RISC-V Controlled M-PESTI Initiator

Reference Design

FPGA-RD-02312-1.1

September 2025

Disclaimers

Inclusive Language

Contents

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
AHB	Advanced High-performance Bus
APB	Advanced Peripheral Bus
EBR	Embedded Block RAM
FIFO	First In, First Out
FPGA	Field Programmable Gate Array
JTAG	Joint Test Action Group
LED	Light-Emitting Diode
LUT	Look-Up Table
OCP	Open Compute Project
RAM	Random Access Memory
ROM	Read-Only Memory
UART	Universal Asynchronous Receiver-Transmitter
VW	Virtual Wire

1. Introduction

M-PESTI is a generic and extensible 1-wire, bidirectional circuit and protocol for applications such as cabled high-speed I/O interposers, managed power distribution, cooling subsystems, and control panels. This reference design provides M-PESTI as an SoC-based project with a RISC-V core and this reference design is OCP Ready™.

This version of the M-PESTI Initiator reference design provides a Lattice Propel™ Builder software solution template that uses the M-PESTI Initiator IP core with a reference driver or firmware, RISCV-MC core with other required IP modules, and M-PESTI target test component. The reference design is compliant with the M-PESTI Base Specification, which is part of the DC-MHS version 1.0 specification. This reference design includes the following collaterals:

- The SoC design project on which you can open, view, and modify the design through the Lattice Propel Builder software. The bitstream can be generated using the Lattice Diamond™ software.
- The SoC workspace, which includes necessary drivers for the connectivity and peripheral intellectual property (IP) cores and firmware for the demonstration.
- The CPLD demonstration bitstream.

1.1. Quick Facts

Download the reference design files from the Lattice reference design web page.

Table 1.1. Summary of the Reference Design

General	Target devices	MachXO3LF, MachXO5-NX.
	Source code format	Verilog, System Verilog, and Embedded C.
Simulation	Functional simulation	Performed.
	Timing simulation	Not performed.
	Testbench	Available.
	Testbench format	System Verilog.
Software Requirements	Software tool and version	• Lattice Radiant™ software 2024.1 or higher. • Lattice Diamond™ 3.13 or higher. • Lattice Diamond™ Programmer version 3.11 or higher. • Lattice Propel™ 2024.1 or higher.
	IP version (if applicable)	IP core v1.1.0.
Hardware Requirements	Board	• MachXO3LF Starter Kit (LCMXO3LF-6900C-S-EVN) or MachXO5-NX Development Board (LFMXO5-25-EVN). • MachXO3D Breakout Board (LCMXO3D-9400HC-B-EVN).
	Cable	• Two USB Type-A male to USB Mini-B male. • Five female-to-female jumper wires.

1.2. Features

Key features of the RISC-V Controlled M-PESTI Initiator reference design include:

- Communicates via half-duplex bidirectional UART protocol at 250k baud rate, 8-bit data, 1-bit odd parity, 1 start bit, and 1 stop bit on the M-PESTI port.
- Supports static discovery payload with CRC-8 payload checksum.
- Supports one initiator to many targets system.
- Supports a configurable number of M-PESTI devices, up to 64 targets.
- Supports autonomous static discovery payload request command to all targets in a round-robin manner during the Discovery phase.
- Supports static discovery payload request command retry. If the payload is not successfully received after an initial attempt, two more retries per target are triggered before proceeding to the next target(s). When the turn returns to the target, the initiator persistently sends command attempts as a set of initial and two retry attempts until the successful payload is received.
- Supports target reset at any time.

- Supports sending broadcast commands.
- Supports aborting an ongoing discovery or active phase command to insert a broadcast command.
- Supports source and destination cable coupling discovery.
- Supports custom user commands.

1.3. Limitations

- The M-PESTI Initiator IP core supports MachXO5™-NX, MachXO3D™, MachXO3L™, MachXO3LF™, Certus™-N2 and Avant™ devices. However, this reference design only includes an M-PESTI Initiator project for MachXO3LF and MachXO5-NX devices.
- Setting the APEN register to 1 is only allowed on targets that have a single VWIN and VWOUT bytes.

2. Directory Structure and Files

The following figure shows the directory structure of the reference design.

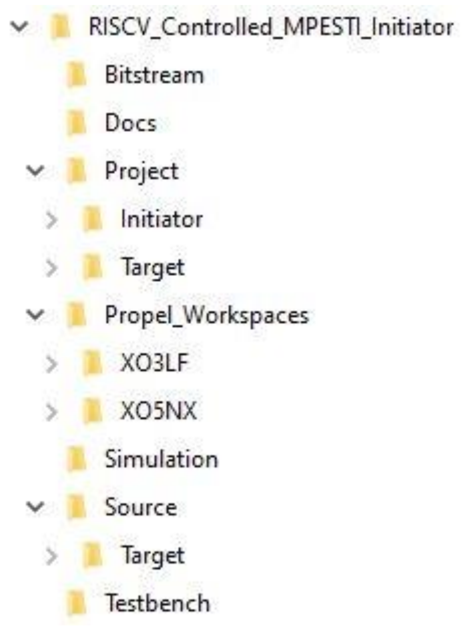


Figure 2.1. Directory Structure

The RISC-V Controlled M-PESTI Initiator reference design package includes the following subfolders:

- **Bitstream**—Contains the programming files for the MachXO3LF Starter Kit (Initiator), MachXO5-NX Development Board (Initiator), and MachXO3D Breakout Board (Target).
- **Project**—Contains the Diamond projects for the M-PESTI Initiator and the M-PESTI target model.
- **Propel_Workspace**—Contains the M-PESTI Initiator Propel Builder SoC project and source files under the XO3LF_SOC and XO5NX_SOC folders. This also contains the Lattice Propel SDK Embedded C project under the XO3LF_SW and XO5NX_SW folders.
- **Simulation**—Contains the simulation file (*.do*) used to run RTL simulation on the ModelSim™ Lattice FPGA Edition software.
- **Source**—Contains the source files for the M-PESTI target model.
- **Testbench**—Contains the testbench used in simulation.

The following table shows the list of files included in the reference design package.

Table 2.1. File List

Attribute	Description
mpesti_m0.ipx	This file contains the information on the files associated to the generated IP.
mpesti_m0.cfg	This file contains the parameter values used in IP configuration.
mpesti_m0.sv	This file provides an example RTL top file that instantiates the module.
mpesti_m0_bb.v	This file provides the synthesis closed-box.
mpesti_m0_tmpl.v *mpesti_m0_tmpl.vhd*	These files provide instance templates for the module.

3. Functional Description

The following figure shows the high-level block diagram of the reference design, which consists of three key hardware components:

- M-PESTI Initiator SoC design that is programmed into a MachXO3LF Starter Kit or MachXO5-NX Development Board
- Emulated multiple target devices that are programmed into a MachXO3D Breakout Board
- A PC-based UART terminal such as PuTTY or the built-in UART Terminal console within the Lattice Propel SDK

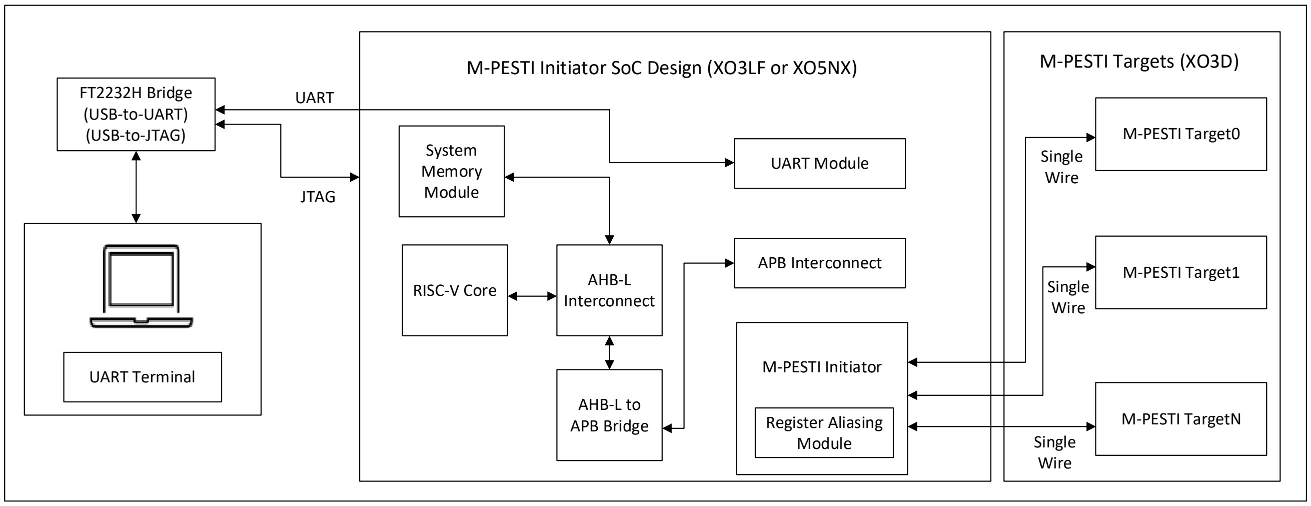


Figure 3.1. Reference Design Block Diagram

3.1. Design Components

The RISC-V Controlled M-PESTI Initiator reference design includes the following blocks:

- M-PESTI Initiator SoC design
  - M-PESTI IP core including the embedded C drivers
  - RISCV-V module
  - AHB-L interconnect module
  - AHB-L to APB bridge module
  - APB interconnect module
  - System memory module
  - UART module
- M-PESTI target module with emulated multiple target devices
- UART terminal

3.1.1. M-PESTI Initiator SoC Design

The following figure shows that the M-PESTI Initiator Reference IP is instantiated in the SoC design and uses all the necessary components to enable the RISC-V core to orchestrate the IP functionality. This Lattice Propel Builder SoC design can be used to generate the Diamond™ software project, which is used to generate the programming bitstreams. For more information, refer to the Lattice Propel 2024.1 Builder User Guide. Pre-generated bitstreams are already included in the reference design package, and regeneration is only required when you need to modify the design.

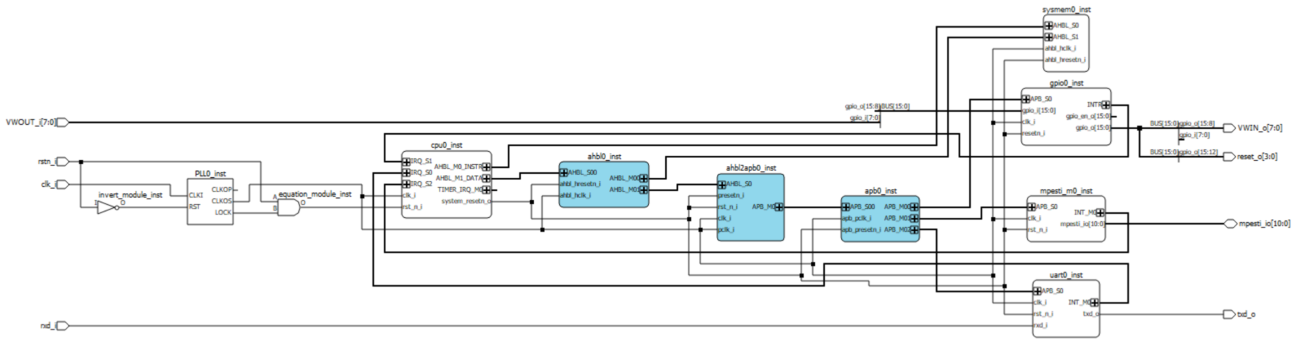


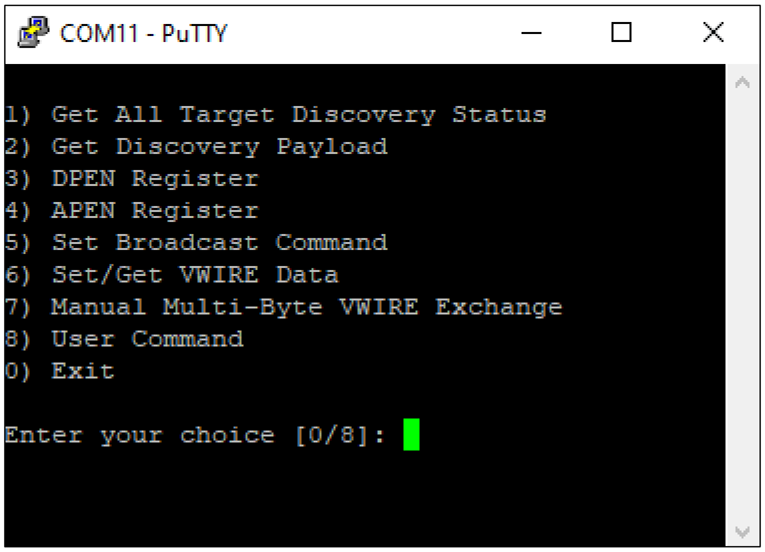
Figure 3.2. M-PESTI Initiator Reference Design Interface Diagram

3.1.2. M-PESTI Target Module

The M-PESTI target device is programmed into a MachXO3D Breakout Board. Several instantiations of the M-PESTI target module are programmed onto a separate FPGA board to emulate multiple M-PESTI target devices. These modules are connected to the initiator through the 1-wire interface. This target module has an internal loopback that receives the VWOUT value sent by the initiator and sends the value back as the VWIN data received by the initiator.

3.1.3. M-PESTI Initiator UART Terminal

To control the M-PESTI Initiator IP, the design uses a built-in UART-based Test Suite to manage IP functionalities and display messages on a UART terminal such as PuTTY, Tera Term, or the built-in terminal of the Lattice Propel™ SDK.



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]:
```

Figure 3.3. Test Suite

3.2. M-PESTI Protocol Phases

3.2.1. Discovery Phase

After power-on, the M-PESTI target enters an initialization state (break event) that lasts for T_DBREAK_NS (50 µs). After that, the M-PESTI target releases the M-PESTI line (break release) and gets ready to respond to the Discovery Payload request command. The following figure shows three targets undergoing the Discovery Phase in a round-robin fashion. The first 8-bit transaction on each of the targets is the Discovery Payload Request command (0x00) sent by the initiator. The next 8-bit transactions are the actual Discovery Payload sent by the targets. The total number of bytes sent by the target may vary and is determined by the third byte of the payload. In the same figure, there are three targets with three different payload sizes: 16 bytes, 24 bytes, and 32 bytes.

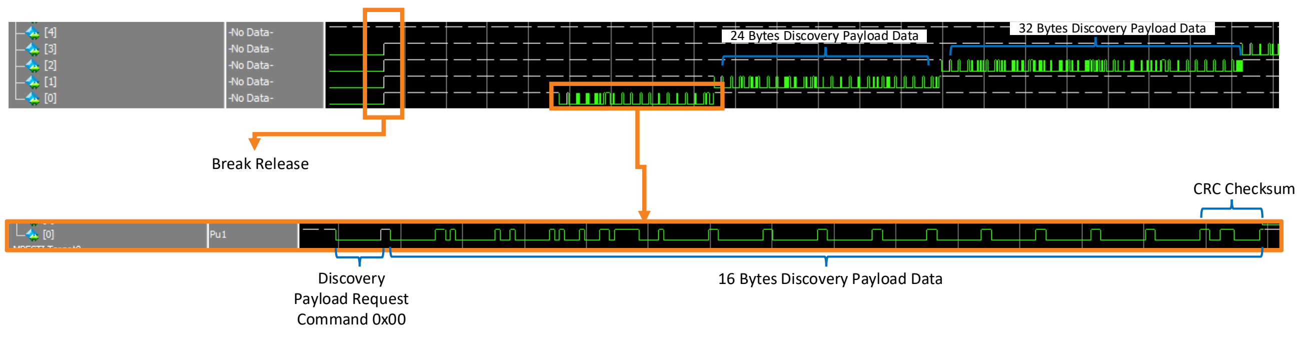

[4]
[3]
[2]
[1]
[0]
-No Data-
24 Bytes Discovery Payload Data
32 Bytes Discovery Payload Data
Break Release
CRC Checksum
Pu1
Discovery Payload Request Command 0x00
16 Bytes Discovery Payload Data

Figure 3.4. Discovery Phase

In the zoomed out image of Target0 above, the target responds with 16 discovery payload data bytes that matches the third header byte value (STATIC_PAYLOAD_SIZE[7:0] x 8) of the discovery payloads. According to the M-PESTI specification, the value of STATIC_PAYLOAD_SIZE represents the number of [discovery payload data bytes]/8. For example, STATIC_PAYLOAD_SIZE = 0x02h indicates the size as 2 × 8 = 16 discovery payload data bytes.

3.2.2. Active Phase

During the active phase, the initiator issues a VW exchange request command (0x01) to the target and starts sending the VWOUT bytes. After that, the target sends the VWIN bytes. The number of VWOUT and VWIN bytes is determined by the fourth byte of the discovery payload. Bit[3:0] of the fourth byte indicates the number of VW input bytes from the target to the initiator (NUM_VW_IN_BYTE). Bit[7:4] of the fourth byte indicates the number of VW output bytes from the initiator to the target (NUM_VW_OUT_BYTE). After finishing the VW exchange on Target0, the sequence is then repeated for the other targets in a round-robin fashion. The following figure shows the VW data exchange activity.

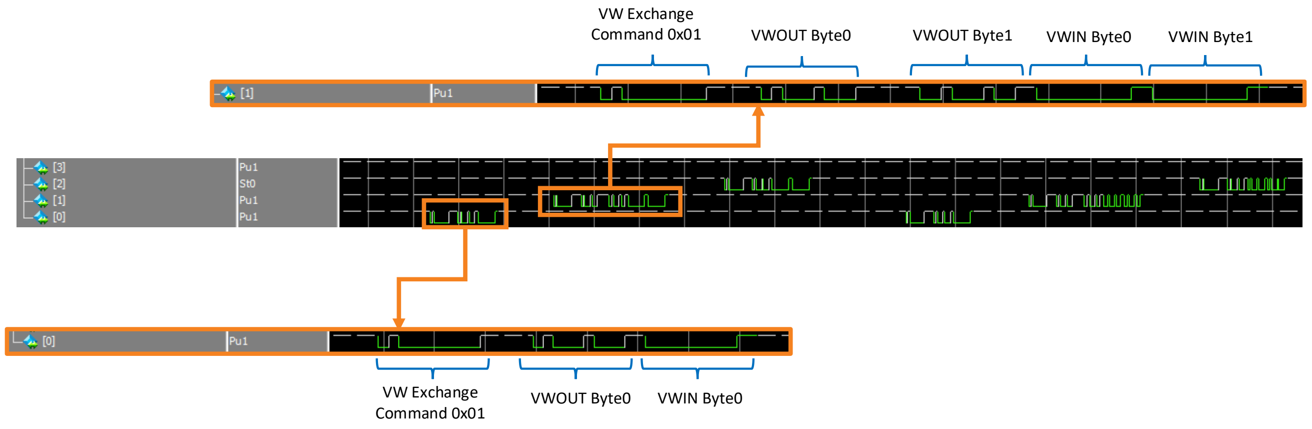


Figure 3.5. Active Phase Virtual Wire Exchange Data

3.2.3. Broadcast Transaction

The broadcast request can only be triggered during the active phase. However, if the broadcast request is triggered when VW exchange is in progress, the request is delayed until the VW exchange completes. This scenario is shown in the following figure, where the broadcast request command (0xFF) only happens at the M-PESTI line after the VW exchange completes.

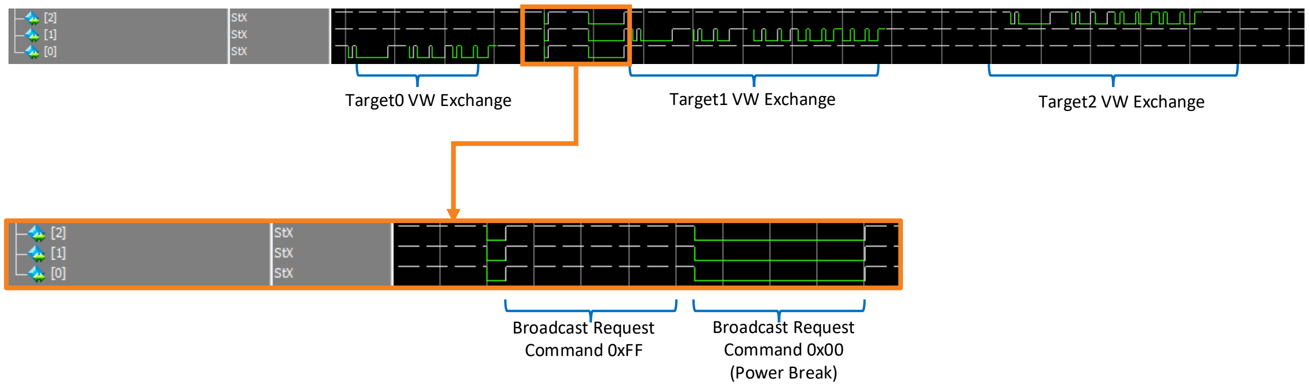


Figure 3.6. Broadcast Request

4. Pinout

4.1. M-PESTI Initiator

Table 4.1. MachXO3LF Starter Kit and MachXO5-NX Development Board Pinout

Name	Width	Direction	MachXO3LF Ball	MachXO3LF Starter Kit Component	MachXO5-NX Ball	MachXO5-NX Development Board Component	Description
Clock and Reset Interface							
clk_i	1	input	C8	—	V1	N/A	12 MHz clock input for MachXO3LF. 125 MHz clock input for MachXO5-NX.
rstn_i	1	input	B3	SW1	G20	SW5	Active low reset signal.
M-PESTI Initiator Interface							
mpesti_io[0]	1	inout	D9	J3 (29)	E4	J9 (4)	M-PESTI line. Each pin can be left floating or connected to an M-PESTI target.
mpesti_io[1]	1	inout	A10	J3 (27)	C2	J9 (6)	
mpesti_io[2]	1	inout	E6	J3 (26)	E5	J9 (8)	
mpesti_io[3]	1	inout	C5	J3 (25)	C5	J9 (10)	
mpesti_io[4]	1	inout	D7	J3 (19)	A2	J9 (12)	
mpesti_io[5]	1	inout	E7	J3 (17)	A3	J9 (14)	
mpesti_io[6]	1	inout	B9	J3 (30)	D5	J9 (16)	
mpesti_io[7]	1	inout	F7	J3 (28)	B5	J9 (18)	
mpesti_io[8]	1	inout	C4	J3 (26)	D3	J9 (3)	
mpesti_io[9]	1	inout	D6	J3 (24)	C3	J9 (5)	
mpesti_io[10]	1	inout	C7	J3 (20)	A4	J9 (7)	
Virtual Wire Interface (Optional)							
VWIN_o[0]	1	output	P16	LED D2	R3	LED D1	These optional pins can be used as outputs for VWIN values received from the targets. This can be utilized through firmware.
VWIN_o[1]	1	output	N15	LED D6	R2	LED D2	
VWIN_o[2]	1	output	L13	LED D4	R1	LED D3	
VWIN_o[3]	1	output	K11	LED D5	P7	LED D4	
VWIN_o[4]	1	output	L12	LED D6	H12	LED D5	
VWIN_o[5]	1	output	J11	LED D7	H11	LED D6	
VWIN_o[6]	1	output	J13	LED D8	G13	LED D7	
VWIN_o[7]	1	output	H11	LED D9	G12	LED D8	
VWOUT_i[0]	1	input	N2	DIP_SW1	T1	DIP_SW1	These optional pins can be used as inputs for VWOUT values to be sent by the initiator. This can be utilized through firmware.
VWOUT_i[1]	1	input	P1	DIP_SW2	T2	DIP_SW2	
VWOUT_i[2]	1	input	M3	DIP_SW3	T3	DIP_SW3	
VWOUT_i[3]	1	input	N1	DIP_SW4	T4	DIP_SW4	
VWOUT_i[4]	1	input	L7	J6 (33)	C20	J8 (4)	
VWOUT_i[5]	1	input	N6	J6 (35)	G14	J8 (6)	
VWOUT_i[6]	1	input	R4	J6 (37)	G18	J8 (8)	
VWOUT_i[7]	1	input	T3	J6 (39)	G19	J8 (10)	
UART Interface							
txd_o	1	output	C11	—	B12	—	This is directly connected to the FT2232HL chip.
rxd_i	1	input	A11	—	B11	—	This is directly connected to the FT2232HL chip.

4.2. M-PESTI Target

Table 4.2. MachXO3D Breakout Board Pinout

Name	Width	Direction	MachXO3LF Ball	MachXO3LF Starter Kit Component	Description
Clock and Reset Interface					
clk_i	1	input	C8	—	12 MHz clock input.
rstn_i	1	input	B3	—	Active low reset signal. SW1 tact switch.
M-PESTI Target Interface					
mpesti_io[0]	1	inout	D1	J3 (29)	M-PESTI line. Each pin can be left floating or connected to an M-PESTI Initiator.
mpesti_io[1]	1	inout	E1	J3 (27)	
mpesti_io[2]	1	inout	F1	J3 (26)	
mpesti_io[3]	1	inout	G1	J3 (25)	
Others					
led_o	1	output	H11	LED D9	LED blink indicator.

5. Implementing the Reference Design on Board

This reference design requires two boards: the MachXO3LF Starter Kit or MachXO5-NX Development Board for the M-PESTI Initiator SoC Design and the MachXO3D Breakout Board for the M-PESTI target module. In Figure 5.1, the board on the left is the MachXO3LF Starter Kit, acting as the M-PESTI Initiator, while the board on the right is the MachXO3D Breakout Board, acting as the emulated M-PESTI targets. Alternatively, Figure 5.2 shows a setup wherein the initiator is a MachXO5-NX Development Board. Jumper wire connections are required to bridge the two boards. Although there is only one target board, the design programmed into the board and emulates four independent target devices connected through each of the jumper wire connections. It is also good practice to have a ground connection between the two boards.

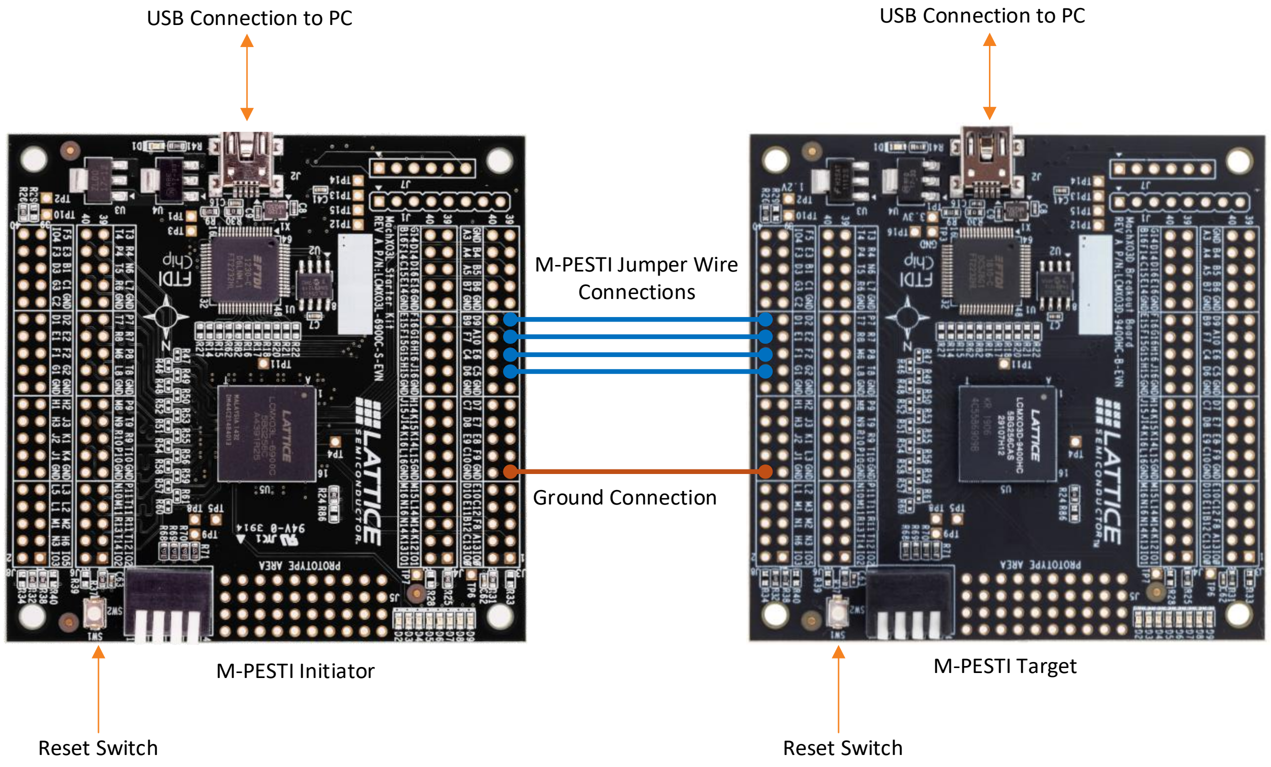


Figure 5.1. Hardware Connections for MachXO3LF and MachXO3D Devices

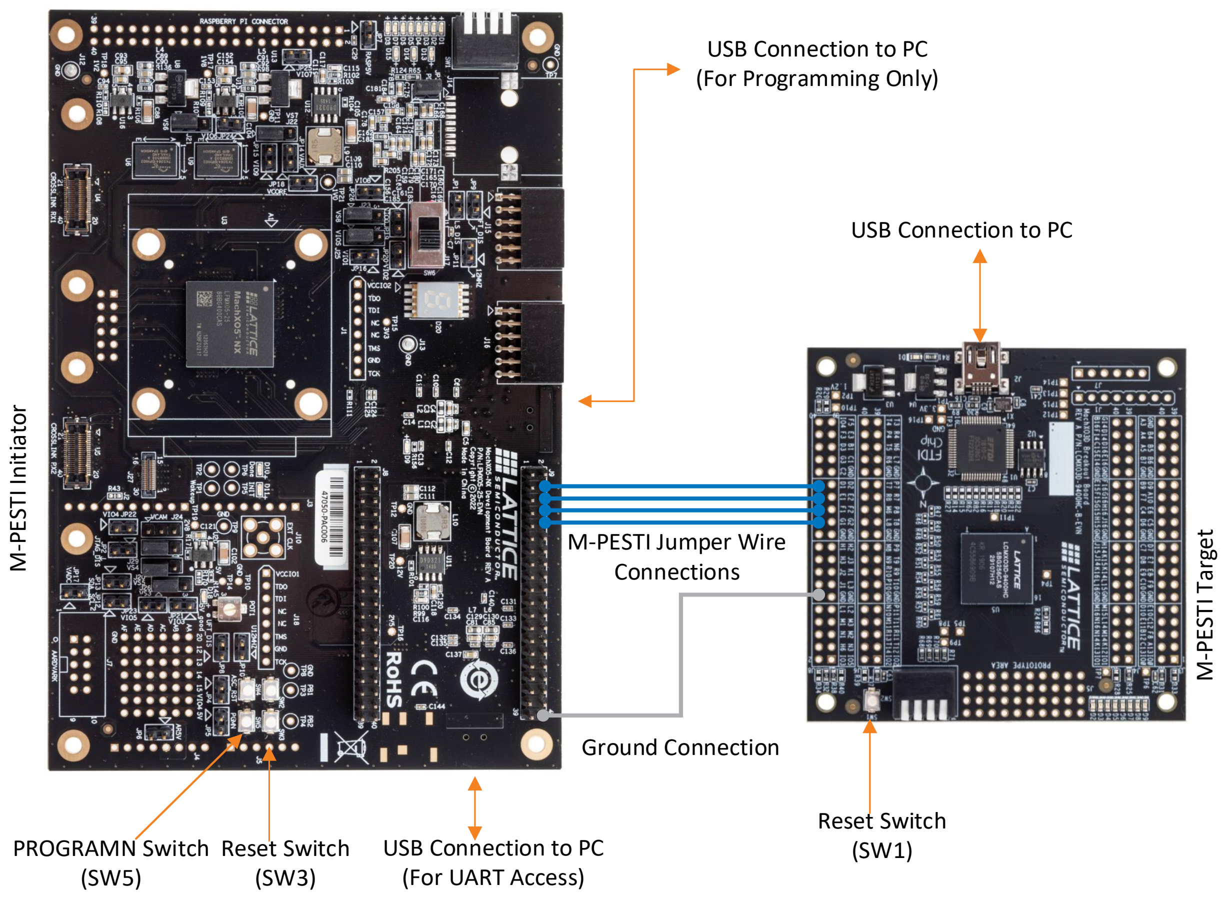


Figure 5.2. Hardware Connections for MachXO5-NX and MachXO3D Devices

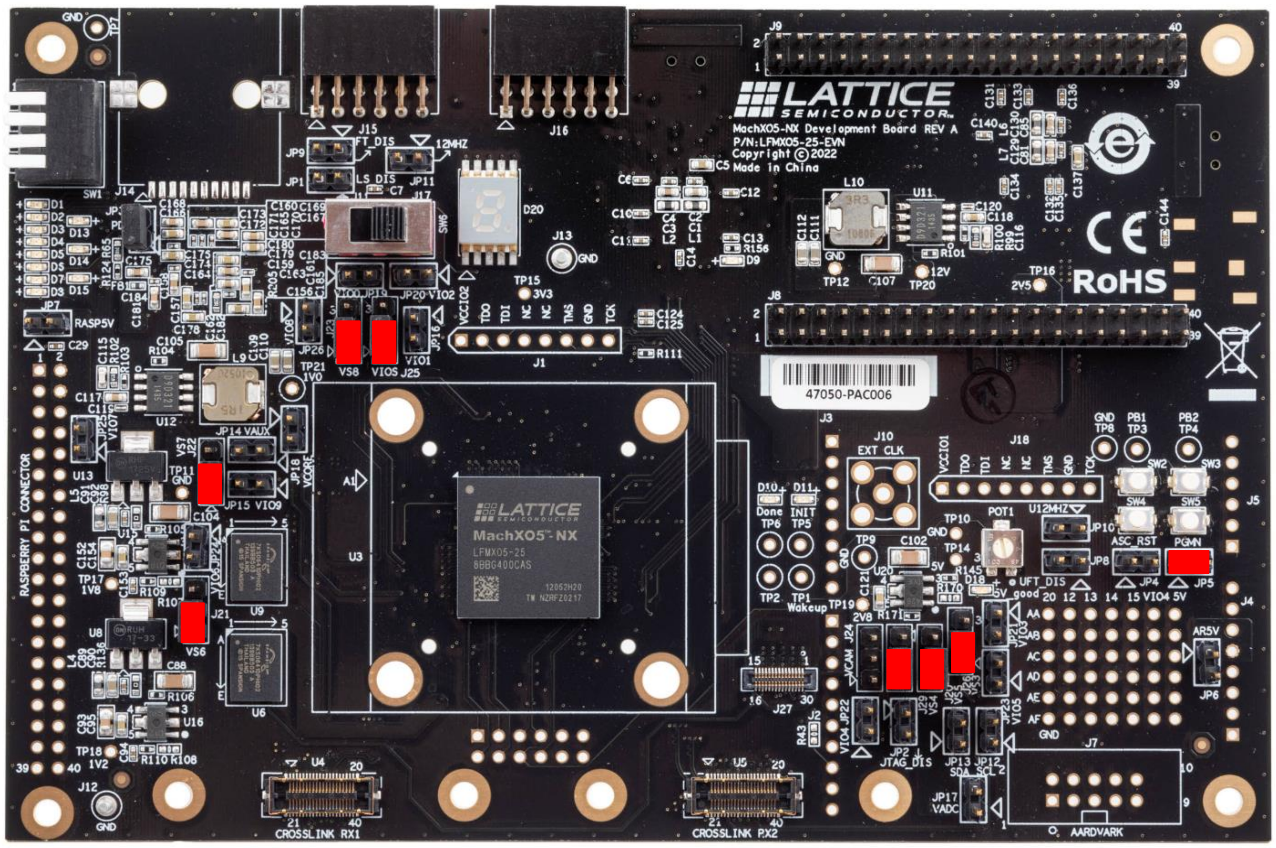
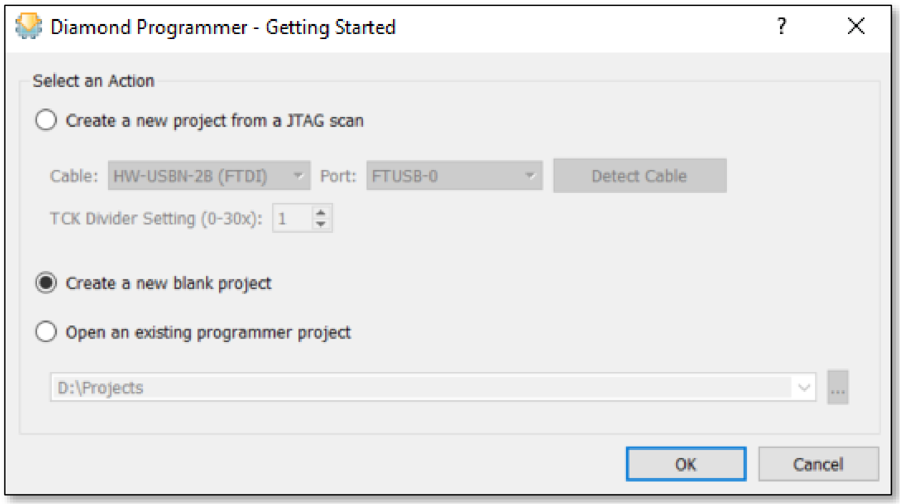


Figure 5.3. Jumper Settings for MachXO5-NX Development Board

5.1. Programming the M-PESTI Initiator SoC Design to the MachXO3LF Starter Kit

To program the M-PESTI Initiator design to the MachXO3LF Starter Kit, follow these steps:

1. Connect the MachXO3LF Starter Kit to the USB port.
2. Launch the Diamond Programmer. Select **Create a new blank project** as shown in the following figure.



Diamond Programmer - Getting Started

Select an Action

( ) Create a new project from a JTAG scan

Cable: HW-USBN-2B (FTDI) Port: FTUSB-0

TCK Divider Setting (0-30x): 1

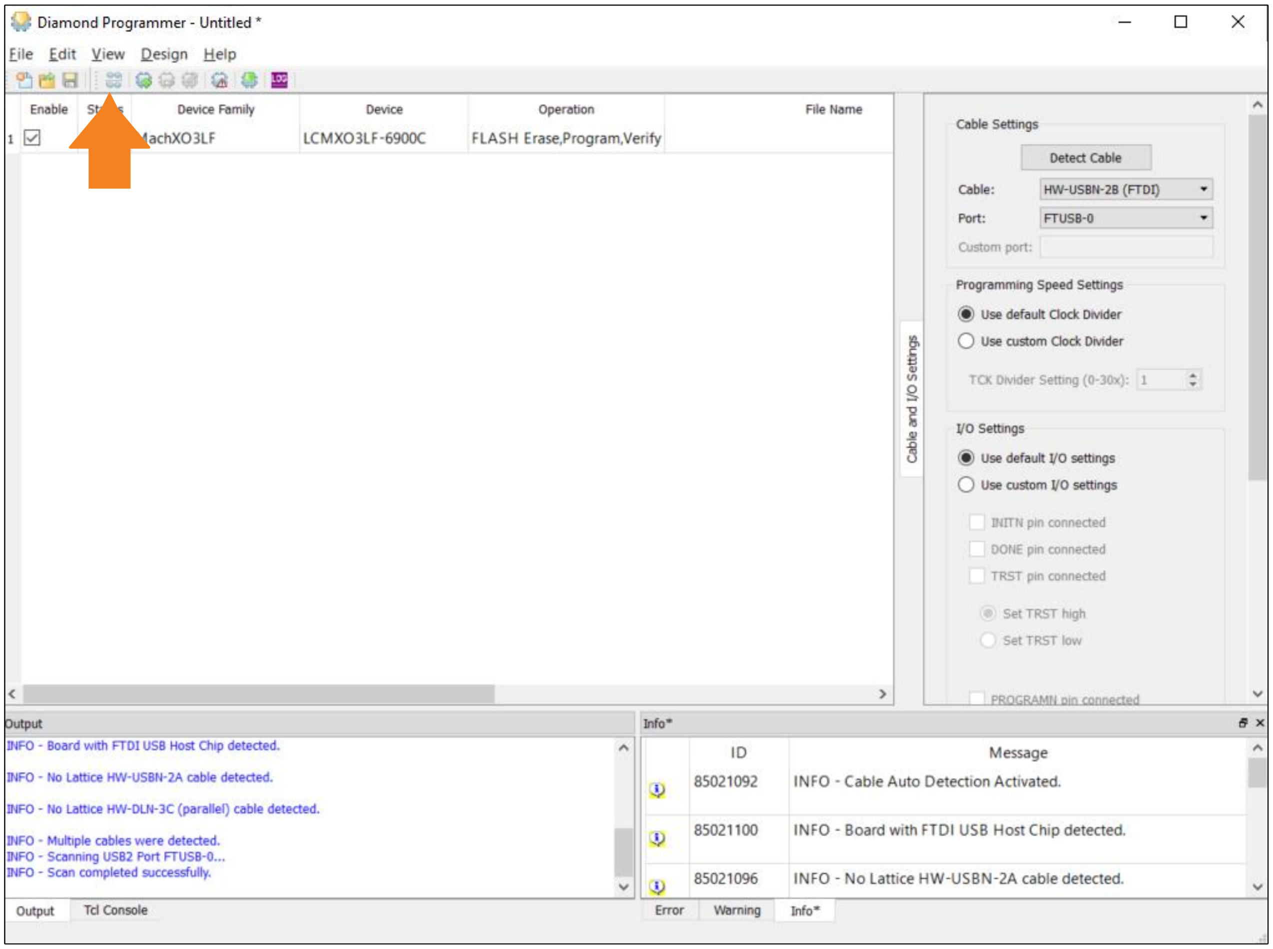
(●) Create a new blank project

( ) Open an existing programmer project

D:\Projects

Figure 5.4. Create a New Blank Project

3. Click the JTAG Scan button to detect the LCMXO3LF-6900C device found on the MachXO3LF Starter Kit board.



	Enable	Status	Device Family	Device	Operation	File Name
1	☑		MachXO3LF	LCMXO3LF-6900C	FLASH Erase,Program,Verify	

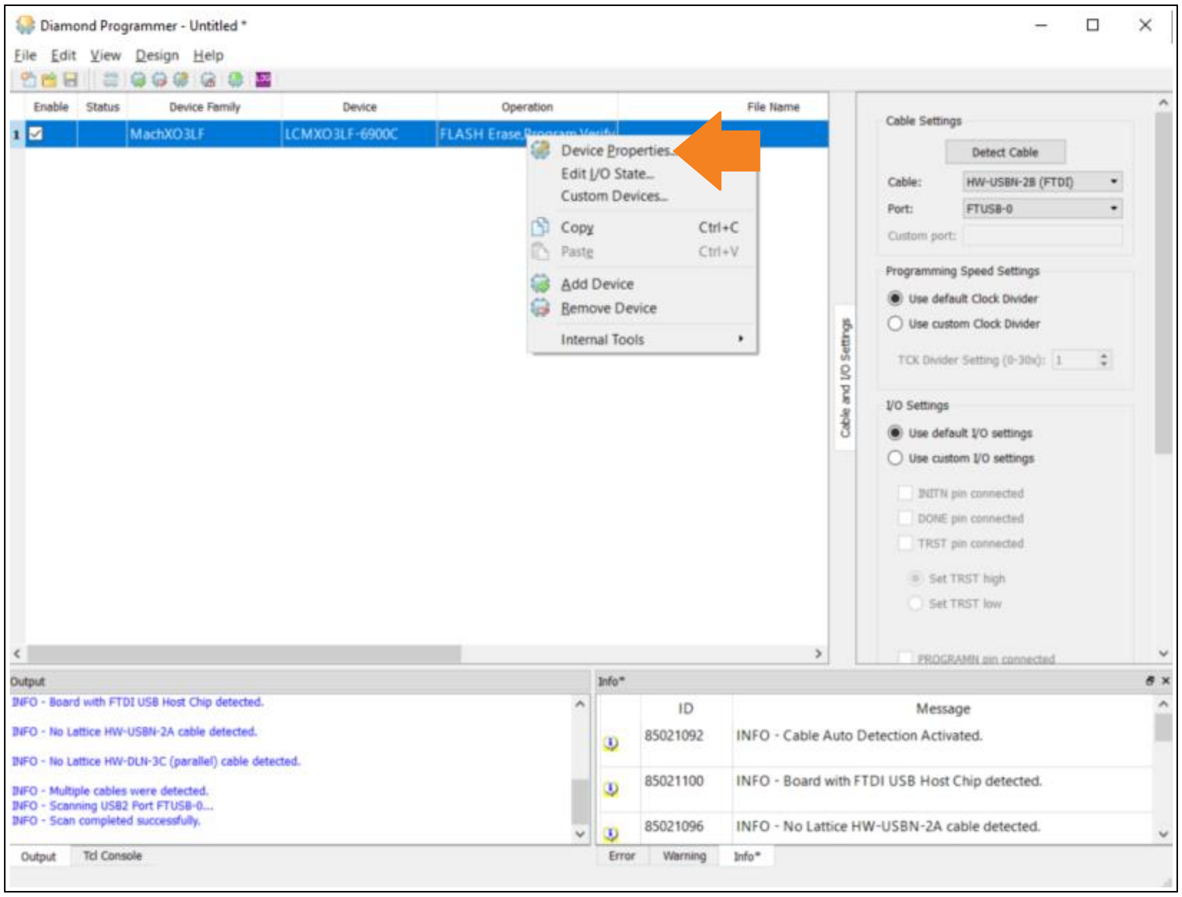
Cable: HW-USBN-2B (FTDI)
Port: FTUSB-0
Custom port:
[✓ Use default Clock Divider]
Use custom Clock Divider
TCK Divider Setting (0-30x): 1
[✓ Use default I/O settings]
Use custom I/O settings
INITN pin connected
DONE pin connected
TRST pin connected
Set TRST high
Set TRST low
PROGRAMN pin connected

```
INFO - Board with FTDI USB Host Chip detected.

INFO - No Lattice HW-USBN-2A cable detected.

INFO - No Lattice HW-DLN-3C (parallel) cable detected.

INFO - Multiple cables were detected.
INFO - Scanning USB2 Port FTUSB-0...
INFO - Scan completed successfully.
```

ID	Message
85021092	INFO - Cable Auto Detection Activated.
85021100	INFO - Board with FTDI USB Host Chip detected.
85021096	INFO - No Lattice HW-USBN-2A cable detected.

Figure 5.5. JTAG Scan

4. Right-click on the device and select **Device Properties** as shown in the following figure.



Enable	Status	Device Family	Device	Operation	File Name
1 ✓		MachXO3LF	LCMXO3LF-6900C	FLASH Erase,Program,Verify	

Cable Settings
Cable: HW-USBN-2B (FTDI)
Port: FTUSB-0
Custom port:

Programming Speed Settings
● Use default Clock Divider
○ Use custom Clock Divider
TCK Divider Setting (0-30x): 1

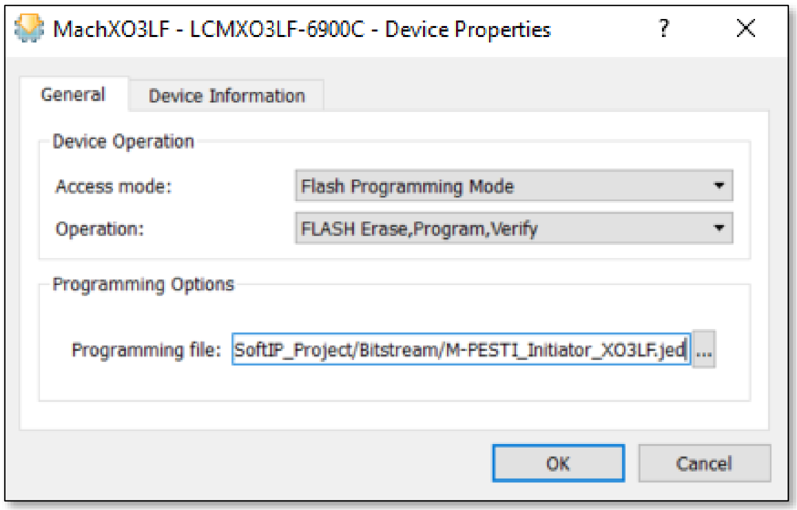
I/O Settings
● Use default I/O settings
○ Use custom I/O settings
☐ INITN pin connected
☐ DONE pin connected
☐ TRST pin connected
● Set TRST high
○ Set TRST low
☐ PROGRAMN pin connected

```
INFO - Board with FTDI USB Host Chip detected.
INFO - No Lattice HW-USBN-2A cable detected.
INFO - No Lattice HW-DLN-3C (parallel) cable detected.
INFO - Multiple cables were detected.
INFO - Scanning USB2 Port FTUSB-0...
INFO - Scan completed successfully.
```

ID	Message
85021092	INFO - Cable Auto Detection Activated.
85021100	INFO - Board with FTDI USB Host Chip detected.
85021096	INFO - No Lattice HW-USBN-2A cable detected.

Figure 5.6. Device Properties

5. Do the following settings as shown in the following figure and click **OK**.
   - Access Mode: **Flash Programming Mode**.
   - Operation: **FLASH Erase, Program, Verify**.
   - Programming File: *M-PESTI_Initiator_XO3LF.jed*

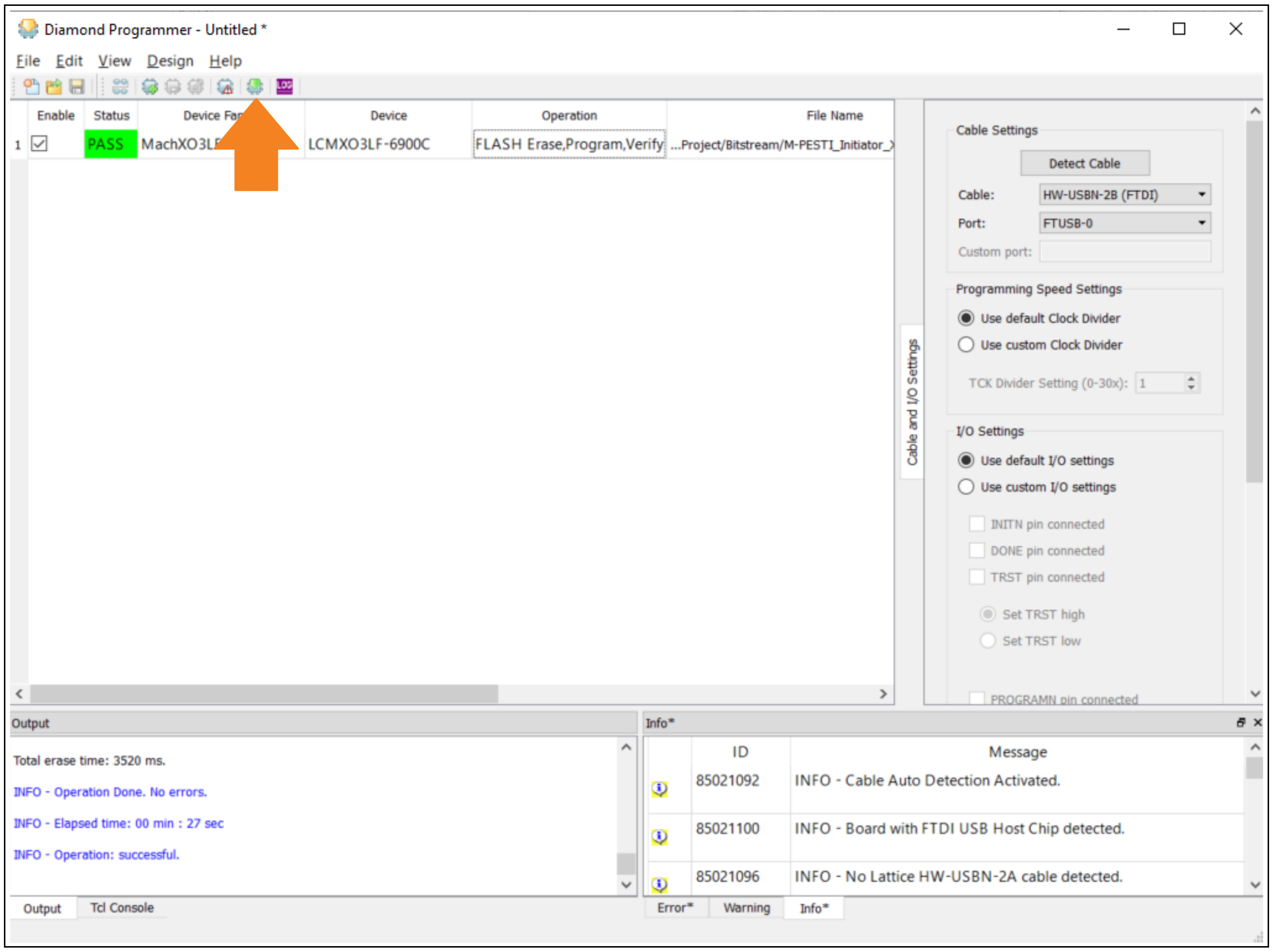


Device Operation
Access mode: Flash Programming Mode
Operation: FLASH Erase,Program,Verify

Programming Options
Programming file: SoftIP_Project/Bitstream/M-PESTI_Initiator_XO3LF.jed

Figure 5.7. MachXO3LF Device Properties Settings

6. Click the Program button as shown in the following figure.



	Enable	Status	Device Family	Device	Operation	File Name
1	✓	PASS	MachXO3LF	LCMXO3LF-6900C	FLASH Erase,Program,Verify	...Project/Bitstream/M-PESTI_Initiator_

Cable Settings
Cable: HW-USBN-2B (FTDI)
Port: FTUSB-0
Custom port:

Programming Speed Settings
● Use default Clock Divider
○ Use custom Clock Divider
TCK Divider Setting (0-30x): 1

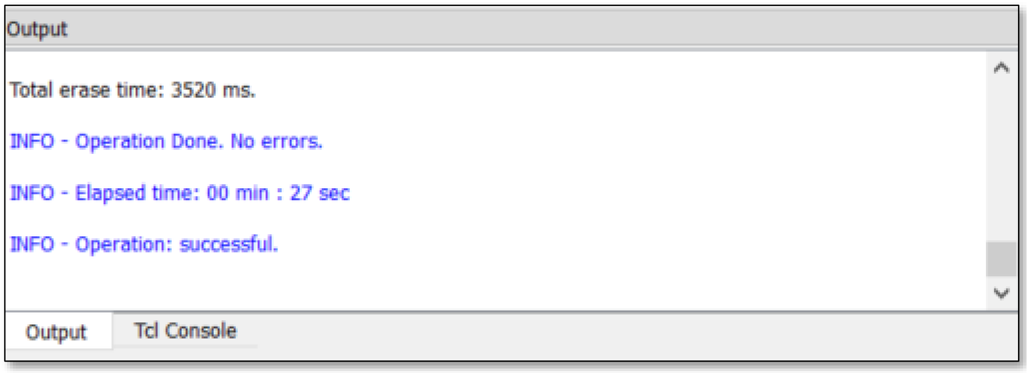
I/O Settings
● Use default I/O settings
○ Use custom I/O settings
INITN pin connected
DONE pin connected
TRST pin connected
Set TRST high
Set TRST low
PROGRAMN pin connected

```
Total erase time: 3520 ms.
INFO - Operation Done. No errors.
INFO - Elapsed time: 00 min : 27 sec
INFO - Operation: successful.
```

	ID	Message
ⓘ	85021092	INFO - Cable Auto Detection Activated.
ⓘ	85021100	INFO - Board with FTDI USB Host Chip detected.
ⓘ	85021096	INFO - No Lattice HW-USBN-2A cable detected.

Figure 5.8. Program Button

7. When programming is successful, the output console displays the message shown in the following figure.



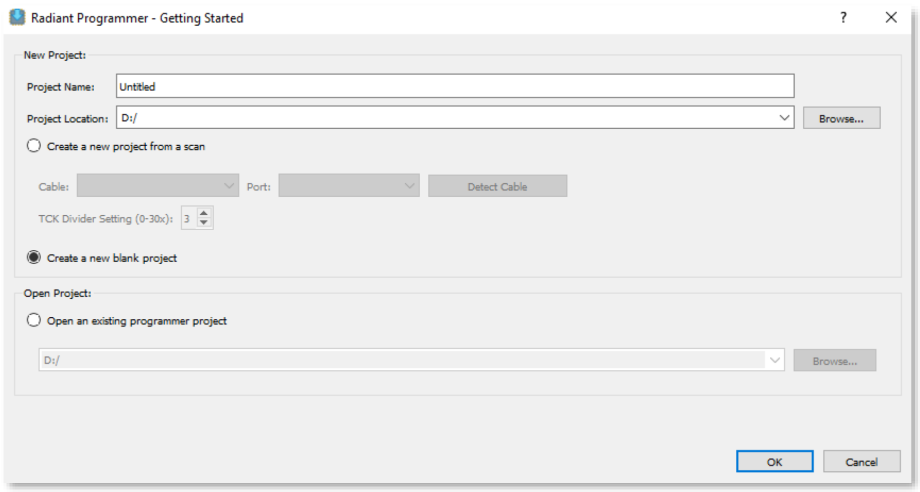
```
Total erase time: 3520 ms.
INFO - Operation Done. No errors.
INFO - Elapsed time: 00 min : 27 sec
INFO - Operation: successful.
```

Figure 5.9. Operation Successful

5.2. Programming the M-PESTI Initiator SoC Design to the MachXO5-NX Development Board

To program the M-PESTI Initiator design to the MachXO5-NX Development Board, follow these steps:

1. Connect the MachXO5-NX Development Board to the USB port.
2. Launch the Radiant Programmer. Select **Create a new blank project** as shown in the following figure.



Radiant Programmer - Getting Started

New Project:

Project Name: Untitled

Project Location: D:/

( ) Create a new project from a scan

Cable: Port: Detect Cable

TCK Divider Setting (0-30x): 3

(●) Create a new blank project

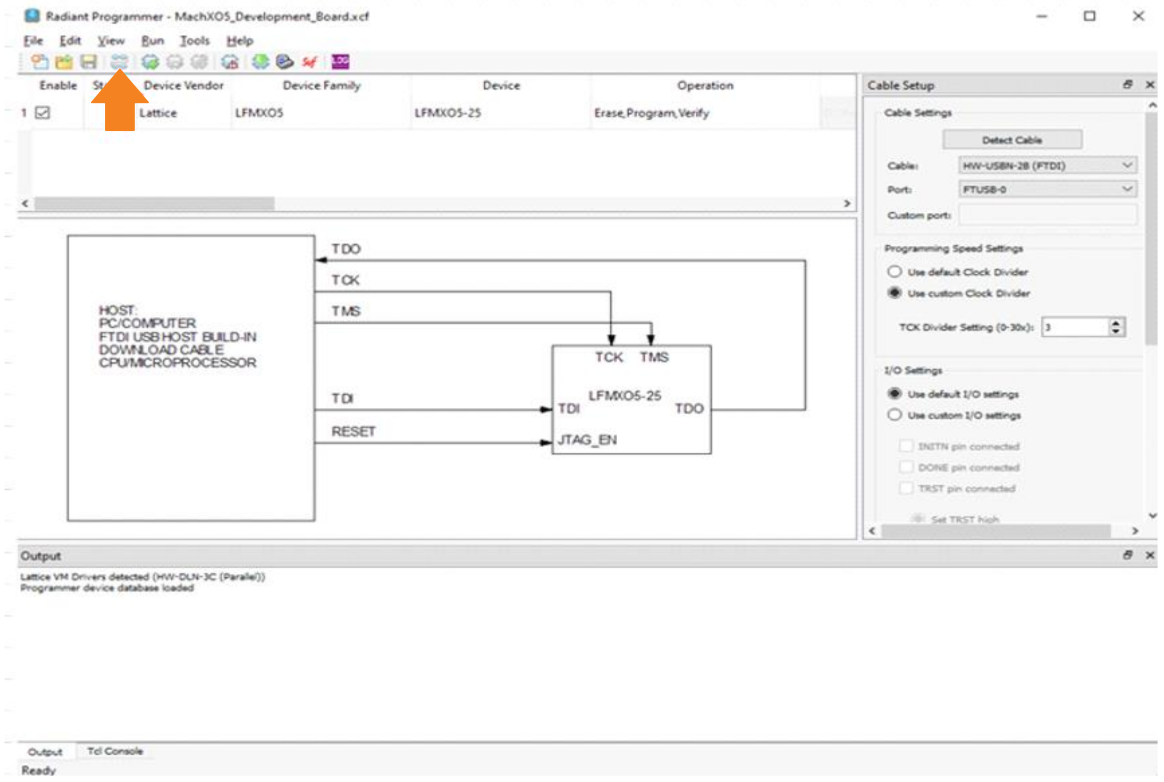
Open Project:

( ) Open an existing programmer project

D:/

Figure 5.10. Create a New Blank Project

3. Click the JTAG Scan button to detect the LFMXO5-25 device found on the MachXO5-NX Development Board.



Radiant Programmer - MachXO5_Development_Board.xcf

	Enable	St	Device Vendor	Device Family	Device	Operation
1	☑		Lattice	LFMXO5	LFMXO5-25	Erase,Program,Verify

Cable Setup

Cable Settings

Cable: HW-USBN-2B (FTDI)

Port: FTUSB-0

Custom port:

Programming Speed Settings

( ) Use default Clock Divider

(●) Use custom Clock Divider

TCK Divider Setting (0-30x): 3

I/O Settings

(●) Use default I/O settings

( ) Use custom I/O settings

☐ INITN pin connected

☐ DONE pin connected

☐ TRST pin connected

Set TRST high

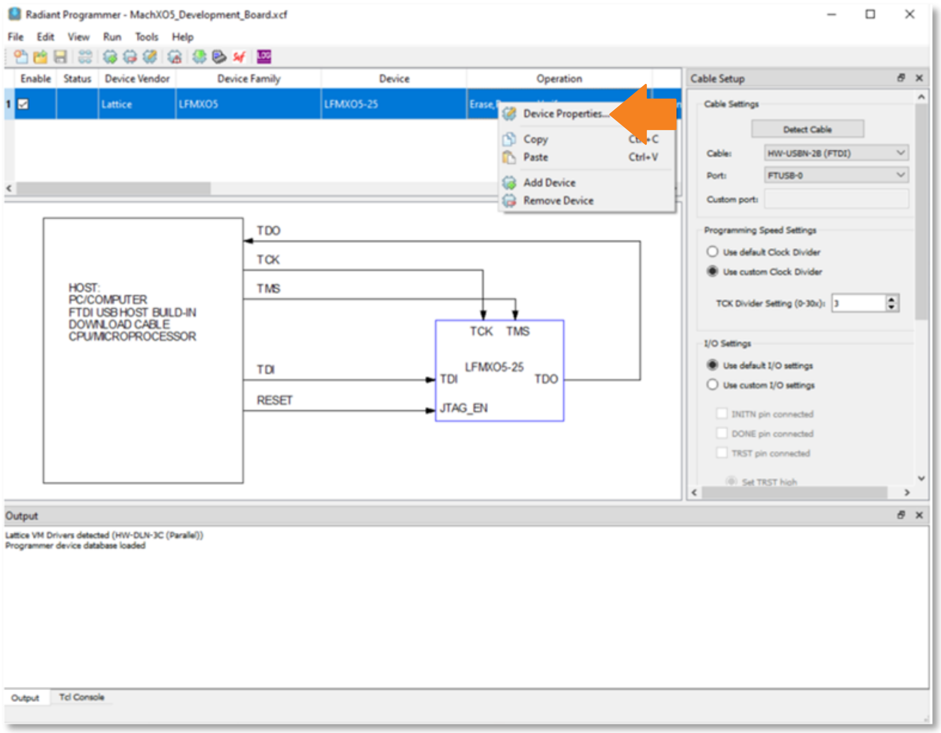
Output

```
Lattice VM Drivers detected (HW-DLN-3C (Parallel))
Programmer device database loaded
```

Ready

Figure 5.11. JTAG Scan

4. Right-click on the device and select **Device Properties** as shown in the following figure.



	Enable	Status	Device Vendor	Device Family	Device	Operation
1	✓		Lattice	LFMXO5	LFMXO5-25	Erase,

Cable Settings
Cable: HW-USBN-2B (FTDI)
Port: FTUSB-0
Custom port:

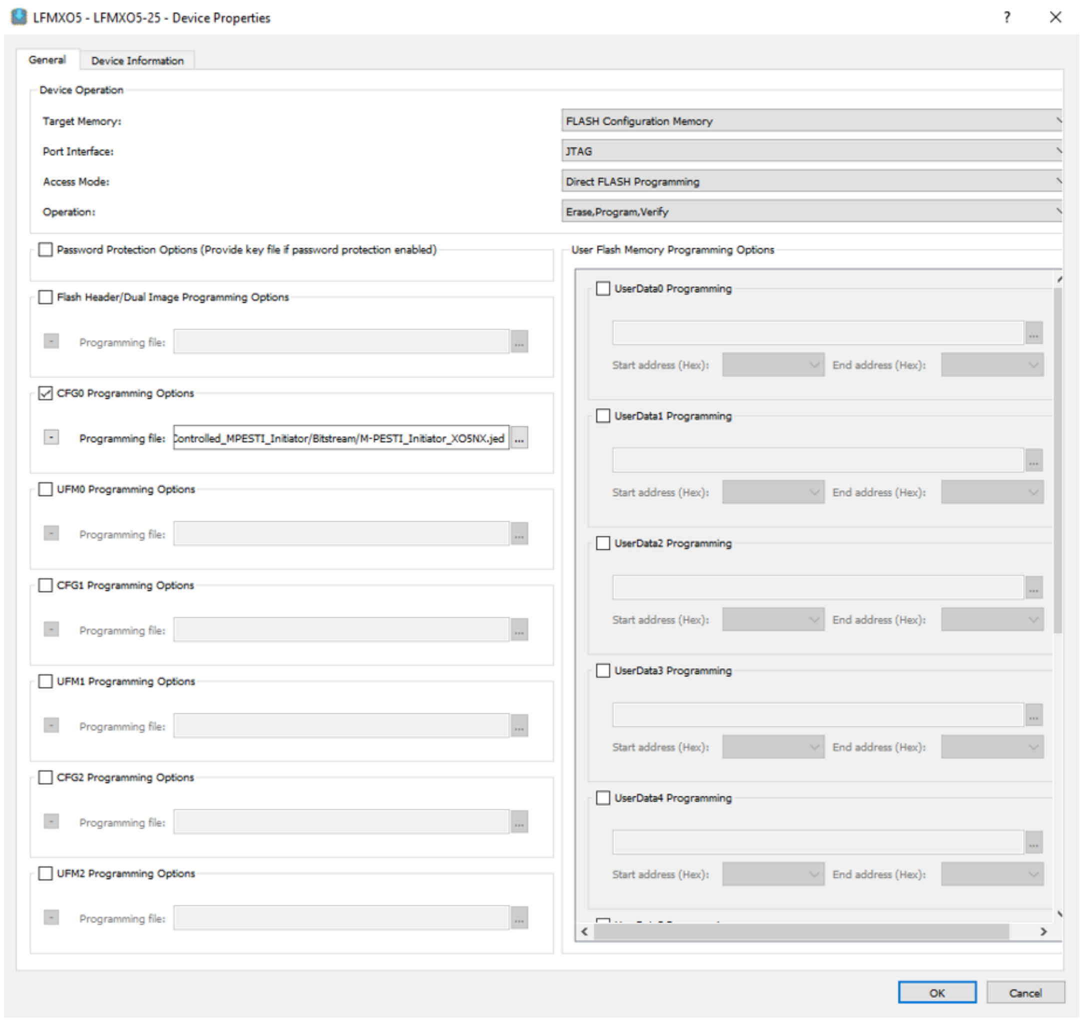
Programming Speed Settings
○ Use default Clock Divider
● Use custom Clock Divider
TCK Divider Setting (0-30x): 3

I/O Settings
● Use default I/O settings
○ Use custom I/O settings
☐ INITN pin connected
☐ DONE pin connected
☐ TRST pin connected
Set TRST high

```
Lattice VM Drivers detected (HW-DLN-3C (Parallel))
Programmer device database loaded
```

Figure 5.12. Device Properties

5. Do the following settings as shown in the following figure and click **OK**.
    - Target Memory: **Flash Configuration Memory**
    - Port Interface: **JTAG**
    - Access Mode: **Direct Flash Programming**
    - Operation: **Erase, Program, Verify**
    - CFG0 Programming Options: Checked
    - Programming File: *M-PESTI_Initiator_XO5NX.jed*


LFMXO5 - LFMXO5-25 - Device Properties

General | Device Information

Device Operation

Target Memory:	FLASH Configuration Memory
Port Interface:	JTAG
Access Mode:	Direct FLASH Programming
Operation:	Erase,Program,Verify

☐ Password Protection Options (Provide key file if password protection enabled)

☐ Flash Header/Dual Image Programming Options

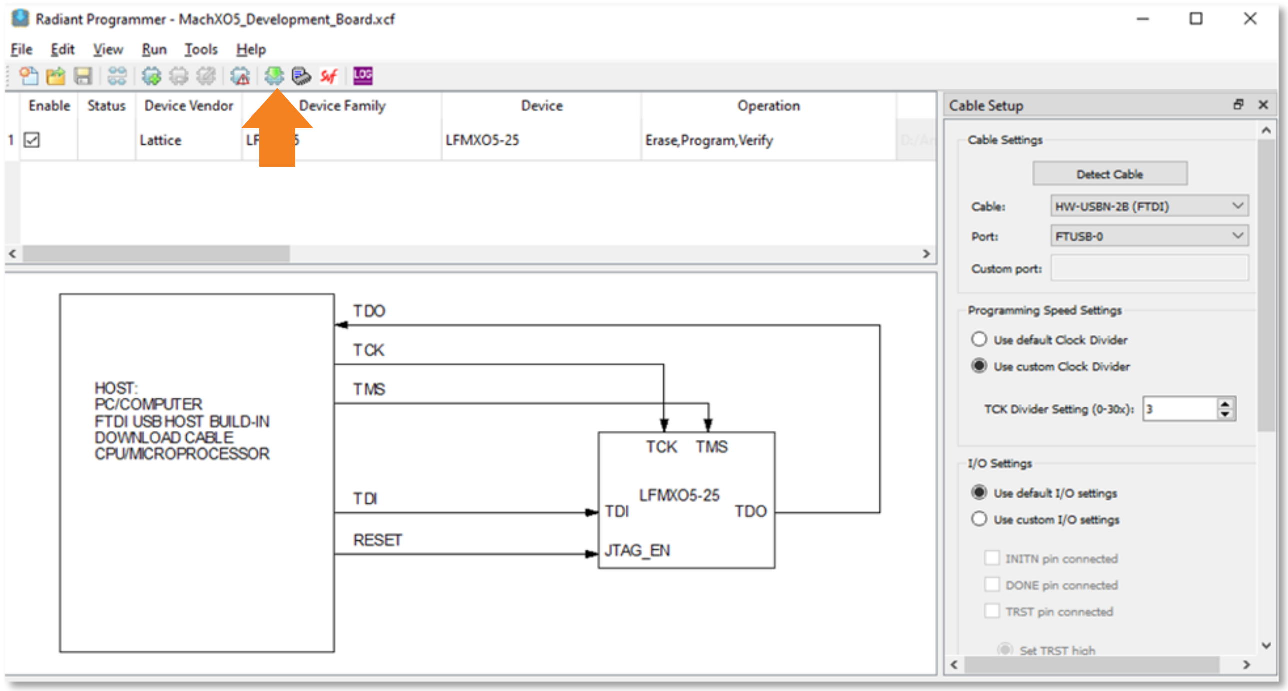
☑ CFG0 Programming Options
Programming file: ontrolled_MPESTI_Initiator/Bitstream/M-PESTI_Initiator_XO5NX.jed

☐ UFM0 Programming Options
☐ CFG1 Programming Options
☐ UFM1 Programming Options
☐ CFG2 Programming Options
☐ UFM2 Programming Options

User Flash Memory Programming Options
☐ UserData0 Programming
☐ UserData1 Programming
☐ UserData2 Programming
☐ UserData3 Programming
☐ UserData4 Programming

Figure 5.13. MachXO5-NX Device Properties Settings

6. Click the Program button as shown in the following figure.



	Enable	Status	Device Vendor	Device Family	Device	Operation
1	☑		Lattice	LF...5	LFMXO5-25	Erase,Program,Verify

Cable Setup

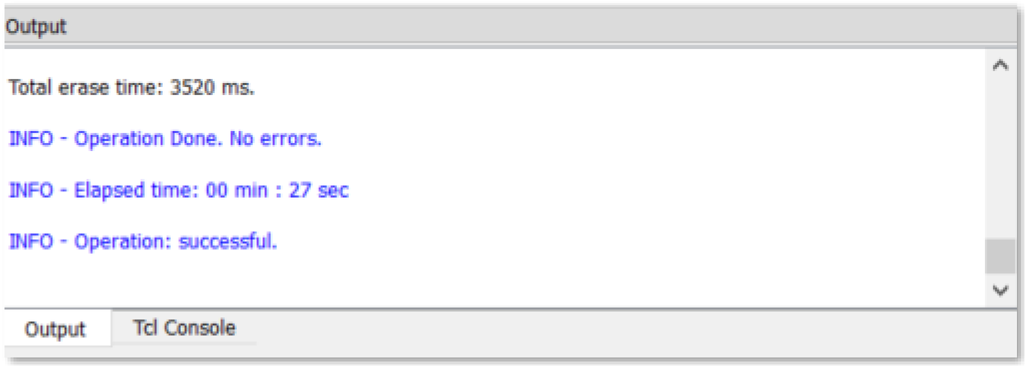
Cable Settings
Cable: HW-USBN-2B (FTDI)
Port: FTUSB-0
Custom port:

Programming Speed Settings
○ Use default Clock Divider
● Use custom Clock Divider
TCK Divider Setting (0-30x): 3

I/O Settings
● Use default I/O settings
○ Use custom I/O settings
☐ INITN pin connected
☐ DONE pin connected
☐ TRST pin connected
Set TRST high

Figure 5.14. Program Button

7. When programming is successful, the output console displays the message shown in the following figure.



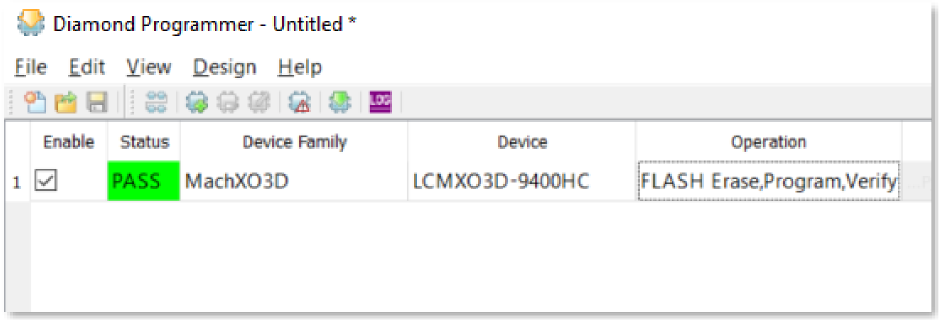
```
Total erase time: 3520 ms.
INFO - Operation Done. No errors.
INFO - Elapsed time: 00 min : 27 sec
INFO - Operation: successful.
```

Figure 5.15. Operation Successful

5.3. Programming the M-PESTI Target Module to the MachXO3D Starter Kit

To program the M-PESTI target module to the MachXO3D Starter Kit, follow these steps:

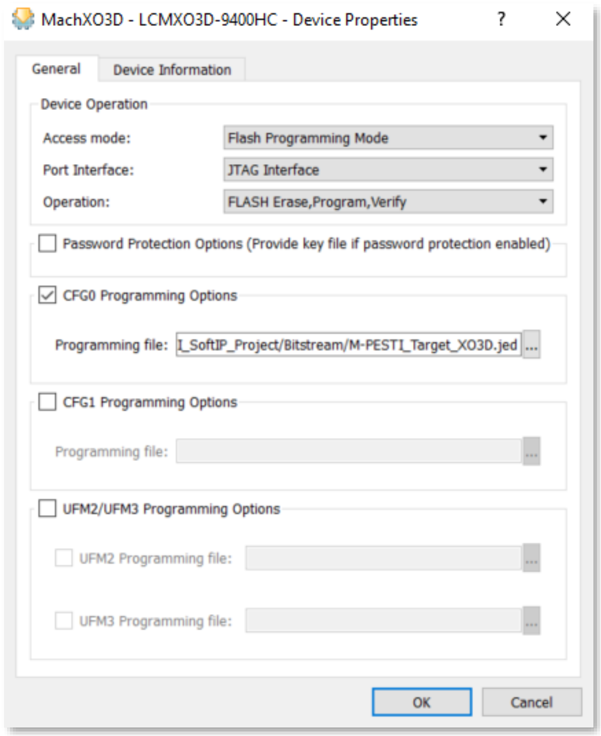
1. Connect the MachXO3D Starter Kit to the USB port.
2. Launch the Diamond Programmer **with Create a new blank project**.
3. Click the **JTAG Scan** button to detect the LCMXO3D-9400HC device found on the MachXO3D Starter Kit board.



	Enable	Status	Device Family	Device	Operation
1	✓	PASS	MachXO3D	LCMXO3D-9400HC	FLASH Erase,Program,Verify

Figure 5.16. LCMXO3D-9400HC Device Selection

4. Do the following settings and click **OK**:
   - Access Mode: **Flash Programming Mode**
   - Port Interface: **JTAG Interface**
   - Operation: **FLASH Erase, Program, Verify**
   - CFG0 Programming Options: Checked
   - Programming File: *M-PESTI_Target_XO3D.jed*



Access mode: Flash Programming Mode
Port Interface: JTAG Interface
Operation: FLASH Erase,Program,Verify
☐ Password Protection Options (Provide key file if password protection enabled)
☑ CFG0 Programming Options
Programming file: I_SoftIP_Project/Bitstream/M-PESTI_Target_XO3D.jed
☐ CFG1 Programming Options
Programming file:
☐ UFM2/UFM3 Programming Options
☐ UFM2 Programming file:
☐ UFM3 Programming file:

Figure 5.17. MachXO3D Device Properties Settings

6. IP Demonstration

6.1. Enabling the UART Controller

6.1.1. Board Rework

To enable UART control on the M-PESTI Initiator IP, R14 and R15 of the MachXO3LF Starter Kit must be soldered as shown in the following figure.

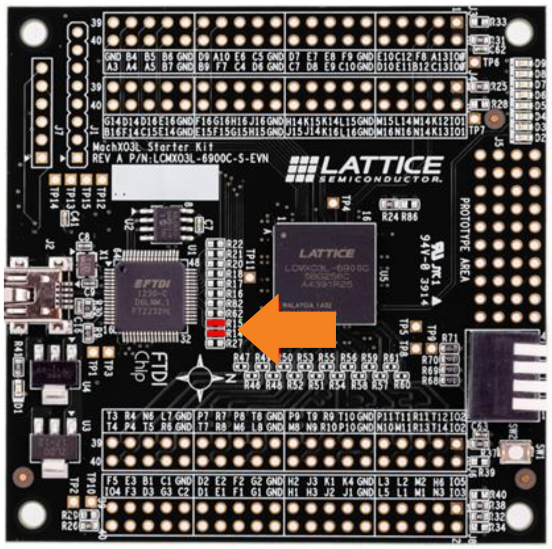
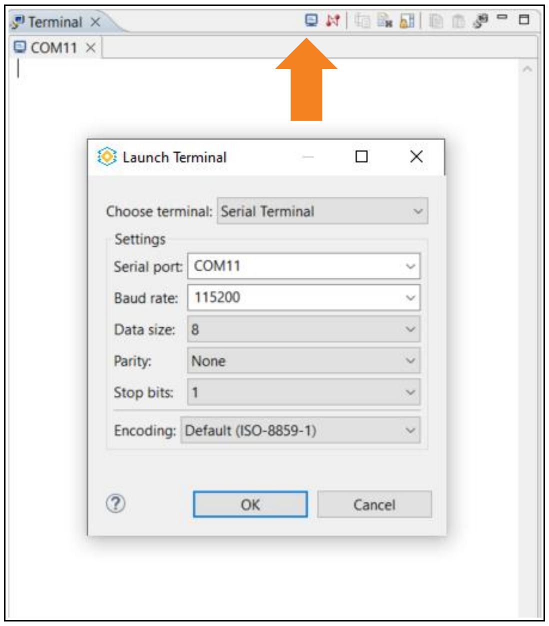


Figure 6.1. Board Rework

6.1.2. COM Port Selection

Any terminal program such as PuTTY or Terra Term can be used. In this reference design, the built-in terminal of the Lattice Propel SDK is used. To open this, go to **Window > Show View > Terminal** and click the Open a Terminal button shown in the following figure and apply the following settings:

- Serial port: (Select the bigger number. In the following figure, COM10 and COM11 are detected so COM11 is chosen.)
- Baud rate: 115,200
- Data size: 8
- Parity: None
- Stop bits: 1

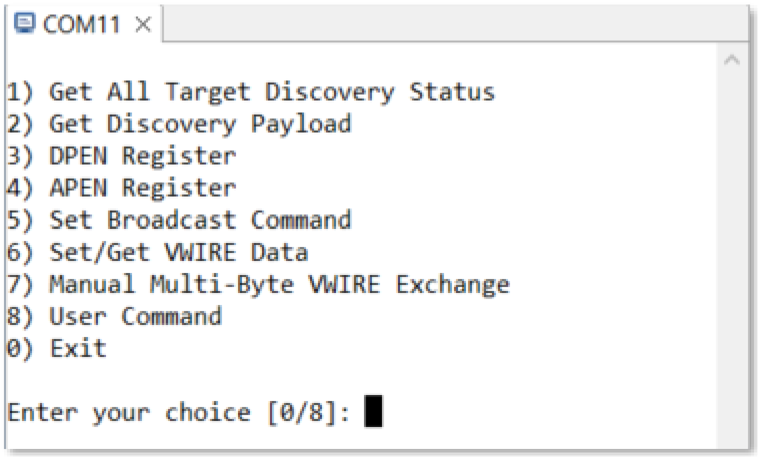


Choose terminal: Serial Terminal
Settings
Serial port: COM11
Baud rate: 115200
Data size: 8
Parity: None
Stop bits: 1
Encoding: Default (ISO-8859-1)

Figure 6.2. Launch Terminal

6.2. Usage Example

When the UART connection is established, a selection menu appears on the screen as shown in the following figure. This menu allows you to access various M-PESTI functionalities by selecting a number and following the subsequent instructions for the chosen item. If this menu does not show up yet, you can press the reset switch of the M-PESTI Initiator. Additionally, for the MachXO5-NX Initiator device, you may need to press the PROGRAMN switch or do a power cycle before you can press the reset switch.



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]:
```

Figure 6.3. Test Suite Menu Selection

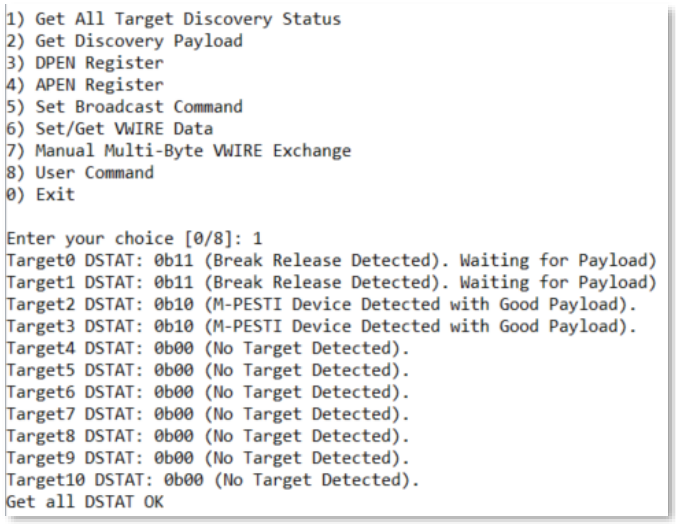
6.2.1. Get All Target Discovery Status

This selection allows printing of the DSTAT register of each of the M-PESTI port. It can be any of the following statuses:

- 0b00 - No Target Detected
- 0b01 - Simple Presence
- 0b10 - M-PESTI Device Detected with Good Payload
- 0b11 - Break Release Detected Waiting for Payload

As shown in the following figure, these statuses are observed:

- Target0 and Target1 already provided a Break Release. However, the payload has not been sent yet.
- Target2 and Target3 already provided Good Payloads.
- Target4 to Target10 shows that there is no device connected to them.

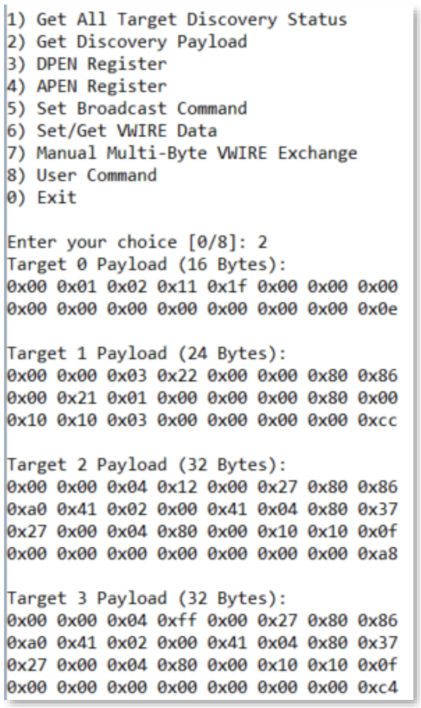


```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 1
Target0 DSTAT: 0b11 (Break Release Detected). Waiting for Payload)
Target1 DSTAT: 0b11 (Break Release Detected). Waiting for Payload)
Target2 DSTAT: 0b10 (M-PESTI Device Detected with Good Payload).
Target3 DSTAT: 0b10 (M-PESTI Device Detected with Good Payload).
Target4 DSTAT: 0b00 (No Target Detected).
Target5 DSTAT: 0b00 (No Target Detected).
Target6 DSTAT: 0b00 (No Target Detected).
Target7 DSTAT: 0b00 (No Target Detected).
Target8 DSTAT: 0b00 (No Target Detected).
Target9 DSTAT: 0b00 (No Target Detected).
Target10 DSTAT: 0b00 (No Target Detected).
Get all DSTAT OK
```

Figure 6.4. DSTAT Values

6.2.2. Get Discovery Payload

As shown in the following figure, when payloads are successfully sent by the targets, a summary is printed on the screen, including the payload size and the actual payload read.

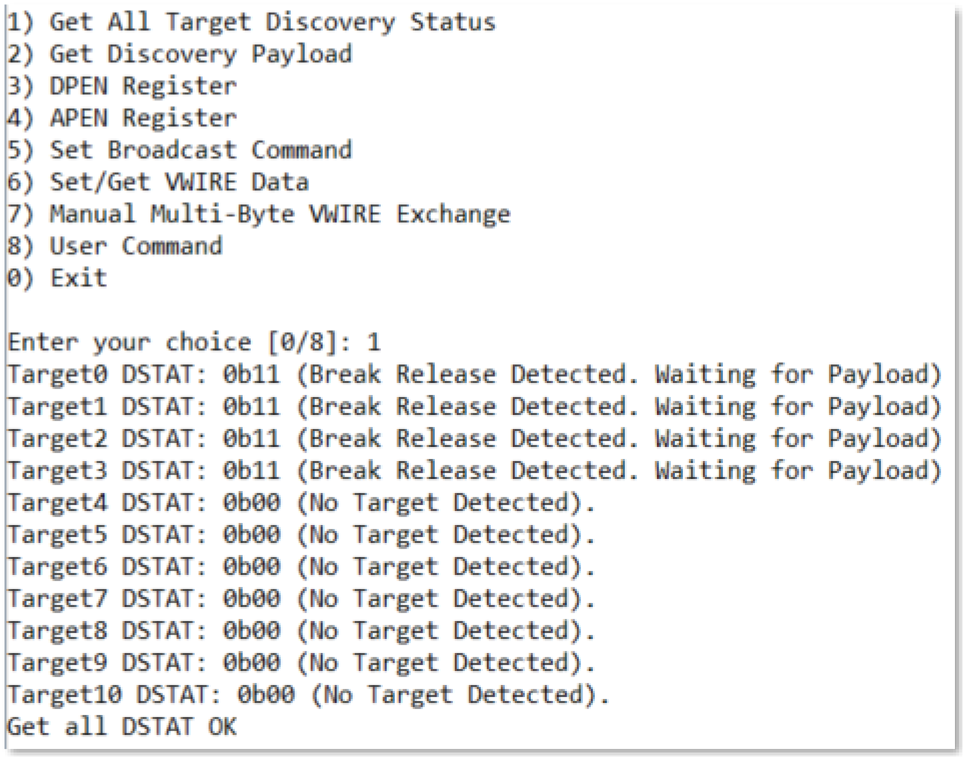


```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 2
Target 0 Payload (16 Bytes):
0x00 0x01 0x02 0x11 0x1f 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x0e

Target 1 Payload (24 Bytes):
0x00 0x00 0x03 0x22 0x00 0x00 0x80 0x86
0x00 0x21 0x01 0x00 0x00 0x00 0x80 0x00
0x10 0x10 0x03 0x00 0x00 0x00 0x00 0xcc

Target 2 Payload (32 Bytes):
0x00 0x00 0x04 0x12 0x00 0x27 0x80 0x86
0xa0 0x41 0x02 0x00 0x41 0x04 0x80 0x37
0x27 0x00 0x04 0x80 0x00 0x10 0x10 0x0f
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0xa8

Target 3 Payload (32 Bytes):
0x00 0x00 0x04 0xff 0x00 0x27 0x80 0x86
0xa0 0x41 0x02 0x00 0x41 0x04 0x80 0x37
0x27 0x00 0x04 0x80 0x00 0x10 0x10 0x0f
0x00 0x00 0x00 0x00 0x00 0x00 0x00 0xc4
```

Figure 6.5. Payload Values

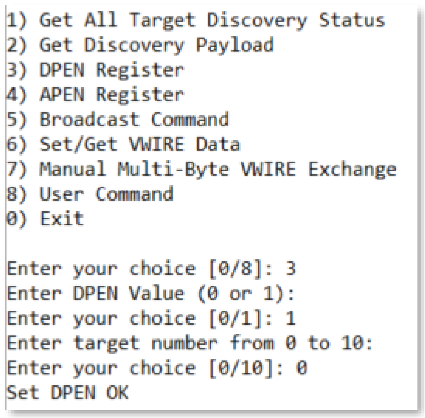
6.2.3. DPEN Register

This menu selection allows individual control of DPEN registers per target and can be enabled at any time. You can check the Discovery Status to determine whether Break Release has been achieved as shown in Figure 6.6. If this has not yet been achieved, you may need to press the reset switch on the target to provide the Break Release again. An example of how to enable DPEN on a specific target is shown in Figure 6.7. After setting DPEN to 1 and selecting Target0, the actual payload is captured using a logic analyzer, as shown in Figure 6.8.



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 1
Target0 DSTAT: 0b11 (Break Release Detected. Waiting for Payload)
Target1 DSTAT: 0b11 (Break Release Detected. Waiting for Payload)
Target2 DSTAT: 0b11 (Break Release Detected. Waiting for Payload)
Target3 DSTAT: 0b11 (Break Release Detected. Waiting for Payload)
Target4 DSTAT: 0b00 (No Target Detected).
Target5 DSTAT: 0b00 (No Target Detected).
Target6 DSTAT: 0b00 (No Target Detected).
Target7 DSTAT: 0b00 (No Target Detected).
Target8 DSTAT: 0b00 (No Target Detected).
Target9 DSTAT: 0b00 (No Target Detected).
Target10 DSTAT: 0b00 (No Target Detected).
Get all DSTAT OK
```

Figure 6.6. Break Release Detected



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 3
Enter DPEN Value (0 or 1):
Enter your choice [0/1]: 1
Enter target number from 0 to 10:
Enter your choice [0/10]: 0
Set DPEN OK
```

Figure 6.7. DPEN Enabled for Target0

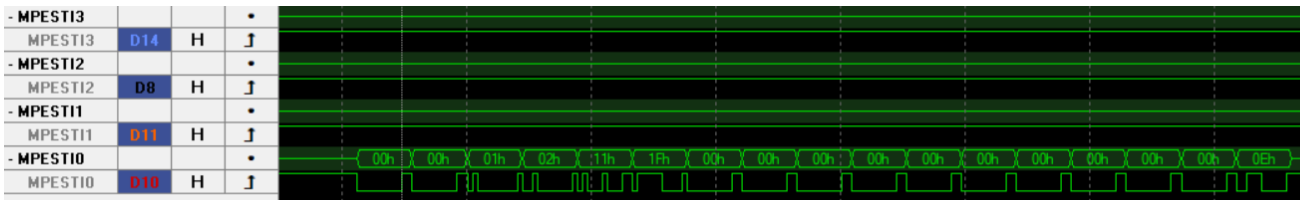


Figure 6.8. Logic Analyzer Capture of Target0 Payload

6.2.4. APEN Register

This menu selection allows individual control of the APEN register. In the current IP version 1.1.0, it is not recommended to set APEN=1 unless all the targets connected to the M-PESTI initiator only has a single byte VWIN and VWOUT bytes.

6.2.5. Set Broadcast Command

After setting the desired Broadcast Command using this menu selection, the specific broadcast command can be sent by pushing down the switch shown in the following figures during Multi-Byte VWIRE Exchange.

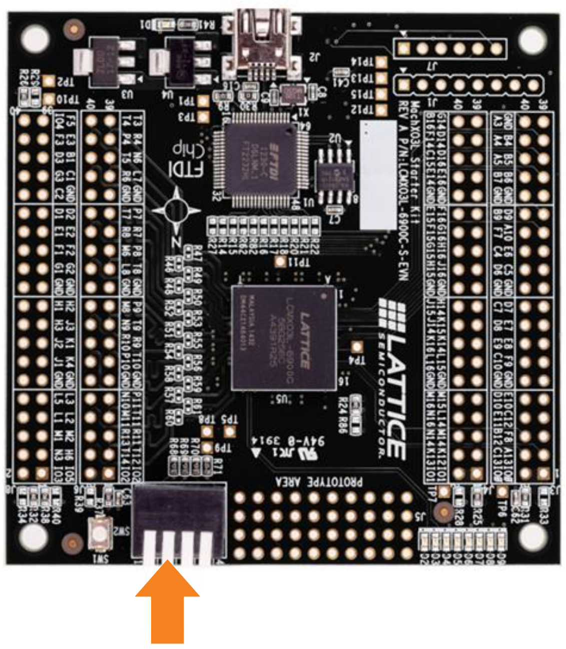


Figure 6.9. Switch to Send Broadcast Command Using the MachXO3LF Initiator

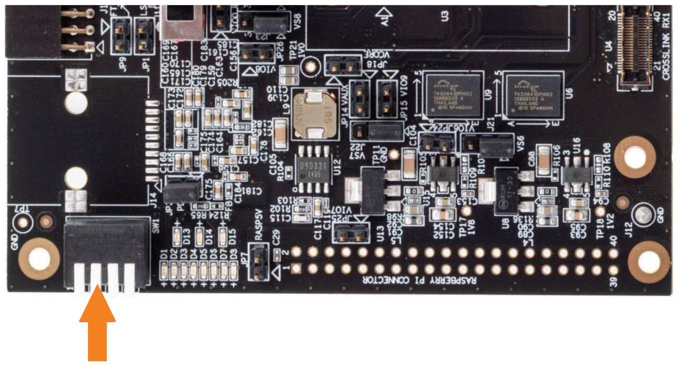
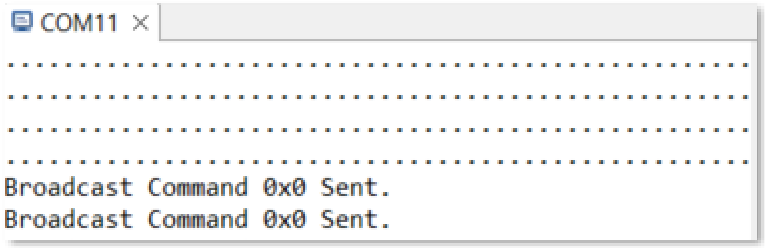


Figure 6.10. Switch to Send Broadcast Command Using the MachXO5-NX Initiator

This can be confirmed by either checking the message shown in Figure 6.11, or by checking the logic analyzer waveform as shown in Figure 6.12.



```
...............................................................
...............................................................
...............................................................
...............................................................
Broadcast Command 0x0 Sent.
Broadcast Command 0x0 Sent.
```

Figure 6.11. Broadcast Command Message

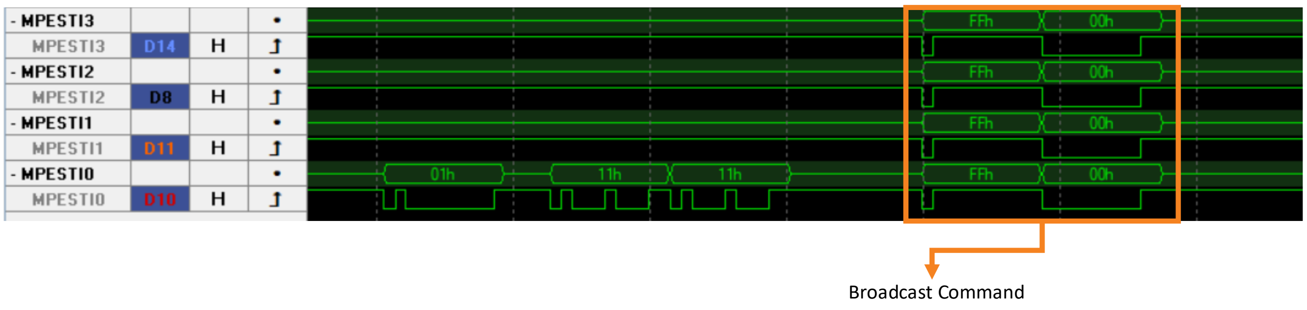


Figure 6.12. Broadcast Command Waveform

6.2.6. SET/GET VWIRE Data

This menu selection allows you to set the VWOUT data when APEN is set to 1. As shown in Figure 6.13, the default VWOUT value of the SoC design is 0x00. After setting the VWOUT to the decimal value 170 (equivalent to the hexadecimal value 0xAA) as shown in Figure 6.14, the value 170 is also received as the VWIN data because of the target internal loopback mechanism. Figure 6.15 shows the actual waveform after modifying the VWOUT data.

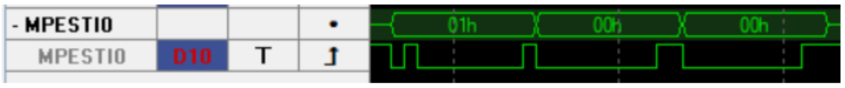
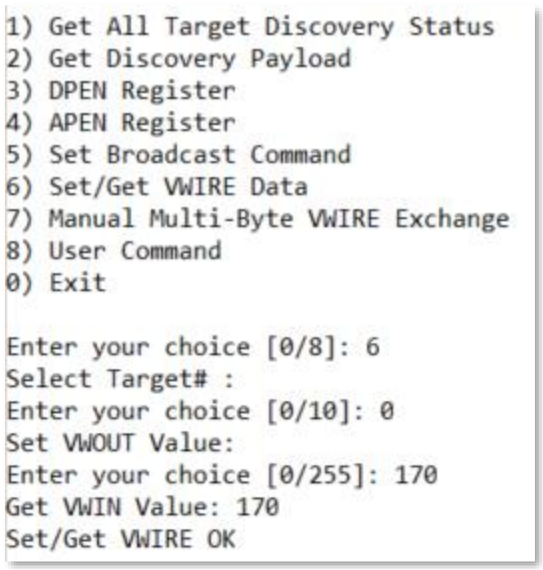


Figure 6.13. Default VWOUT/VWIN Waveform



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 6
Select Target# :
Enter your choice [0/10]: 0
Set VWOUT Value:
Enter your choice [0/255]: 170
Get VWIN Value: 170
Set/Get VWIRE OK
```

Figure 6.14. Set/Get VWIRE Data

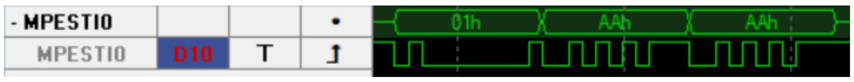
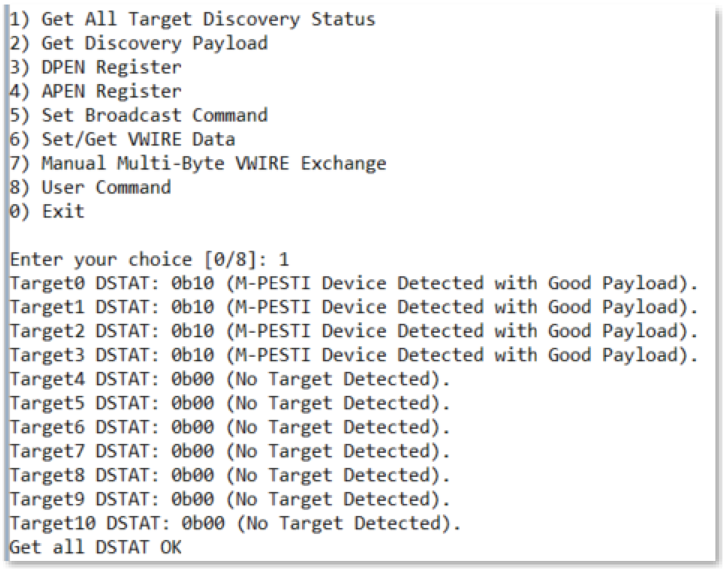


Figure 6.15. Modified VWOUT/VWIN Waveform

6.2.7. Manual Multi-Byte VWIRE Exchange

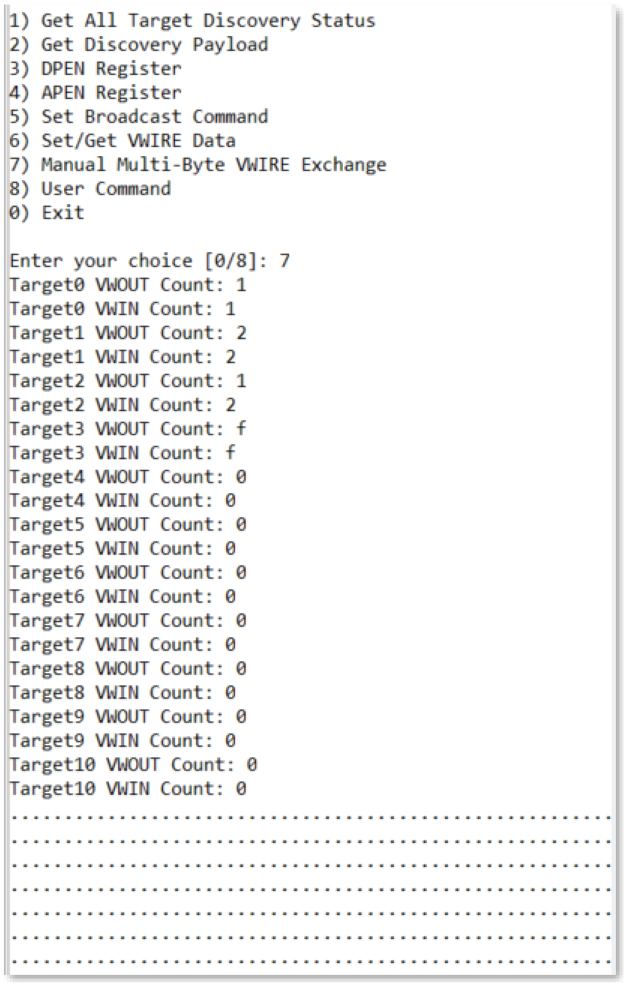
Before running this menu selection, make sure the DSTAT of the desired targets have already achieved a 0b10 (Good Payload) status as shown in the following figure.



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 1
Target0 DSTAT: 0b10 (M-PESTI Device Detected with Good Payload).
Target1 DSTAT: 0b10 (M-PESTI Device Detected with Good Payload).
Target2 DSTAT: 0b10 (M-PESTI Device Detected with Good Payload).
Target3 DSTAT: 0b10 (M-PESTI Device Detected with Good Payload).
Target4 DSTAT: 0b00 (No Target Detected).
Target5 DSTAT: 0b00 (No Target Detected).
Target6 DSTAT: 0b00 (No Target Detected).
Target7 DSTAT: 0b00 (No Target Detected).
Target8 DSTAT: 0b00 (No Target Detected).
Target9 DSTAT: 0b00 (No Target Detected).
Target10 DSTAT: 0b00 (No Target Detected).
Get all DSTAT OK
```

Figure 6.16. Good Payload Status

After selecting the Manual Multi-Byte VWIRE Exchange, the expected VWOUT and VWIN bytes for each target is printed, as shown in Figure 6.17. The actual round-robin waveform of the VWIRE Exchanges can be observed using the logic analyzer, as shown in Figure 6.18. You can stop the VWIRE Exchange by pushing the switch shown in Figure 6.19 and Figure 6.20.



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 7
Target0 VWOUT Count: 1
Target0 VWIN Count: 1
Target1 VWOUT Count: 2
Target1 VWIN Count: 2
Target2 VWOUT Count: 1
Target2 VWIN Count: 2
Target3 VWOUT Count: f
Target3 VWIN Count: f
Target4 VWOUT Count: 0
Target4 VWIN Count: 0
Target5 VWOUT Count: 0
Target5 VWIN Count: 0
Target6 VWOUT Count: 0
Target6 VWIN Count: 0
Target7 VWOUT Count: 0
Target7 VWIN Count: 0
Target8 VWOUT Count: 0
Target8 VWIN Count: 0
Target9 VWOUT Count: 0
Target9 VWIN Count: 0
Target10 VWOUT Count: 0
Target10 VWIN Count: 0
........................................................
........................................................
........................................................
........................................................
........................................................
........................................................
........................................................
```

Figure 6.17. Manual Multi-Byte VWIRE Exchange

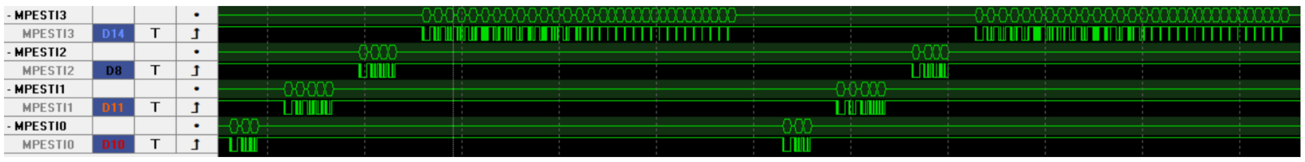


Figure 6.18. Round-Robin VWIRE Exchange Waveform

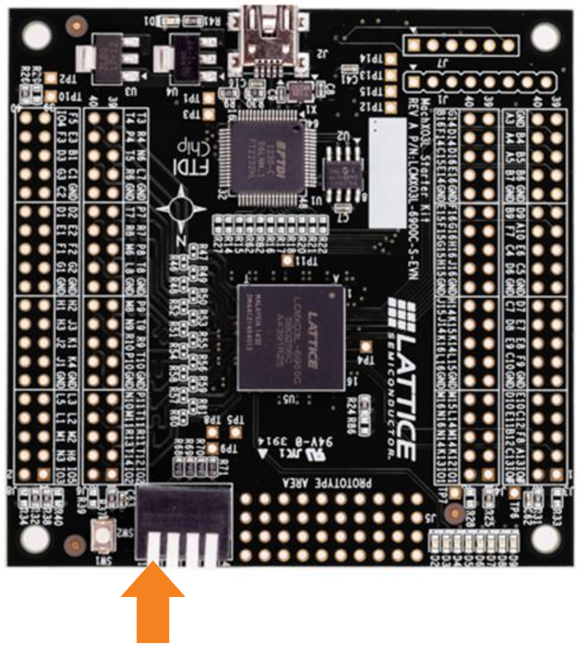


Figure 6.19. Stop VWIRE Exchange Using the MachXO3LF Initiator

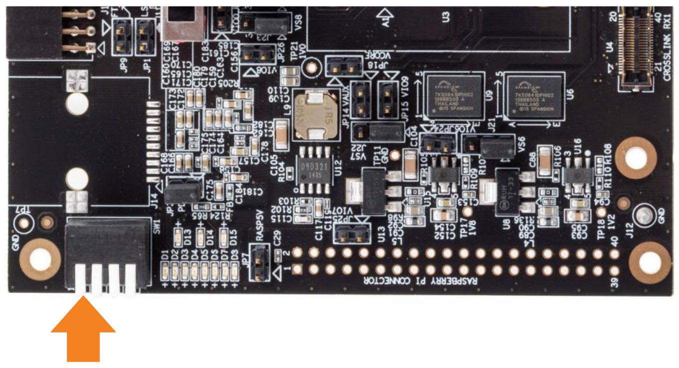
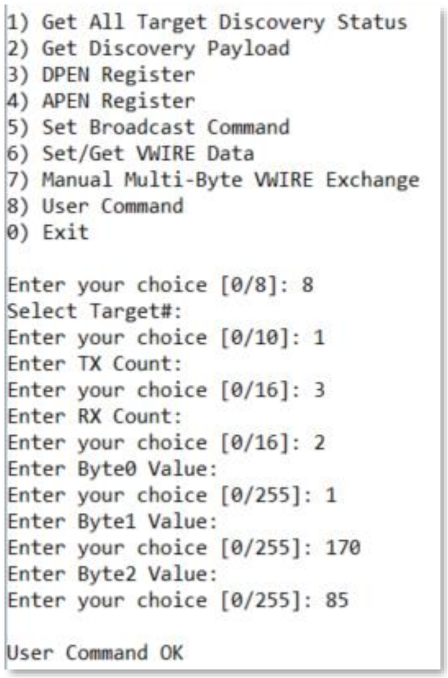


Figure 6.20. Stop VWIRE Exchange Using the MachXO5-NX Initiator

6.2.8. User Command

This menu selection allows you to customize a data frame structure by defining the number of bytes to transmit, the data to transmit, and the number of bytes to be received. As an example, the VWIRE exchange structure of Target1 is manually defined, as shown in Figure 6.21. This data frame is customized to have 3 bytes to send and 2 bytes to be received. The data sent, 1, 170, and 85, are the hexadecimal values 0x1, 0xAA, and 0x55, as shown in Figure 6.22.



```
1) Get All Target Discovery Status
2) Get Discovery Payload
3) DPEN Register
4) APEN Register
5) Set Broadcast Command
6) Set/Get VWIRE Data
7) Manual Multi-Byte VWIRE Exchange
8) User Command
0) Exit

Enter your choice [0/8]: 8
Select Target#:
Enter your choice [0/10]: 1
Enter TX Count:
Enter your choice [0/16]: 3
Enter RX Count:
Enter your choice [0/16]: 2
Enter Byte0 Value:
Enter your choice [0/255]: 1
Enter Byte1 Value:
Enter your choice [0/255]: 170
Enter Byte2 Value:
Enter your choice [0/255]: 85

User Command OK
```

Figure 6.21. User Command Example

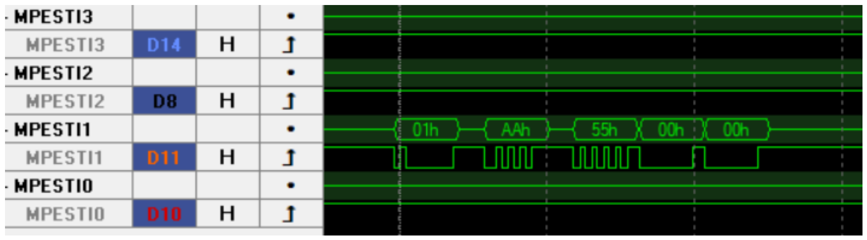


Figure 6.22. User Command Waveform

7. Simulating the Reference Design

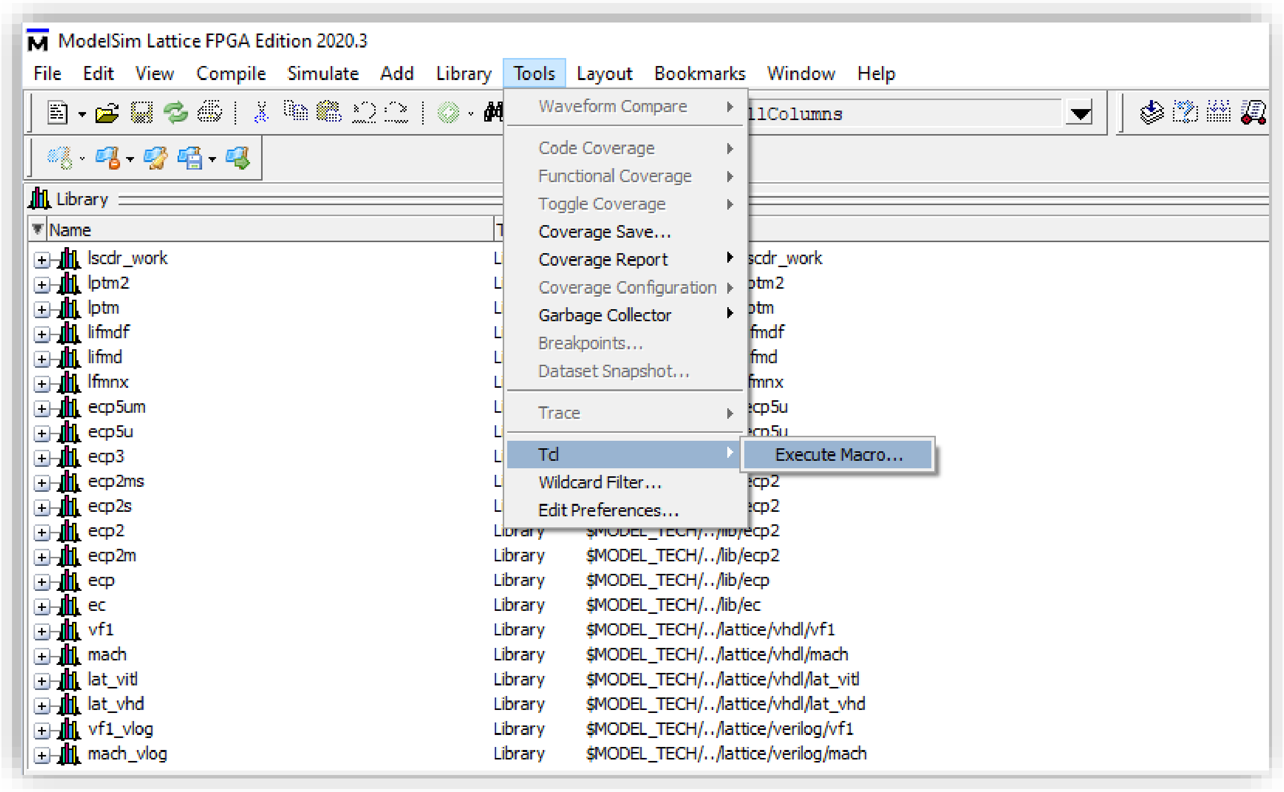
This reference design includes pre-generated RTL files that can be directly simulated within the ModelSim™ Lattice FPGA Edition software or QuestaSim™ Lattice FPGA Edition software.
To simulate the design, perform the following steps:

1. Unzip the reference design *.zip* file.
2. Open the *.do* file on a text editor and replace the text SET THE SIMULATION PATH HERE from Line 2 with the directory path of the simulation file. An example is seen on Line 5 of the file.

```
2     set SIM_DIR "SET THE SIMULATION PATH HERE"
3
4     # Example:
5     # set SIM_DIR "D:/AISLA/FPGA-XX-XXXXX-1-0-RISC-V_Controlled_MPESTI-Initiator-Reference-Design/Simulation"
```

Figure 7.1. Changing the Simulation Directory

3. Launch the ModelSim Lattice FPGA Edition software and select **Tools > TCL> Execute Macro.**



Name	T	
lscdr_work	L	scdr_work
lptm2	L	ptm2
lptm	L	ptm
lifmdf	L	fmdf
lifmd	L	fmd
lfmnx	L	fmnx
ecp5um	L	cp5u
ecp5u	L	cp5u
ecp3	L	
ecp2ms	L	cp2
ecp2s	L	cp2
ecp2	Library	$MODEL_TECH/../lib/ecp2
ecp2m	Library	$MODEL_TECH/../lib/ecp2
ecp	Library	$MODEL_TECH/../lib/ecp
ec	Library	$MODEL_TECH/../lib/ec
vf1	Library	$MODEL_TECH/../lattice/vhdl/vf1
mach	Library	$MODEL_TECH/../lattice/vhdl/mach
lat_vitl	Library	$MODEL_TECH/../lattice/vhdl/lat_vitl
lat_vhd	Library	$MODEL_TECH/../lattice/vhdl/lat_vhd
vf1_vlog	Library	$MODEL_TECH/../lattice/verilog/vf1
mach_vlog	Library	$MODEL_TECH/../lattice/verilog/mach

Figure 7.2. Running the Simulation Script File

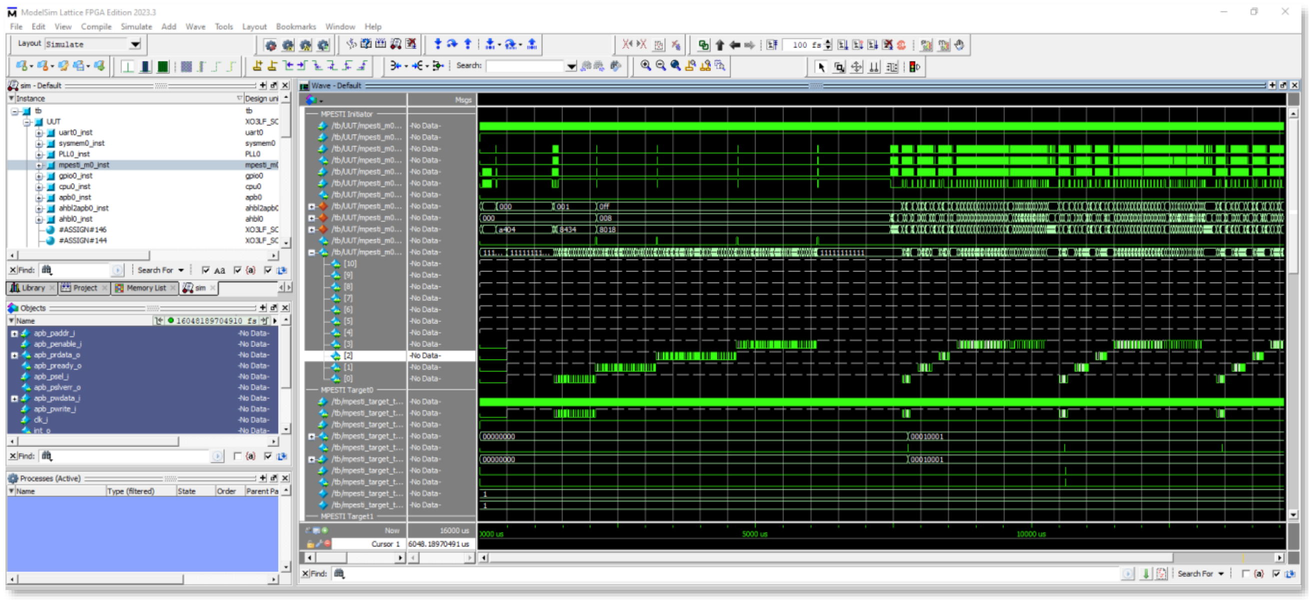

ModelSim Lattice FPGA Edition 2023.3
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
sim - Default
Objects
Processes (Active)
Wave - Default
MPESTI Initiator
MPESTI Target0
MPESTI Target1
5000 us
10000 us

Figure 7.3. Simulation Waveform

8. Resource Utilization

The following table lists the resource utilization information of the reference design.

Table 8.1. Resource Utilization

Module or IP	MachXO3LF Utilization		MachXO5-NX Utilization		Notes
	LUTs	EBR	LUTs	EBR	
RISC-V MC (cpu0)	1559	4	—	—	Use either RISC-V MC or RISC-V SM with the Debug Mode enabled.
RISC-V SM (cpu0)	—	—	1584	2	
System Memory (sysmem0)	255	20	230	10	Approximately 20 KB. This is not a minimum requirement and can require less depending on the design.
M-PESTI IP	1323	1	1377	1	The current design allows up to 11 targets. This can be increased through the IP GUI, which requires approximately 80 LUTs for each additional target.
GPIO0	193	0	188	0	Optional GPIO pins and can be removed safely.
UART	200	0	211	0	For debugging and demonstration purposes.
PLL	1	0	1	0	Used to generate the 25 MHz clock from the external 12 MHz oscillator.
AHB-Lite Interconnect	90	0	89	0	Required to connect RISC-V core to the other components of the SoC design.
AHB-Lite to APB Bridge	316	0	294	0	
APB Interconnect	8	0	15	0	

References

- M-PESTI Initiator IP User Guide (FPGA-IPUG-02258)
- M-PESTI Initiator IP Release Notes (FPGA-RN-02005)
- M-PESTI Initiator Driver API Reference (FPGA-TN-02413)
- M-PESTI Initiator IP web page
- RISC-V MC CPU IP User Guide (FPGA-IPUG-02252)
- Open Compute Project® web page
- System Memory Module IP User Guide (FPGA-IPUG-02073)
- UART IP User Guide (FPGA-IPUG-02105)
- AHB-Lite Interconnect Module IP User Guide (FPGA-IPUG-02051)
- AHB Lite to APB Bridge Module IP User Guide (FPGA-IPUG-02053)
- APB Interconnect Module IP User Guide (FPGA-IPUG-02054)
- MachXO3 Family Devices (FPGA-DS-02026)
- MachXO3D Family Devices (FPGA-DS-02032)
- MachXO3LF Starter Kit (FPGA-EB-02036)
- MachXO3D Breakout Board (FPGA-UG-02084)
- MachXO5-NX Development Board (FPGA-EB-02052)
- MachXO3D web page
- MachXO3 web page
- MachXO5-NX web page
- Lattice Radiant FPGA design software
- Lattice Solutions Reference Designs web page
- Lattice Solutions IP Cores web page
- Lattice Propel Design Environment web page
- Lattice Radiant Software User Guide
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

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

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.1, September 2025

Section	Change Summary
All	• Replaced instances of *MachXO5NX* and *XO5NX* with *MachXO5-NX*, except for mentioned design source file and folder names. • Replaced instances of *MPESTI* with *M-PESTI*, except for name attributes. • Added the OCP logo to the cover page.
Abbreviations in This Document	Added *Open Compute Project (OCP)*.
Introduction	• Added a statement indicating that the *reference design is OCP Ready*. • Minor editorial edit to the *Board* entry under *Hardware Requirements* in Table 1.1. Summary of the Reference Design.
Functional Description	Replaced the caption of *Figure 5.17. XO3D Device Properties* with Figure 5.17. MachXO3D Device Properties Settings.
References	Added the following links: • The Open Compute Project web page • M-PESTI Initiator Release Notes (FPGA-RN-02005) • M-PESTI Initiator Driver API Reference (FPGA-RN-02413)

Revision 1.0, April 2025

Section	Change Summary
All	Initial release.

www.latticesemi.com