configured in the SPI tab of the EFB GUI. The calculation is based on the WISHBONE Clock Frequency and the SPI Frequency, both of which are user-entered values. The digital value of the divider is programmed in the MachXO4 device during device programming. After power-up or device reconfiguration, the data is loaded into the SPIBR register.

Register SPIBR has read/write access from the WISHBONE interface. The clock pre-scale register can be dynamically updated during device operation.

**Table 5.17. SPI Controller Chip Select**

| SPICSR  |       |       |       |       |       |       |       | 0x58  |
|---------|-------|-------|-------|-------|-------|-------|-------|-------|
| Bit     | 7     | 6     | 5     | 4     | 3     | 2     | 1     | 0     |
| Name    | CSN_7 | CSN_6 | CSN_5 | CSN_4 | CSN_3 | CSN_2 | CSN_1 | CSN_0 |
| Default | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |
| Access  | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |

**CSN\_[7:0]**

SPI Controller Chip Selects. Used in controller mode for asserting a specific Controller Chip Select line. The register has eight bits, enabling the SPI core to control up to eight external SPI target devices. Each bit represents one controller chip select line (Active-Low). Bits [7:1] may be connected to any I/O pin through the FPGA fabric. Bit 0 has a pre-assigned pin location. The register has read/write access from the WISHBONE interface. A write operation on this register causes the SPI core to reset.

**Table 5.18. SPI Transmit Data Register**

| SPITXDR |                        |   |   |   |   |   |   | 0x59 |
|---------|------------------------|---|---|---|---|---|---|------|
| Bit     | 7                      | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | SPI_Transmit_Data[7:0] |   |   |   |   |   |   |      |
| Default | —                      | — | — | — | — | — | — | —    |
| Access  | W                      | W | W | W | W | W | W | W    |

**SPI\_Transmit\_Data[7:0]**

SPI Transmit Data. This register holds the byte to be transmitted on the SPI bus. Bit 0 in this register is LSB, and is transmitted last when SPICR2[LSBF]=0 or first when SPICR2[LSBF]=1.

**Note:** When operating as a target, SPITXDR must be written when SPISR[TRDY] is 1 and at least 0.5 CCLKs before the first bit is to appear on SO. For example, when CPOL = CPHA = TXEDGE = LSBF = 0, SPITXDR must be written prior to the CCLK rising edge used to sample the LSB (bit 0) of the previous byte. This timing requires at least one protocol dummy byte be included for all target SPI read operations.

**Table 5.19. SPI Status**

| SPISR   |     |            |   |      |      |            |     | 0x5A |
|---------|-----|------------|---|------|------|------------|-----|------|
| Bit     | 7   | 6          | 5 | 4    | 3    | 2          | 1   | 0    |
| Name    | TIP | (Reserved) |   | TRDY | RRDY | (Reserved) | ROE | MDF  |
| Default | 0   | —          | — | 0    | 0    | —          | 0   | 0    |
| Access  | R   | —          | — | R    | R    | —          | R   | R    |

|             |  |
|-------------|--|
| <b>TIP</b>  | SPI Transmitting In Progress. Indicates the SPI port is actively transmitting/receiving data.<br>0: SPI Transmitting completed<br>1: SPI Transmitting in progress  |
| <b>TRDY</b> | SPI Transmit Ready. Indicates the SPI transmit data register (SPITXDR) is empty. This bit is cleared by a write to SPITXDR. This bit is capable of generating an interrupt.<br>0: SPITXDR is not empty<br>1: SPITXDR is empty  |
| <b>RRDY</b> | SPI Receive Ready. Indicates the receive data register (SPIRXDR) contains valid receive data. This bit is cleared by a read access to SPIRXDR. This bit is capable of generating an interrupt.<br>0: SPIRXDR does not contain data<br>1: SPIRXDR contains valid receive data |
| <b>ROE</b>  | Receive Overrun Error. Indicates SPIRXDR receives new data before the previous data is read. The previous data is lost. This bit is capable of generating an interrupt.<br>0: Normal<br>1: Receiver Overrun detected   |
| <b>MDF</b>  | Mode Fault. Indicates the Target SPI chip select (spi_scsn) is driven low while SPICR2[MSTR]=1. This bit is cleared by any write to SPICR0, SPICR1, or SPICR2. This bit is capable of generating an interrupt.<br>0: Normal<br>1: Mode Fault detected                        |

**Table 5.20. SPI Receive Data Register**

| SPIRXDR |                       |   |   |   |   |   |   | 0x5B |
|---------|-----------------------|---|---|---|---|---|---|------|
| Bit     | 7                     | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | SPI_Receive_Data[7:0] |   |   |   |   |   |   |      |
| Default | 0                     | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | R                     | R | R | R | R | R | R | R    |

**SPI\_Receive\_Data[7:0]** SPI Receive Data. This register holds the byte captured from the SPI bus. Bit 0 in this register is LSB and is received last when LSBF=0 or first when LSBF=1.

**Table 5.21. SPI Interrupt Status**

| SPIIRQ  |            |   |   |         |         |            |        | 0x5C   |
|---------|------------|---|---|---------|---------|------------|--------|--------|
| Bit     | 7          | 6 | 5 | 4       | 3       | 2          | 1      | 0      |
| Name    | (Reserved) |   |   | IRQTRDY | IRQRRDY | (Reserved) | IRQROE | IRQMDF |
| Default | —          | — | — | 0       | 0       | —          | 0      | 0      |
| Access  | —          | — | — | R/W     | R/W     | —          | R/W    | R/W    |



|                |   |
|----------------|---|
| <b>IRQTRDY</b> | Interrupt Status for SPI Transmit Ready. When enabled, indicates SPISR[TRDY] is asserted. Write a 1 to this bit to clear the interrupt.<br>1: SPI Transmit Ready Interrupt<br>0: No interrupt |
| <b>IRQRRDY</b> | Interrupt Status for SPI Receive Ready. When enabled, indicates SPISR[RRDY] is asserted. Write a 1 to this bit to clear the interrupt.<br>1: SPI Receive Ready Interrupt<br>0: No interrupt   |
| <b>IRQROE</b>  | Interrupt Status for Receive Overrun Error. When enabled, indicates ROE is asserted. Write a 1 to this bit to clear the interrupt.<br>1: Receive Overrun Error Interrupt<br>0: No interrupt   |
| <b>IRQMDF</b>  | Interrupt Status for Mode Fault. When enabled, indicates MDF is asserted. Write a 1 to this bit to clear the interrupt.<br>1: Mode Fault Interrupt<br>0: No interrupt                         |

**Table 5.22. SPI Interrupt Enable**

| SPIIRQEN |            |   |   |           |           |            |          | 0x5D     |
|----------|------------|---|---|-----------|-----------|------------|----------|----------|
| Bit      | 7          | 6 | 5 | 4         | 3         | 2          | 1        | 0        |
| Name     | (Reserved) |   |   | IRQTRDYEN | IRQRRDYEN | (Reserved) | IRQROEEN | IRQMDFEN |
| Default  | —          | — | — | 0         | 0         | —          | 0        | 0        |
| Access   | —          | — | — | R/W       | R/W       | —          | R/W      | R/W      |

|                  |  |
|------------------|--|
| <b>IRQTRDYEN</b> | Interrupt Enable for SPI Transmit Ready.<br>1: Interrupt generation enabled<br>0: Interrupt generation disabled    |
| <b>IRQRRDYEN</b> | Interrupt Enable for SPI Receive Ready.<br>1: Interrupt generation enabled<br>0: Interrupt generation disabled     |
| <b>IRQROEEN</b>  | Interrupt Enable for Receive Overrun Error.<br>1: Interrupt generation enabled<br>0: Interrupt generation disabled |
| <b>IRQMDFEN</b>  | Interrupt Enable for Mode Fault.<br>1: Interrupt generation enabled<br>0: Interrupt generation disabled            |

## 5.3. Timer/Counter Registers

**Table 5.23. Timer/Counter Functions Register Map**

| Offset | Register Name | Description                              | Access | Default           |
|--------|---------------|--|--------|-------------------|
| 0x5E   | TCCR0         | Control Register 0                       | RW     | 0x00 <sup>1</sup> |
| 0x5F   | TCCR1         | Control Register 1                       | RW     | 0x00 <sup>1</sup> |
| 0x60   | TCTOPSET0     | Set Top Counter Value [7:0]              | WO     | 0x00 <sup>1</sup> |
| 0x61   | TCTOPSET1     | Set Top Counter Value [15:8]             | WO     | 0x00 <sup>1</sup> |
| 0x62   | TCOCRSET0     | Set Compare Counter Value [7:0]          | WO     | 0x00 <sup>1</sup> |
| 0x63   | TCOCRSET1     | Set Compare Counter Value [15:8]         | WO     | 0x00 <sup>1</sup> |
| 0x64   | TCCR2         | Control Register 2                       | RW     | 0x00 <sup>1</sup> |
| 0x65   | TCCNT0        | Counter Value [7:0]                      | RO     | —                 |
| 0x66   | TCCNT1        | Counter Value [15:8]                     | RO     | —                 |
| 0x67   | TCTOP0        | Current Top Counter Value [7:0]          | RO     | —                 |
| 0x68   | TCTOP1        | Current Top Counter Value [15:8]         | RO     | —                 |
| 0x69   | TCOCR0        | Current Compare Counter Value [7:0]      | RO     | —                 |
| 0x6A   | TCOCR1        | Current Compare Top Counter Value [15:8] | RO     | —                 |
| 0x6B   | TCICR0        | Current Capture Counter Value [7:0]      | RO     | —                 |
| 0x6C   | TCICR1        | Current Capture Counter Value [15:8]     | RO     | —                 |
| 0x6D   | TCSR0         | Status Register                          | RW     | —                 |
| 0x6E   | TCIRQ         | Interrupt Request                        | RW     | —                 |
| 0x6F   | TCIRQEN       | Interrupt Request Enable                 | RW     | 0x00 <sup>1</sup> |

**Note:**

1. The hardware default is overridden by parameter configurations.

**Table 5.24. Timer/Counter Control**

| TCCR0   |       |            |               |   |   |         | 0x5E   |            |
|---------|-------|------------|---------------|---|---|---------|--------|------------|
| Bit     | 7     | 6          | 5             | 4 | 3 | 2       | 1      | 0          |
| Name    | RSTEN | (Reserved) | PRESCALE[2:0] |   |   | CLKEDGE | CLKSEL | (Reserved) |
| Default | 0     | 0          | 0             |   |   | 0       | 0      | 0          |
| Access  | R/W   | —          | R/W           |   |   | R/W     | R/W    | R/W        |

**RSTEN** Enables the reset signal (tc\_rstn) to enter the Timer/Counter core from the PLD logic.

- 1: External reset enabled
- 0: External reset disabled

**PRESCALE[2:0]** Used to divide the clock input to the Timer/Counter.

- 000: Static (clock disabled)
- 001: Divide by 1
- 010: Divide by 8
- 011: Divide by 64
- 100: Divide by 256
- 101: Divide by 1024
- 110: (Reserved setting)
- 111: (Reserved setting)

**CLKEDGE** Used to select the edge of the input clock source. The Timer/Counter updates states on the edge of the input clock source.

- 0: Rising Edge
- 1: Falling Edge

**CLKSEL** Defines the source of the input clock.

0: Clock Tree

1: On-chip Oscillator

**Table 5.25. Timer/Counter Control 1**

| TCCR1   |            |        |      |      |          |   |          | 0x5F |
|---------|------------|--------|------|------|----------|---|----------|------|
| Bit     | 7          | 6      | 5    | 4    | 3        | 2 | 1        | 0    |
| Name    | (Reserved) | SOVFEN | ICEN | TSEL | OCM[1:0] |   | TCM[1:0] |      |
| Default | 0          | 0      | 0    | 0    | 0        |   | 0        |      |
| Access  | —          | R/W    | R/W  | R/W  | R/W      |   | R/W      |      |

**SOVFEN** Enables the overflow flag to be used with the interrupt output signal. It is set when the Timer/Counter is standalone, with no WISHBONE interface.

0: Disabled

1: Enabled

**Note:** When this bit is set, other flags such as the OCRF and ICRF is not routed to the interrupt output signal.

**ICEN** Enables the ability to perform a capture operation of the counter value. You can assert the *tc\_ic* signal and load the counter value onto the TCICR0/1 registers. The captured value can serve as a timer stamp for a specific event.

0: Disabled

1: Enabled

**TSEL** Enables the auto-load of the counter with the value from TCTOPSET0/1. When disabled, the value 0xFFFF is auto-loaded.

0: Disabled

1: Enabled

**OCM[1:0]** Select the function of the output signal of the Timer/Counter. The available functions are Static, Toggle, Set/Clear and Clear/Set.

All Timer/Counter modes:

00: The output is static low

In non-PWM modes:

01: Toggle on TOP match

In Fast PWM mode:

10: Clear on TOP match, Set on OCR match.

11: Set on TOP match, Clear on OCR match.

In Phase and Frequency Correct PWM mode:

10: Clear on OCR match when the counter is incrementing.

Set on OCR match when counter is decrementing.

11: Set on OCR match when the counter is incrementing.

Clear on OCR match when the counter is decrementing.

**TCM[1:0]** Timer Counter Mode. Defines the mode of operation for the Timer/Counter.

00: Watchdog Timer Mode

01: Clear Timer on Compare Match Mode

10: Fast PWM Mode

11: Phase and Frequency Correct PWM Mode

**Table 5.26. Timer/Counter Set Top Counter Value 0**

| TCTOPSET0            |               |     |     |     |     |     |     | 0x60 |
|----------------------|---------------|-----|-----|-----|-----|-----|-----|------|
| Bit                  | 7             | 6   | 5   | 4   | 3   | 2   | 1   | 0    |
| Name                 | TCTOPSET[7:0] |     |     |     |     |     |     |      |
| Default <sup>1</sup> | 1             | 1   | 1   | 1   | 1   | 1   | 1   | 1    |
| Access               | R/W           | R/W | R/W | R/W | R/W | R/W | R/W | R/W  |

**Note:**

1. Hardware default value may be overridden by EFB component instantiation parameters.

**Table 5.27. Timer/Counter Set Top Counter Value 1**

| TCTOPSET1            |                |     |     |     |     |     |     | 0x61 |
|----------------------|----------------|-----|-----|-----|-----|-----|-----|------|
| Bit                  | 7              | 6   | 5   | 4   | 3   | 2   | 1   | 0    |
| Name                 | TCTOPSET[15:8] |     |     |     |     |     |     |      |
| Default <sup>1</sup> | 1              | 1   | 1   | 1   | 1   | 1   | 1   | 1    |
| Access               | R/W            | R/W | R/W | R/W | R/W | R/W | R/W | R/W  |

**Note:**

1. Hardware default value may be overridden by EFB component instantiation parameters.

The value from TCTOPSET0/1 is loaded to the TCTOP0/1 registers once the counter has completed the current counting cycle. Refer to the [Timer/Counter](#) section for more details.

TCTOPSET0 register holds the lower eight bits [7:0] of the top value. TCTOPSET1 register holds the upper eight bits [15:8] of the top value.

**Table 5.28. Timer/Counter Set Compare Counter Value 0**

| TCOCRSET0            |               |     |     |     |     |     |     | 0x62 |
|----------------------|---------------|-----|-----|-----|-----|-----|-----|------|
| Bit                  | 7             | 6   | 5   | 4   | 3   | 2   | 1   | 0    |
| Name                 | TCOCRSET[7:0] |     |     |     |     |     |     |      |
| Default <sup>1</sup> | 1             | 1   | 1   | 1   | 1   | 1   | 1   | 1    |
| Access               | R/W           | R/W | R/W | R/W | R/W | R/W | R/W | R/W  |

**Note:**

1. Hardware default value may be overridden by EFB component instantiation parameters.

**Table 5.29. Timer/Counter Set Compare Counter Value 1**

| TCOCRSET1            |                |     |     |     |     |     |     | 0x63 |
|----------------------|----------------|-----|-----|-----|-----|-----|-----|------|
| Bit                  | 7              | 6   | 5   | 4   | 3   | 2   | 1   | 0    |
| Name                 | TCOCRSET[15:8] |     |     |     |     |     |     |      |
| Default <sup>1</sup> | 1              | 1   | 1   | 1   | 1   | 1   | 1   | 1    |
| Access               | R/W            | R/W | R/W | R/W | R/W | R/W | R/W | R/W  |

**Note:**

1. Hardware default value may be overridden by EFB component instantiation parameters.

The value from TCOCRSET0/1 is loaded to the TCOCR0/1 registers once the counter has completed the current counting cycle. Refer to the [Timer/Counter](#) section for more details.

TCOCRSET0 register holds the lower 8-bit value [7:0] of the compare value. TCOCRSET1 register holds the upper 8-bit value[15:8] of the compare value.

**Table 5.30. Timer/Counter Control 2**

| TCCR2   |            |   |   |   |   |         |         | 0x64    |
|---------|------------|---|---|---|---|---------|---------|---------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2       | 1       | 0       |
| Name    | (Reserved) |   |   |   |   | WBFORCE | WBRESET | WBPAUSE |
| Default | 0          | 0 | 0 | 0 | 0 | 0       | 0       | 0       |
| Access  | —          | — | — | — | — | R/W     | R/W     | R/W     |

|                 |  |
|-----------------|--|
| <b>WBFORCE</b>  | In non-PWM modes, forces the output of the counter, as if the counter value matches the compare (TCOCR) value or it matches the top value (TCTOP).   |
| 0:              | Disabled   |
| 1:              | Enabled  |
| <b>WBRESET</b>  | Reset the counter from the WISHBONE interface by writing a 1 to this bit. Manually reset to 0. The rising edge is detected in the WISHBONE clock domain, and the counter is reset synchronously on the next tc_clki. Due to the clock domain crossing, there is a one-clock uncertainty when the reset is effective. This bit has higher priority than WBPAAUSE. |
| 0:              | Disabled   |
| 1:              | Enabled  |
| <b>WBPAAUSE</b> | Pause the 16-bit counter   |
| 1:              | Pause  |
| 0:              | Normal   |

**Table 5.31. Timer/Counter Counter Value 0**

| TCCNT0  |            |   |   |   |   |   |   | 0x65 |
|---------|------------|---|---|---|---|---|---|------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCCNT[7:0] |   |   |   |   |   |   |      |
| Default | 0          | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | R          | R | R | R | R | R | R | R    |

**Table 5.32. Timer/Counter Counter Value 1**

| TCCNT1  |             |   |   |   |   |   |   | 0x66 |
|---------|-------------|---|---|---|---|---|---|------|
| Bit     | 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCCNT[15:8] |   |   |   |   |   |   |      |
| Default | 0           | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | R           | R | R | R | R | R | R | R    |

Registers TCCNT0 and TCCNT1 are 8-bit registers, which combined, hold the counter value. The WISHBONE host has read-only access to these registers.

TCCNT0 register holds the lower 8-bit value [7:0] of the counter value. TCCNT1 register holds the upper 8-bit value [15:8] of the counter value.

**Table 5.33. Timer/Counter Current Top Counter Value 0**

| TCTOP0  |            |   |   |   |   |   |   | 0x67 |
|---------|------------|---|---|---|---|---|---|------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCTOP[7:0] |   |   |   |   |   |   |      |
| Default | 1          | 1 | 1 | 1 | 1 | 1 | 1 | 1    |
| Access  | R          | R | R | R | R | R | R | R    |

**Table 5.34. Timer/Counter Current Top Counter Value 1**

| TCTOP1  |             |   |   |   |   |   |   | 0x68 |
|---------|-------------|---|---|---|---|---|---|------|
| Bit     | 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCTOP[15:8] |   |   |   |   |   |   |      |
| Default | 1           | 1 | 1 | 1 | 1 | 1 | 1 | 1    |
| Access  | R           | R | R | R | R | R | R | R    |

Registers TCTOP0 and TCTOP1 are 8-bit registers, which combined, receive a 16-bit value from the TCTOP-SET0/1. The data stored in these registers represents the top value of the counter. The registers update once the counter has completed the current counting cycle. The WISHBONE host has read-only access to these registers. Refer to the [Timer/Counter](#) section for more details.

TCTOP0 register holds the lower 8-bit value [7:0] of the top value. TCTOP1 register holds the upper 8-bit value [15:8] of the top value.

**Table 5.35. Timer/Counter Current Compare Counter Value 0**

| TCOCR0  |            |   |   |   |   |   |   | 0x69 |
|---------|------------|---|---|---|---|---|---|------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCOCR[7:0] |   |   |   |   |   |   |      |
| Default | 1          | 1 | 1 | 1 | 1 | 1 | 1 | 1    |
| Access  | R          | R | R | R | R | R | R | R    |

**Table 5.36. Timer/Counter Current Compare Counter Value 1**

| TCOCR1  |             |   |   |   |   |   |   | 0x6A |
|---------|-------------|---|---|---|---|---|---|------|
| Bit     | 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCOCR[15:8] |   |   |   |   |   |   |      |
| Default | 1           | 1 | 1 | 1 | 1 | 1 | 1 | 1    |
| Access  | R           | R | R | R | R | R | R | R    |

Registers TCOCR0 and TCOCR1 are 8-bit registers, which combined, receive a 16-bit value from the TCO-CRSET0/1. The data stored in these registers represents the compare value of the counter. The registers update once the counter has completed the current counting cycle. The WISHBONE host has read-only access to these registers. Refer to the [Timer/Counter](#) section for more details.

TCOCR0 register holds the lower 8-bit value [7:0] of the compare value. TCOCR1 register holds the upper 8-bit value [15:8] of the compare value.

**Table 5.37. Timer/Counter Current Capture Counter Value 0**

| TCICR0  |            |   |   |   |   |   |   | 0x6B |
|---------|------------|---|---|---|---|---|---|------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCICR[7:0] |   |   |   |   |   |   |      |
| Default | 0          | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | R          | R | R | R | R | R | R | R    |

**Table 5.38. Timer/Counter Current Capture Counter Value 1**

| TCICR1  |             |   |   |   |   |   |   | 0x6C |
|---------|-------------|---|---|---|---|---|---|------|
| Bit     | 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | TCICR[15:8] |   |   |   |   |   |   |      |
| Default | 0           | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | R           | R | R | R | R | R | R | R    |

Registers TCICR0 and TCICR1 are 8-bit registers, and when combined, they can hold the counter value. The counter value is loaded onto these registers once a trigger event, tc\_ic IP signal, is asserted. The capture value is commonly used as a timestamp for a specific system event. The WISHBONE host has read-only access to these registers.

TCICR0 register holds the lower 8-bit value [7:0] of the counter value. TCICR1 register holds the upper 8-bit value [15:8] of the counter value.

**Table 5.39. Timer/Counter Status Register**

| TCSR0   |            |   |   |   |     |      |      | 0x6D |
|---------|------------|---|---|---|-----|------|------|------|
| Bit     | 7          | 6 | 5 | 4 | 3   | 2    | 1    | 0    |
| Name    | (Reserved) |   |   |   | BTF | ICRF | OCRf | OVF  |
| Default | —          | — | — | — | 0   | 0    | 0    | 0    |
| Access  | —          | — | — | — | R   | R    | R    | R    |

- BTF** Bottom Flag. Asserted when the counter reaches the value of zero. A write operation to this register clears this flag.
- 1: Counter has reached zero  
0: Counter has not reached zero
- ICRF** Capture Counter Flag. Asserted when the TC\_IC input signal is asserted. The counter value is captured into the TCICR0/1 registers. A write operation to this register clears this flag. This bit is capable of generating an interrupt.
- 1: TC\_IC signal asserted  
0: Normal
- OCRf** Compare Match Flag. Asserted when counter matches the TCOCR0/1 register value. A write operation to this register clears this flag. This bit is capable of generating an interrupt.
- 1: Counter match  
0: Normal
- OVF** Overflow Flag. Asserted when the counter matches the TCTOP0/1 register value. A write operation to this register clears this flag. This bit is capable of generating an interrupt.
- 1: Counter match  
0: Normal

**Table 5.40. Timer/Counter Interrupt Status**

| TCIRQ   |            |   |   |   |   |         |         | 0x6E   |
|---------|------------|---|---|---|---|---------|---------|--------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2       | 1       | 0      |
| Name    | (Reserved) |   |   |   |   | IRQICRF | IRQOCRf | IRQOVF |
| Default | 0          | 0 | 0 | 0 | 0 | 0       | 0       | 0      |
| Access  | —          | — | — | — | — | R/W     | R/W     | R/W    |

- IRQICRF** Interrupt Status for Capture Counter Flag. When enabled, indicates ICRF is asserted. Write a 1 to this bit to clear the interrupt.
- 1: Capture Counter Flag Interrupt  
0: No interrupt
- IRQOCRf** Interrupt Status for Compare Match Flag. When enabled, indicates OCRf is asserted. Write a 1 to this bit to clear the interrupt.
- 1: Compare Match Flag Interrupt  
0: No interrupt
- IRQOVF** Interrupt Status for Overflow Flag. When enabled, indicates OVF is asserted. Write a 1 to this bit to clear the interrupt.
- 1: Overflow Flag Interrupt  
0: No interrupt

**Table 5.41. Timer/Counter Interrupt Enable**

| TCIRQEN |            |   |   |   |   |           |           | 0x6F     |
|---------|------------|---|---|---|---|-----------|-----------|----------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2         | 1         | 0        |
| Name    | (Reserved) |   |   |   |   | IRQICRFEN | IRQOCRFEN | IRQOVFEN |
| Default | 0          | 0 | 0 | 0 | 0 | 0         | 0         | 0        |
| Access  | —          | — | — | — | — | R/W       | R/W       | R/W      |

**IRQICRFEN** Interrupt Enable for Capture Counter Flag.

- 1: Interrupt generation enabled
- 0: Interrupt generation disabled

**IRQOCRFEN** Interrupt Enable for Compare Match Flag.

- 1: Interrupt generation enabled
- 0: Interrupt generation disabled

**IRQOVFEN** Interrupt Enable for Overflow Flag.

- 1: Interrupt generation enabled
- 0: Interrupt generation disabled

## 5.4. Flash Access Registers

The WISHBONE target interface of the EFB module enables you to access the flash interface directly from the FPGA core logic. The WISHBONE bus signals are utilized by a WISHBONE host that you can implement using the general-purpose FPGA resources.

The WISHBONE interface communicates to the configuration logic through a set of data, control, and status registers. The following tables show the register names and their functions.

**Table 5.42. Flash Memory Functions Register Map**

| Offset | Register Name | Description              | Access | Default |
|--------|---------------|--------------------------|--------|---------|
| 0x70   | CFGCR         | Control                  | RW     | 0x00    |
| 0x71   | CFGTXDR       | Transmit Data            | WO     | 0x00    |
| 0x72   | CFGSR         | Status                   | RO     | 0x00    |
| 0x73   | CFGRXDR       | Receive Data             | RO     | 0x00    |
| 0x74   | CFGIRQ        | Interrupt Request        | RW     | 0x00    |
| 0x75   | CFGIRQEN      | Interrupt Request Enable | RW     | 0x00    |

**Table 5.43. Flash Memory Control**

| CFGCR   |      |      |            |   |   |   |   | 0x70 |
|---------|------|------|------------|---|---|---|---|------|
| Bit     | 7    | 6    | 5          | 4 | 3 | 2 | 1 | 0    |
| Name    | WBCE | RSTE | (Reserved) |   |   |   |   |      |
| Default | 0    | 0    | 0          | 0 | 0 | 0 | 0 | 0    |
| Access  | R/W  | R/W  | —          | — | — | — | — | —    |

**WBCE**

WISHBONE Connection Enable. Enables the WISHBONE to establish the read/write connection to the Flash logic. This bit must be set prior to executing any command through the WISHBONE port. Likewise, this bit must be cleared to terminate the command. For more information on framing WISHBONE commands, see the [Command and Data Transfers to Flash Space](#) section.

- 1: Enabled
- 0: Disabled



**RSTE** WISHBONE Connection Reset. Resets the input/output FIFO logic. The reset logic is level sensitive. After setting this bit to 1, it must be cleared to 0 for normal operation.

1: Reset

0: Normal operation

**Table 5.44. Flash Memory Transmit Data**

| CFGTXDR |                        |   |   |   |   |   |   | 0x71 |
|---------|------------------------|---|---|---|---|---|---|------|
| Bit     | 7                      | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | CFG_Transmit_Data[7:0] |   |   |   |   |   |   |      |
| Default | 0                      | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | W                      | W | W | W | W | W | W | W    |

**CFG\_Transmit\_Data[7:0]** CFG Transmit Data. This register holds the byte to be written to the Flash logic. Bit 0 is the LSB.

**Table 5.45. Flash Memory Status**

| CFGSR   |        |            |      |      |      |      |         | 0x72   |
|---------|--------|------------|------|------|------|------|---------|--------|
| Bit     | 7      | 6          | 5    | 4    | 3    | 2    | 1       | 0      |
| Name    | WBCACT | (Reserved) | TXFE | TXFF | RXFE | RXFF | SSPIACT | I2CACT |
| Default | 0      | 0          | 0    | 0    | 0    | 0    | 0       | 0      |
| Access  | R      | —          | R    | R    | R    | R    | R       | R      |

**WBCACT** WISHBONE Bus to Configuration Logic Active. Indicates that the WISHBONE to configuration interface is active and the connection is established.

1: WISHBONE Active

0: WISHBONE not Active

**TXFE** Transmit FIFO Empty. Indicates that the Transmit Data register is empty. This bit can generate an interrupt.

1: FIFO empty

0: FIFO not empty

**TXFF** Transmit FIFO Full. Indicates that the Transmit Data register is full. This bit can generate an interrupt.

1: FIFO full

0: FIFO not full

**RXFE** Receive FIFO Empty. Indicates that the Receive Data register is empty. This bit can generate an interrupt.

1: FIFO empty

0: FIFO not empty

**RXFF** Receive FIFO Full. Indicates that the Transmit Data register is full. This bit can generate an interrupt.

1: FIFO full

0: FIFO not full

**SSPIACT** Target SPI Active. Indicates the Target SPI port has started actively communicating with the Configuration Logic while WBCE is enabled. This port has priority over the I2C and WISHBONE ports. It pre-empts any existing, and prohibits any new, lower priority transaction. This bit can generate an interrupt. SPI Access is currently not supported in this IP.

- 1: Target SPI port active
- 0: Target SPI port not active

**I2CACT** I2C Active. Indicates the I2C port has started actively communicating with the Configuration Logic while WBCE is enabled. This port has priority over the WISHBONE ports. It pre-empts any existing, and prohibits any new WISHBONE transaction. This bit can generate an interrupt.

- 1: I2C port active
- 0: I2C port not active

**Table 5.46. Flash Memory Receive Data**

| CFGRXDR |                       |   |   |   |   |   |   | 0x73 |
|---------|-----------------------|---|---|---|---|---|---|------|
| Bit     | 7                     | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | CFG_Receive_Data[7:0] |   |   |   |   |   |   |      |
| Default | 0                     | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | R                     | R | R | R | R | R | R | R    |

**CFG\_Receive\_Data[7:0]** CFG Receive Data. This register holds the byte read from the Flash logic. Bit 0 in this register is the LSB.

**Table 5.47. Flash Memory Interrupt Status**

| CFGIRQ  |            |   |         |         |         |         |            | 0x74      |
|---------|------------|---|---------|---------|---------|---------|------------|-----------|
| Bit     | 7          | 6 | 5       | 4       | 3       | 2       | 1          | 0         |
| Name    | (Reserved) |   | IRQTXFE | IRQTXFF | IRQRXFE | IRQRXFF | IRQSSPIACT | IRQI2CACT |
| Default | 0          | 0 | 0       | 0       | 0       | 0       | 0          | 0         |
| Access  | —          | — | R/W     | R/W     | R/W     | R/W     | R/W        | R/W       |

**IRQTXFE** Interrupt Status for Transmit FIFO Empty. When enabled, indicates TXFE is asserted. Write a 1 to this bit to clear the interrupt.

- 1: Transmit FIFO Empty Interrupt
- 0: No interrupt

**IRQTXFF** Interrupt Status for Transmit FIFO Full. When enabled, indicates TXFF is asserted. Write a 1 to this bit to clear the interrupt.

- 1: Transmit FIFO Full Interrupt
- 0: No interrupt

**IRQRXFE** Interrupt Status for Receive FIFO Empty. When enabled, indicates RXFE is asserted. Write a 1 to this bit to clear the interrupt.

- 1: Receive FIFO Empty Interrupt
- 0: No interrupt

**IRQRXFF** Interrupt Status for Receive FIFO Full. When enabled, indicates RXFF is asserted. Write a 1 to this bit to clear the interrupt.

- 1: Receive FIFO Full Interrupt
- 0: No interrupt

**IRQSSPIACT** Interrupt Status for Target SPI Active. When enabled, indicates SSPIACT is asserted. Write a 1 to this bit to clear the interrupt. SPI Access is currently not supported in this IP.

1: Target SPI Active Interrupt

0: No interrupt

**IRQI2CACT** Interrupt Status for I2C Active. When enabled, indicates I2CACT is asserted. Write a 1 to this bit to clear the interrupt.

1: I2C Active Interrupt

0: No interrupt

**Table 5.48. Flash Memory Interrupt Enable**

| CFGIRQEN |            |   |           |           |           |           |              | 0x75        |
|----------|------------|---|-----------|-----------|-----------|-----------|--------------|-------------|
| Bit      | 7          | 6 | 5         | 4         | 3         | 2         | 1            | 0           |
| Name     | (Reserved) |   | IRQTXFEEN | IRQTXFFEN | IRQRXFEEN | IRQRXFFEN | IRQSSPIACTEN | IRQI2CACTEN |
| Default  | 0          | 0 | 0         | 0         | 0         | 0         | 0            | 0           |
| Access   | —          | — | R/W       | R/W       | R/W       | R/W       | R/W          | R/W         |

**IRQTXFEEN** Interrupt Enable for Transmit FIFO Empty.

1: Interrupt generation enabled

0: Interrupt generation disabled

**IRQTXFFEN** Interrupt Enable for Transmit FIFO Full.

1: Interrupt generation enabled

0: Interrupt generation disabled

**IRQRXFEEN** Interrupt Enable for Receive FIFO Empty.

1: Interrupt generation enabled

0: Interrupt generation disabled

**IRQRXFFEN** Interrupt Enable for Receive FIFO Full.

1: Interrupt generation enabled

0: Interrupt generation disabled

**IRQSSPIACTEN** Interrupt Enable for Target SPI Active.

1: Interrupt generation enabled

0: Interrupt generation disabled

**IRQI2CACTEN** Interrupt Enable for I2C Active.

1: Interrupt generation enabled

0: Interrupt generation disabled

**Table 5.49. Unused (Reserved) Register**

| UNUSED  |            |   |   |   |   |   |   | 0x76 |
|---------|------------|---|---|---|---|---|---|------|
| Bit     | 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
| Name    | (Reserved) |   |   |   |   |   |   |      |
| Default | 0          | 0 | 0 | 0 | 0 | 0 | 0 | 0    |
| Access  | —          | — | — | — | — | — | — | —    |

**Table 5.50. EFB Interrupt Source**

| EFBIRQ  |            |   |   |         |        |         |          | 0x77     |
|---------|------------|---|---|---------|--------|---------|----------|----------|
| Bit     | 7          | 6 | 5 | 4       | 3      | 2       | 1        | 0        |
| Name    | (Reserved) |   |   | CFG_INT | TC_INT | SPI_INT | I2C2_INT | I2C1_INT |
| Default | 0          | 0 | 0 | 0       | 0      | 0       | 0        | 0        |
| Access  | R          | R | R | R       | R      | R       | R        | R        |

**CFG\_INT** Flash Interrupt Source. Indicates EFB interrupt source is from the Flash Block. Use CFGIRQ for further source resolution.

- 1: A bit is set in register CFGIRQ
- 0: No interrupt

**TC\_INT** Timer/Counter Interrupt Source. Indicates EFB interrupt source is from the Timer/Counter Block. Use TCIRQ for further source resolution.

- 1: A bit is set in register TCIRQ
- 0: No interrupt

**SPI\_INT** SPI Interrupt Source. Indicates EFB interrupt source is from the SPI Block. Use SPI-IRQ for further source resolution. SPI Access is currently not supported in this IP.

- 1: A bit is set in register SPIIRQ
- 0: No interrupt

**I2C2\_INT** I2C2 Interrupt Source. Indicates EFB interrupt source is from the Secondary I2C Block. Use I2C\_2\_IRQ for further source resolution.

- 1: A bit is set in register I2C\_2\_IRQ
- 0: No interrupt

**I2C1\_INT** I2C1 Interrupt Source. Indicates EFB interrupt source is from the Primary I2C Block. Use I2C\_1\_IRQ for further source resolution.

- 1: A bit is set in register I2C\_1\_IRQ
- 0: No interrupt

## 6. Command and Data Transfers to Flash Space

The command and data transferred to the flash memory are identical for all the access ports, regardless of their different physical interfaces. The flash memory is organized in pages. Therefore, you address a specific page for read or write operations to that page. Each page has 128 bits (16 bytes). Transfers are based on a set of instructions and page addresses. Erase operations are sector based (Configuration Flash, UFM, and/or Feature Row).

### 6.1. Command Summary by Application

**Table 6.1. UFM (Sector 1) Commands**

| Command Name                                      | Command MSB LSB | SVF Command Name  | Description   |
|---|-----------------|-------------------|---|
| Read Status Register                              | 0x3C            | LSC_READ_STATUS   | Read the 4-byte Configuration Status register.  |
| Check Busy Flag                                   | 0xF0            | LSC_CHECK_BUSY    | Read the Configuration Busy Flag status.  |
| Bypass  | 0xFF            | ISC_NOOP          | Null operation.   |
| Enable Configuration Interface (Transparent Mode) | 0x74            | ISC_ENABLE_X      | Enable Transparent UFM access – All user I/Os (except the primary user ports) are governed by the user logic. The device remains in user mode. The subsequent commands in this table require the interface to be enabled.   |
| Enable Configuration Interface (Offline Mode)     | 0xC6            | ISC_ENABLE        | Enable offline access – All user I/O (except persisted sysCONFIG ports) are tri-stated. User logic ceases to function, UFM remains accessible, and the device enters <i>Offline</i> access mode. The subsequent commands in this table require the interface to be enabled. |
| Disable Configuration Interface                   | 0x26            | ISC_DISABLE       | Disable the configuration UFM access.   |
| Set Address                                       | 0xB4            | LSC_WRITE_ADDRESS | Set the UFM sector 14-bit Address register.   |
| Reset UFM Address                                 | 0x47            | LSC_INIT_ADDR_UFM | Reset the address to point to Sector 1, Page 0 of the UFM.  |
| Read UFM  | 0xCA            | LSC_READ_TAG      | Read the UFM data. Operand specifies the number of pages to read. Address Register is post-incremented.   |
| Erase UFM   | 0xCB            | LSC_ERASE_TAG     | Erase the UFM sector only.  |
| Program UFM Page                                  | 0xC9            | LSC_PROG_TAG      | Write one page of data to the UFM. Address register is post-incremented.  |

**Table 6.2. Configuration Flash (Sector 0) Commands**

| Command Name                                      | Command MSB LSB | SVF Command Name | Description  |
|---|-----------------|------------------|--|
| Read Device ID                                    | 0xE0            | IDCODE_PUB       | Read the 4-byte Device ID. See <a href="#">Table 6.28</a> for the expected Device IDs.   |
| Read USERCODE                                     | 0xC0            | USERCODE         | Read the 32-bit USERCODE.  |
| Read Status Register                              | 0x3C            | LSC_READ_STATUS  | Read the 4-byte Configuration Status register.   |
| Read Busy Flag                                    | 0xF0            | LSC_CHECK_BUSY   | Read the Configuration Busy Flag status.   |
| Refresh1  | 0x79            | LSC_REFRESH      | Launch boot sequence.  |
| STANDBY   | 0x7D            | LSC_DEVICE_CTRL  | Trigger the Power Controller to enter or wake from standby mode.   |
| Bypass  | 0xFF            | ISC_NOOP         | Null operation.  |
| Enable Configuration Interface (Transparent Mode) | 0x74            | ISC_ENABLE_X     | Enable Transparent Configuration Flash access – All user I/O (except hardened primary user I2C ports) are governed by the user logic; the device remains in user mode. |

| Command Name                                  | Command MSB LSB | SVF Command Name     | Description  |
|---|-----------------|----------------------|--|
|   |                 |                      | The subsequent commands in this table require the interface to be enabled.   |
| Enable Configuration Interface (Offline Mode) | 0xC6            | ISC_ENABLE           | Enable Offline Configuration Flash access – All user I/O (except persisted sysCONFIG ports) are tri-stated. User logic ceases to function, and the device enters <i>Offline</i> access mode.<br>The subsequent commands in this table require the interface to be enabled. |
| Disable Configuration Interface               | 0x26            | ISC_DISABLE          | Exit access mode.  |
| Set Configuration Flash Address               | 0xB4            | LSC_WRITE_ADDRESS    | Set the Configuration Flash 14-bit Page Address.   |
| Verify Device ID                              | 0xE2            | VERIFY_ID            | Verify device ID with 32-bit input, set Fail flag if mismatched.   |
| Reset Configuration Flash Address             | 0x46            | LSC_INIT_ADDRESS     | Reset the address to point to Sector 0, Page 0 of the Configuration Flash.   |
| Read Flash                                    | 0x73            | LSC_READ_INCR_NV     | Read the flash data. Operand specifies number of the pages to read. Address Register is post-incremented.  |
| Erase   | 0x0E            | ISC_ERASE            | Erase the Config Flash, FEATURE Row, FEABITS, Done bit, Security bits, and USERCODE.   |
| Program Page                                  | 0x70            | LSC_PROG_INCR_NV     | Write one page of data to the Flash. Address Register is post-incremented.   |
| Program DONE                                  | 0x5E            | ISC_PROGRAM_DONE     | Program the Done bit.  |
| Program SECURITY                              | 0xCE            | ISC_PROGRAM_SECURITY | Program the Security bit (Secures Configuration Flash sector).   |
| Program SECURITY PLUS                         | 0xCF            | ISC_PROGRAM_SECPLUS  | Program the Security Plus bit (Secures Configuration Flash and UFM Sectors).<br><b>Note:</b> SECURITY and SECURITY PLUS commands are mutually exclusive.   |
| Program USERCODE                              | 0xC2            | ISC_PROGRAM_USERCODE | Program 32-bit USERCODE.   |
| Read Feature Row                              | 0xE7            | LSC_READ_FEATURE     | Read Feature Row.  |
| Program Feature Row                           | 0xE4            | LSC_PROG_FEATURE     | Program Feature Row.   |
| Read FEABITS                                  | 0xFB            | LSC_READ_FEABITS     | Read FEA bits.   |
| Program FEABITS                               | 0xF8            | LSC_PROG_FEABITS     | Program the FEA bits.  |

**Table 6.3. Non-Volatile Register (NVR) Commands**

| Command Name       | Command MSB LSB | SVF Command Name | Description          |
|--------------------|-----------------|------------------|----------------------|
| Read Trace ID code | 0x19            | UIDCODE_PUB      | Read 64-bit TraceID. |

When using the WISHBONE bus interface, the commands, operand, and data are written to the CFGTXDR register. The target SPI or I2C interface shifts the most significant bit (MSB) first into the MachXO4 device. This is required only when communicating with the configuration logic inside the MachXO4 device.

## 6.2. Command Descriptions by Command Code

**Table 6.4. Erase Flash (0x0E)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| x   | x         | —   | Y           | 0E        | See below      | —         | —         | —           |

Operand 0000 udfs 0000 0000 0000 0000(binary)

where:

- u: Erase UFM sector
  - 0: No action
  - 1: Erase
- c: Erase CFG Flash sector (Configuration Flash, DONE, security bits, USERCODE)
  - 0: No action
  - 1: Erase
- f: Erase Feature sector (Target I2C address, sysCONFIG port persistence, Boot mode, and others)
  - 0: No action
  - 1: Erase
- s: Erase SRAM
  - 0: No action
  - 1: Erase

**Notes:** Poll the BUSY bit after issuing this command for erasure to complete before issuing a subsequent command other than Read Status or Check Busy.

Erased condition for Flash bits = 0.

Examples:

- 0x0E 04 00 00  
Erase Flash sector.
- 0x0E 08 00 00  
Erase UFM sector.
- 0x0E 0C 00 00  
Erase UFM and Flash sectors.

**Table 6.5. Read TraceID Code (0x19)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | —         | x   | N           | 19        | 00 00 00       | R         | 8B        | —           |

Example: 0x19 00 00 00  
Read 8-byte TraceID.

**Note:** First byte read is user portion. Next seven bytes are unique to each silicon die.

**Table 6.6. Disable Configuration Interface (0x26)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| x   | x         | —   | —           | 26        | 00 00          | —         | —         | —           |

Example: 0x26 00 00  
Disable Flash interface for change access.

**Note:** Must have only two operands.

The interface cannot be disabled while the Configuration Status Register Busy bit is asserted. After commands (for example, Erase, Program) verify Busy is clear before issuing the Disable command.

This command should be followed by Command 0xFF (BYPASS) to complete the Disable operation. The BYPASS command is required to restore Power Controller, GSR, Hardened User I2C port operation.

SRAM must be erased before exiting Offline (0xC6) Mode.

**Table 6.7. Read Status Register (0x3C)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary)                    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|---|
| x   | x         | —   | N           | 3C        | 00 00 00       | R         | 4B        | xxxx IxEE Exxx xxxx xxFB xxCD xxxx xxxx |

Data Format: The most significant byte of SR is received first, the LSB last.

- D Bit 8 Flash or SRAM Done Flag
  - When C = 0 SRAM Done bit has been programmed
    - D = 1 Successful Flash to SRAM transfer
    - D = 0 Failure in the Flash to SRAM transfer
  - When C=1 CFG Flash Done bit has been programmed
    - D = 1 Programmed
    - D = 0 Not Programmed
- C Bit 9 Enable Configuration Interface (1=Enable, 0=Disable)
- B Bit 12: Busy Flag (1 = busy)
- F Bit 13: Fail Flag (1 = operation failed)
- I I = 0 Device verified correct; I = 1 Device failed to verify

EEE bits[25:23]: Configuration Check Status

- 000: No Error
- 001: ID ERR
- 010: CMD ERR
- 011: CRC ERR
- 100: Preamble ERR
- 101: Abort ERR
- 110: Overflow ERR
- 111: SDM EOF

(all other bits reserved)

Usage: The BUSY bit should be checked following all Enable, Erase or Program operations.

**Note:** Wait at least 1us after power-up or asserting wb\_rst\_i before accessing the EFB.

Example: 0x3C 00 00 00  
Read 4-byte Status Register for example, 0x00 00 20 00 (fail flag set).

**Table 6.8. Reset Flash Address (0x46)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | x         | —   | Y           | 46        | 00 00 00       | —         | —         | —           |



### Table 6.9. Reset UFM Address (0x47)

Example: 0x47 00 00 00  
Set Address register to Flash Sector 1, page 0.

Example:            0x5E 00 00 00  
                      Set the DONE bit.

### Table 6.11. Program Configuration Flash (0x70)

Example: 0x70 00 00 00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

Write one page of data, pointed to by the Address Register.

**Note:** 16 data bytes must be written following the command and operand bytes to ensure proper data alignment. The Address Register is auto-incremented following the page write.  
Use 0x0E to erase CFG Flash sector prior to executing this command.  
Poll the BUSY bit (or wait 200  $\mu$ s) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

**Note:** This applies when Configuration Flash is read through I2C.

- |               |   |
|---------------|---|
| 1. Operand:   | 0000 0000 00pp pppp pppp pppp (binary)                                      |
|               | pp..pp: num_pages      Number of CFG Flash pages to read when num_pages = 1 |
|               | Number of CFG Flash pages to read + 1 when num_pages > 1                    |
| 2. Data Size: | (num_pages × 16) bytes    when num_pages = 1                                |
|               | 32 bytes + (num_pages) × (16 + 4) bytes    when num_pages > 1               |

**Note:** Read CFG Flash may be aborted at any time. Any data remaining in the read FIFO will be discarded. Any read data beyond the prescribed read size will be indeterminate. The Address Register is auto-incremented after each page read.

- ### Table 6.13. Read Configuration Flash (0x73) (WISHBONE)

### Table 6.14. Enable Configuration Interface (Transparent) (0x74)

Examples: 0x74 08 00 00  
Enable Flash interface for change access through a non-I2C interface.

**Table 6.15. Refresh (0x79)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | —         | —   | —           | 79        | 00 00          | —         | —         | —           |

Example: 0x79 00 00

Issue Refresh command

**Note:** The Refresh command launches Boot sequence

Must have only two operands

After completing the Refresh command (for example, I2C stop), further bus accesses are prohibited for the duration of tREFRESH. Violating this requirement causes the Refresh process to abort and leaves the MachXO4 device in an unprogrammed state.

Occasionally, following a device REFRESH or PROGRAMN pin toggle, the secondary I2C port may be left in an undefined (non-idle) state. The likelihood of this condition is design and route dependent. To positively return the Secondary I2C port to the idle state, write a value of 0x44 to register I2C\_2\_CMDR through WISHBONE immediately after device reset is released. This causes a short low-pulse on SCK as the hard-block signals a STOP on the bus then returns to the idle state. Failure to manually return the Secondary I2C port to the idle state may result in an I2C bus lock-up condition. Normal I2C activity can be commenced without additional delay.

**Table 6.16. STANDBY (0x7D)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | x         | —   | N           | 7D        | 0y 00          | —         | —         | —           |

**Notes:** Power Controller is currently not supported in this IP

**Table 6.17. Set Address (0xB4)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary)                    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|---|
| x   | x         | —   | Y           | B4        | 00 00 00       | W         | 4B        | 0s00 0000 0000 0000 00aa aaaa aaaa aaaa |

Data Format: s: sector

0: CFG Flash

1: UFM

aa..aa:address14-bit page address

Example: 0xB4 00 00 00 40 00 00 0A

Set Address register to UFM sector, page 10 decimal

**Table 6.18. Read USERCODE (0xC0)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | x         | —   | Y/N         | C0        | 00 00 00       | R         | 4B        | —           |

Example: 0xC0 00 00 00

EN Required = Y Read 4-byte USERCODE from CFG sector

EN Required = N Read 4-byte USERCODE from SRAM

**Table 6.19. Program USERCODE (0xC2)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | x         | —   | Y           | C2        | 00 00 00       | W         | 4B        | —           |

Example: 0xC2 00 00 00 10 20 30 40

Sets USERCODE with 32-bit input 0x10 20 30 40.

**Note:** Poll the BUSY bit (or wait 200  $\mu$ s) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

**Table 6.20. Enable Configuration Interface (Offline) (0xC6)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | x         | —   | —           | C6        | 0y 00 00       | —         | —         | —           |

Operand: 08 00 00 – Enable Flash Normal mode. Normal edit mode for Offline configuration. Used for all offline commands described in this document, including Erase SRAM.

00 00 00 – Enable SRAM mode. Optional edit mode. Supports Erase SRAM command only.

Example: 0xC6 08 00 00 Enable Flash interface for offline change access.

**Notes:** Use this command to enable offline modification of the Flash, or non-volatile registers (NVR). SRAM must be erased exiting Offline mode. When exiting Offline mode follow the command 0x26 with the command 0xFF. Exercising this command will tristate all user I/O (except persisted sysCONFIG ports). User logic ceases to function. UFM remains accessible.

Poll the BUSY bit (or wait 5  $\mu$ s) after issuing this command for the Flash pumps to fully charge.

**Table 6.21. Program UFM (0xC9)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary)    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------------------|
| x   | —         | —   | Y           | C9        | 00 00 01       | W         | 16B       | 16 bytes UFM write data |

Example: 0xC9 00 00 01 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F Write one page of data, pointed to by Address Register.

**Notes:** 16 data bytes must be written following the command and operand bytes to ensure proper data alignment. The Address Register is auto-incremented following the page write.

Use 0x0E or 0xCB to erase UFM sector prior to executing this command.

Poll the BUSY bit (or wait 200  $\mu$ s) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.

**Table 6.22. Read UFM (0xCA) (I2C)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex)       | Data Mode | Data Size            | Data Format          |
|-----|-----------|-----|-------------|-----------|----------------------|-----------|----------------------|----------------------|
| x   | —         | —   | Y           | CA        | (below) <sup>1</sup> | R         | (below) <sup>2</sup> | (below) <sup>3</sup> |

1. Operand: 0000 0000 00pp pppp pppp pppp (binary)

where: pp..pp: num\_pages Number of UFM pages to read when num\_pages = 1  
Number of UFM pages to read +1 when num\_pages > 1

2. Data Size: (num\_pages  $\times$  16) bytes when num\_pages = 1  
32 bytes + (num\_pages  $\times$  16 + 4) bytes when num\_pages > 1

**Note:** Read UFM may be aborted at any time. Any data remaining in the read FIFO is discarded. Any read data beyond the prescribed read size is indeterminate. The Address Register is auto-incremented after each page read.

3. Examples: 0xCA 00 00 01

Read 0-byte dummy followed by one page UFM data (16 bytes total).

0xCA 00 00 04

Read two-page dummy followed by three sets – one page UFM data, followed by four bytes dummy (five pages total and 12 dummy bytes).

### Table 6.23. Read UFM (0xCA) (WISHBONE)

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex)       | Data Mode | Data Size            | Data Format          |
|-----|-----------|-----|-------------|-----------|----------------------|-----------|----------------------|----------------------|
| x   | —         | —   | Y           | CA        | (below) <sup>1</sup> | R         | (below) <sup>2</sup> | (below) <sup>3</sup> |

1. Operand: 0001 0000 00pp pppp pppp pppp (binary), or

0000 0000 00pp pppp pppp pppp (binary)

where:                    pp..pp:   num\_pages    Number of UFM pages to read when num\_pages = 1

Number of UFM pages to read +1 when  $1 < \text{num\_pages} \leq 12$

Set to 0x3FFF when num pages > 12

2. Data Size: (num\_pages × 16) bytes when num\_pages = 1

32 bytes + (num\_pages × 16 + 4) bytes      when num\_pages > 1

**Note:** When reading more than 12 pages, the `num_pages` argument is intentionally oversized. It is not necessary to read the extra pages. Read UFM may be aborted at any time. Any data remaining in the read fifo is discarded. Any read data beyond the prescribed read size is indeterminate. The Address Register is auto-incremented after each page read.

3. Examples:            0xCA 00 00 01

Read 0 bytes dummy followed by one page UFM data (16 bytes total).

0xCA 10 00 04

Read 1 page dummy followed by three pages of CFG data (four pages total).

0xCA 00 00 04

Read two-page dummy followed by three sets – one page UFM data, followed by four bytes dummy (five pages total and 12 dummy bytes).

**Note:** The maximum WISHBONE clock speed with which one page of data (num\_page = 1) can be read using WISHBONE and no wait states is 16.6 MHz. Faster WISHBONE clock speeds are supported by inserting WB wait states to observe the retrieval delay timing requirement.

### Table 6.24. Erase UFM (0xCB)

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| x   | —         | —   | y           | CB        | 00 00 00       | —         | —         | —           |

**Notes:** Erased condition for UFM bits = 0.

Poll the BUSY bit after issuing this command for erasure to complete before issuing a subsequent command other than Read Status or Check Busy.

Example:            0xCB 00 00 00

Erase UFM sector.

**Table 6.25. Program SECURITY (0xCE)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | x         | —   | Y           | CE        | 00 00 00       | —         | —         | —           |

Example: 0xCE 00 00 00  
Set the SECURITY bit.

**Note:** Poll the BUSY bit (or wait 200  $\mu$ s) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.  
SECURITY and SECURITY PLUS commands are mutually exclusive.

**Table 6.26. Program SECURITY PLUS (0xCF)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| —   | x         | —   | Y           | CF        | 00 00 00       | —         | —         | —           |

Example: 0xCF 00 00 00  
Set the SECURITY PLUS bit.

**Note:** Poll the BUSY bit (or wait 200  $\mu$ s) after issuing this command for programming to complete before issuing a subsequent command other than Read Status or Check Busy.  
SECURITY and SECURITY PLUS commands are mutually exclusive.

**Table 6.27. Read Device ID Code (0xE0)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format                    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|--------------------------------|
| —   | x         | —   | N           | E0        | 00 00 00       | R         | 4B        | See <a href="#">Table 6.28</a> |

Example: 0xE0 00 00 00  
Read 4-byte device ID.

**Table 6.28. Device ID**

| Device Name           | E Devices     | C Devices     |
|-----------------------|---------------|---------------|
| LFMXO4-010            | 0x61 2B 20 43 | 0xE1 2B B0 43 |
| LFMXO4-015            | 0x61 2B 20 43 | 0xE1 2B B0 43 |
| LFMXO4-025            | 0x61 2B 30 43 | 0x61 2B B0 43 |
| LFMXO4-050            | 0x61 2B 40 43 | 0x61 2B C0 43 |
| LFMXO4-050 (256-Ball) | 0x61 2B 50 43 | 0x61 2B D0 43 |
| LFMXO4-080            | 0x61 2B 50 43 | 0x61 2B D0 43 |
| LFMXO4-110            | 0x61 2B 60 43 | 0x61 2B E0 43 |

Example: 0xE0 00 00 00  
Read 4-byte device ID.

**Table 6.29. Verify Device ID Code (0xE2)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format                    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|--------------------------------|
| —   | x         | —   | Y           | E2        | 00 00 00       | W         | 4B        | See <a href="#">Table 6.28</a> |

Example: 0xE2 00 00 00 01 2B 20 43  
Verify device ID with 32-bit input, sets ID Error bit 27 in SR if mismatched.

**Table 6.30. Program Feature (0xE4)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary)    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------------------|
| —   | —         | —   | Y           | E4        | 00 00 00       | —         | 8B        | 00 00 ss uu cc cc cc cc |

Data Format: ss: 8 bits for the user programmable I2C Target Address uu: 8 bits for the user programmable TraceID  
cc cc cc cc: 32 bits of Custom ID code

**Note:** It is not recommended to reprogram the Feature Row in system as it should be programmed ideally during manufacturing.

Example: 0xE4 00 00 00 00 00 01 00 00 00 12 34  
Program Feature Row with User I2C address set to 1, default user TraceID string, Custom ID code of 12 34.

**Table 6.31. Read Feature Row (0xE7)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary)    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------------------|
| —   | —         | x   | Y           | E7        | 00 00 00       | R         | 8B        | 00 00 ss uu cc cc cc cc |

Data Format: ss: 8 bits for the user programmable I2C Target Address  
uu: 8 bits for the user programmable TraceID  
cc cc cc cc: 32 bits of Custom ID code

Example: 0xE7 00 00 00  
Reads the Feature Row.

**Table 6.32. Check Busy Flag (0xF0)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary) |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|----------------------|
| x   | x         | —   | N           | F0        | 00 00 00       | R         | 1B        | Bxxx xxxx            |

Data Format: b: bit 7: Busy Flag (1 = busy)  
(all other bits reserved)

Example: 0xF0 00 00 00  
Read one byte, for example, 0x80 (busy flag set)

**Table 6.33. Program FEABITs (0xF8)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary)    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------------------|
| —   | —         | x   | Y           | F8        | 00 00 00       | W         | 2B        | 00 bb mi sj di pa vv 00 |

Data Format: bb: Boot Sequence

1. If b = 00 (Default) and m = 0 then Single Boot from Flash.
2. If b = 00 and m = 1 then Dual Boot from Flash then External, if there is a failure.
3. If b = 01 and m = 1 then Single Boot from External Flash.
4. If b = 10 and m = 1 then Dual Boot from External then NVMC/Flash, if there is a failure.

m: Controller SPI Port Persistence  
0 = Disabled (Default), 1 = Enabled

i: I2C Port Persistence

- 0 = Enabled (Default), 1 = Disabled
- s: Target SPI Port Persistence
  - 0 = Enabled (Default), 1 = Disabled
- j: JTAG Port Persistence
  - 0 = Enabled (Default), 1 = Disabled
- d: DONE Persistence
  - 0 = Disabled (Default), 1 = Enabled
- i: INITN Persistence
  - 0 = Disabled (Default), 1 = Enabled
- p: PROGRAMN Persistence
  - 0 = Enabled (Default), 1 = Disabled
- a: my\_ASSP Enabled
  - 0 = Disabled (Default), 1 = Enabled
- w: Password (Flash Protect Key) Protect All Enabled
  - 0 = Disabled (Default), 1 = Enabled
- v: Password (Flash Protect Key) Enabled
  - 0 = Disabled (Default), 1 = Enabled

**Note:** It is not recommended to reprogram the FEABITs in system as it should be programmed ideally during manufacturing.

**Example:** 0xF8 00 00 00 0D 20  
Programs the FEABITs

**Table 6.34. Read FEABITs (0xFB)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format (Binary)    |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------------------|
| —   | —         | x   | Y           | FB        | 00 00 00       | R         | 2B        | 00 bb mi sj di pa wv 00 |

- Data Format:**
- bb: Boot Sequence
    1. If b = 00 (Default) and m=0 then Single Boot from Flash.
    2. If b = 00 and m = 1 then Dual Boot from Flash then External, if there is a failure.
    3. If b = 01 and m = 1 then Single Boot from External Flash.
    4. If b = 10 and m = 1 then Dual Boot from External then NVMC/Flash, if there is a failure.
  - m: Controller SPI Port Persistence
    - 0 = Disabled (Default), 1 = Enabled
  - i: INITN Persistence
    - 0 = Disabled (Default), 1 = Enabled
  - s: Target SPI Port Persistence
    - 0 = Enabled (Default), 1 = Disabled
  - j: JTAG Port Persistence
    - 0 = Enabled (Default), 1 = Disabled
  - d: DONE Persistence
    - 0 = Disabled (Default), 1 = Enabled
  - i: INITN Persistence
    - 0 = Disabled (Default), 1 = Enabled
  - p: PROGRAMN Persistence



- 0 = Enabled (Default), 1 = Disabled
- a: my\_ASSP Enabled
- 0 = Disabled (Default), 1 = Enabled
- w: Password (Flash Protect Key) Protect All Enabled
- 0 = Disabled (Default), 1 = Enabled
- v: Password (Flash Protect Key) Enabled
- 0 = Disabled (Default), 1 = Enabled

**Table 6.35. Bypass (Null Operation) (0xFF)**

| UFM | CFG Flash | NVR | EN Required | CMD (Hex) | Operands (Hex) | Data Mode | Data Size | Data Format |
|-----|-----------|-----|-------------|-----------|----------------|-----------|-----------|-------------|
| x   | x         | x   | N           | FF        | FF FF FF       | —         | —         | —           |

**Note:** Operands are optional.

Example: 0xFF FF FF FF Bypass

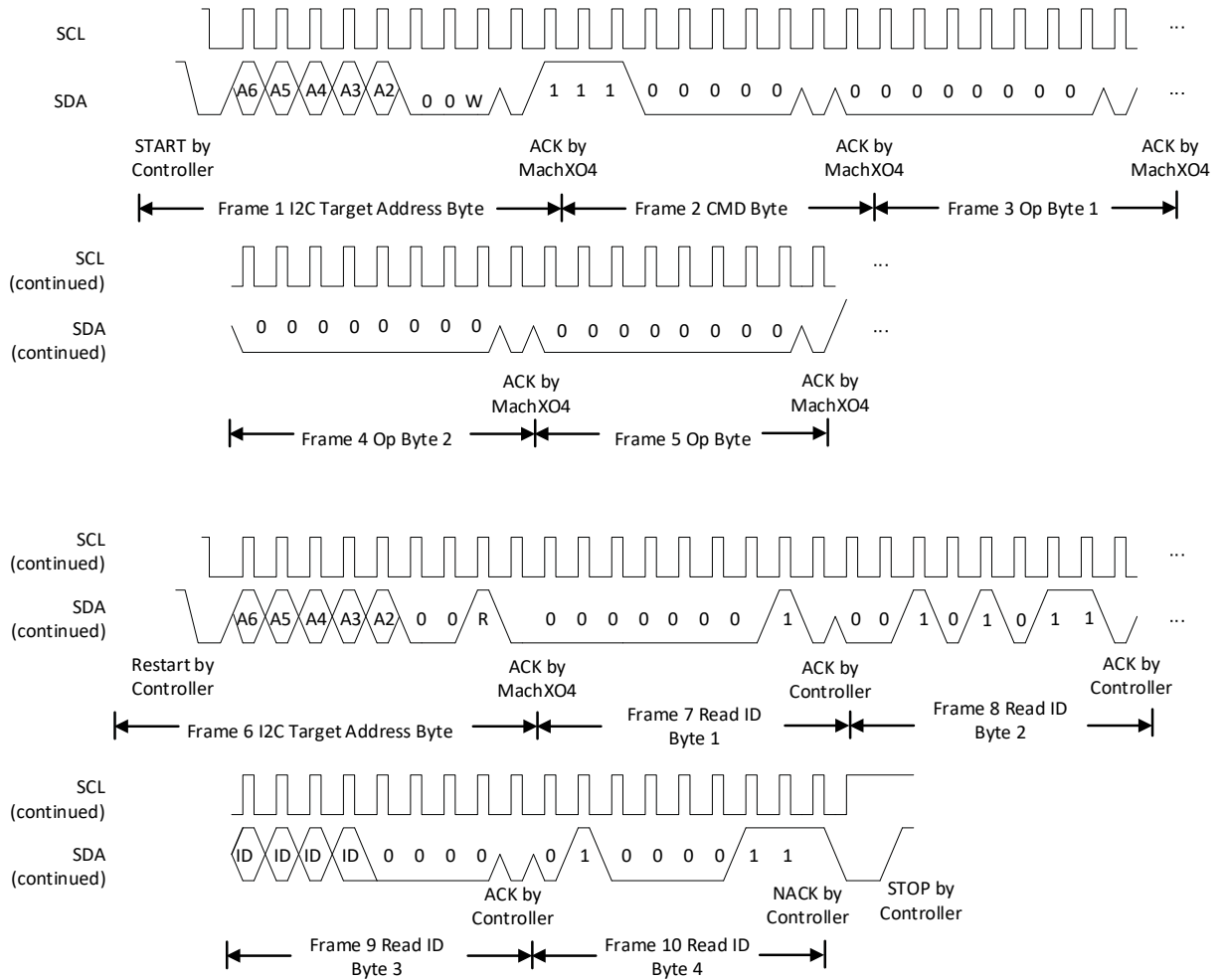
## 6.3. Command Framing

### 6.3.1. I2C Framing

Each command string sent to the I2C EFB port must be correctly framed using the protocol defined for each interface. In the case of I2C, the protocol is well known and defined by the industry as shown below.

**Table 6.36. Command Framing Protocol by Interface**

| Interface | Pre-op (+) | Command String          | Post-op (–) |
|-----------|------------|-------------------------|-------------|
| I2C       | Start      | (Command/Operands/Data) | Stop        |



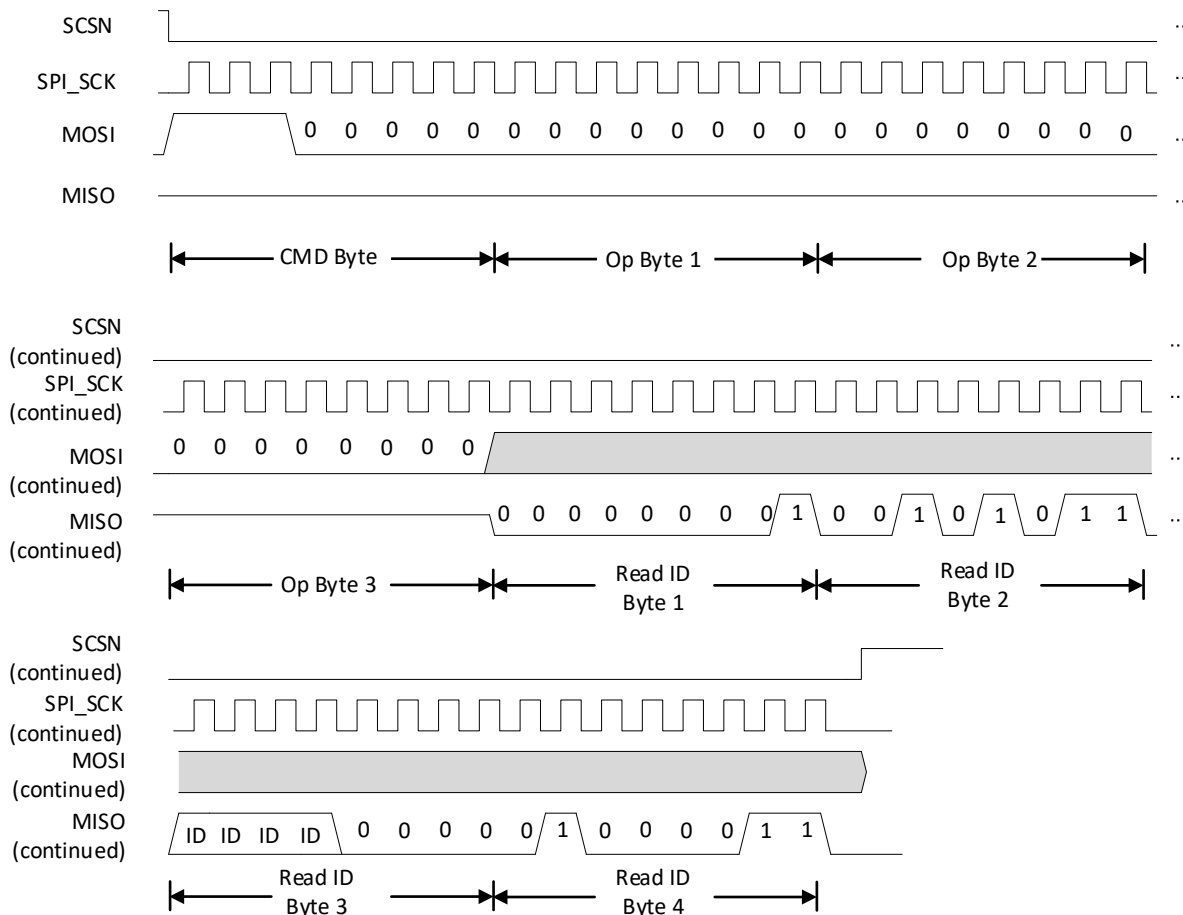
**Figure 6.1. I2C Read Device ID Example**

### 6.3.2. SPI Framing

Each command string sent to the SPI EFB port must be correctly framed using the protocol defined for each interface. For SSPI, the protocol is well known and defined by the industry, as shown below.

**Table 6.37. Command Framing Protocol, by Interface**

| Interface | Pre-op (+) | Command String          | Post-op (-)  |
|-----------|------------|-------------------------|--------------|
| SPI       | Assert CS  | (Command/Operands/Data) | De-assert CS |



**Figure 6.2. SSPI Read Device ID Example**

### 6.3.3. WISHBONE Framing

To access the flash memory, each command string sent to the WISHBONE EFB ports must be correctly framed using the protocol defined for each interface. For the internal WISHBONE port, each command string is preceded by setting CFGCR[WBCE]. Similarly, each command string is followed by clearing the CFGCR[WBCE] bit.

**Table 6.37. Command Framing Protocol by Interface**

| Interface | Pre-op (+)         | Command String          | Post-op (-)           |
|-----------|--------------------|-------------------------|-----------------------|
| WISHBONE  | Assert CFGCR[WBCE] | (Command/Operands/Data) | De-assert CFGCR[WBCE] |

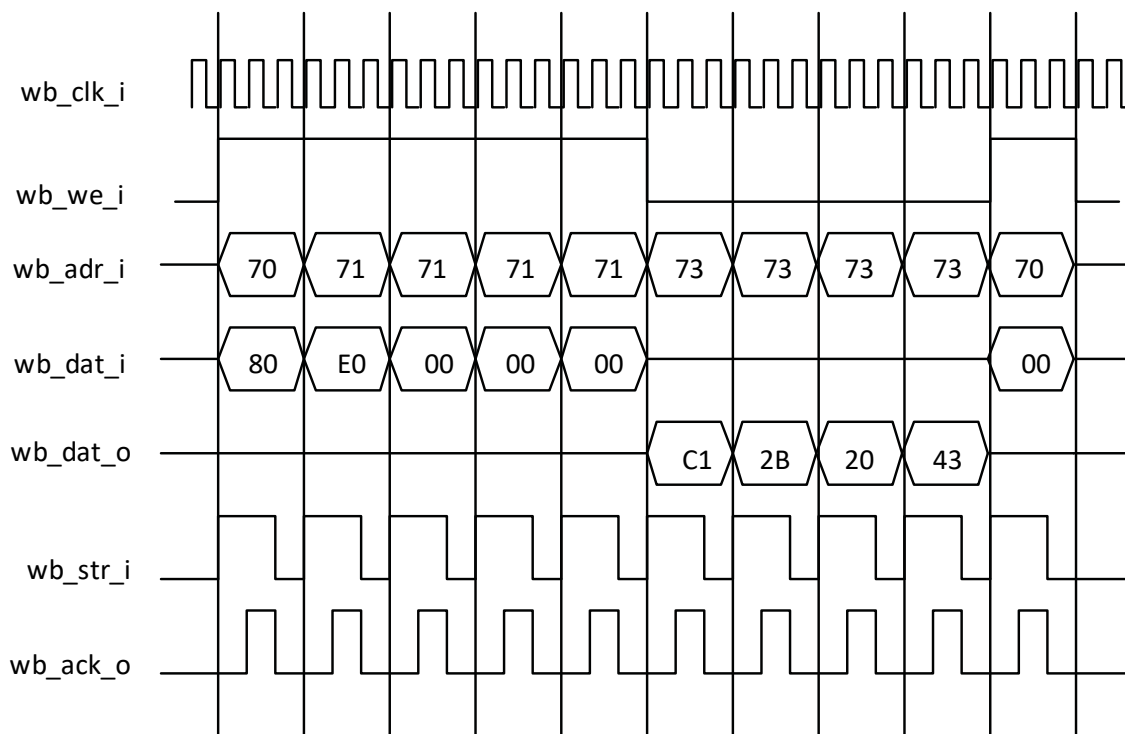


Figure 6.3. WISHBONE Read Device ID Example

## 7. Designing with the IP

This section provides information on how to generate the IP core using the Lattice Radiant software and how to run simulation and synthesis. For more details on the Lattice Radiant software, refer to the Lattice Radiant Software User Guide.

**Note:** The screenshots provided are for reference only. Details may vary depending on the version of the IP or software being used. If there have been no significant changes to the GUI, a screenshot may reflect an earlier version of the IP.

### 7.1. Generating and Instantiating the IP

You can use the Lattice Radiant software to generate IP modules and integrate them into the device architecture. The steps below describe how to generate the EFB Module IP in the Lattice Radiant software.

To generate the EFB Module IP:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **Embedded Functional Block** under **IP, Architecture Modules** category. The **Module/IP Block Wizard** opens as shown in [Figure 7.1](#). Enter values in the **Component name** and the **Create in** fields and click **Next**.

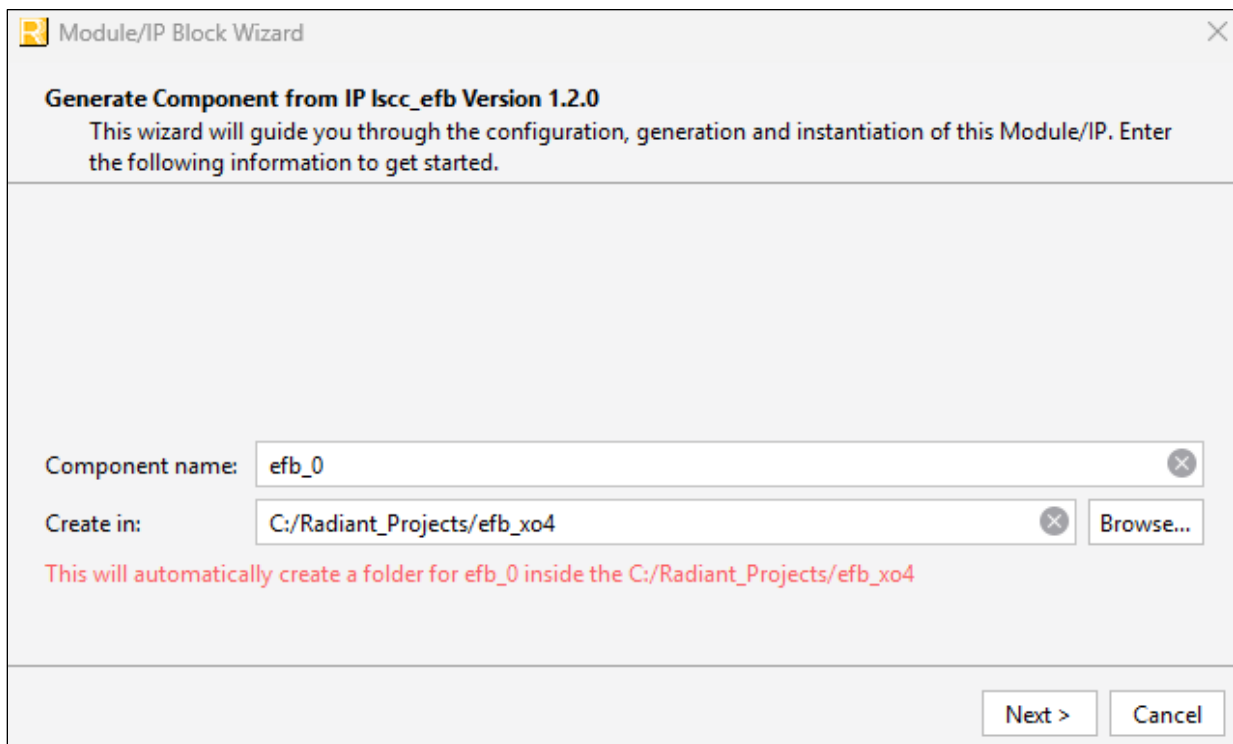
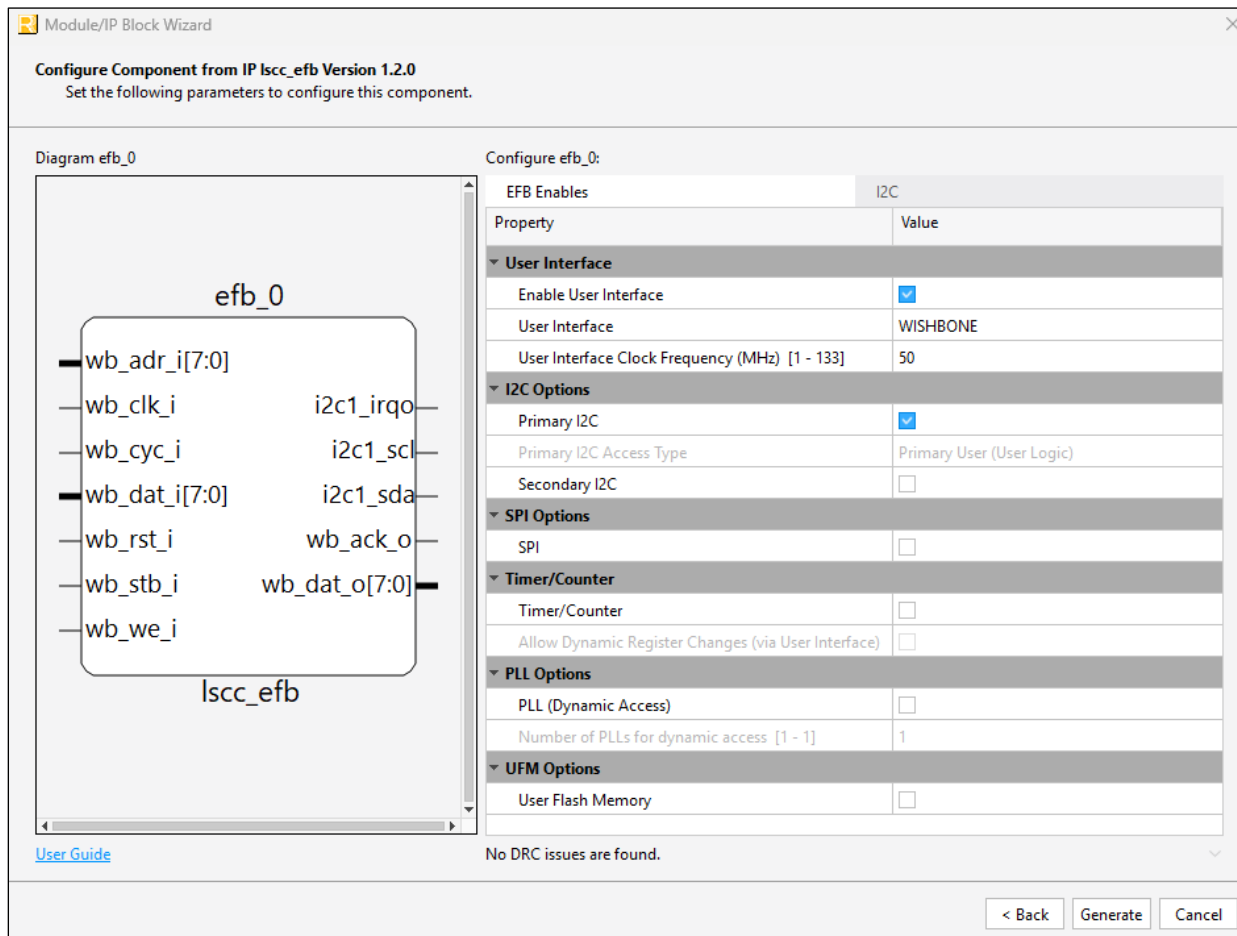


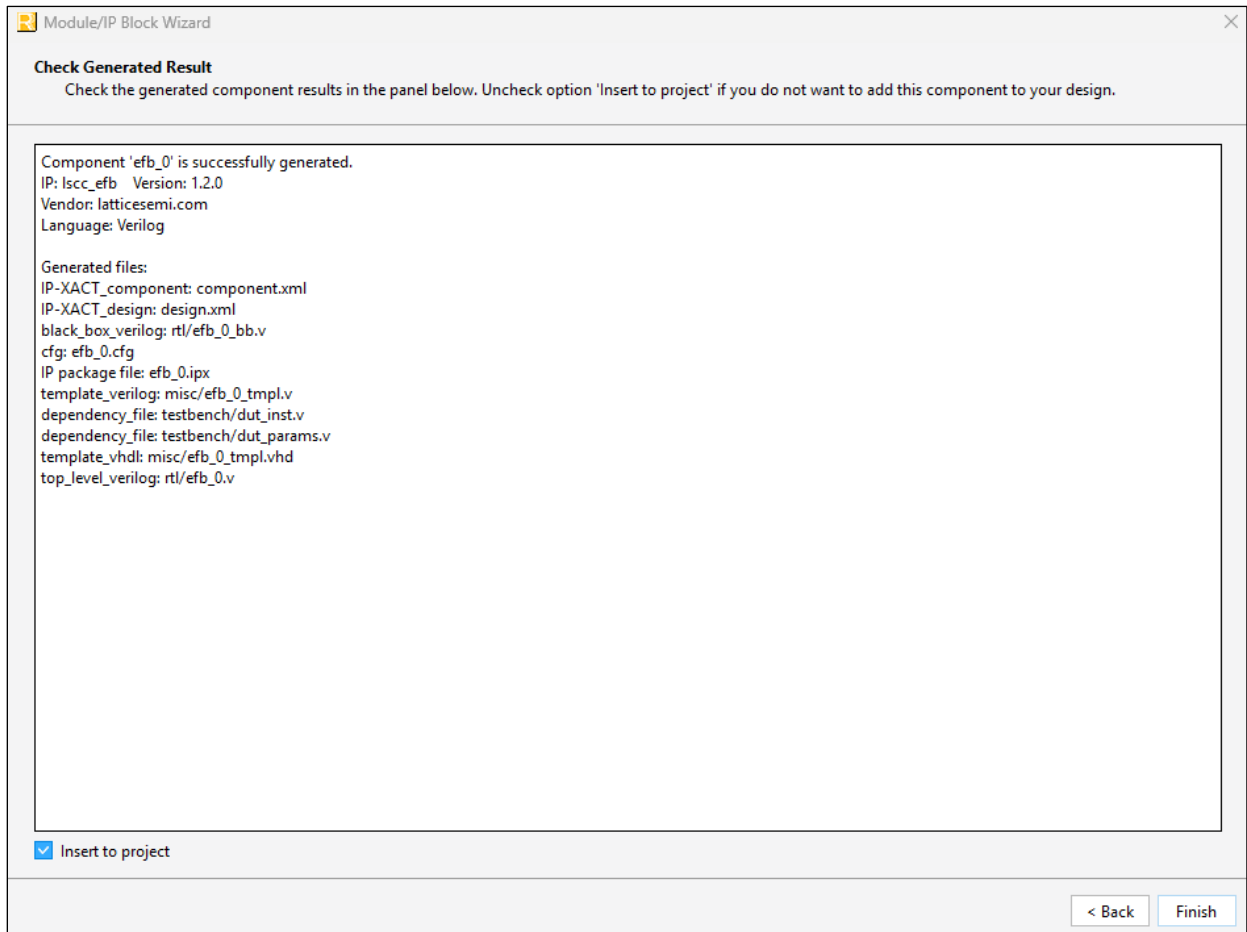
Figure 7.1. Module/IP Block Wizard

3. In the next **Module/IP Block Wizard** window, customize the selected EFB Module IP using drop-down lists and check boxes. [Figure 7.2](#) shows an example configuration of the EFB Module IP. For details on the configuration options, refer to the [IP Parameter Description](#) section. For sample configurations relating to the EFB features, specifically for I2C and UFM functionality, refer to the [I2C Core Transaction Example](#) and [Flash Write and Read Example](#) sections.



**Figure 7.2. IP Configuration**

4. Click **Generate**. The **Check Generated Result** dialog box opens, showing design block messages and results as shown in [Figure 7.3](#).



**Figure 7.3. Check Generated Result**

5. Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields shown in [Figure 7.1](#).

### 7.1.1. Generated Files and File Structure

The generated EFB module package includes the closed-box (<Component name>\_bb.v) and instance templates (<Component name>\_tmpl.v/vhd) that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file (<Component name>.v) that can be used as an instantiation template for the module is also provided. You may also use this top-level reference as the starting template for the top-level for their complete design. The generated files are listed in [Table 7.1](#).

**Table 7.1. Generated File List**

| Attribute   | Description   |
|---|---|
| <Component name>.ipx  | This file contains the information on the files associated to the generated IP. |
| <Component name>.cfg  | This file contains the parameter values used in IP configuration.               |
| component.xml   | Contains the ipxact: component information of the IP.                           |
| design.xml  | Documents the configuration parameters of the IP in IP-XACT 2014 format.        |
| rtl/<Component name>.v  | This file provides an example RTL top file that instantiates the module.        |
| rtl/<Component name>_bb.v                                       | This file provides the synthesis closed-box.                                    |
| misc/<Component name>_tmpl.v<br>misc /<Component name>_tmpl.vhd | These files provide instance templates for the module.                          |

## 7.2. Design Implementation

Completing your design includes additional steps to specify analog properties, pin assignments, and timing and physical constraints. You can add and edit the constraints using the Device Constraint Editor or by manually creating a PDC File.

Post-Synthesis constraint files (.pdc) contain both timing and non-timing constraint.pdc source files for storing logical timing/physical constraints. Constraints that are added using the Device Constraint Editor are saved to the active .pdc file. The active post-synthesis design constraint file is then used as input for post-synthesis processes.

Refer to the relevant sections in the Lattice Radiant Software User Guide for more information on how to create or edit constraints and how to use the Device Constraint Editor.

## 7.3. Timing Constraints

The timing constraints are based on the clock frequency used. The timing constraints for the IP are defined in relevant constraint files. The example below shows the IP timing constraints generated for the EFB Module IP.

```
create_clock -name {wb_clk_i} -period 20 [get_ports wb_clk_i]
```

## 7.4. Physical Constraints

For the primary I2C core signals (i2c1\_scl and i2c1\_sda), the Radiant software tool automatically routes these signals to their pre-assigned pins (no user pin location constraint is necessary). However, for the secondary I2C core signals (i2c2\_scl and i2c2\_sda), these can be routed to any GPIO of the MachXO4 device.

The SPI core signals (spi\_clk, spi\_miso, spi\_mosi, spi\_csn[0], and ufm\_sn) are automatically routed by the Radiant software tool to their pre-assigned pins. The remaining SPI signals (spi\_csn[7:1]) can be routed to any GPIO on the MachXO4 device.

## 7.5. Specifying the Strategy


The Radiant software provides two predefined strategies: Area and Timing. It also enables you to create customized strategies. For details on how to create a new strategy, refer to the *Strategies* section in the Lattice Radiant Software user guide.

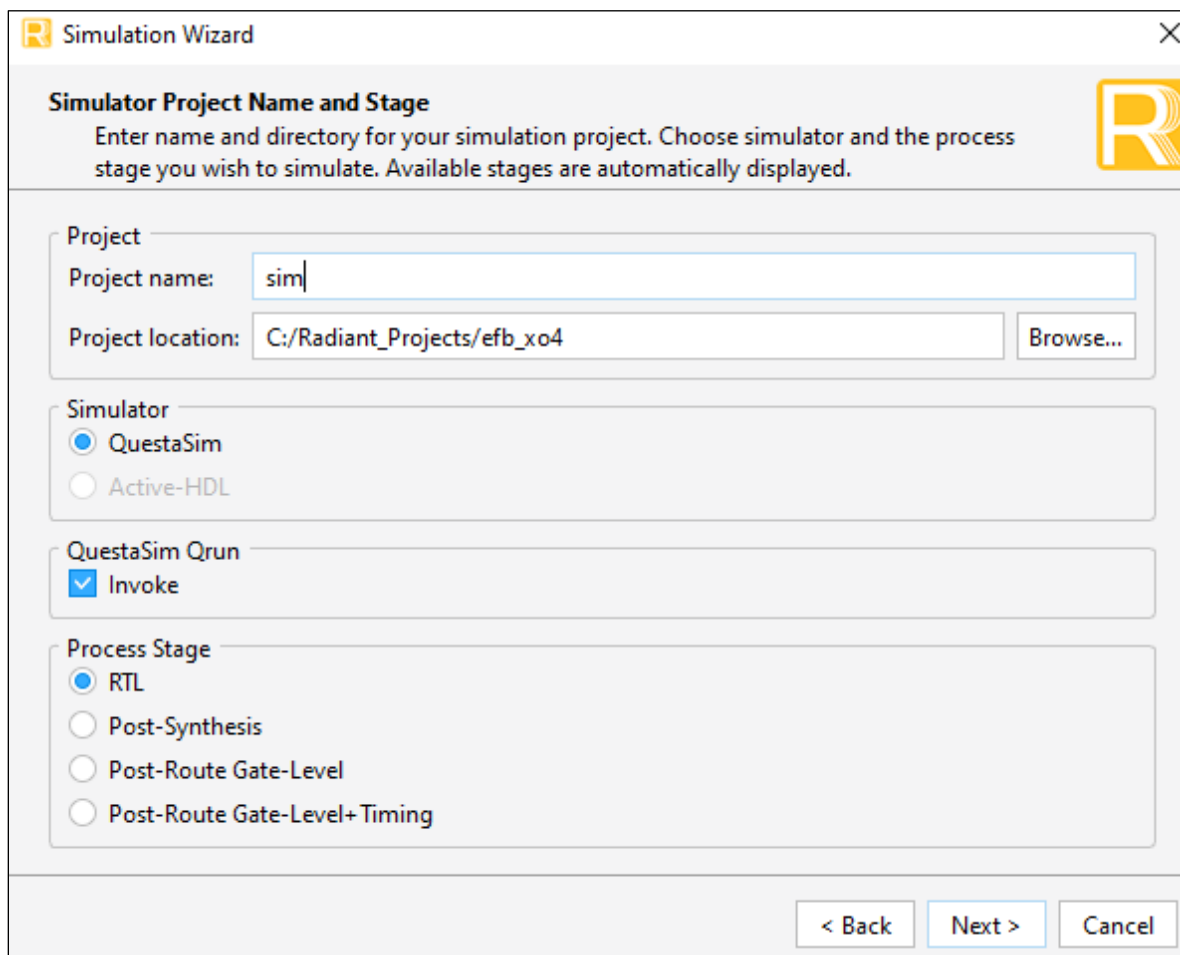


## 7.6. Running Functional Simulation

You can run functional simulation after the IP is generated.

To run functional simulation:

1. Click the  button located on the **Toolbar** to initiate the **Simulation Wizard** shown in [Figure 7.4](#).



The **Simulation Wizard** dialog box is shown. It has a title bar with the Lattice logo and a close button. The main content area is titled **Simulator Project Name and Stage** and includes the instruction: "Enter name and directory for your simulation project. Choose simulator and the process stage you wish to simulate. Available stages are automatically displayed." The dialog is divided into four sections:   
1. **Project**: Contains a "Project name:" text box with "sim" entered and a "Project location:" text box with "C:/Radiant\_Projects/efb\_xo4" entered, followed by a "Browse..." button.   
2. **Simulator**: Contains two radio buttons, "QuestaSim" (selected) and "Active-HDL".   
3. **QuestaSim Qrun**: Contains a checked checkbox labeled "Invoke".   
4. **Process Stage**: Contains four radio buttons: "RTL" (selected), "Post-Synthesis", "Post-Route Gate-Level", and "Post-Route Gate-Level+Timing".   
At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

Figure 7.4. Simulation Wizard

2. Click **Next** to open the **Add and Reorder Source** window as shown in Figure 7.5.

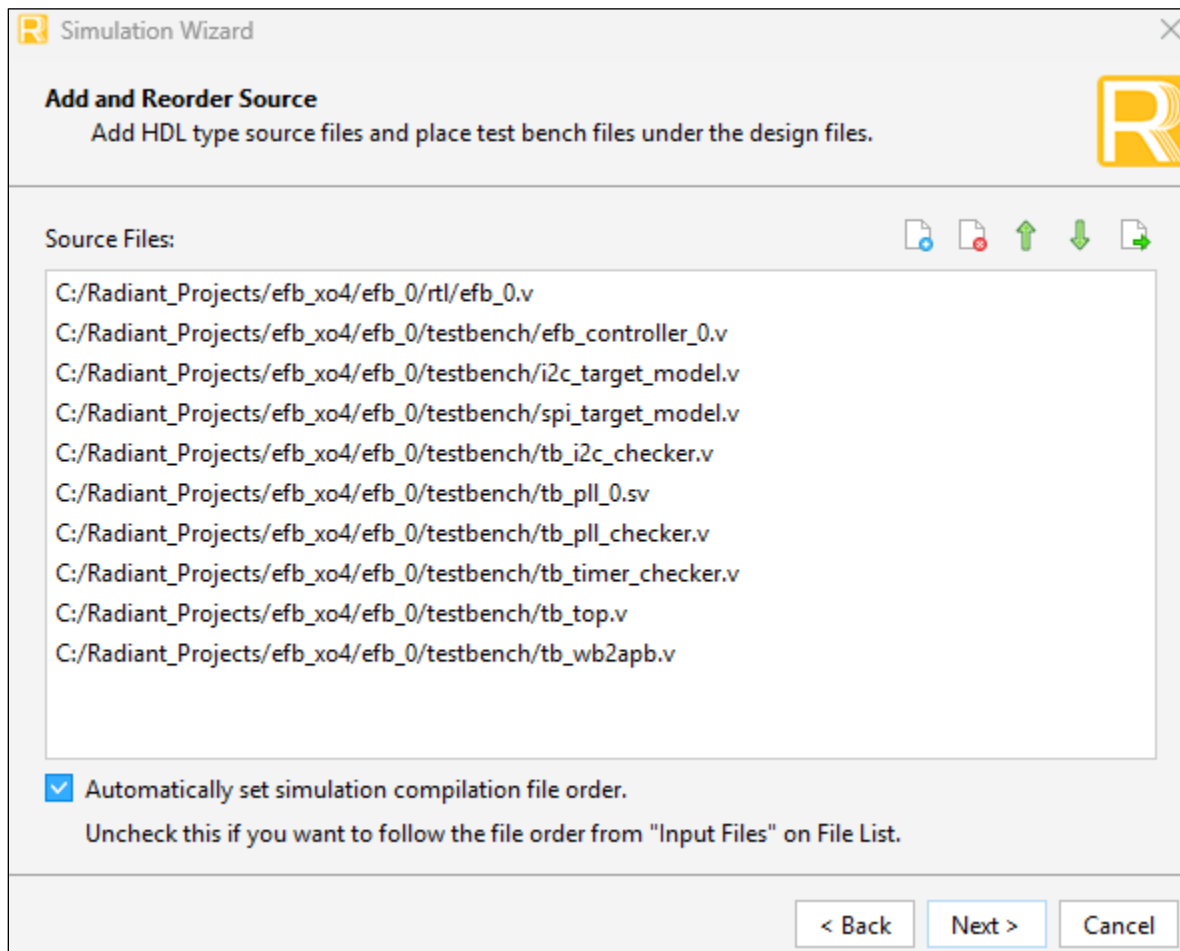


Figure 7.5. Add and Reorder Source

3. Click **Next**. The **Summary** window is shown.
4. Click **Finish** to run the simulation.

The waveform in Figure 7.6 shows an example simulation result.

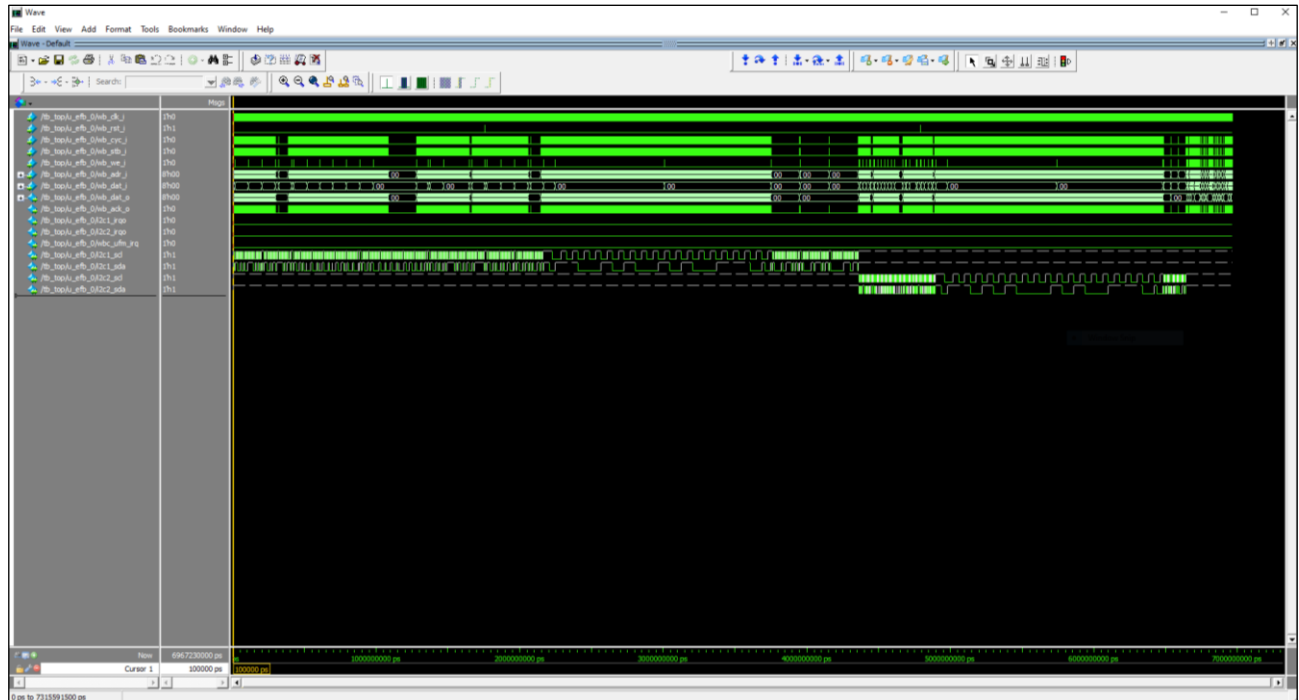


Figure 7.6. Simulation Waveform

## 7.6.1. Simulation Results

The provided testbench includes basic functional tests for all EFB features. A WISHBONE controller is used to access registers and generate the required sequences. If APB support is enabled, a WISHBONE-to-APB bridge is included to interface with the EFB.

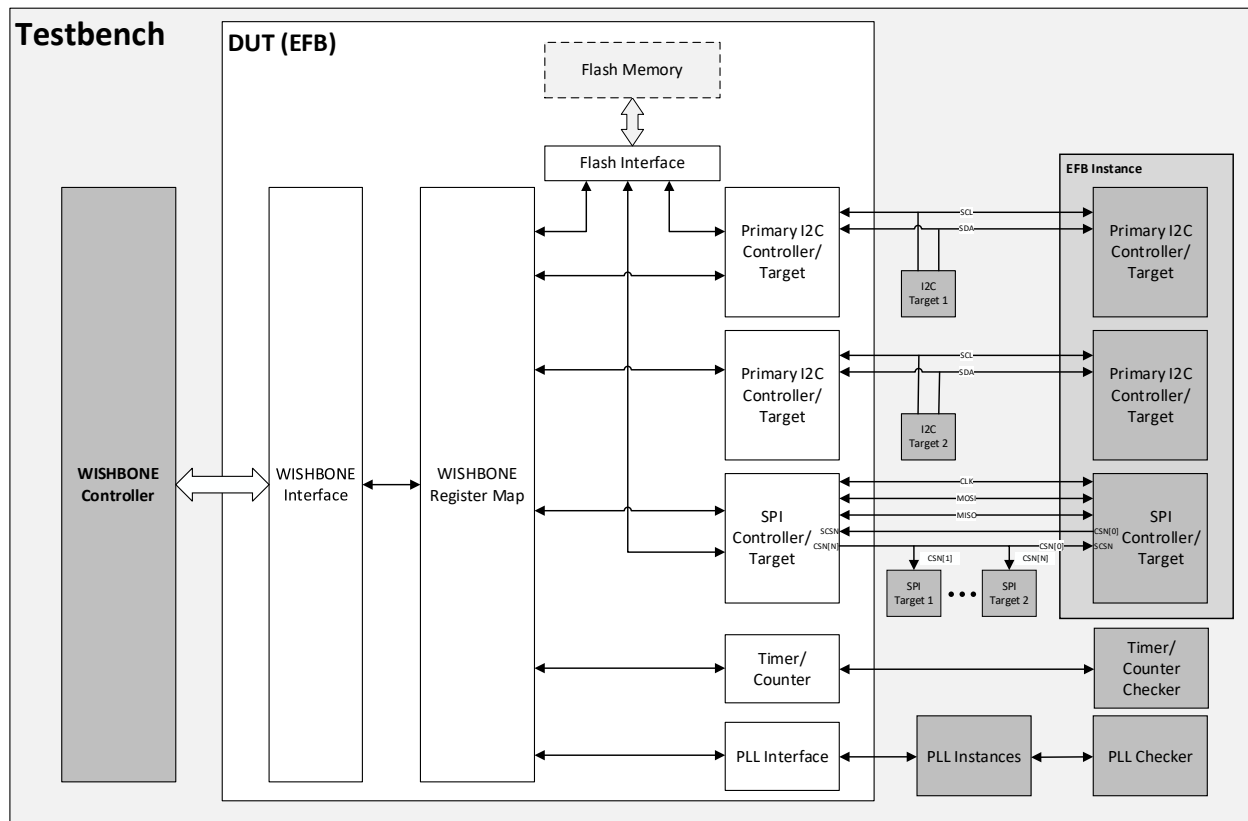
The following describe how each EFB feature is exercised and checked through simulation using the provided testbench:

- I2C Core Simulation:
  - Controller Mode – A custom I2C target module with simple memory is used to confirm basic write and read behavior.
  - Target Mode – An EFB instance with I2C enabled acts as the controller to exercise target functionality.
- SPI Core Simulation:
  - Controller Mode – A custom SPI target module is connected to observe basic controller behavior.
  - Target Mode – An EFB instance with SPI enabled is used to simulate a controller and check target-side operation.
- Timer/Counter Simulation:
  - Timer/Counter functionality is exercised by:
    - Observing output timing and interrupt signals.
    - Accessing Timer/Counter registers to adjust and monitor behavior dynamically.
- PLL Interface Simulation:
  - The PLL interface is exercised by:
    - Instantiating a PLL IP in the testbench.
    - Observing clock timing behavior.

**Note:** Only basic PLL configuration changes are included in the simulation.

- UFM Access:
  - Basic flash memory operations are performed using WISHBONE register access:
    - Initialization
    - Erase operation
    - Status flag polling
    - Write and read operations
  - Additionally, read/write operations through I2C and SPI interfaces are included if those interfaces are enabled.

The following figure provides an overview of a sample testbench.



**Figure 7.7. Sample Testbench Overview**

Depending on the features enabled during IP configuration, the corresponding test is included in the simulation. The simulation logs are expected to indicate **SIMULATION PASSED**, along with the corresponding waveform behavior.

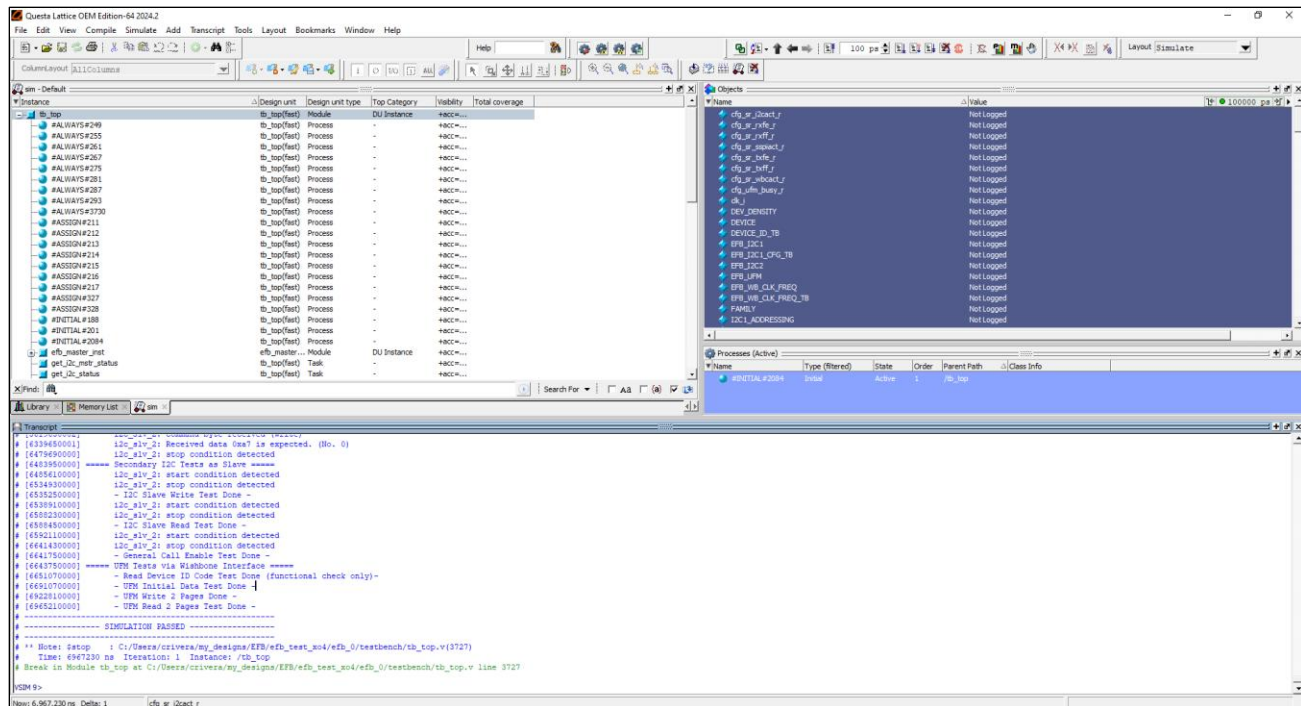


Figure 7.8. Sample Testbench Output

## 7.7. Example Designs (Simulation)

### 7.7.1. I2C Core Transaction Example

Figure 7.9 and Figure 7.10 show flow diagrams for controlling and receiving I2C reads and writes initiated through the WISHBONE interface. The following sequence is for the primary I2C, but the same sequence applies to the secondary I2C. This sequence is implemented in the included sample testbench to confirm the basic operations of the Primary and Secondary I2C. Figure 7.11 shows an example configuration to enable both Primary and Secondary I2C.

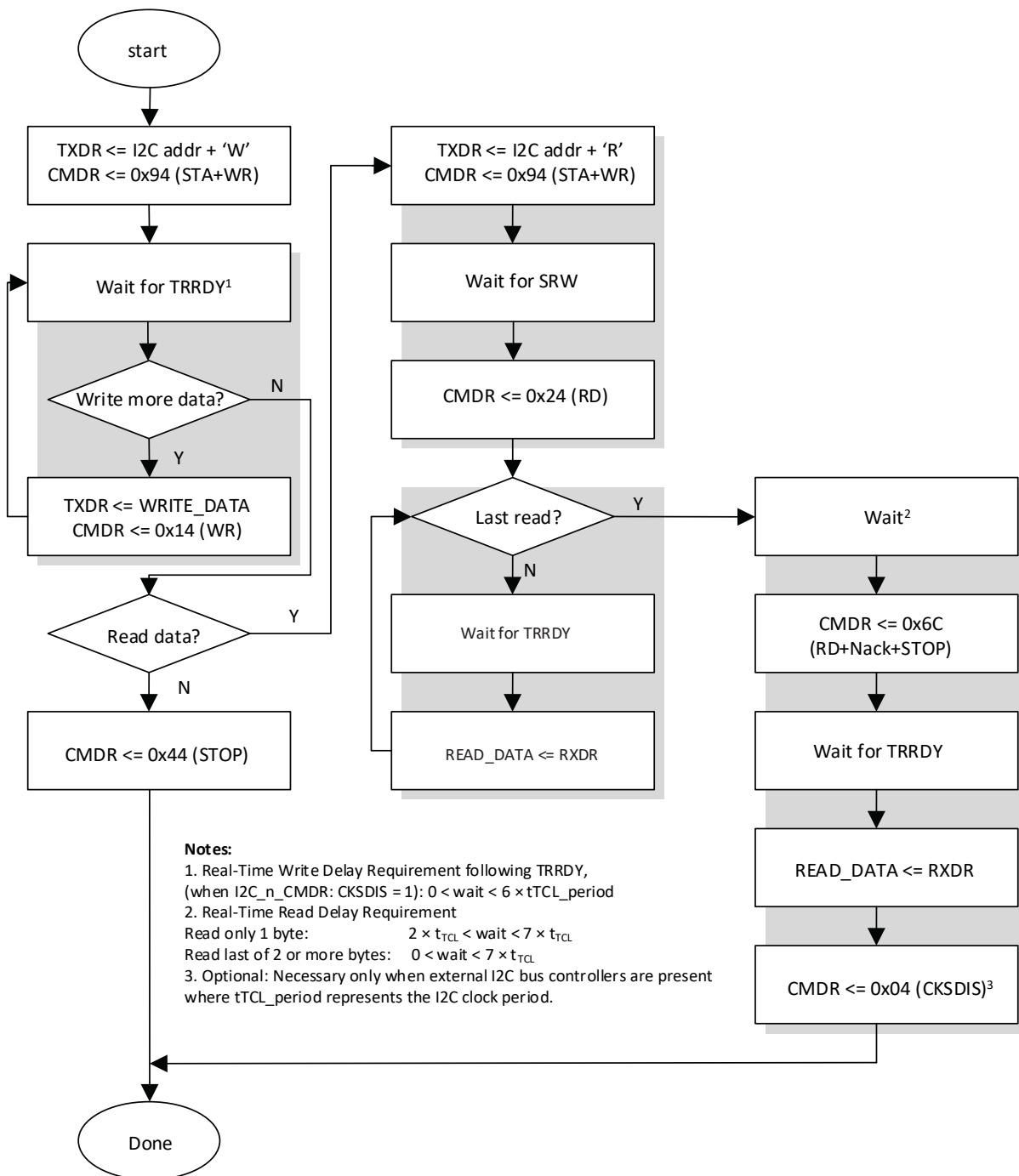
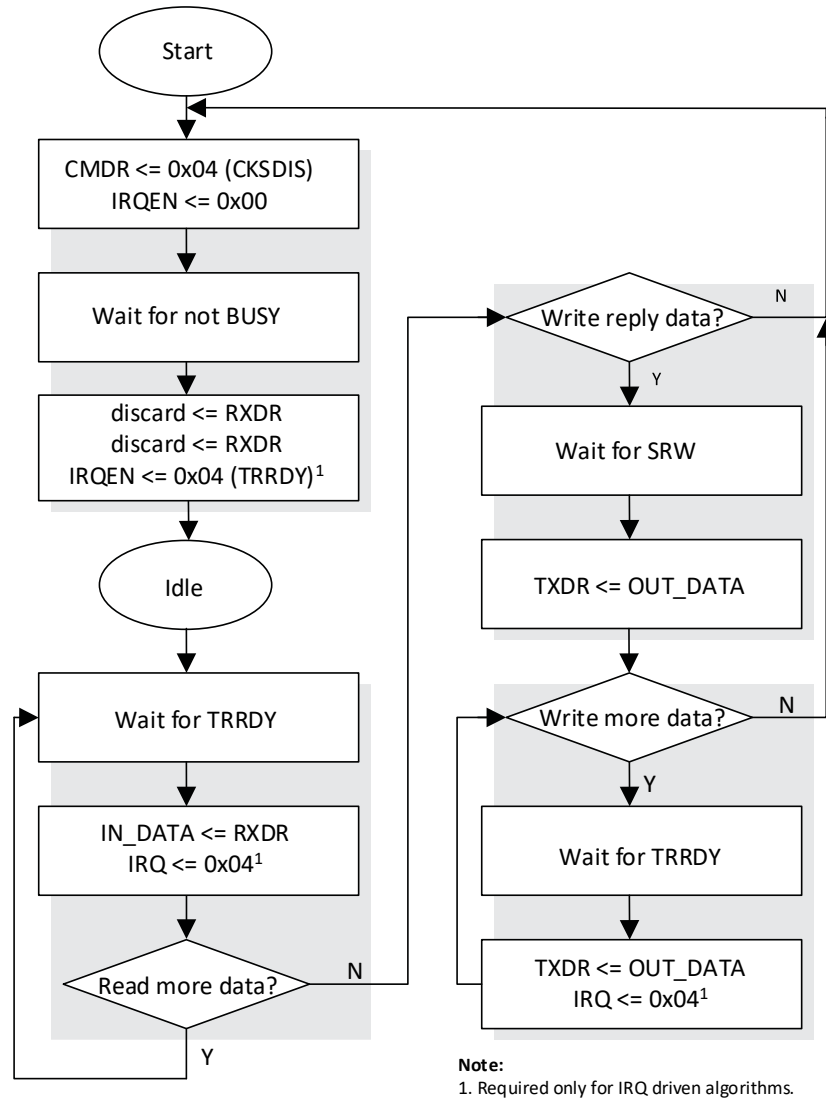


Figure 7.9. I2C Controller Read/Write Example (through WISHBONE)



**Figure 7.10. I2C Target Read/Write Example (through WISHBONE)**

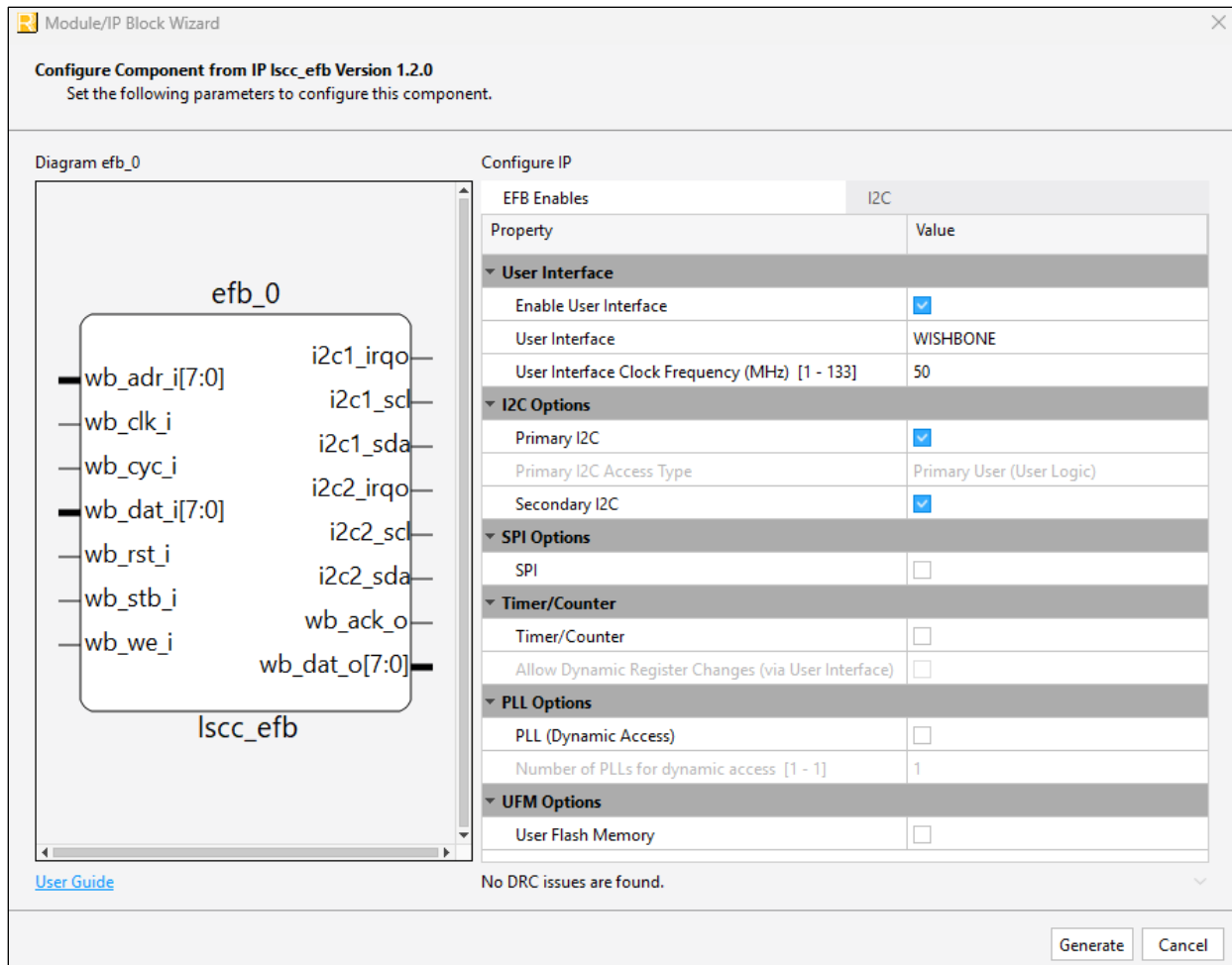


Figure 7.11. I2C Sample Configuration (Primary and Secondary I2C Enabled)

## 7.7.2. Flash Write and Read Example

Table 7.2, Table 7.3, and Table 7.4 show example write, erase, and read operations on a flash sector. The following sequence is for UFM access, but similar sequences can be followed for other sectors using their corresponding commands. This was implemented in the included sample testbench to confirm the basic operations of the UFM.

Table 7.2. Write Two UFM Pages (WISHBONE)

| Instruction Number | R/W1 | CMD2 | Operand  | Data        | Comment   |
|--------------------|------|------|----------|-------------|---|
| —                  | —    | —    | —        | —           | Open frame  |
| 1                  | W    | 74   | 08 00 00 | —           | Enable Configuration Interface                                      |
| —                  | —    | —    | —        | —           | Close frame   |
| —                  | —    | +    | —        | —           | —   |
| 2                  | W    | 3C   | 00 00 00 | —           | Poll Configuration Status Register                                  |
| —                  | R    | —    | —        | xx xx bx xx | —   |
| —                  | —    | —    | —        | —           | (Repeat until Busy Flag not set, or wait 5 $\mu$ s if not polling.) |
| —                  | —    | +    | —        | —           | —   |
| 3                  | W    | 47   | 00 00 00 | —           | Init UFM Address to 0000  |
| —                  | —    | —    | —        | —           | —   |
| —                  | —    | +    | —        | —           | —   |
| 4                  | W    | C9   | 00 00 01 | 00 01 02 03 | Write UFM Page 0 Data   |



| Instruction Number | R/W1 | CMD2 | Operand  | Data   | Comment   |
|--------------------|------|------|----------|--|---|
|                    |      |      |          | 04 05 06 07<br>08 09 0A 0B<br>0C 0D 0E 0F                |   |
| —                  | —    | —    | —        | —  | —   |
| —                  | —    | +    | —        | —  | —   |
| 5                  | W    | 3C   | 00 00 00 | —  | Poll Configuration Status Register                                      |
| —                  | R    | —    | —        | xx xx bx xx  | —   |
| —                  | —    | —    | —        | —  | (Repeat until Busy Flag not set)  |
| —                  | —    | +    | —        | —  | —   |
| 6                  | W    | C9   | 00 00 01 | 10 11 12 13<br>14 15 16 17<br>18 19 1A 1B<br>1C 1D 1E 1F | Write UFM Page 1 Data<br><b>Note:</b> Address automatically incremented |
| —                  | —    | —    | —        | —  | —   |
| —                  | —    | +    | —        | —  | —   |
| 7                  | W    | 3C   | 00 00 00 | —  | Poll Configuration Status Register                                      |
| —                  | R    | —    | —        | xx xx bx xx  | —   |
| —                  | —    | —    | —        | —  | (poll until Busy Flag clear)  |
| —                  | —    | +    | —        | —  | —   |
| 8                  | W    | 26   | 00 00    | —  | Disable Configuration Interface   |
| —                  | —    | —    | —        | —  | —   |
| —                  | —    | +    | —        | —  | —   |
| 9                  | W    | FF   | —        | —  | Bypass (NOP)  |
| —                  | —    | —    | —        | —  | —   |

**Notes:**

- When accessing the flash through WISHBONE, use CFGTXDR (0x71) to write data and CFDRXDR (0x73) to read data.
- + and – refer to the command framing protocol appropriate for the interface discussed in the [Command Framing](#) section.

**Table 7.3. Read One UFM Page (WISHBONE)**

| Instruction Number | R/W1 | CMD2 | Operand  | Data   | Comment                            |
|--------------------|------|------|----------|--|------------------------------------|
| —                  | —    | +    | —        | —  | Open frame                         |
| 1                  | W    | 74   | 08 00 00 | —  | Enable Configuration Interface     |
| —                  | —    | —    | —        | —  | Close frame                        |
| —                  | —    | +    | —        | —  | —                                  |
| 2                  | W    | 3C   | 00 00 00 | —  | Poll Configuration Status Register |
| —                  | R    | —    | —        | xx xx bx xx  | —                                  |
| —                  | —    | —    | —        | —  | (Repeat until Busy Flag not set)   |
| —                  | —    | +    | —        | —  | —                                  |
| 3                  | W    | B4   | 00 00 00 | 40 00 00 01  | Set UFM Address to 0001            |
| —                  | —    | —    | —        | —  | —                                  |
| —                  | —    | +    | —        | —  | —                                  |
| 4                  | W    | CA   | 10 00 01 | —  | Read one page UFM (page 1) data    |
| —                  | R    | —    | —        | 10 11 12 13<br>14 15 16 17<br>18 19 1A 1B<br>1C 1D 1E 1F | —                                  |
| —                  | —    | —    | —        | —  | —                                  |
| —                  | —    | +    | —        | —  | —                                  |

| Instruction Number | R/W1 | CMD2 | Operand | Data | Comment                         |
|--------------------|------|------|---------|------|---------------------------------|
| 5                  | W    | 26   | 00 00   | —    | Disable Configuration Interface |
| —                  | —    | —    | —       | —    | —                               |
| —                  | —    | +    | —       | —    | —                               |
| 6                  | W    | FF   | —       | —    | Bypass (NOP)                    |
| —                  | —    | —    | —       | —    | —                               |

**Notes:**

- When accessing the Flash through WISHBONE, use CFGTXDR (0x71) to write data and CFDRXDR (0x73) to read data.
- + and – refer to the command framing protocol appropriate for the interface discussed in the [Command Framing](#) section.

**Table 7.4. Read Two UFM Pages (WISHBONE)**

| Instruction Number | R/W1 | CMD2 | Operand  | Data   | Comment   |
|--------------------|------|------|----------|--|---|
| —                  | —    | +    | —        | —  | Open frame  |
| 1                  | W    | 74   | 08 00 00 | —  | Enable Configuration Interface  |
| —                  | —    | —    | —        | —  | Close frame   |
| —                  | —    | +    | —        | —  | —   |
| 2                  | W    | 3C   | 00 00 00 | —  | Poll Configuration Status Register                                      |
| —                  | R    | —    | —        | xx xx bx xx  | —   |
| —                  | —    | —    | —        | —  | (Repeat until Busy Flag not set, or wait 5 $\mu$ s if not polling.)     |
| —                  | —    | +    | —        | —  | —   |
| 3                  | W    | 47   | 00 00 00 | —  | Init UFM address to 0000  |
| —                  | —    | —    | —        | —  | —   |
| —                  | —    | +    | —        | —  | —   |
| 4                  | W    | CA   | 10 00 03 | —  | Read two pages of UFM data, after one page of dummy bytes. <sup>3</sup> |
| —                  | R    | —    | —        | xx xx xx xx xx<br>xx xx xx xx xx<br>xx xx xx xx xx<br>xx<br>00 01 02 03<br>04 05 06 07<br>08 09 0A 0B<br>0C 0D 0E 0F<br>10 11 12 13<br>14 15 16 17<br>18 19 1A 1B<br>1C 1D 1E 1F | —   |
| —                  | —    | —    | —        | —  | —   |
| —                  | —    | +    | —        | —  | —   |
| 5                  | W    | 26   | 00 00    | —  | Disable Configuration Interface   |
| —                  | —    | —    | —        | —  | —   |
| —                  | —    | +    | —        | —  | —   |
| 6                  | W    | FF   | —        | —  | Bypass (NOP)  |
| —                  | —    | —    | —        | —  | —   |

**Notes:**

- When accessing the Flash through WISHBONE, use CFGTXDR (0x71) to write data and CFDRXDR (0x73) to read data.
- + and – refer to the command framing protocol appropriate for the interface.
- num\_pages count must include dummy page.

To use the I2C ports to access the flash memory, both *Primary I2C* and *User Flash Memory* must be enabled. Then, select *Primary Config (Flash Access)* as the *Primary I2C Access Type*. This sets the Primary I2C Address to *0bxxxxx00*. Figure 7.12 shows an example configuration to enable Flash Access, through both Wishbone Access and Primary I2C ports.

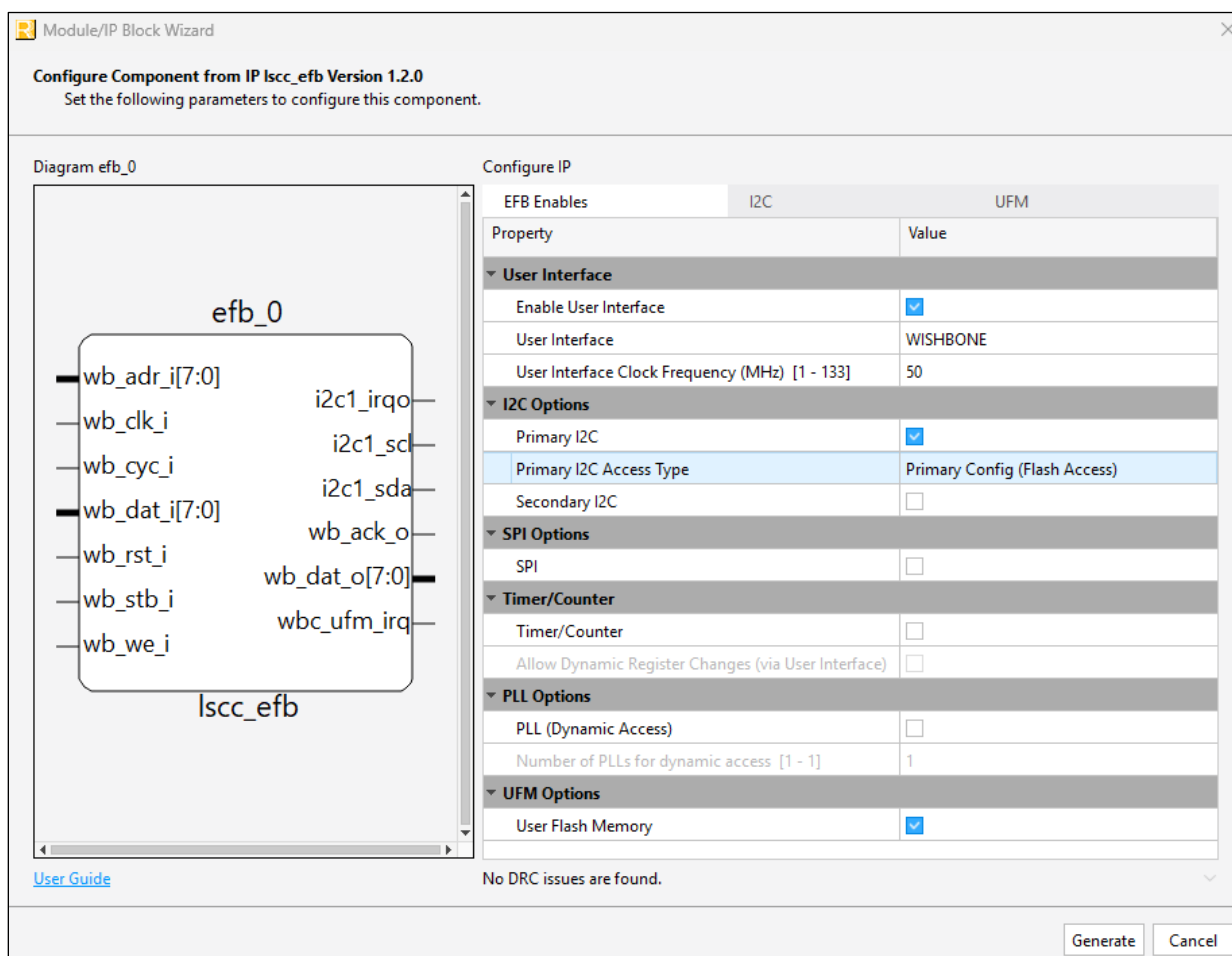
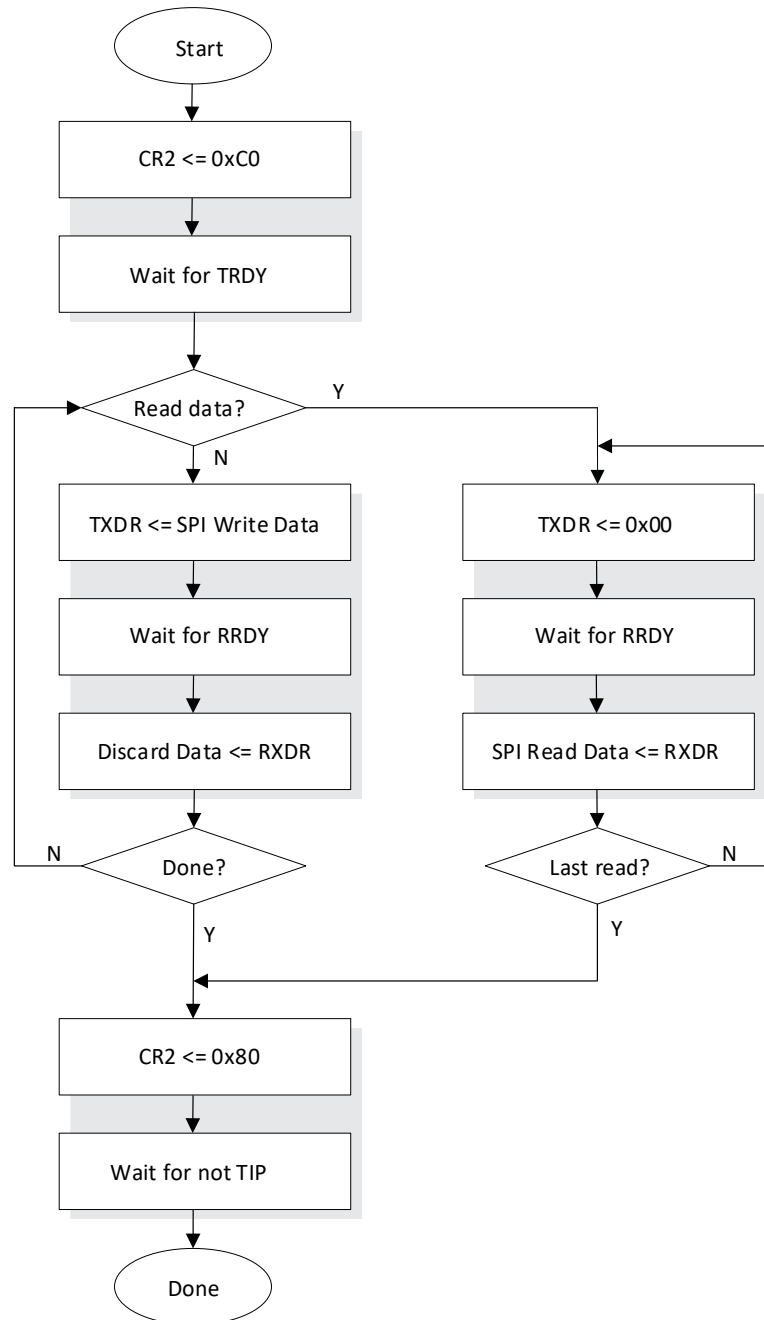


Figure 7.12. UFM Access Sample Configuration (through Wishbone and I2C Access)

### 7.7.3. SPI Core Transaction Example

Figure 7.13 shows flow diagrams for controlling and receiving SPI reads and writes initiated through the WISHBONE interface. This sequence is implemented in the sample testbench to confirm the basic operations of the SPI core.

Figure 7.14 shows an SPI example configuration.



**Figure 7.13. SPI Controller Read/Write Example (through WISHBONE)**

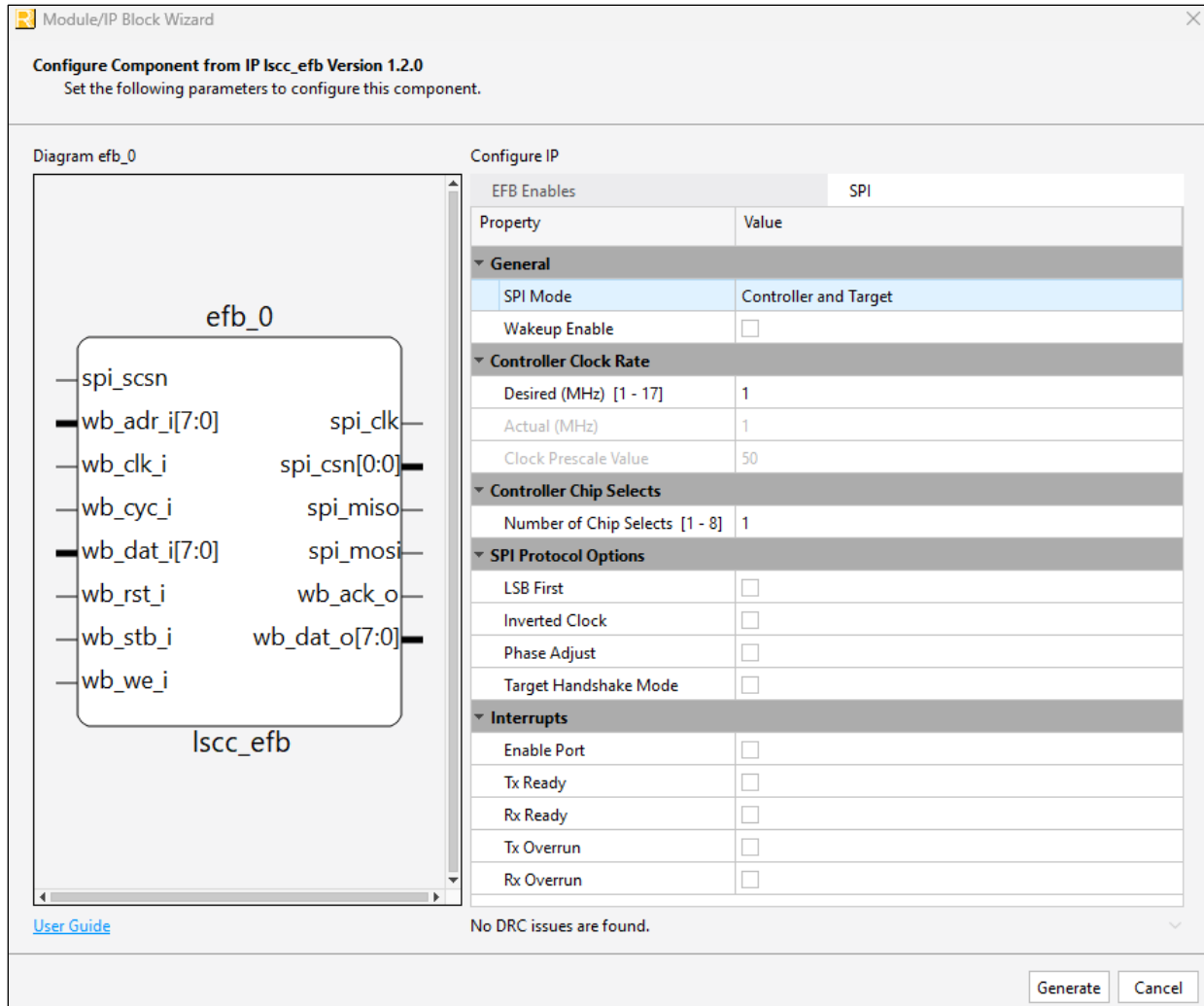


Figure 7.14. SPI Sample Configuration

## 8. Design Considerations

Refer to the design tips in the [MachXO4 Hardened Control Functions User Guide \(FPGA-TN-02403\)](#) for suggestions on using the various MachXO4 EFB functions.

## Appendix A. Resource Utilization

Table A.1 shows a sample resource utilization of the EFB Module IP core on the LFMXO4-110HE-6BBG484I device.

**Table A.1. LFMXO4-110HE-6BBG484I Device Resource Utilization**

| IP Configuration  | Registers | LUTs | EFB | Fmax (MHz) <sup>1</sup> |
|---|-----------|------|-----|-------------------------|
| (Default)<br>User Interface Enabled (WISHBONE)<br>• Interface Clock: 133 MHz<br>I2C Cores Enabled (Primary I2C: Flash Access)<br>User Flash Memory Enabled                      | 0         | 0    | 1   | 133.014<br>(wb_clk_i)   |
| User Interface Enabled (WISHBONE)<br>• Interface Clock: 133 MHz<br>SPI Core Enabled (Controller and Target)<br>• SPI Clock: 45 MHz<br>• CS Pins: 8<br>User Flash Memory Enabled | 0         | 0    | 1   | 133.014<br>(wb_clk_i)   |
| User Interface Disabled<br>Timer/Counter Enabled<br>Pre-scale Divider Value: 8  | 0         | 0    | 1   | 133.014 (tc_clk_i)      |

**Note:**

1. The maximum WISHBONE clock that can be configured is 133 MHz. Fmax is generated when the FPGA design contains only the EFB Module IP core and the target frequency is 133 MHz. These values may be reduced when user logic is added to the FPGA design.

## References

- [MachXO4 EFB Module IP Release Notes \(FPGA-RN-02079\)](#)
- [MachXO4 Hardened Control Functions User Guide \(FPGA-TN-02403\)](#)
- [MachXO4 Hardened Control Functions Reference Guide \(FPGA-TN-02404\)](#)
- [Lattice Radiant Timing Constraints Methodology \(FPGA-AN-02059\)](#)
- [MachXO4](#) web page
- [Lattice Radiant Software](#) web page
- [Lattice Solutions IP Cores](#) web page
- [Lattice Solutions Reference Designs](#) web page
- [Lattice Insights](#) web page for Lattice Semiconductor training courses and learning plans



## Technical Support Assistance

Submit a technical support case through [www.latticesemi.com/techsupport](http://www.latticesemi.com/techsupport).

For frequently asked questions, refer to the Lattice Answer Database at [www.latticesemi.com/Support/AnswerDatabase](http://www.latticesemi.com/Support/AnswerDatabase).

## Revision History

**Note:** In some instances, the IP may be updated without changes to the user guide. The user guide may reflect an earlier IP version but remains fully compatible with the later IP version. Refer to the IP Release Notes for the latest updates.

### Revision 1.1, IP v1.2.0, December 2025

| Section              | Change Summary   |
|----------------------|--|
| Register Description | Updated the <i>DIVIDER</i> description for <a href="#">Table 5.16. SPI Clock Pre-scale</a> . |

### Revision 1.0, IP v1.2.0, December 2025

| Section                                   | Change Summary  |
|---|---|
| All                                       | <ul style="list-style-type: none"> <li>Updated the IP version information on the cover page.</li> <li>Added a note on the IP version in the <i>Quick Facts</i> and <i>Revision History</i> sections.</li> <li>Made editorial fixes.</li> </ul>  |
| Abbreviations in This Document            | Added <i>ACK</i> , <i>APB</i> , <i>CTCM</i> , <i>GUI</i> , <i>GPIO</i> , <i>LSE</i> , <i>MDF</i> , <i>NACK</i> , <i>PFCPWM</i> , <i>PLL</i> , and <i>PWM</i> .  |
| Introduction                              | <ul style="list-style-type: none"> <li>In the Overview of the IP section, added the following control functions: <ul style="list-style-type: none"> <li>One SPI core</li> <li>One 16-bit Timer/Counter</li> <li>Interface to the dynamic PLL configuration settings</li> <li>Interface to the on-chip power controller through I2C and SPI</li> </ul> </li> <li>Updated <i>Resource Utilization</i> and <i>Lattice Implementation</i> in Table 1.1. Summary of the EFB Module IP.</li> <li>Updated Table 1.2. EFB Module IP Support Readiness.</li> <li>Updated the Features section and added the Attribute Names section.</li> </ul>  |
| Functional Description                    | <ul style="list-style-type: none"> <li>Updated Figure 2.1. EFB Module IP Block Diagram – Figure 2.5. EFB Module I2C Core Interface and Figure 2.8. EFB Module Flash Interface.</li> <li>Updated Table 2.1. User Interfaces and Supported Protocols.</li> <li>Added Register Access to the WISHBONE Bus Interface (Register Access) section title.</li> <li>Added SPI IP Core and WISHBONE Controller (for PLL) sections.</li> <li>Updated the Flash Access Interface section.</li> </ul>  |
| IP Parameter Description                  | Updated this section.   |
| Signal Description                        | <ul style="list-style-type: none"> <li>Replaced the <i>Interrupt Ports</i> subtable header with <i>Flash Memory</i>.</li> <li>Added <i>SPI Interface</i>, <i>Timer/Counter</i>, <i>PLL0 (WISHBONE)</i>, <i>PLL1 (WISHBONE)</i>, and <i>Power Controller</i> ports.</li> </ul>   |
| Register Description                      | Added the SPI Registers and Timer/Counter Registers sections.   |
| Command and Data Transfers to Flash Space | Added the SPI Framing section.  |
| Designing with the IP                     | <ul style="list-style-type: none"> <li>Added a note on screenshots in this section.</li> <li>Updated the Simulation Results section.</li> <li>Added SPI core signals description to the Physical Constraints section.</li> <li>Add the SPI Core Transaction Example section.</li> <li>Updated the following figures: <ul style="list-style-type: none"> <li>Figure 7.1. Module/IP Block Wizard</li> <li>Figure 7.2. IP Configuration</li> <li>Figure 7.3. Check Generated Result</li> <li>Figure 7.5. Add and Reorder Source</li> <li>Figure 7.7. Sample Testbench Overview</li> <li>Figure 7.11. I2C Sample Configuration (Primary and Secondary I2C Enabled)</li> <li>Figure 7.12. UFM Access Sample Configuration (through Wishbone and I2C Access)</li> </ul> </li> </ul> |
| Design Considerations                     | Added this section.   |

| Section              | Change Summary  |
|----------------------|---|
| Resource Utilization | Updated this section.   |
| References           | Added <i>MachXO4 EFB Module IP Release Notes (FPGA-RN-02079)</i> , <i>MachXO4 Hardened Control Functions User Guide (FPGA-TN-02403)</i> , <i>MachXO4 Hardened Control Functions Reference Guide (FPGA-TN-02404)</i> , and <i>MachXO4 web page</i> . |

**Revision 0.80, IP v1.0.0, June 2025**

| Section | Change Summary   |
|---------|------------------|
| All     | Initial release. |



[www.latticesemi.com](http://www.latticesemi.com)