



M-PESTI Initiator IP

IP Version: v2.0.0

User Guide

FPGA-IPUG-02258-1.4

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.....	6
1. Introduction	7
1.1. Overview of the IP	7
1.2. Quick Facts	7
1.3. IP Support Summary	7
1.4. Features	8
1.5. Licensing and Ordering Information	8
1.5.1. Ordering Part Number.....	8
1.6. Hardware Support.....	8
1.7. Minimum Device Requirements.....	9
1.8. Naming Conventions	9
1.8.1. Nomenclature.....	9
1.8.2. Signal Names	9
2. Functional Description.....	10
2.1. IP Architecture Overview	10
2.2. Clocking	10
2.3. Reset.....	11
2.4. User Interfaces	11
2.5. M-PESTI Protocol.....	11
2.5.1. M-PESTI UART Frame	11
2.5.2. M-PESTI Protocol Phase	12
2.5.3. Optional Source to Destination Detection	16
2.5.4. Fanout Feature	19
3. IP Parameter Description.....	20
3.1. General.....	20
3.2. Virtual Wire Setting.....	22
4. Signal Description	23
5. Register Description	25
5.1. Global Registers (0x0_0000)	27
5.1.1. IP Information 0x00.....	27
5.1.2. Configuration 0 0x04	28
5.1.3. Configuration 1 0x08	28
5.1.4. Global Software Reset 0x10	28
5.1.5. Discovery Payload Page Select 0x14	28
5.1.6. User Command Status 0x20	28
5.1.7. User Command 0x24	29
5.1.8. User Write Data 0x28	30
5.1.9. User Read Data 0x2C	30
5.2. Global Optional Registers (0x0_1000).....	30
5.2.1. Clock Configuration 0x00	30
5.2.2. Software Reset 0x04.....	30
5.2.3. Global Interrupt Status 0x08	31
5.2.4. Global Interrupt Enable 0x0C	32
5.2.5. Global Interrupt Set 0x10	32
5.2.6. User Command Interrupt Enable 0x14.....	33
5.2.7. User Command Interrupt Set 0x18	33
5.2.8. Secondary Wire Control Status 0x40.....	33
5.2.9. Secondary Wire Select 0x44.....	34
5.3. Target Required Registers (0x1_0000)	34
5.3.1. M-PESTI Target N Error, Status, Control and Configuration 0x00+(N*4)	34
5.4. Target Payload (0x2_0000)	37

5.5.	Target Virtual Wire Registers (0x3_0000)	37
5.5.1.	M-PESTI Target N Virtual Wire Input 0x00+(N*32)	38
5.5.2.	M-PESTI Target N Virtual Wire Output 0x10+(N*32)	38
5.6.	Target Optional Registers (0x4_0000).....	38
5.6.1.	M-PESTI Target N Status 0x00+(N*16)	38
5.6.2.	M-PESTI Target N Virtual Wire Configuration 0x04+(N*16).....	39
6.	Example Design.....	40
6.1.	Example Design Supported Configuration	40
6.2.	Overview of the Example Design and Features.....	41
6.3.	Example Design Components.....	42
6.4.	Generating the Example Design	42
6.5.	Simulating the Example Design	46
6.6.	Hardware Testing	48
7.	Designing with the IP	49
7.1.	Generating and Instantiating the IP	49
7.1.1.	Generated Files and File Structure	51
7.2.	Design Implementation.....	52
7.3.	Timing Constraints	52
7.4.	Running Functional Simulation	52
	Appendix A. Resource Utilization	56
	References.....	59
	Technical Support Assistance	60
	Revision History.....	61

Figures

Figure 2.1.	Lattice M-PESTI Initiator IP Core Block Diagram.....	10
Figure 2.2.	M-PESTI UART Frame.....	11
Figure 2.3.	M-PESTI Protocol Timing Diagram	12
Figure 2.4.	Single Initiator Multiple Targets Transaction.....	12
Figure 2.5	Virtual Wire Exchange	13
Figure 2.6	Virtual Wire Exchange with PEC	13
Figure 2.7.	Manual Multiple Virtual Bytes Program Flow.....	14
Figure 2.8.	Broadcast Command	15
Figure 2.9.	User Command Program Flow.....	16
Figure 2.10.	Secondary Wire Capability Program Flow	18
Figure 2.11.	Fanout Feature Flow	19
Figure 5.1.	Memory Map Allocation	25
Figure 5.2.	Target Payload Allocation	37
Figure 5.3.	Target Virtual Wire Register Allocation	37
Figure 6.1.	M-PESTI Initiator IP Core in an SoC Project	41
Figure 6.2.	Sample C Code Test Routine.....	42
Figure 6.3.	M-PESTI Initiator IP Example Design Block Diagram.....	42
Figure 6.4.	Select Template	43
Figure 6.5.	Select Template	44
Figure 6.6.	Create SoC Project	45
Figure 6.7.	Define Instance	45
Figure 6.8.	Reload sbx.....	46
Figure 6.9.	Verification Project Schematic.....	47
Figure 6.10.	SoC and Simulation Model Instantiation	48
Figure 7.1.	Module/IP Block Wizard	49
Figure 7.2.	M-PESTI Initiator IP Core Configuration Example	50
Figure 7.3.	Check Generated Result	51

Figure 7.4. Simulation Wizard.....	52
Figure 7.5. Add and Reorder Source.....	53
Figure 7.6. Parse HDL Files for Simulation.....	53
Figure 7.7. Summary Window.....	54
Figure 7.8. Transcript of the Simulation Result.....	54
Figure 7.9. Simulation Waveform.....	55

Tables

Table 1.1. Summary of the M-PESTI Initiator IP	7
Table 1.2. IP Support Summary of the M-PESTI Initiator IP.....	7
Table 1.3. Ordering Part Number	8
Table 2.1. User Interfaces and Supported Protocols.....	11
Table 3.1. General Attributes	20
Table 3.2. Virtual Wire Settings	22
Table 4.1. Ports Description.....	23
Table 5.1. Register Access Types	25
Table 5.2. Register Address Map	26
Table 5.3. IP Information 0x00	27
Table 5.4. Configuration 0 0x04.....	28
Table 5.5. Configuration 1 0x08.....	28
Table 5.6. Global Software Reset 0x10.....	28
Table 5.7. Discovery Payload Page Select 0x14	28
Table 5.8. User Command Status 0x20.....	28
Table 5.9. User Command 0x24.....	29
Table 5.10. User Write Data 0x28.....	30
Table 5.11. User Read Data 0x2C.....	30
Table 5.12. Clock Configuration 0x00	30
Table 5.13. Software Reset 0x04	30
Table 5.14. Global Interrupt Status 0x08.....	31
Table 5.15. User Command Interrupt Enable 0x14	33
Table 5.16. User Command Interrupt Set 0x18	33
Table 5.17. Secondary Wire Control Status 0x40	33
Table 5.18. Secondary Wire Select 0x44.....	34
Table 5.19. M-PESTI Target N Error, Status, Control and Configuration 0x00+(N*4).....	34
Table 5.20. M-PESTI Target N Virtual Wire Input 0x00+(N*32).....	38
Table 5.21. M-PESTI Target N Virtual Wire Output 0x10+(N*32).....	38
Table 5.22. M-PESTI Target N Status 0x00+(N*16).....	38
Table 5.23. M-PESTI Target N Virtual Wire Configuration 0x04+(N*16)	39
Table 6.1. M-PESTI Initiator IP Configuration Supported by the Example Design	40
Table 7.1. Generated File List	51
Table A.1. Resource Utilization for LFMX05-25-7BBG400I Device (IP v1.1.0 ³)	56
Table A.2. Resource Utilization for LFMX05-25-7BBG400I Device (IP v1.2.0 ³)	56
Table A.3. Resource Utilization for LAV-AT-E70-1LFG676I Device (IP v1.2.0 ³)	56
Table A.4. Resource Utilization for LN2-CT-20ES-1ASG410I Device (IP v1.2.0 ³).....	57
Table A.5. Resource Utilization for LFMX05-25-7BBG400I Device (IP v2.0.0 ³)	57
Table A.6. Resource Utilization for LAV-AT-E70-1LFG676I Device (IP v2.0.0 ³)	57
Table A.7. Resource Utilization for LN2-CT-20ES-1ASG410I Device (IP v2.0.0 ³).....	58

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
AHB	Advanced High-performance Bus
APB	Advanced Peripheral Bus
CPU	Central Processing Unit
CMD	Command
CRC	Cyclic Redundancy Check
FPGA	Field Programmable Gate Array
GPIO	General Purpose I/O
GUI	Graphical User Interface
IP	Intellectual Property
LUT	Look Up Table
M-PESTI	Modular-Peripheral Sideband Tunneling Interface
OCP	Open Compute Project
PDC	Physical Design Constraint
PEC	Packet Error Checking
PLL	Phase-Locked Loop
RAM	Random-Access Memory
SoC	System-on-a-Chip
SRAM	Static Random-Access Memory
UART	Universal Asynchronous Receiver Transmitter

1. Introduction

1.1. Overview of the IP

The Modular-Peripheral Sideband Tunneling Interface (M-PESTI) Initiator IP core provides early peripheral presence detection and attribute collection before system boot up. This IP core supports bidirectional communication with initiator command and target response structure.

It is designed to comply both with the [Modular-Peripheral Sideband Tunneling Interface \(M-PESTI\) Base Specification Version 1.0 Release Candidate 2](#) and [Modular-Peripheral Sideband Tunneling Interface \(M-PESTI\) Base Specification Version 1.2 Release Candidate 2](#). This IP is OCP Ready™.

1.2. Quick Facts

Table 1.1. Summary of the M-PESTI Initiator IP

IP Requirements	Supported Devices	MachXO3™, MachXO3D™, Mach™-NX, MachXO5™-NX, Lattice Avant™, and Certus™-N2.
	IP Changes ¹	For a list of changes to the IP, refer to the M-PESTI Initiator IP Release Notes (FPGA-RN-02005) .
Resource Utilization	Supported User Interface	Advanced Peripheral Bus (APB)
	Resources	See Appendix A. Resource Utilization
Design Tool Support	Lattice Implementation ²	IP core v2.0.0 - Lattice Radiant software 2025.2
	Synthesis	Lattice Synthesis Engine (LSE) Synopsys® Synplify Pro® for Lattice
	Simulation	For the list of supported simulators, see the Lattice Radiant Software User Guide .
Driver Support	API Reference	Refer to the M-PESTI Initiator Driver API Reference (FPGA-TN-02413) .

Notes:

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.
2. Lattice Implementation indicates the IP version release coinciding with the software version release. Check the software for IP version compatibility with earlier or later software versions.

1.3. IP Support Summary

Table 1.2. IP Support Summary of the M-PESTI Initiator IP

Device Family	System Clock	Number of Target Devices	Radiant Timing Model	Hardware Validated
MachXO5-NX	25 MHz	2	Final	Yes
	100 MHz	2	Final	No
MachXO3D	25 MHz	2	Final	No
	100 MHz	2	Final	No
MachXO3LF	25 MHz	2	Preliminary	No
	100 MHz	2	Preliminary	No
MachXO3L	25 MHz	2	Preliminary	No
	100 MHz	2	Preliminary	No
Certus-N2	25 MHz	2	Preliminary	No
	100 MHz	2	Preliminary	No
Lattice Avant	25 MHz	2	Preliminary	No
	100 MHz	2	Preliminary	No

1.4. Features

The following lists the key features of the M-PESTI Initiator IP core:

- 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 M-PESTI port.
- Supports Static Discovery payload with CRC-8 payload checksum.
- Supports one initiator to many targets system.
- Supports configurable number of M-PESTI devices up to 64 targets.
- Supports autonomous Static Discovery Payload request command to all Targets on round robin manner during 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 send command attempts as a set of initial and two retry attempts until successful payload is received.
- Supports Target reset at any time.
- Supports sending broadcast command.
- Supports an aborting of ongoing discovery or active phase command to insert a broadcast command.
- Supports source and destination cable coupling discovery.
- Supports optional discovery phase bypass.
- Supports optional active phase Packet Error Checking (PEC).

1.5. Licensing and Ordering Information

An IP-specific license string is required to enable full use of the M-PESTI Initiator IP core in a complete, top-level design.

The IP can be fully evaluated through functional simulation and implementation (synthesis, map, place and route) without an IP license string. This IP supports Lattice's IP hardware evaluation capabilities. You can create versions of the IP to operate in hardware for a limited time (approximately four hours) without requiring an IP license string. A license string is required to enable timing simulation and to generate a bitstream file that does not include the hardware evaluation timeout limitation.

For more information about pricing and availability of the M-PESTI Initiator IP core, contact your [local Lattice Sales Office](#).

1.5.1. Ordering Part Number

Table 1.3. Ordering Part Number

Device Family	Part Number	
	Single Seat Annual	Single Seat Perpetual
MachXO5-NX	MPESTI-I-XO5-US	MPESTI-I-XO5-UT
MachXO3D	MPESTI-I-XO3D-U	MPESTI-I-XO3D-UT
MachXO3	MPESTI-I-XO3-US	MPESTI-I-XO3-UT
Mach-NX	MPESTI-I-MNX-US	MPESTI-I-MNX-UT
Certus-N2	MPESTI-I-CN2-US	MPESTI-I-CN2-UT
Avant-AT-G	MPESTI-I-AVG-US	MPESTI-I-AVG-UT
Avant-AT-X	MPESTI-I-AVX-US	MPESTI-I-AVX-UT
Avant-AT-E	MPESTI-I-AVE-US	MPESTI-I-AVE-UT

1.6. Hardware Support

Refer to the [Example Design](#) section for more information on the boards used.

1.7. Minimum Device Requirements

There is no limitation in device speed grade for M-PESTI Initiator IP core. For the minimum required resources to instantiate this IP and maximum clock frequency supported, see [Appendix A. Resource Utilization](#) section.

1.8. Naming Conventions

1.8.1. Nomenclature

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

1.8.2. Signal Names

Signal names that end with:

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

2. Functional Description

2.1. IP Architecture Overview

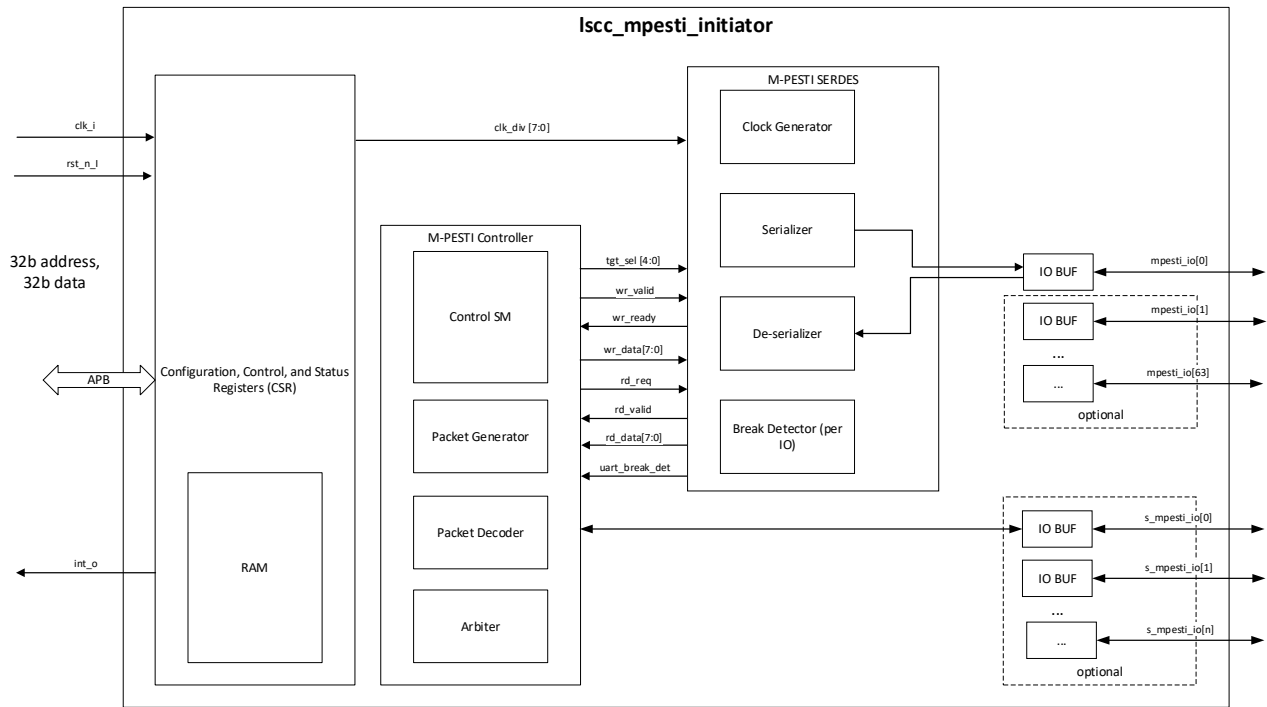


Figure 2.1. Lattice M-PESTI Initiator IP Core Block Diagram

The M-PESTI Initiator IP core includes the following layers:

- APB Interface
- Configuration, Control and Status Registers
- This block includes the registers for user command and configurations and target/s status and the internal RAM for the payload allocation.
- M-PESTI Initiator Top
- This block includes logics related to input and output ports and blocks instantiation.
- M-PESTI Controller
- This block includes the implementation of the process flow as well as the device status tracking and the generation and decoding of the necessary packet format. This also includes the implementation of the round robin servicing for multiple targets sharing one initiator.
- M-PESTI SERDES
- This block includes the implementation of the low level half-duplex UART protocol and the generation of the divided clock and cyclic redundancy check (CRC).

2.2. Clocking

There is one clock source for the M-PESTI Initiator IP core.

clk_i: System clock. Set the value in the *System Clock Frequency* attribute. This clock is used to generate the M-PESTI baud rate of 250 kHz for the M-PESTI wire transactions.

2.3. Reset

There is one hardware reset for the M-PESTI Initiator IP core.

rst_n_i: An asynchronous active low reset.

Note: This asynchronous reset is synchronized with the system clock. Upon the de-assertion of *rst_n_i*, three (3) clock cycles are needed to propagate the reset in the IP core blocks.

2.4. User Interfaces

The following table shows the user interfaces and supported protocols. The memory-mapped interface of M-PESTI Initiator IP core is APB interface only.

Table 2.1. User Interfaces and Supported Protocols

User Interface	Supported Protocols	Description
Memory-Mapped Interface	APB	<p>Data width is set to 32 bits.</p> <p>The following describes the read and write transactions of the APB interface:</p> <ul style="list-style-type: none"> Write transaction has no wait states Read transaction has one wait states <p>For more information the APB interface and the timing diagrams, refer to the AMBA 3 APB Protocol v1.0 Specification.</p>

2.5. M-PESTI Protocol

2.5.1. M-PESTI UART Frame

The M-PESTI wire is a simple half-duplex UART. Start bit is determined when the wire is pulled from high to low since the frame is asynchronous to the receiving clock. Following the start bit is the 8 bits data, odd parity, and one stop bit running at the 250 kHz +/- 3% baud rate. Note that 1 baud is 4μs. Stop bit is determined when the wire is pulled from low to high. When stop bit is zero (STOP_BIT=0), framing error will be flagged.

The next frame may start after stop bit of the previous frame is sampled at the positive edge of the receiving clock.

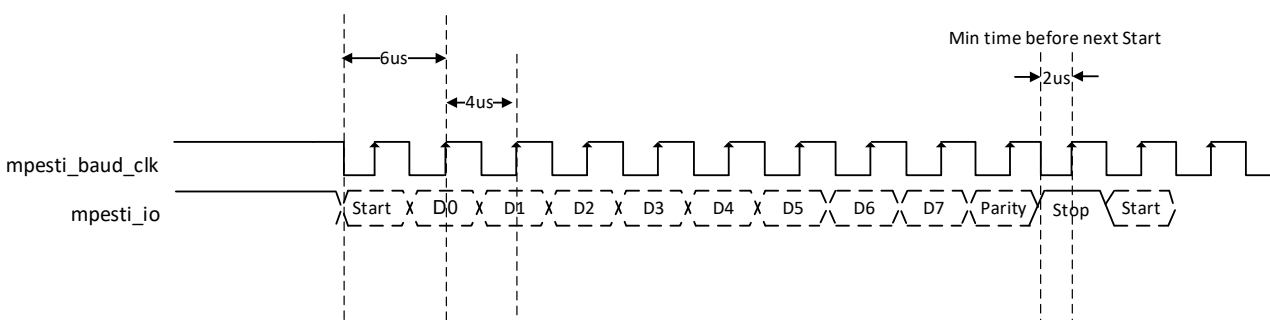


Figure 2.2. M-PESTI UART Frame

2.5.2. M-PESTI Protocol Phase

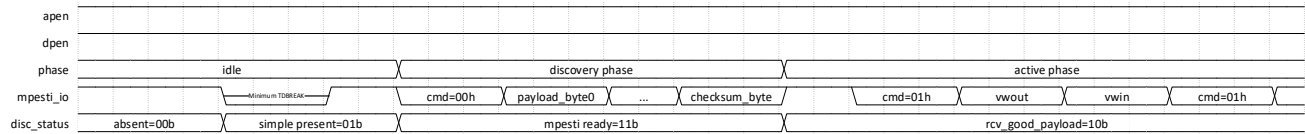


Figure 2.3. M-PESTI Protocol Timing Diagram

2.5.2.1. Target Presence Detection

Before the frame-based discovery, the initiator must detect the target presence. The event of target holding the M-PESTI wire low for greater than the frame length is called UART BREAK. The target must meet the minimum target discovery break width of 50us to guarantee presence detection. The maximum target discovery break width is configured through the *Presence Detection Timeout* attribute. Exceeding the maximum limit will trigger a target presence detection timeout. UART break after presence detection timeout is still valid and will proceed accordingly.

2.5.2.2. Target Setting Reconfiguration

The initiator samples the configurations of the selected target during the transition of target select. Reconfiguration in between transactions take effect after the current transaction.

2.5.2.3. Discovery Phase

After the target releases the UART BREAK, the initiator autonomously sends the discovery phase command, 8'h00, to M-PESTI ready target given that the *Discovery Phase Enable* attribute is checked. On the other hand, when the *Discovery Phase Enable* attribute is unchecked, you can program the *Discovery Phase Enable* register to manually perform the discovery phase. When the target did not send a response the initiator resends the discovery phase command until good payload is successfully received. Good payload means that there is no CRC error, parity error, or framing error. Re-discovery after a good payload is received can be done by writing 0 then 1 to the *Discovery Phase Enable* register or by hardware reset. After three unsuccessful attempts (initial try plus two retries), the Discovery Retry Rollover bit in the Target Control Status register is asserted, and the initiator proceeds to the next target.

Target payload size is determined by its payload information. Refer to *Maximum Number of Static Payload Size* attribute for the supported sizes.

A single M-PESTI initiator can handle up to 64 targets. In the case of instantiating a single initiator to multiple targets, round robin servicing is implemented.

The following figure shows the transition of the target arbitration given that there are three (3) M-PESTI ready targets instantiated, both discovery phase and active phase are configured as enabled and all targets release break at the same time. At the first arbitration cycle, initiator transacts to each target for the discovery phase. In this example, target select is at Target 0 and this target is M-PESTI ready. Target 0 reached 3 discovery retries. The initiator proceeds to the next target, Target 1. While Target 1 and Target 2 is doing the discovery phase, Target 0 stays at discovery phase waiting for payload request command. Then, Target 1 and Target 2 transitioned to an active phase after receiving a good payload and then waited for its turn to start virtual wire exchange.

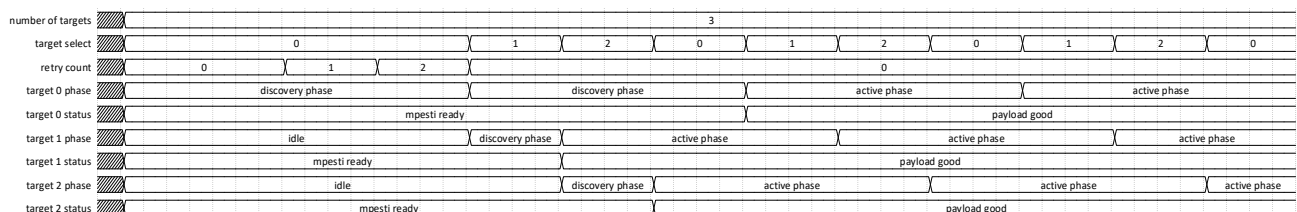


Figure 2.4. Single Initiator Multiple Targets Transaction

In Specification version 1.2, there is an optional Discovery Phase Bypass feature which allows virtual wire exchange without prior good payload received. This is configurable through the *Discovery Phase Bypass Enable* attribute and can be reconfigured through the Discovery Phase Bypass Enable register.

2.5.2.4. Active Phase

When APEN is enabled in the *Active Phase Enable* attribute and the discovery phase is successful, this means the initiator receives good checksum, the initiator autonomously sends the active phase command, 8'h01, then exchanges data to each target through the M-PESTI interface. When the *Active Phase Enable* attribute is unchecked, you can program the *Active Phase Enable* register to manually perform the virtual wire exchange. The initiator expects a respond within 500us. An active phase receive timeout error is flagged when there is no response from the target.

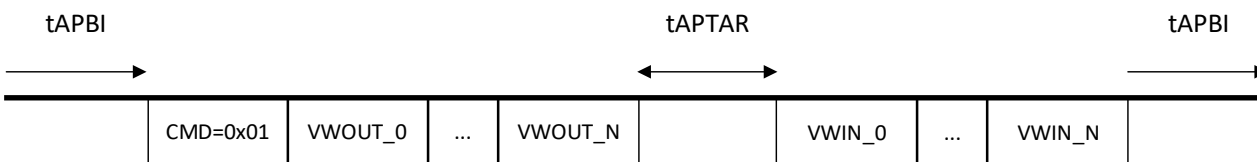


Figure 2.5 Virtual Wire Exchange

In Specification version 1.2, Packet Error Checking (PEC) can be enabled through the *Active Phase PEC Enable* attribute and can be configured through Active Phase PEC Enable register. The initiator parses the payload information, Active Phase PEC Support, and any mismatch from the configuration will trigger an Active Phase PEC Configuration Mismatch status. PEC is calculated using CRC-8.

Active phase command with PEC, 8'h81, is sent. Initiator and target must append PEC byte after the virtual wire output data and virtual wire input data, respectively.

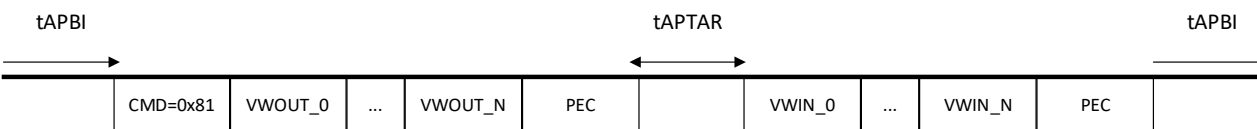


Figure 2.6 Virtual Wire Exchange with PEC

The M-PESTI Initiator IP continuously send active phase command through round robin servicing to all targets unless Active Phase is disabled.

Current autonomous virtual wire exchange transacts virtual wire input and output data based on the *Number of Virtual Wire Input and Output Bytes* attribute per target.

When *Enable Programmable Virtual Wire* attribute is checked, this allows reconfiguration of the number of virtual wire bytes per target. However, the allowable reconfigurable values must not be greater than the values set on the *Number of Virtual Wire Input Bytes* and *Number of Virtual Wire Output Bytes* attributes.

If the number of virtual wire output bytes is zero, the initiator sends 8'h00 data. If the number of virtual wire input bytes is zero, the target is expected to send 8'h00 as an acknowledgement response.

To support manual multiple virtual wire bytes transactions, refer to the following figure that shows the manual program flow through user command.

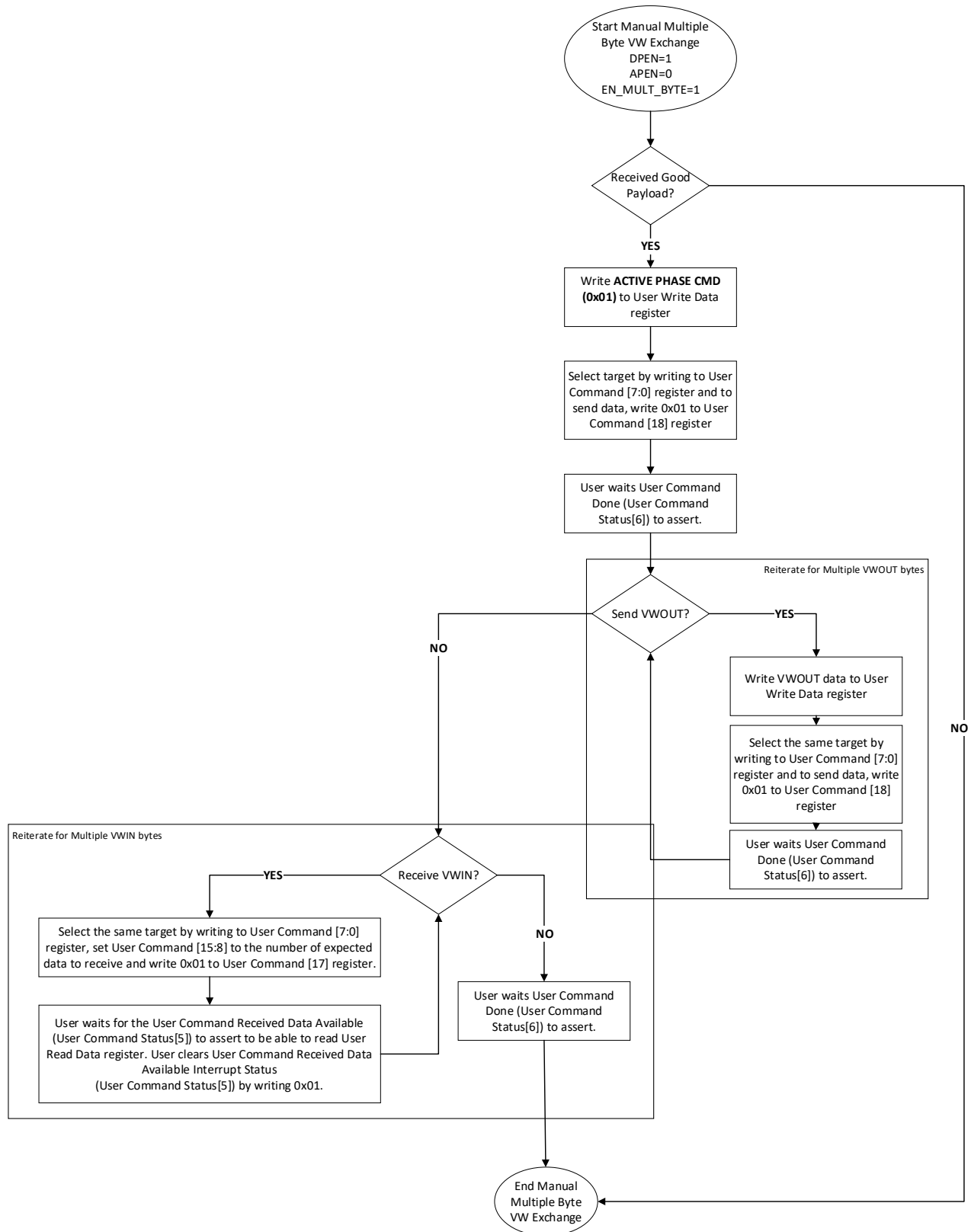


Figure 2.7. Manual Multiple Virtual Bytes Program Flow

2.5.2.5. User Command, Broadcast, and Abort Mechanism

For independent commands, User Command register can be configured to transact with the target. Enabling send data in the User Command register sends the data to the User Write data register. Additionally, enabling the expected data in the User Command register expects response from the target, which can be read in the User Read data register.

When the *Allow Multiple User Receive Bytes* attribute is enabled, the initiator can receive up to 16 bytes of data from the target. These bytes are read sequentially from the User Read Data register. The User Command Received Data Available status remains asserted until all bytes are read. The initiator expects a response within 250 ms; otherwise, a receive timeout is flagged in the Interrupt Status register, and the user command is terminated.

Because multiple targets can share a single initiator, this limits transactions at the same time. The broadcast command enables the initiator to send commands to multiple targets simultaneously. The initiator sends a broadcast command, 8'hFF, and the data to be broadcasted. This command has no target response.

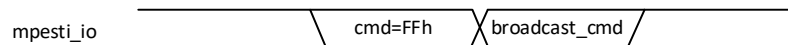


Figure 2.8. Broadcast Command

The M-PESTI initiator has an option to abort the current transaction or to continue before sending the broadcast command. Aborted transactions may happen during the initiator transmission and target transmission.

During the initiator transmission, the initiator may assert a BREAK condition to cancel a sequence. If the start bit has already been sent, the initiator must wait until the target idle (time after stop bit prior to start bit) before asserting a BREAK condition. The target must ignore any previously received bytes and prepare for a new sequence.

During target transmission, the initiator may assert a BREAK condition at any point during the target's frame transmission or during the turnaround period (between initiator transmission and target transmission). The target must detect the BREAK condition and must not attempt to transmit.

User command can be programmed as shown in the following figure.

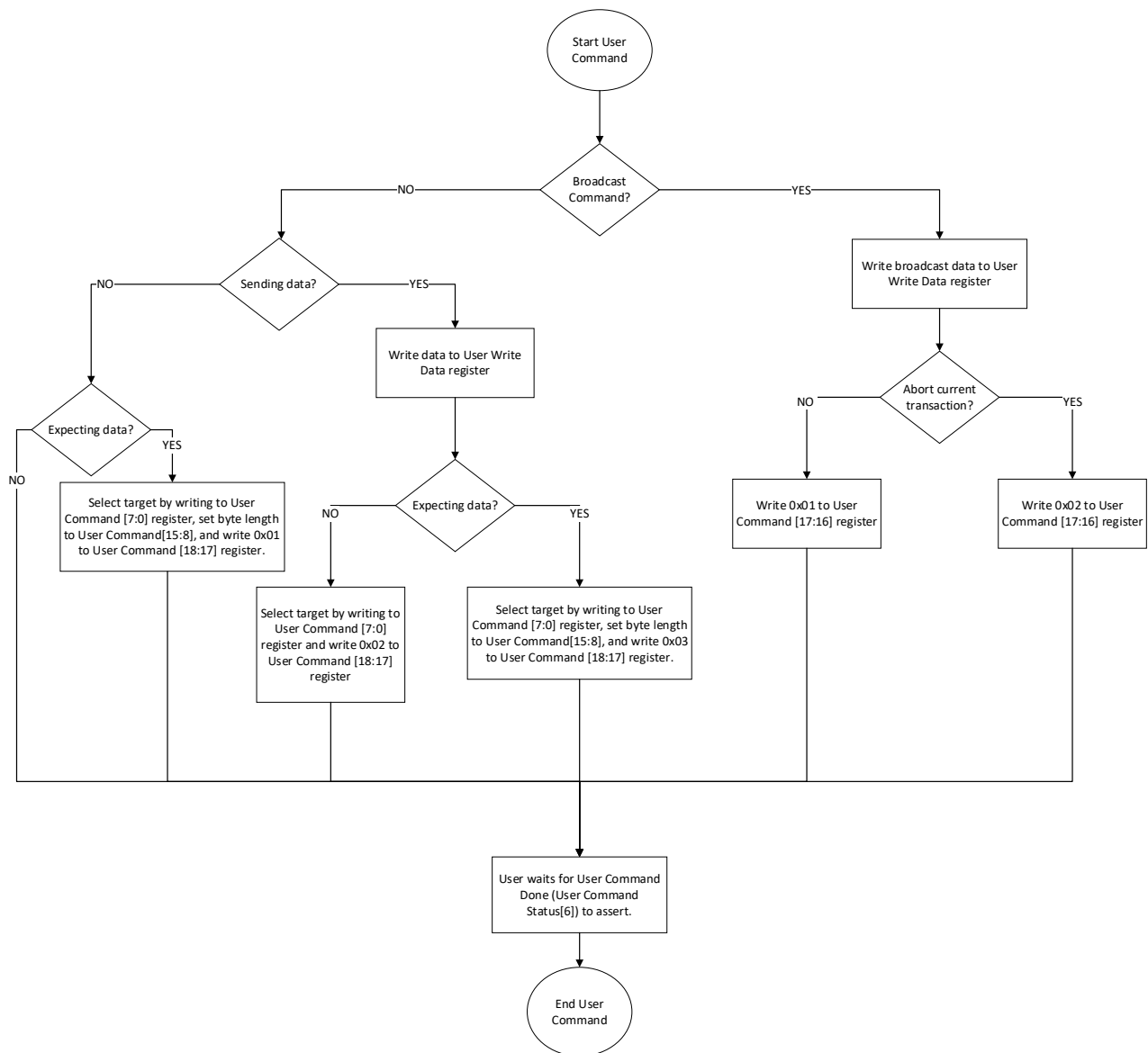


Figure 2.9. User Command Program Flow

2.5.2.6. Target Reset

If the target resets prior to a successful discovery phase, the initiator will autonomously send a discovery phase command to retry discovery given that the discovery phase is enabled.

If the target resets after a successful discovery phase, the discovery status and payload information is locked. The initiator will send an active phase command to initiate a virtual wire exchange. Because of the target resets, the active phase command is not observed and flags an active phase error due to timeout.

2.5.3. Optional Source to Destination Detection

2.5.3.1. Secondary Wire Capability with Stimulus and Response

This is only applicable when the selected Version Support is 1.0. The primary source wire is used for frame-based communication. The initial discovery payload determines if there are additional source to destination coupling needed to be identified. These couplings are defined to be the secondary M-PESTI wires.

You reiterate the process of selecting wires and sending low (secondary wire stimulus is zero) stimulus to each secondary wire to detect all wires associated with that target.

The selectable value using the wire select is based on the configuration in the Total Number of Secondary Wire attribute. Selecting values greater than the configuration set in the Total Number of Secondary Wire attribute will be ignored. The commands are sent to the previously selected wire.

The following figure shows the Source Discovery Detection Flow of the secondary wires.

Note: N is the target number.

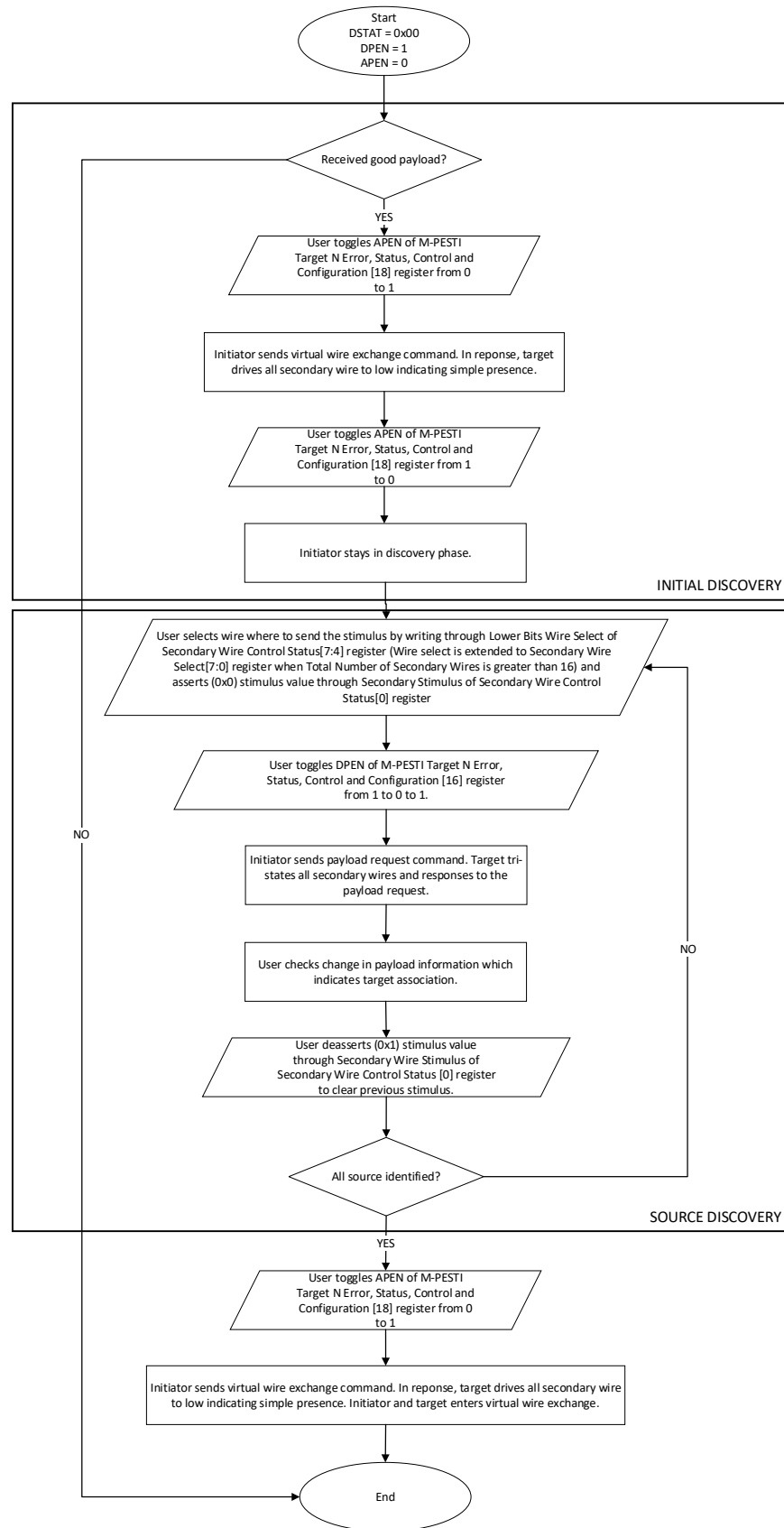


Figure 2.10. Secondary Wire Capability Program Flow

2.5.3.2. Use of I2C GPIO Expander

In this optional source to destination detection, once payload good is received, the software handles the process of determining the power and data couplings. See Section 7 of [Modular-Peripheral Sideband Tunneling Interface \(M-PESTI\) Base Specification Version 1.2 Release Candidate 2](#).

2.5.4. Fanout Feature

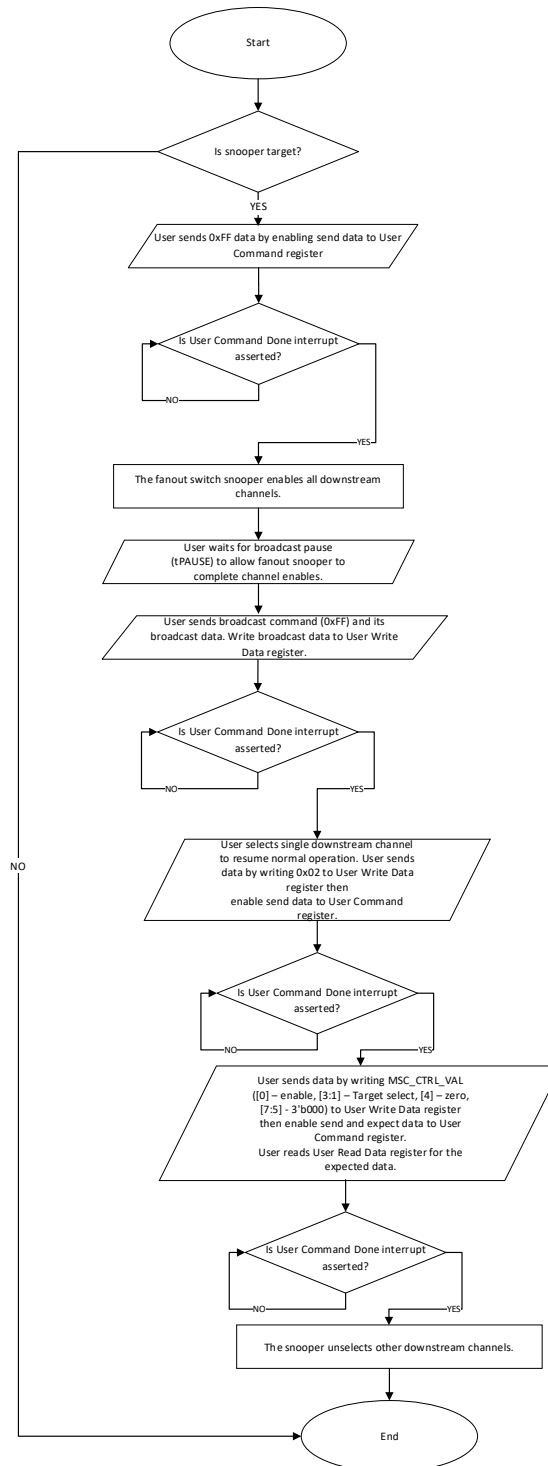


Figure 2.11. Fanout Feature Flow

3. IP Parameter Description

The configurable attributes of the M-PESTI Initiator IP core 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. General

Table 3.1. General Attributes

Attribute	Selectable Values	Description
IP Settings		
Specification Version Support	1.0, 1.2	Specifies the supported specification version.
IP Block ID	0	Specifies the IP Block ID. Display only.
IP Block Version (Major)	2	Specifies the IP Block Major Version
IP Block Version (Minor)	0	Specifies the IP Block Minor Version
IP Block Version	—	Displays the IP Block Version (IP Block Major Version. IP Block Minor Version)
Clock Settings		
System Clock Frequency (MHz)	1-100, 25	Specifies the input system clock of the IP.
System Clock Period (ns)	—	Displays the input system clock period of the IP.
M-PESTI Baud Rate (kHz)	—	Displays the M-PESTI baud rate of the IP.
M-PESTI Baud Period (ns)	—	Displays the M-PESTI period of the IP.
Calculated Clock Divider Setting	—	Displays the calculated clock divider based on the system clock frequency and M-PESTI baud rate.
APB Interface Setting		
APB Data Width	—	Specifies the data width of the APB interface. Display only.
General Settings		
Enable Optional Software Reset Control	Checked, Unchecked	When checked, this enables the optional software reset control. This generates Software Reset register under Global Optional Registers.
Presence Detection Timeout Enable	Checked, Unchecked	When checked, this enables the flag for presence detection timeout.
Presence Detection Timeout (us) (multiples of 2us)	1,000 -2,000	Specifies the presence detection timeout. Applicable only when Presence Detection Timeout Enable is checked.
Number of Target Devices	1-64, 8	Specifies the number of targets sharing one initiator.
Maximum Static Discovery Payload Size	16 , 32, 64, 128	Specifies the maximum size of the discovery payload among all targets.
Total Number of Secondary Wires	0 -(24*Number of Target Devices)	Specifies the total number of secondary wires of all targets. The maximum number of secondary wires per target is 24. Example: Two targets are instantiated. Target 0 has two secondary wires and Target 1 has one secondary wire. Total Number of Secondary Wires attribute must be configured to 3. Note that each device has limited number of input or output ports. Only applicable when selected Specification Version Support is 1.0.
Enable CSR for Secondary Wire	Checked, Unchecked	When checked, this generates a register— Secondary Wire Control Status [2] register—that contains secondary wire value and can be read. Only applicable when selected Specification Version Support is 1.0 and <i>Total Number of Secondary Wires</i> > 0.

Attribute	Selectable Values	Description
Enable Glitch Filter	Checked, Unchecked	When checked, pulses below 50 ns are suppressed.
Allow Multiple User Receive Bytes	Checked, Unchecked	If checked and the expect_data/abort (User Command[17]) is enabled, this parameter allows you to receive multiple bytes of data. Otherwise, user command receives 1 byte.
Maximum User Receive Byte Size	—	Specifies the maximum byte size of user data that can be stored. It is recommended to use the default value to minimize LUT size. However, it is necessary to read received data within 88us. Only applicable when <i>Allow Multiple User Receive Bytes</i> is enabled. Display only.
Target Settings		
Discovery Phase Enable	Checked, Unchecked	When checked, the discovery phase is enabled. This can be re-configured by writing to the Discovery Phase Enable (M-PESTI Target N Error, Status, Control and Configuration 0x00+(N*4)[16]) register.
Discovery Bypass Enable (in hex)	0 – Maximum Number of Target Devices	This is a bitwise configuration for enabling discovery phase bypass. Input value must not be greater than the <i>Maximum Number of Target Devices</i> attribute. Only applicable when selected Specification Version Support is 1.2.
Active Phase Enable	Checked, Unchecked	When checked, the active phase is enabled. This can be re-configured by writing to the Active Phase Enable (M-PESTI Target N Error, Status, Control and Configuration 0x00+(N*4)[18]) register.
Active Phase PEC Enable (in hex)	0 – Maximum Number of Target Devices	This is a bitwise configuration for enabling active phase PEC which PEC byte for each virtual wire input data and virtual wire output data. Input value must not be greater than the <i>Maximum Number of Target Devices</i> attribute. Only applicable when selected Specification Version Support is 1.2.
Memory Settings		
Payload Memory Allocation Option (for single target only)	RAM, Register	Specifies where to store the payload if a single target is selected. Otherwise, the RAM is used for the payload allocation.
Payload Memory Allocation	—	Displays the memory allocation of the payload information.
I/O Setting		
Include I/O Primitive	Checked, Unchecked	When checked, this includes the I/O Primitive instance. The M-PESTI ports, both primary and secondary, are seen as bidirectional I/O. Otherwise, they are seen as tristate M-PESTI ports.

3.2. Virtual Wire Setting

Table 3.2. Virtual Wire Settings

Attribute	Selectable Values	Description
Virtual Wire Settings		
Enable Programmable Virtual Wire	Checked , Unchecked	Allows reconfiguration of Number of Virtual Wire Input and Output Bytes per target. Enabling this attribute generates M-PESTI Target Virtual Wire Configuration registers.
Enable Virtual Wire as Ports (in hex)	0 – Maximum Number of Target Devices	This is a bitwise configuration for enabling virtual wire as physical input and output ports. Input value must not be greater than the <i>Maximum Number of Target Devices</i> attribute.
Target x¹		
Enable Virtual Wire as Ports x ¹	—	Reflects Enable Virtual Wire as Port for Target x. Display only.
Number of Virtual Wire Input Bytes	0-16, 1	Specifies how many virtual wire input bytes for Target x.
Number of Virtual Wire Output Bytes	0-16, 1	Specifies how many virtual wire output bytes for Target x.

Note:

- Attributes below Target x group are configurable per target device where x is the device number which is limited by the *Maximum Number of Target Devices* attribute.

4. Signal Description

The following table lists the input and output signals for the M-PESTI Initiator IP core along with their descriptions.

Table 4.1. Ports Description

Port	Type	Width	Description
System Clock and Reset			
clk_i	Input	1	This signal is the system clock in MHz defined in the <i>System Clock Frequency</i> attribute.
rst_n_i	Input	1	This signal is an asynchronous active low reset.
APB Interface			
apb_psel_i	Input	1	This is the APB interface select signal.
apb_paddr_i	Input	32	This is the APB interface address signal.
apb_pwdata_i	Input	32	This is the APB interface write data signal.
apb_pwrite_i	Input	1	This is the APB interface direction signal. <ul style="list-style-type: none"> 1'b0: Read 1'b1: Write
apb_penable_i	Input	1	This is the APB interface enable signal.
apb_pslverr_o	Output	1	This is the APB error signal. This signal is tied to 1'b0.
apb_pready_o	Output	1	This is the APB interface ready signal. Indicates transfer completion. The completer uses this signal to extend an APB transfer.
apb_prdata_o	Output	32	This is the APB interface read data signal.
M-PESTI Interface			
mpesti_io ¹	Inout	<i>Maximum Number of Target Devices</i>	This signal is a bidirectional M-PESTI signal.
mpesti_i ¹	Input	<i>Maximum Number of Target Devices</i>	This signal is an M-PESTI input signal.
mpesti_oe ¹	Input	<i>Maximum Number of Target Devices</i>	This signal is an M-PESTI output enable signal.
mpesti_o ¹	Output	<i>Maximum Number of Target Devices</i>	This signal is an M-PESTI output signal.
s_mpesti_io ¹	Inout	<i>Total Number of Secondary Wires</i>	Generated when the <i>Total Number of Secondary Wires</i> attribute is not zero. This signal is a bidirectional secondary M-PESTI signal.
s_mpesti_i ¹	Input	<i>Total</i>	Generated when the <i>Total Number of Secondary Wires</i> attribute is not

		<i>Number of Secondary Wires</i>	zero. This signal is a secondary M-PESTI input signal.
s_mpesti_oe ¹	Input	<i>Total Number of Secondary Wires</i>	Generated when the <i>Total Number of Secondary Wires</i> attribute is not zero. This signal is a secondary M-PESTI output enable signal.
s_mpesti_o ¹	Output	<i>Total Number of Secondary Wires</i>	Generated when the <i>Total Number of Secondary Wires</i> attribute is not zero. This signal is a secondary M-PESTI output signal.
Virtual Wire Interface²			
vwire_out_tgt0	Input	<i>Number of Virtual Wire Output Bytes for Target 0</i>	This signal is the Virtual Wire Output of Target 0. Applicable when <i>Enable Virtual Wire as Ports</i> == Checked for Target 0. Note that the bus width is based on the <i>Number of Virtual Wire Output Bytes</i> attribute of the specific target.
vwire_in_tgt0	Output	<i>Number of Virtual Wire Input Bytes for Target 0</i>	This signal is the Virtual Wire Input of Target 0. Applicable when <i>Enable Virtual Wire as Ports</i> == Checked for Target 0. Note that the bus width is based on the <i>Number of Virtual Wire Input Bytes</i> attribute of the specific target.
Interrupt Interface			
int_o	Output	1	This signal is the interrupt signal.

Notes:

1. The *Include I/O Primitive* attribute states the M-PESTI interface ports.
2. This interface is generated when *Enable Virtual Wire as Ports* == Checked. This attribute can be set per target.
3. Applicable only when the Specification Version selected is version 1.0 and there is/are a present secondary wire/s.

5. Register Description

The register address map starts at 0x000 and increments by 4 per register. All register access should be DWORD (32 bit) aligned (bit[1:0]=0). For single M-PESTI target, the *Memory Allocation Option* attribute can be set to either **Register** or **RAM**, which indicates the location to store the payload. On the other hand, RAM is used when using multiple targets.

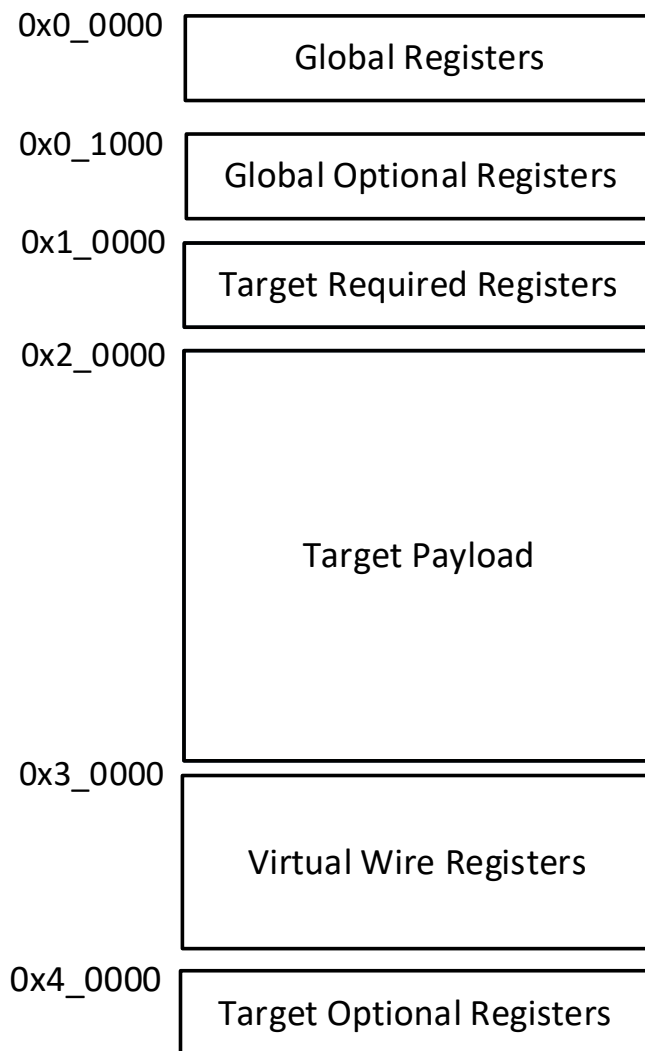


Figure 5.1. Memory Map Allocation

Table 5.1. Register Access Types

Access Type	Access Type Abbreviation	Behavior on Read Access	Behavior on Write Access
Read only	RO	Returns register value.	Ignore write access.
Write only	WO	Returns 0.	Updates register value.
Read and write	RW	Returns register value.	Updates register value.
Read and write 1 to clear	RW1C	Returns register value.	Writing 1'b1 on register bit clears the bit to 1'b0. Writing 1'b0 on register bit is ignored.

Note: All bits without field assignments are treated as Reserved with read-only access.

Table 5.2. Register Address Map

Offset APB	Register Name	Access Type	Description
Global Registers (0x0_0000)			
0x00	IP Information	RO	Specifies IP ID and block version.
0x04	Configuration 0	RO	Specifies configuration of the IP.
0x08	Configuration 1	RO	Specifies configuration of the IP.
0x10	Global Software Reset	RW	Sets the global software reset of both IP core and the CSR.
0x14	Discovery Payload: Page Select	RO	Selects discovery payload page.
0x18-0x1C	Reserved	RO	Reserved addresses.
0x20	User Command Status	RW1C	Sets status of the user command.
0x24	User Command	RW	Sets the command from the user.
0x28	User Write Data	RW	Indicates the data to be sent to the Target.
0x2C	User Read Data	RO	Indicates the data received from the Target.
Global Optional Registers (0x0_1000)			
0x00	Clock Configuration	RW	Sets the configuration of the clock divider.
0x04	Software Reset	RW	Sets the separate optional CSR and IP core software reset. Note that this is only generated when <i>Enable Optional Software Reset Control</i> attribute is checked.
0x08	Global Interrupt Status	RW1C	Indicates the status of the global interrupts.
0x0C	Global Interrupt Enable	RW	Indicates the global interrupts that can trigger the interrupt output port.
0x10	Global Interrupt Set	WO	Sets the global interrupt status.
0x14	User Command Interrupt Enable	RW	Indicates the user command interrupts that can trigger the interrupt output port.
0x18	User Command Interrupt Set	WO	Sets the user command interrupt status.
0x1C-0x3C	Reserved	RO	Reserved addresses.
0x40	Secondary Wire Control Status	RW	Sets the secondary wire stimulus, status and wire select for secondary wires less than 16. Note that this is generated when there is an existing secondary wire that is determined by the <i>Total Number of Secondary Wires</i> attribute.
0x44	Secondary Wire Select	RW	Sets the upper bits of the wire select. This is generated when the number of secondary wires is greater than 16. Note that this is generated when there is an existing secondary wire that is determined by the <i>Total Number of Secondary Wires</i> attribute.
Target Required Registers (0x1_0000)			
0x00+(4*N)	MPESTI Target N Error, Status, Control and Configuration	RW	Indicates the error, status, control and configuration of each M-PESTI target device.
Payload (0x2_0000)			
0x2_0000+(Maximum Static Discovery Payload Size * N)	Target N Payload	RO	Indicates device information of each M-PESTI target.

Offset APB	Register Name	Access Type	Description
Target Virtual Wire Registers^{1,2} (0x3_0000)			
0x00+(N*32)	M-PESTI Target N Virtual Wire Input	RO	Indicates the data received from the Target via Virtual Wire. Each target has an allocation of a maximum of 16 bytes of virtual wire input data.
0x10+(N*32)	M-PESTI Target N Virtual Wire Output	RW	Indicates the data to be sent to the Target via Virtual Wire. Each target has an allocation of a maximum of 16 bytes of virtual wire output data.
Target Optional Registers (0x4_0000)			
0x4_0000+(N*4)	M-PESTI Target N Virtual Wire Configuration	RW	Sets the number of virtual input and output bytes of the M-PESTI target device. Applicable when <i>Enable Programmable Virtual Wire Settings</i> == Checked.

Note:

1. N is the target number
2. Virtual Wire registers are generated based on the *Number of Virtual Wire Input Bytes* and *Number of Virtual Wire Output Bytes* attributes per target. Unused registers are considered as reserved addresses.

5.1. Global Registers (0x0_0000)

5.1.1. IP Information 0x00

Table 5.3. IP Information 0x00

Field	Name	Access	Width	Reset
[31:16]	IP Block version	RO	16	Refer to <i>IP Block version</i> attribute.
[15:0]	IP Block ID	RO	16	Refer to <i>IP Block ID</i> attribute.

- IP Block version [31:16]
Format is Major. Minor in Binary-Coded Decimal (BCD) coding.
- IP Block ID [15:0]
Not a GUID – just uniquely ID that this is M-PESTI block to connect the dots between JSON block ID and actual hardware

5.1.2. Configuration 0 0x04

Table 5.4. Configuration 0 0x04

Field	Name	Access	Width	Reset
[7:0]	Maximum Number of Targets Supported	RO	8	Refer to <i>Maximum Number of Target Devices</i> attribute.

Maximum Number of Targets Supported [7:0]

Selected maximum number of targets supported (e.g. 0x01 == 1 target, ..., 0x00 == 256 targets). Refer to *Maximum Number of Target Devices* attribute for the maximum number of targets supported.

5.1.3. Configuration 1 0x08

Table 5.5. Configuration 1 0x08

Field	Name	Access	Width	Reset
[7:0]	Maximum Payload Size Supported	RO	8	Refer to <i>Maximum Static Discovery Payload Size</i> attribute.

Maximum Payload Size Supported [7:0]

Selected values are multiples of 8 bytes, minimum set to 16 bytes (e.g. 0x2 == 16 bytes, ..., 0x20 == 256 bytes). Refer to *Maximum Static Discovery Payload Size* attribute for the maximum number of targets supported.

5.1.4. Global Software Reset 0x10

Table 5.6. Global Software Reset 0x10

Field	Name	Access	Width	Reset
[31:1]	Reserved	RO	31	0x00
[0]	Global Software Reset	RW	1	0x0

Global Software Reset [0]

This global software reset clears both the CSR and IP Core and sets its reset value. This does not clear the virtual wire data and payload information stored in the memory module.

5.1.5. Discovery Payload Page Select 0x14

Table 5.7. Discovery Payload Page Select 0x14

Field	Name	Access	Width	Reset
[31:4]	Reserved	RO	28	0x00
[3:0]	Discovery Payload Page Select	RO	4	0x00

Discovery Payload Page Select [3:0]

This is used when Discovery payload is greater than 128 bytes (1 Page = 128 bytes), maximum of 16 windows per target.

5.1.6. User Command Status 0x20

Table 5.8. User Command Status 0x20

Field	Name	Access	Width	Reset
[7]	User Command Receive Timeout	RW1C	1	0x0
[6]	User Command Done	RW1C	1	0x0
[5]	User Command Received Data Available	RW1C	1	0x0
[4:0]	Reserved	RO	5	0x00

User Receive Timeout [7]

- 0 – No timeout to flag.
- 1 – Initiator did not receive the expected response from target. Receive timeout is 250 ms.

User Command Done [6]

- 0 – No user command to execute or user command is not yet executed.
- 1 – Flags that the user command has been executed.

User Command Received Data Available [5]

- 0 – No updated data in the user read data register.
- 1 – An updated data is available in the user read data register.

5.1.7. User Command 0x24

Table 5.9. User Command 0x24

Field	Name	Access	Width	Reset
[31:19]	Reserved	RO	13	0x00
[18]	Send Data	RW	1	0x0
[17]	Expect Data/ Abort	RW	1	0x0
[16]	Broadcast	RW	1	0x0
[15:8]	Byte Length	RW	8	0x00
[7:0]	Target Select	RW	8	0x00

Send Data [18]

This bit auto-clears once done

- 0 – Initiator is not sending data to any target.
- 1 – Initiator is sending data to the target/s.

Expect Data/ Abort [17]

Note: This bit auto-clears once done when Allow Multiple User Receive Bytes parameter is disabled. Otherwise, it is necessary to write 0 to clear.

When broadcast = 1,

- 0 – Wait for the current transaction to finish before proceeding to broadcast.
- 1 – Abort the current transaction then proceed to broadcast.

When broadcast = 0,

- 0 – No response is expected from the target.
- 1 – Response is expected from the target after sending the command.

Broadcast [16]

- 0 – Sends broadcast command to a single target.
- 1 – Sends broadcast command to all targets.

Byte Length [15:8]

Specifies the number of expected read data bytes.

Note: If the *Allow Multiple User Receive Bytes* attribute is enabled, up to 16 bytes can be received; otherwise, only a single byte is received regardless of the specified length.

Target Select [7:0]

Selects a specific target location to send the broadcast command. Refer to *Maximum Number of Target Devices* attribute for the allowable supported targets.

5.1.8. User Write Data 0x28

Table 5.10. User Write Data 0x28

Field	Name	Access	Width	Reset
[31:8]	Reserved	RO	24	0x00
[7:0]	User Write Data	RW	8	0x00

User Write Data [7:0]

When sent data (User Command [18] register) is set to 1, this register stores the data to be transmitted to the target/s.

5.1.9. User Read Data 0x2C

Table 5.11. User Read Data 0x2C

Field	Name	Access	Width	Reset
[31:8]	Reserved	RO	24	0x00
[7:0]	User Read Data	RO	8	0x00

User Read Data [7:0]

This register stores the data received from the target.

5.2. Global Optional Registers (0x0_1000)

5.2.1. Clock Configuration 0x00

Table 5.12. Clock Configuration 0x00

Field	Name	Access	Width	Reset
[7:0]	Clock Pulse Width	RW	8	Refer to the <i>Calculated Clock Divider Setting</i> attribute.

Clock Pulse Width [7:0]

- Default – calculated divider setting based on the default *System Clock Frequency* attribute.
- Ranged from clock divider setting 2-200.

Notes:

1. Bit 0 is RO. Given that, clock pulse width configuration can only be set as multiples of 2.
2. Writing in this register changes the default clock divider setting calculated from the System Clock Frequency attribute and may cause improper behavior when wrongly configured. This affects the M-PESTI baud rate.

5.2.2. Software Reset 0x04

Table 5.13. Software Reset 0x04

Field	Name	Access	Width	Reset
[7]	No Auto Clear	RW	1	0x0
[6:2]	Reserved	RO	5	0x00
[1]	CSR Reset	RW	1	0x0
[0]	IP Core Reset	RW	1	0x0

Note: This register is generated when *Enable Optional Software Reset Control* attribute is checked. Otherwise, this register is treated as reserved and will have a RO access.

No Auto Clear [7]

- 0 – Software reset is set to auto clear.
- 1 – Software reset value remains until it is reprogrammed.

CSR Reset [1]

Active high software reset that resets the Control, Configuration, and Status Registers to its default values. This does not clear the virtual wire data and payload information stored in the memory module.

IP Core Reset [0]

Active high software reset that resets all blocks except the Control, Configuration, and Status Registers. This does not clear the virtual wire data and payload information stored in the memory module.

5.2.3. Global Interrupt Status 0x08

Table 5.14. Global Interrupt Status 0x08

Field	Name	Access	Width	Reset
[31:7]	Reserved	RO	25	0x00
[6]	Virtual Wire Input Update Detected	RW1C	1	0x0
[5]	Payload Good Detected	RW1C	1	0x0
[4]	M-PESTI Ready Detected	RW1C	1	0x0
[3]	Target Presence Detected	RW1C	1	0x0
[2]	Active Phase Error Detected	RW1C	1	0x0
[1]	Discovery Phase Error Detected	RW1C	1	0x0
[0]	Presence Timeout Detected	RW1C	1	0x0

Virtual Wire Input Update Detected [6]

This bit indicates that at least one of the targets has an updated virtual wire input register.

- 0 – Virtual Wire Input data is not updated.
- 1 – Virtual Wire Input data is updated.

Payload Good Detected [5]

This bit indicates that at least one of the targets received a good payload.

- 0 – Initiator received a payload with a bad checksum or wrong parity bit from the target.
- 1 – Initiator received a payload with a good checksum from the target.

M-PESTI Ready Detected [4]

This bit indicates that at least one of the targets is M-PESTI ready.

- 0 – Initiator did not detect any M-PESTI ready target.
- 1 – Initiator detects one or more M-PESTI ready target/s.

Target Presence Detected [3]

This bit indicates that at least one of the targets did UART break release and is present.

- 0 – Initiator did not detect any target presence.
- 1 – Initiator detects one or more target/s is/are present.

Active Phase Error Detected [2]

This bit indicates that at least one of the targets has an active phase error.

- 0 – No errors are detected from any targets.
- 1 – Error (this flags active phase receive timeout, parity error and/or framing error) is detected from target(s).

Discovery Phase Error Detected [1]

This bit indicates that at least one of the targets has a discovery phase error.

- 0 – No errors are detected from any targets.

- 1 – Error (this flags CRC error, parity error and/or framing error) is detected from target(s).

Presence Timeout Detected [0]

Note: This bit is generated when Presence Detection Timeout Enable attribute is checked. Otherwise, this bit is treated as reserved and will have a RO access.

This bit indicates that at least one of the targets reached the presence timeout set in the *Presence Detection Timeout* attribute.

- 0 – Did not reached the set presence timeout.
- 1 – Reached the set presence timeout.

5.2.4. Global Interrupt Enable 0x0C

Table 5.15. Global Interrupt Enable 0x0C

Field	Name	Access	Width	Reset
[31:7]	Reserved	RO	25	0x00
[6]	Virtual Wire Input Update Interrupt Enable	RW	1	0x0
[5]	Payload Good Interrupt Enable	RW	1	0x0
[4]	M-PESTI Ready Interrupt Enable	RW	1	0x0
[3]	Target Presence Interrupt Enable	RW	1	0x0
[2]	Active Phase Error Interrupt Enable	RW	1	0x0
[1]	Discovery Phase Error Interrupt Enable	RW	1	0x0
[0]	Presence Timeout Interrupt Enable	RW	1	0x0

Global Interrupt Enable [6:0]

When enabled, the corresponding interrupt status asserts the interrupt port.

5.2.5. Global Interrupt Set 0x10

Table 5.16. Global Interrupt Set 0x10

Field	Name	Access	Width	Reset
[31:7]	Reserved	RO	25	0x00
[6]	Virtual Wire Input Update Interrupt Set	WO	1	0x0
[5]	Payload Good Interrupt Set	WO	1	0x0
[4]	M-PESTI Ready Interrupt Set	WO	1	0x0
[3]	Target Presence Interrupt Set	WO	1	0x0
[2]	Active Phase Error Interrupt Set	WO	1	0x0
[1]	Discovery Phase Error Interrupt Set	WO	1	0x0
[0]	Presence Timeout Interrupt Set	WO	1	0x0

Global Interrupt Set [6:0]

This register is mainly for testing purposes.

When enabled, the corresponding interrupt status is asserted. Writing zero in this register is ignored.

5.2.6. User Command Interrupt Enable 0x14

Table 5.15. User Command Interrupt Enable 0x14

Field	Name	Access	Width	Reset
[7]	User Command Receive Timeout Interrupt Enable	RW	1	0x0
[6]	User Command Done Interrupt Enable	RW	1	0x0
[5]	User Command Received Data Available Interrupt Enable	RW	1	0x0
[4:0]	Reserved	RO	5	0x00

User Command Interrupt Enable [7:5]

When enabled, the corresponding user command status asserts the interrupt port.

5.2.7. User Command Interrupt Set 0x18

Table 5.16. User Command Interrupt Set 0x18

Field	Name	Access	Width	Reset
[7]	User Command Receive Timeout Interrupt Set	WO	1	0x0
[6]	User Command Done Interrupt Set	WO	1	0x0
[5]	User Command Received Data Available Interrupt Set	WO	1	0x0
[4:0]	Reserved	RO	5	0x00

User Command Interrupt Set [7:5]

This register is mainly for testing purposes.

When enabled, the corresponding user command status is asserted. Writing zero in this register is ignored.

5.2.8. Secondary Wire Control Status 0x40

Table 5.17. Secondary Wire Control Status 0x40

Field	Name	Access	Width	Reset
[7:4]	Lower Bits Wire Select	RW	4	0x0
[3]	Reserved	RO	1	0x0
[2]	Secondary Wire	RO	1	0x1
[1]	Reserved	RO	1	0x0
[0]	Secondary Wire Stimulus	RW	1	0x1

Note: This register is generated when there is/are configured existing secondary wires in the *Total Number of Secondary Wires* attribute. Otherwise, this bit is treated as reserved and will have RO access. The reset value of the register is 0x00.

Lower Bits Wire Select [7:4]

This register is generated when the *Total Number of Secondary Wires* attribute is less than 16. This is the lower bits of the wire select from the Secondary Wire Select Register.

Secondary Wire [2]

This is generated when the *Enable CSR for Secondary Wire* attribute is checked. This reflects the value of the selected secondary wire.

Secondary Wire Stimulus [0]

This bit is an active low signal and sets the value of the selected secondary wire stimulus. This register must be cleared by writing 0x1 before sending stimulus to the next secondary wire.

5.2.9. Secondary Wire Select 0x44

Table 5.18. Secondary Wire Select 0x44

Field	Name	Access	Width	Reset
[7:0]	Upper Bits Wire Select	RW	8	0x00

Note: This register is generated when the *Total Number of Secondary Wires* attribute exceeds 16. This is the upper bits of the wire select from the Secondary Wire Select Register.

5.3. Target Required Registers (0x1_0000)

5.3.1. M-PESTI Target N Error, Status, Control and Configuration 0x00+(N*4)

Table 5.19. M-PESTI Target N Error, Status, Control and Configuration 0x00+(N*4)

Field	Name	Access	Width	Reset
[31]	Enable Target VWIRE Input Update Interrupt Status	RW	1	0x0
[30]	Enable Target Payload Good Interrupt Status	RW	1	0x0
[29]	Enable Target M-PESTI Ready Interrupt Status	RW	1	0x0
[28]	Enable Target Presence Interrupt Status	RW	1	0x0
[27]	Enable Active Phase Error Interrupt Status	RW	1	0x0
[26]	Enable Discovery Phase Error Interrupt Status	RW	1	0x0
[25]	Enable Presence Timeout Detection Interrupt Status	RW	1	0x0
[24]	Broadcast subscription	RW	1	0x1
[23:20]	Reserved	RO	4	0x00
[19]	Active Phase PEC Enable	RW	1	Refer to the <i>Active Phase PEC Enable (in hex)</i> attribute.
[18]	Active Phase Enable	RW	1	Refer to the <i>Active Phase Enable</i> attribute.
[17]	Discovery Phase Bypass	RW	1	Refer to the <i>Discovery Bypass Enable (in hex)</i> attribute.
[16]	Discovery Phase Enable	RW	1	Refer to the <i>Discovery Phase Enable</i> attribute.
[15:11]	Reserved	RO	5	0x00
[10]	Active Phase CRC Error	RW1C	1	0x0
[9]	Discovery Phase CRC Error	RW1C	1	0x0
[8]	Parity Error	RW1C	1	0x0
[7]	Discovery Retry Rollover	RW1C	1	0x0
[6]	Discovery Phase Error	RW1C	1	0x0
[5]	Presence Detection Timeout	RW1C	1	0x0
[4]	Active Phase Receive Timeout	RW1C	1	0x0
[3]	Active Phase Error	RW1C	1	0x0
[2]	Active Phase Status	RO	1	0x0

Field	Name	Access	Width	Reset
[1:0]	Discovery Status	RO	2	0x0

Note: N is the target number.

Enable Target VWIRE Input Update Interrupt Status [31]

When set to high, it asserts the corresponding global interrupt status.

- 1 – Target VWIRE Input Update interrupt status is enabled.
- 0 – Target VWIRE Input Update interrupt status is disabled.

Enable Target Payload Good Interrupt Status [30]

When set to high, it asserts the corresponding global interrupt status.

- 1 – Target Payload Good interrupt status is enabled.
- 0 – Target Payload Good interrupt status is disabled.

Enable Target M-PESTI Ready Interrupt Status [29]

When set to high, it asserts the corresponding global interrupt status.

- 1 – Target M-PESTI Ready interrupt status is enabled.
- 0 – Target M-PESTI Ready interrupt status is disabled.

Enable Target Presence Interrupt Status [28]

When set to high, it asserts the corresponding global interrupt status.

- 1 – Target Presence interrupt status is enabled.
- 0 – Target Presence interrupt status is disabled.

Enable Active Phase Error Interrupt Status [27]

When set to high, it asserts the corresponding global interrupt status.

- 1 – Active Phase Error interrupt status is enabled.
- 0 – Active Phase Error interrupt status is disabled.

Enable Discovery Phase Error Interrupt Status [26]

When set to high, it asserts the corresponding global interrupt status.

- 1 – Discovery Phase Error interrupt status is enabled.
- 0 – Discovery Phase Error interrupt status is disabled.

Enable Presence Timeout Detection Interrupt Status [25]

Note: This bit is generated when Presence Detection Timeout Enable attribute is checked. Otherwise, this bit is treated as reserved and will have a RO access.

When set to high, it asserts the corresponding global interrupt status.

- 1 – Presence Timeout Detection interrupt status is enabled.
- 0 – Presence Timeout Detection interrupt status is disabled.

Broadcast subscription [24]

- 1 – Allows target to receive broadcast messages.
- 0 – Disables sending broadcast messages.

Active Phase PEC Enable [19]

- 0 – No PEC during active phase.
- 1 – There is an active phase PEC during the active phase.

Active Phase Enable [18]

- 0 – The initiator will not send active phase command.
- 1 – The initiator will send active phase command to transact through the virtual wire exchange given that the discovery status is M-PESTI is present with good payload (2'b10).

Discovery Phase Bypass [17]

- 0 – Discovery phase bypass is disabled. Successful M-PESTI payload received is required prior to active phase.
- 1 – Discovery phase bypass is enabled. This allows active phase without successful M-PESTI payload received.

Discovery Phase Enable [16]

- 0 – The initiator will not send discovery phase request command.
- 1 – The initiator will send discovery phase request command.

Active Phase CRC Error [10]

- 0 – No CRC error is detected during active phase.
- 1 – CRC error is detected during active phase.

Discovery Phase CRC Error [9]

Note: In discovery phase, this becomes sticky once Discovery Retry Rollover asserted (three unsuccessful retries have been reached).

- 0 – No CRC error is detected during discovery phase.
- 1 – CRC error is detected during discovery phase.

Parity Error [8]

Note: This becomes sticky once Discovery Retry Rollover asserted (three unsuccessful retries have been reached).

- 0 – No parity error is detected.
- 1 – Parity error is detected.

Discovery Retry Rollover [7]

- 0 – Initiator has not reached the three discovery retries.
- 1 – Initiator has attempted discovery for three unsuccessful retries (initial + two retries).

Discovery Phase Error [6]

Note: This becomes sticky once Discovery Retry Rollover asserted (three unsuccessful retries have been reached).

- 0 – No discovery phase error.
- 1 – Discovery phase error has been flagged, either parity, framing, or CRC error.

Presence Timeout Detection [5]

- 0 – No presence timeout detected.
- 1 – Presence timeout reached the set value in the *Presence Detection Timeout* attribute.

Active Phase Receive Timeout [4]

- 0 – No active phase received timeout error is detected.
- 1 – Active phase received timeout error is detected.

Active Phase Error [3]

- 0 – No active phase error is detected.
- 1 – Active phase error is detected (includes receive timeout, parity error, framing error and PEC error if applicable).

Active Phase Status [2]

- 0 – Target device is in the discovery phase.
- 1 – The Initiator and Target communication is in the active phase.

Discovery Status [1:0]

- 00 – Absent.
- 01 – Present.
- 10 – M-PESTI is present with good payload.
- 11 – M-PESTI is present without successful payload yet.

5.4. Target Payload (0x2_0000)

The maximum configurable size of the payload per target is indicated by the *Maximum Static Discovery Payload Size* attribute. This payload information is stored in an uninitialized memory, and reset does not clear its contents.

Target 0	—TARGET_DISCOVERY_BASE
Target 1	—TARGET_DISCOVERY_BASE + (Maximum Static Discovery Payload Size * 1)
Target 2	—TARGET_DISCOVERY_BASE + (Maximum Static Discovery Payload Size * 2)
Target 3	—TARGET_DISCOVERY_BASE + (Maximum Static Discovery Payload Size * 3)
	—TARGET_DISCOVERY_BASE + (Maximum Static Discovery Payload Size * 4)
Target N	—TARGET_DISCOVERY_BASE + (Maximum Static Discovery Payload Size * (N-1))
	—TARGET_DISCOVERY_BASE + (Maximum Static Discovery Payload Size * N)

Figure 5.2. Target Payload Allocation

5.5. Target Virtual Wire Registers (0x3_0000)

Each target has two (2) sixteen (16) bytes allocation for virtual wire input data and virtual wire output data.

Target 0	—TARGET_VIRTUAL_WIRES_BASE
Target 1	—TARGET_VIRTUAL_WIRES_BASE + 32
Target 2	—TARGET_VIRTUAL_WIRES_BASE + 64
Target 3	—TARGET_VIRTUAL_WIRES_BASE + 96
	—TARGET_VIRTUAL_WIRES_BASE + 128
Target N	—TARGET_VIRTUAL_WIRES_BASE + 32 * (N - 1)
	—TARGET_VIRTUAL_WIRES_BASE + 32 * N

Figure 5.3. Target Virtual Wire Register Allocation

5.5.1. M-PESTI Target N Virtual Wire Input 0x00+(N*32)

Table 5.20. M-PESTI Target N Virtual Wire Input 0x00+(N*32)

Field	Name	Access	Width	Reset
[31:0]	Virtual Wire Input	RO	32	This information is stored in an uninitialized memory module.

Note: N is the target number.

Virtual Wire Input [31:0]

This register stores the data sent by the target.

This information is stored in an uninitialized memory module, and reset does not clear its contents.

5.5.2. M-PESTI Target N Virtual Wire Output 0x10+(N*32)

Table 5.21. M-PESTI Target N Virtual Wire Output 0x10+(N*32)

Field	Name	Access	Width	Reset
[31:0]	Virtual Wire Output	if Enable Virtual Wire as Ports, RO, else, RW	32	This information is stored in an uninitialized memory module.

Note: N is the target number.

Virtual Wire Output [31:0]

This register stores the data sent to the target. Unused bits of this register should be ignored.

This information is stored in an uninitialized memory module, and reset does not clear its contents.

5.6. Target Optional Registers (0x4_0000)

5.6.1. M-PESTI Target N Status 0x00+(N*16)

Table 5.22. M-PESTI Target N Status 0x00+(N*16)

Field	Name	Access	Width	Reset
[31:4]	Reserved	RO	30	0x00
[3]	Discovery Payload Size Configuration Overflow	RO	1	0x0
[2]	Active Phase PEC Configuration Mismatch	RO	1	0x0
[1:0]	Target Current Phase	RO	2	0x0

Note: N is the target number.

Discovery Payload Size Configuration Overflow[3]

- 0 – The parsed Discovery Payload Size from the payload information is within the set *Maximum Static Discovery Payload Size* configuration.
- 1 – The parsed Discovery Payload Size from the payload information is greater than the set *Maximum Static Discovery Payload Size* configuration.

Active Phase PEC Configuration Mismatch [2]

- 0 – The parsed Active Phase PEC Enable from the payload information and the Active Phase Enable target configuration are matched.
- 1 – The parsed Active Phase PEC Enable from the payload information, and the Active Phase Enable target configuration are mismatched.

Target Current Phase [1:0]

- 00 – Reset.
- 01 – UNUSED.
- 10 – Discovery phase.
- 11 – Active phase.

5.6.2. M-PESTI Target N Virtual Wire Configuration 0x04+(N*16)

Table 5.23. M-PESTI Target N Virtual Wire Configuration 0x04+(N*16)

Field	Name	Access	Width	Reset
[12:8]	Number of Virtual Wire Output Bytes	RW	5	Refer to the <i>Number of Virtual Wire Output Bytes</i> attribute.
[7:5]	Reserved	RO	3	0x00.
[4:0]	Number of Virtual Wire Input Bytes	RW	5	Refer to the <i>Number of Virtual Wire Input Bytes</i> attribute.

Note: N is the target number. Allowed reconfiguration values must be less than or equal to the set Number of Virtual Wire Output Bytes and Number of Virtual Wire Output Bytes attributes.

This register becomes read-only when *Enable Programmable Virtual Wire* attribute is unchecked.

6. Example Design

The M-PESTI Initiator IP example design allows you to compile, simulate, and test the M-PESTI Initiator IP on the following Lattice evaluation boards:

- MachXO5-NX Development Board

6.1. Example Design Supported Configuration

Table 6.1. M-PESTI Initiator IP Configuration Supported by the Example Design

M-PESTI Initiator IP GUI Parameter	M-PESTI Initiator IP Configuration Supported in the Example Demo Design
IP Settings	
Specification Version Support	1.2
IP Block ID	0 (Display only)
IP Block Version (Major)	2 (Display only)
IP Block Version (Minor)	0 (Display only)
IP Block Version	2.0 (Display only)
Clock Settings	
System Clock Frequency (MHz)	25
System Clock Period (ns)	40 (Display only)
M-PESTI Baud Rate (kHz)	250 (Display only)
M-PESTI Baud Period (ns)	4,000 (Display only)
Calculated Clock Divider Setting	50 (Display only)
APB Interface Setting	
APB Data Width	32 (Display only)
General Settings	
Enable Optional Software Reset Control	Unchecked
Presence Detection Timeout Enable	Unchecked
Presence Detection Timeout (us) (multiples of 2us)	1000
Maximum Number of Target Devices	2
Maximum Static Discovery Payload Size	16
Enable Glitch Filter	Unchecked
Allow Multiple User Receive Bytes	Checked
Maximum User Receive Byte Size	16 (Display only)
Target Settings	
Discovery Phase Enable	Checked
Discovery Bypass Enable (in hex)	0
Active Phase Enable	Checked
Active Phase PEC Enable (in hex)	0
Memory Settings	
Payload Memory Allocation Option (for single target only)	RAM
Payload Memory Allocation	RAM (Display only)
I/O Setting	
Include I/O Primitive	Checked
Virtual Wire Settings	
Enable Programmable Virtual Wire	Checked
Enable Virtual Wire as Ports (in hex)	0
Target 0	
Number of Virtual Wire Input Bytes	1

M-PESTI Initiator IP GUI Parameter	M-PESTI Initiator IP Configuration Supported in the Example Demo Design
Number of Virtual Wire Output Bytes	1
Target 1	
Number of Virtual Wire Input Bytes	2
Number of Virtual Wire Output Bytes	2

6.2. Overview of the Example Design and Features

The example design discussed in this section is created using the RISC-V MC SoC project template in the Lattice Propel Development Suite. The generated project includes the following components:

- Processor – RISC-V MC w/ PIC/TIMER
- GPIO
- Asynchronous SRAM
- UART – Serial port
- Phase-locked loop (PLL)
- Glue logic

M-PESTI Initiator and M-PESTI Targets are instantiated and connected in the project as shown in the following figure. In this example design, both initiator and targets are instantiated in the same system. In actual hardware or use case, the M-PESTI Initiator IP core can be connected to external target devices.

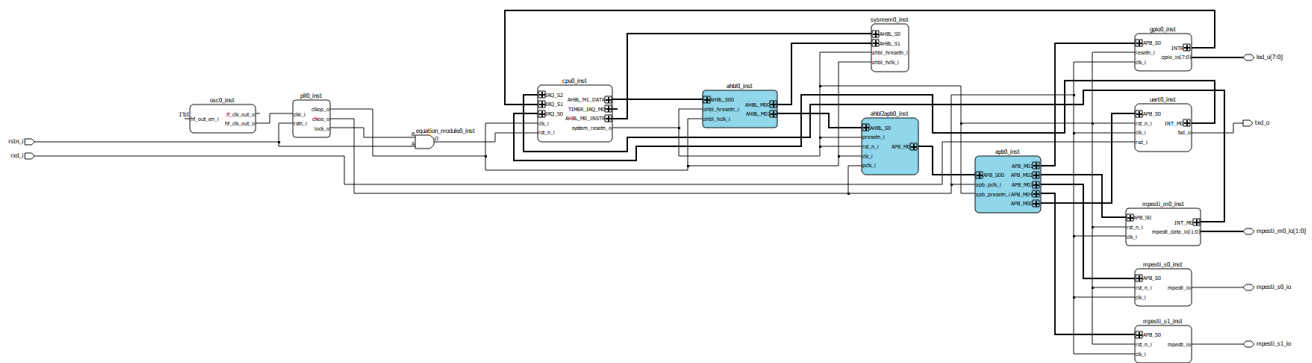


Figure 6.1. M-PESTI Initiator IP Core in an SoC Project

An Embedded C/C++ project is also created in the Propel software to enable the development and debugging application code for different IP features. The M-PESTI Initiator IP core features can be tested by initiating discovery and active phase transactions and sending user commands. Runtime configuration of IP core and feature testing can be done through C-Code Test Routine. The following figure shows an example routine for reading the M-PESTI discovery payload.

```

unsigned int mpesti_mst_rd_disc_payload(mpesti_mst_handle_t *handle, unsigned int *data_buf)
{
    unsigned int status;
    int cnt;

    if(handle->base_addr != ZERO)
    {
        for(cnt=ZERO; cnt<handle->disc_payload_size; cnt++)
        {
            *(unsigned int *)(data_buf + cnt) = *(volatile unsigned int *)(handle->base_addr + MSTADR_INTMEM_BASE + cnt);
        }
        status=SUCCESS;
    }
    else
    {
        status=FAILURE;
    }
    return status;
}

```

Figure 6.2. Sample C Code Test Routine

6.3. Example Design Components

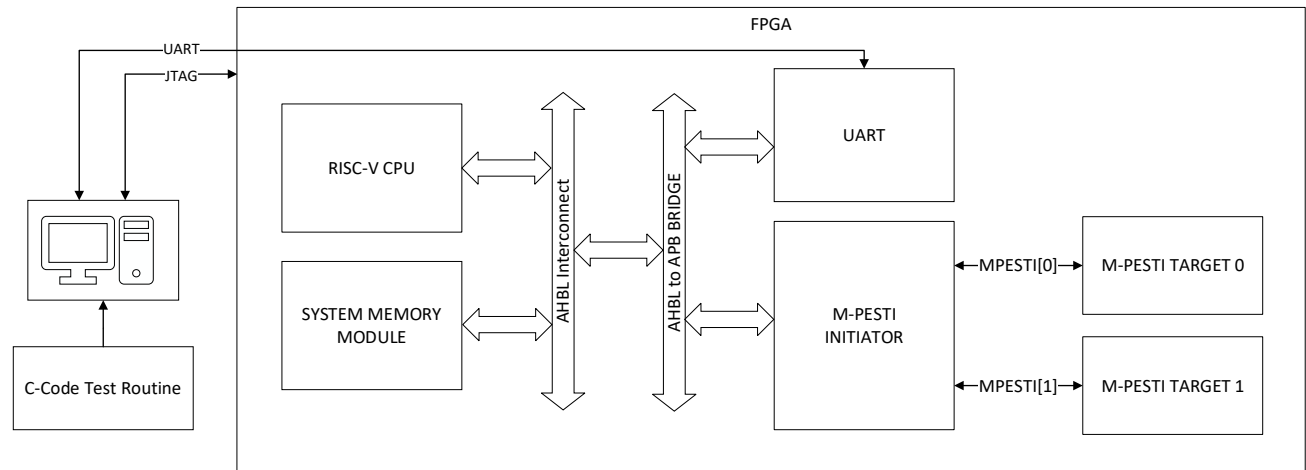


Figure 6.3. M-PESTI Initiator IP Example Design Block Diagram

The M-PESTI Initiator example design includes the following blocks:

- RISC-V CPU – Passes the C-Code Test Routing from system memory to system bus and handles interrupts.
- Memory module – Contains command to be done for testing.
- System bus – AHB-Lite systems bus for transfers between memory and IP.
- M-PESTI Initiator IP.
- M-PESTI target devices.

6.4. Generating the Example Design

To generate the example design, follow these steps:

1. Launch the Lattice Propel software and set your workspace directory.
2. In the Propel software, create a new Lattice SoC Design Project. Click **File > New > Lattice SoC Design Project**.
3. The Create SoC Project window opens.
4. In Template Design window, select Scalable RISC-X SoC Project. Click Next.

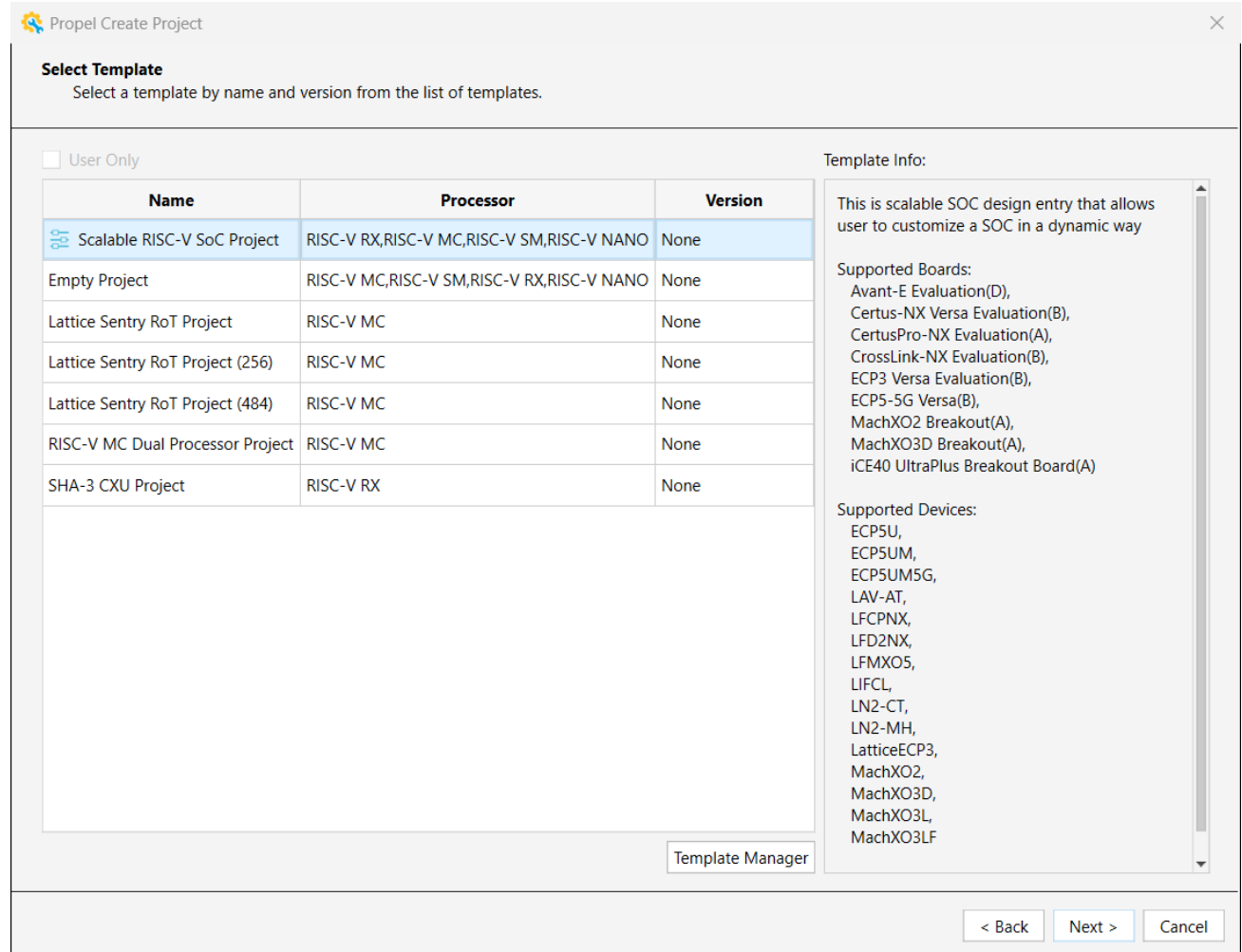


Figure 6.4. Select Template

5. In the **System Application Level** section, select **Generic Micro Controller Project (RISC-V MC SoC) Project**. Click **Next**.

Propel Create Project

System Application Level

(choose an application)

This template enables RTOS level application. The SOC is based on RISC-V RX and uses AXI as main system bus

☐ Real-Time Operation System(RISC-V RX)

This template enables bare metal level application. The SOC is based on RISC-V MC and uses AHBL as main system bus. Considering RISC-V MC is highly configurable, this template is suitable for almost all the available devices

☒ General Micro Controller(RISC-V MC)

This template enables bare metal level application. The SOC is based on RISC-V SM and uses AHBL as main system bus. RISC-V SM is recommended for resource constrained devices

☐ General State Machine(RISC-V SM)

This template enables bare metal level application. The SOC is based on RISC-V NANO and uses AHBL as main system bus. RISC-V NANO is recommended for certain use cases that resource usage is highly sensitive

☐ Glue Logic(RISC-V Nano)

< Back Next > Cancel

Figure 6.5. Select Template

6. In **Device Select** window, select the device or board that you will use. In figure below, the device is set to **LFMX05-25-9BBG400C** as MachXO5-NX Development Board is used in hardware testing. Click **Next**. In the next window, click **Confirm** then set Number of GPIO to x and set UART to 1. Click **Next**. Click **Finish**.

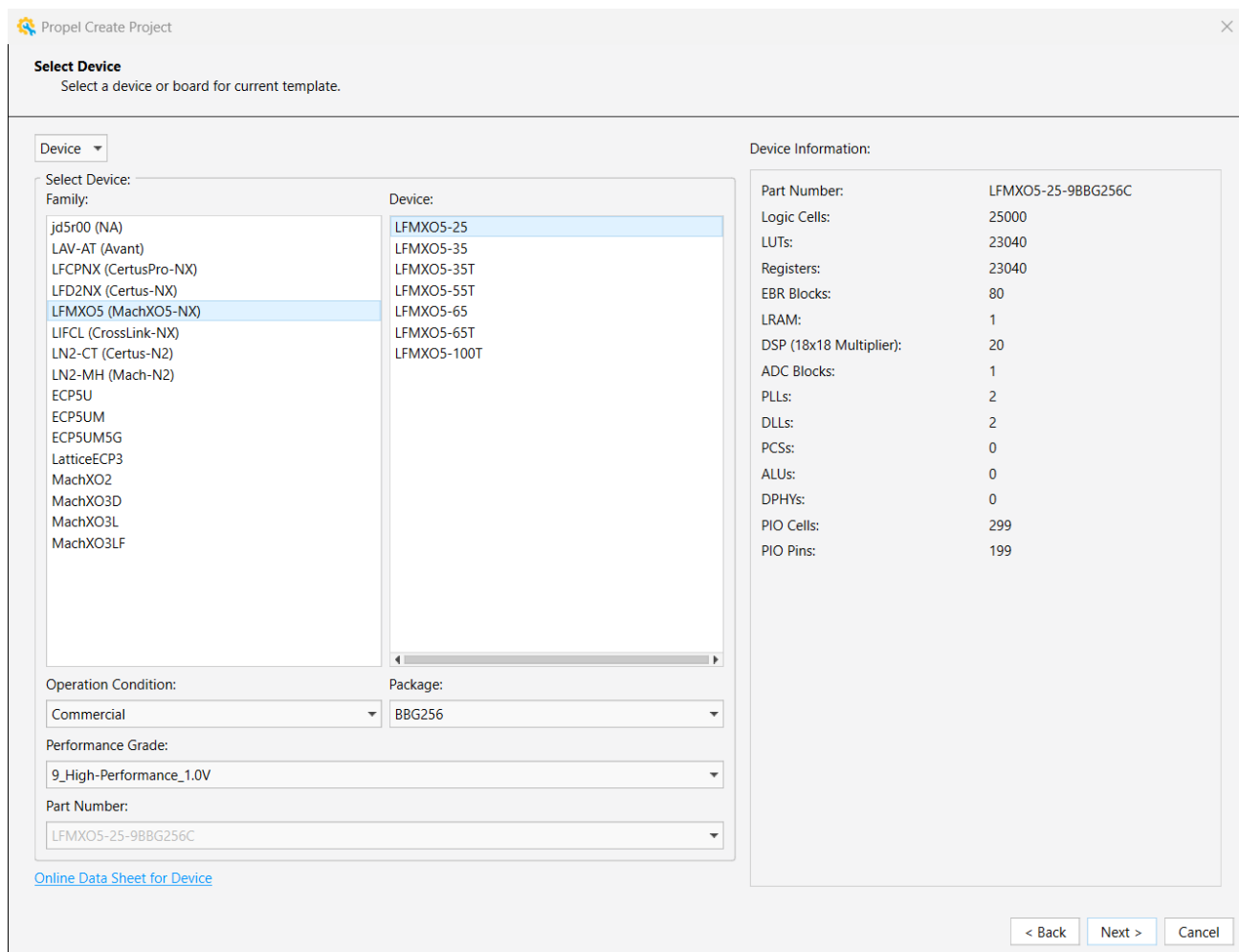



Figure 6.6. Create SoC Project

7. Run the Propel Builder by clicking on the  icon or **LatticeTools > Open Design** in the Propel Builder. The Propel Builder opens and loads the design template.
8. In the **IP Catalog** tab, instantiate the M-PESTI Initiator IP core. For more information, refer to the [Generating and Instantiating the IP](#) section.
9. After generating the IP core, the **Define Instance** window opens. Modify the instance name when needed then click **OK**.

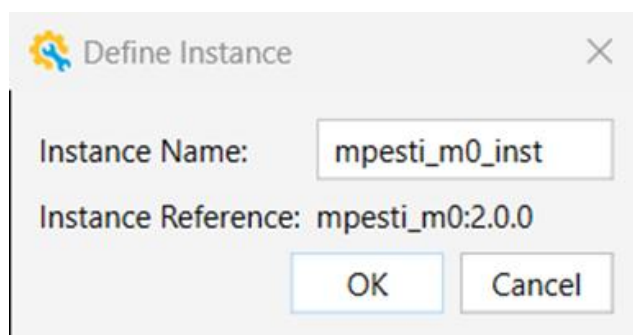


Figure 6.7. Define Instance


10. Instantiate M-PESTI targets as well. Your own target must be provided. For more information, refer to the [Generating and Instantiating the IP](#) section.
11. Connect the instantiated IP cores to the system. Refer to [Figure 6.1](#) for the connections used in this IP. You must update other components of the system for clock and reset sources, interrupt, and bus interface.
12. Click the icon or **Design > Run Radiant** to launch the Lattice Radiant software.
13. Update your constraints file accordingly and generate the programming file.
14. In the Lattice Propel software, build your SoC project to generate the system environment needed for the embedded C/C++ project. Select your SoC project, then click **Project > Build Project**.
15. Check the build result from the **Console** view.
16. Generate a new Lattice C/C++ project by clicking on **File > New > Lattice C/C++ Project**. Update the project name and click **Next > Finish**.

For more information on using the Propel software, refer to the [A Step-By-Step Approach to Lattice Propel Application Note \(FPGA-AN-02052\)](#).

6.5. Simulating the Example Design

The Lattice Propel Builder software also has a verification project mode that can be used to generate a simulation environment. The procedure for generating a verification project for the M-PESTI Initiator IP core is described in the following section.

To simulate the example design, follow these steps:

1. From the SoC Project, perform the pre-simulation requirements enumerated in the Verification Project Flow section of the [A Step-By-Step Approach to Lattice Propel Application Note \(FPGA-AN-02052\)](#).
2. Click on the Switch Verification and SoC design icon  to switch to verification project.
3. Reload dut_inst by double clicking on it. The **Reload sbx** window displays, as shown in the following figure. Click **Yes** to continue.

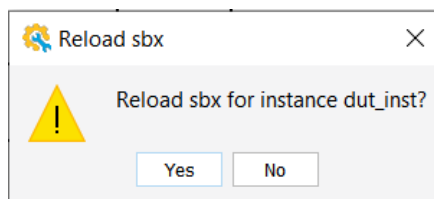


Figure 6.8. Reload sbx

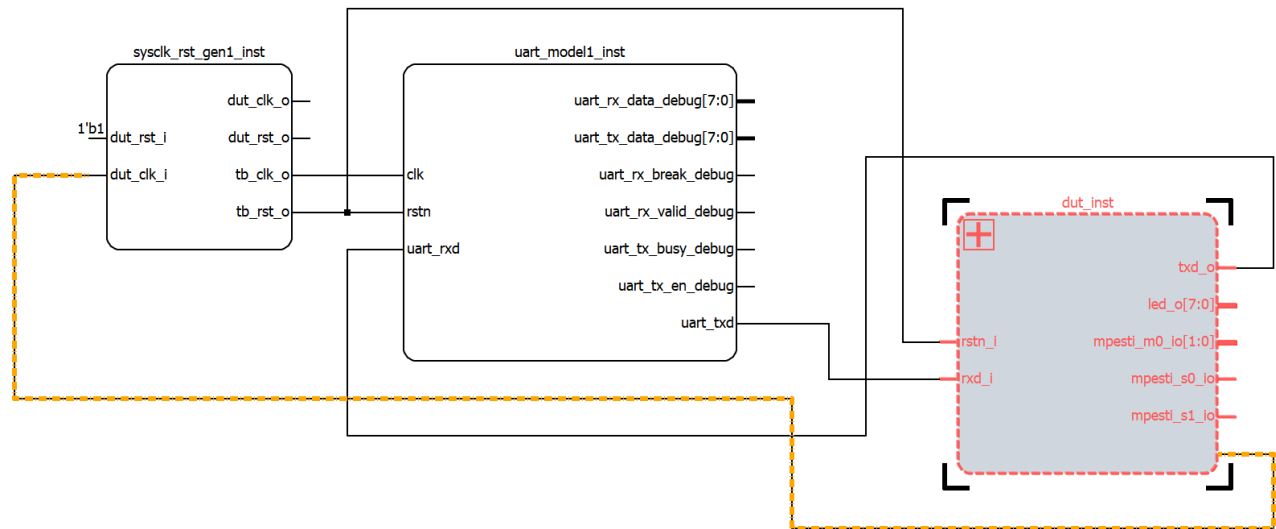



Figure 6.9. Verification Project Schematic

4. Click the Generate icon  to generate the simulation environment. The testbench file includes scripts for the chosen simulator, file lists, and other files are generated in the verification folder of the SoC project.
5. To see the full IP simulation behavior for this example design, connect the M-PESTI model to the IP instance in the generated testbench file as shown in the following figure and add the M-PESTI simulation model in the generated file list.

```

/*-----Instantiation Blocks-----*/
/* This section cover the instantiation of DUT, VIPs, Simulation */
/* Models, and other IP/Components. */
/*-----*/

mpesti_soc
dut_inst
(
    .mpesti_s0_io( ),
    .mpesti_s1_io( ),
    .rstn_i(sysclk_rst_gen1_inst_tb_rst_o_net),
    .rx_d_i(uart_model1_inst_uart_txd_net),
    .tx_d_o(uart_model1_inst_uart_rxd_net),
    .led_o( ),
    .mpesti_m0_io( )
);

initial begin
    sysclk_rst_gen1_inst.SysClk_Rst_GenMon_inst.set_clk_freq_mhz(50);
    sysclk_rst_gen1_inst.SysClk_Rst_GenMon_inst.start_clk();
end


sysclk_rst_gen1
sysclk_rst_gen1_inst
(
    .dut_clk_i(sysclk_rst_gen1_inst_dut_clk_i_net),
    .dut_rst_i(32'd1),
    .tb_clk_o(sysclk_rst_gen1_inst_tb_clk_o_net),
    .tb_rst_o(sysclk_rst_gen1_inst_tb_rst_o_net)
);

uart_model1
uart_model1_inst
(
    .clk(sysclk_rst_gen1_inst_tb_clk_o_net),
    .rstn(sysclk_rst_gen1_inst_tb_rst_o_net),
    .uart_rxd(uart_model1_inst_uart_rxd_net),
    .uart_txd(uart_model1_inst_uart_txd_net)
);

endmodule

```

Figure 6.10. SoC and Simulation Model Instantiation

- Click the Launch Simulation icon  to run the simulation.

6.6. Hardware Testing

The generated bitstream file from the procedure in the [Generating the Example Design](#) section is downloaded to the MachXO5-NX Development Board via the Radiant programmer. The Reveal analyzer is added to the Radiant software project to verify the output behavior of the IP core.

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 following steps describe how to generate the M-PESTI Initiator IP core in the Lattice Radiant software:

1. Create a new Lattice Radiant software project or open an existing project.
2. In the **IP Catalog** tab, double-click **M-PESTI Initiator** under **IP, Processors_Controllers_and_Peripherals** category. The **Module/IP Block Wizard** opens as shown in the following figure. Enter values in the **Component name** and the **Create in** fields and click **Next**.

Note: If the IP is not available on the **IP on Local** tab, download the IP from the **IP on Server** tab.

Figure 7.1. Module/IP Block Wizard

3. In the next **Module/IP Block Wizard** window, customize the selected M-PESTI Initiator IP core using the drop-down lists and check boxes. For details on the configuration options, refer to the [IP Parameter Description](#) section.

Module/IP Block Wizard

Configure Component from IP mpesti_initiator Version 2.0.0
Set the following parameters to configure this component.

Diagram mpesti0

Configure mpesti0:

General		Virtual Wire Setting
Property	Value	
IP Settings		
Specification Version Support	1.2	
IP Block ID	0	
IP Block Version (Major) [1 - 2]	2	
IP Block Version (Minor) [0 - 1]	0	
IP Block Version	2.0	
Clock		
System Clock Frequency (MHz) [1 - 100]	25	
System Clock Period (ns)	40	
MPESTI Baud Rate (kHz)	250	
MPESTI Baud Period (ns)	4000	
Calculated Clock Divider Setting	50	
APB Interface Settings		
APB Data Width	32	
General Settings		
Enable Optional Software Reset Control	<input type="checkbox"/>	
Presence Detection Timeout Enable	<input type="checkbox"/>	
Maximum Number of Target Devices [1 - 64]	8	
Maximum Static Discovery Payload Size	16	
Total Number of Secondary Wires [0 - 192]	0	
Enable Glitch Filter	<input type="checkbox"/>	
Allow Multiple User Receive Bytes	<input checked="" type="checkbox"/>	
Maximum User Receive Byte Size [2 - 16]	16	
Target Settings		
Discovery Phase Enable	<input checked="" type="checkbox"/>	
Discovery Phase Bypass Enable (in hex)	0	
Active Phase Enable	<input checked="" type="checkbox"/>	
Active Phase PEC Enable (in hex)	0	
Memory Settings		
Payload Memory Allocation	RAM	
I/O Settings		
Include I/O Primitive	<input checked="" type="checkbox"/>	

[User Guide](#)

No DRC issues are found.

< Back Generate Cancel

Figure 7.2. M-PESTI Initiator IP Core Configuration Example

- Click **Generate**. The **Check Generated Result** dialog box opens. The following figure shows the design block messages and results.

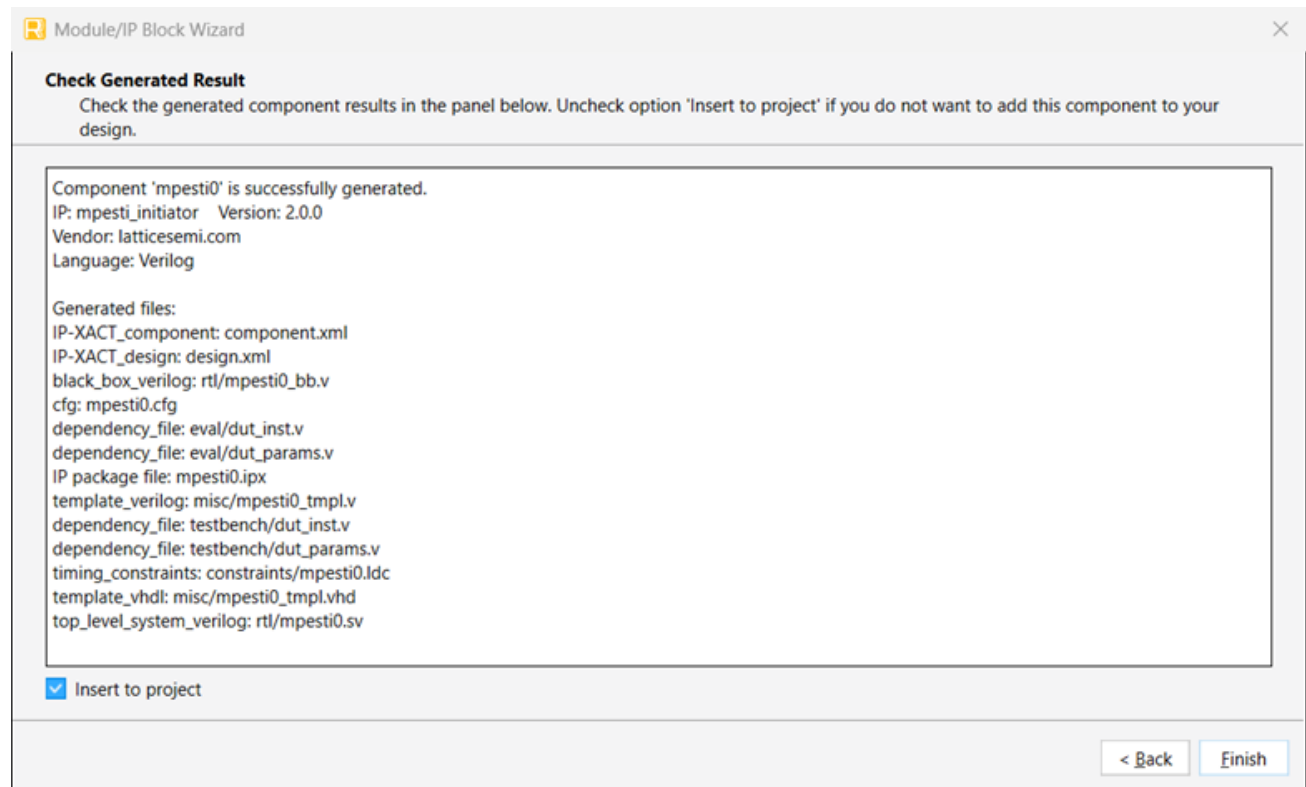


Figure 7.3. Check Generated Result

- Click **Finish**. All the generated files are placed under the directory paths in the **Create in** and the **Component name** fields.

7.1.1. Generated Files and File Structure

The generated M-PESTI Initiator IP 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 example as the starting template for your top-level design.

Table 7.1. Generated File List

Generated File	Description
<i><Component name>.ipx</i>	Contains the information on the files associated to the generated IP.
<i><Component name>.cfg</i>	Contains the parameter values used in IP configuration.
<i>component.xml</i>	Contains the ipxact: component information of the IP.
<i>design.xml</i>	Documents the configuration parameters of the IP in IP-XACT 2014 format.
<i>rtl/<Component name>.v</i>	Provides an example RTL top file that instantiates the module.
<i>rtl/<Component name>_bb.v</i>	Provides the synthesis closed-box.
<i>misc/<Component name>_tmpl.v</i> <i>misc /<Component name>_tmpl.vhd</i>	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

You must provide proper timing and physical design constraints to ensure that your design meets the desired performance goals on the FPGA. Add the content of the following IP constraint file to your design constraints:
<IP_Instance_Path>/<IP_Instance_Name>/eval/constraint.pdc.

The constraint file has been verified during IP evaluation with the IP instantiated directly in the top-level module. You can modify the constraints in this file with thorough understanding of the effect of each constraint.

To use this constraint file, copy the content of *constraint.pdc* to the top-level design constraint for post-synthesis.

For more information on how to constrain your design, refer to the [Lattice Radiant Timing Constraints Methodology Application Note \(FPGA-AN-02059\)](#).

7.4. Running Functional Simulation

You can run functional simulation after the IP core is generated.

To run the functional simulation, follow these steps:

1. Click the  icon located on the **Toolbar** to initiate the **Simulation Wizard**.

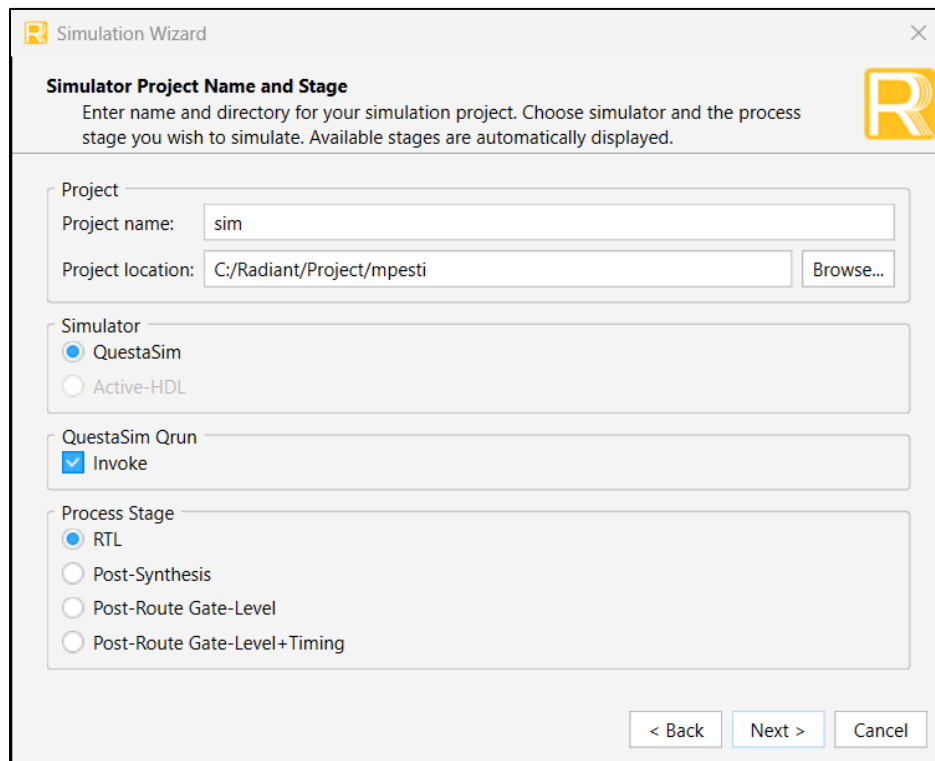


Figure 7.4. Simulation Wizard

- Click **Next** to open the **Add and Reorder Source** window.

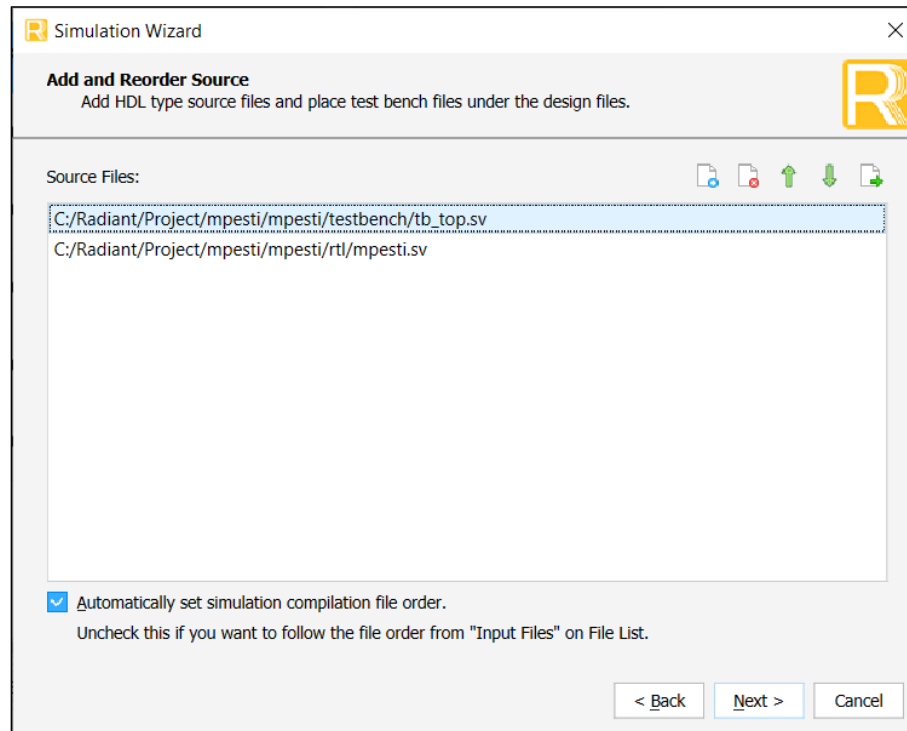


Figure 7.5. Add and Reorder Source

- Click **Next**. The **Summary** window displays.

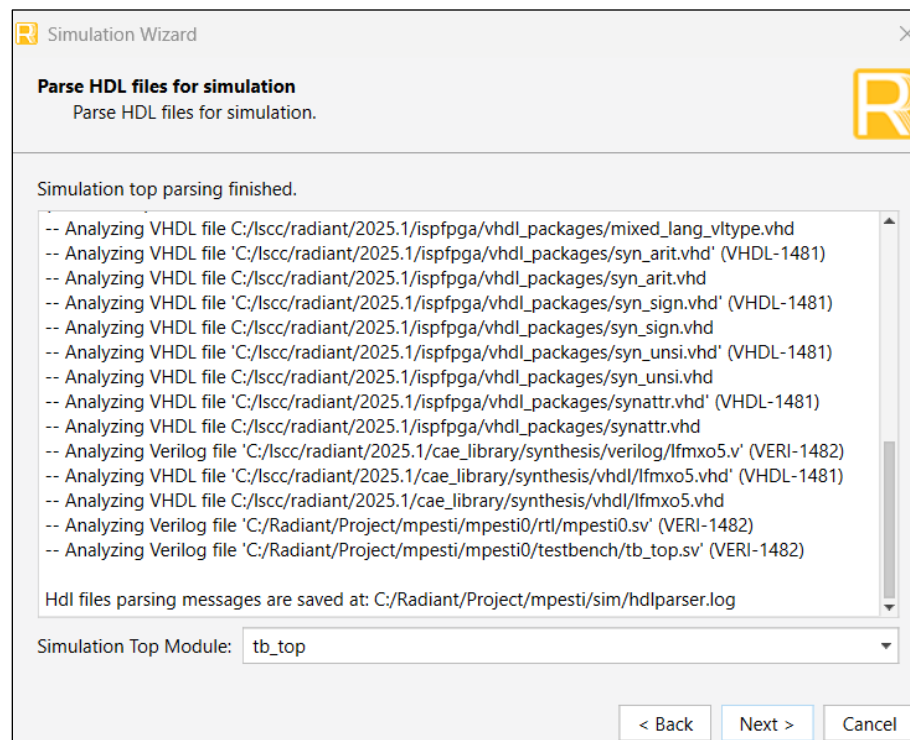


Figure 7.6. Parse HDL Files for Simulation

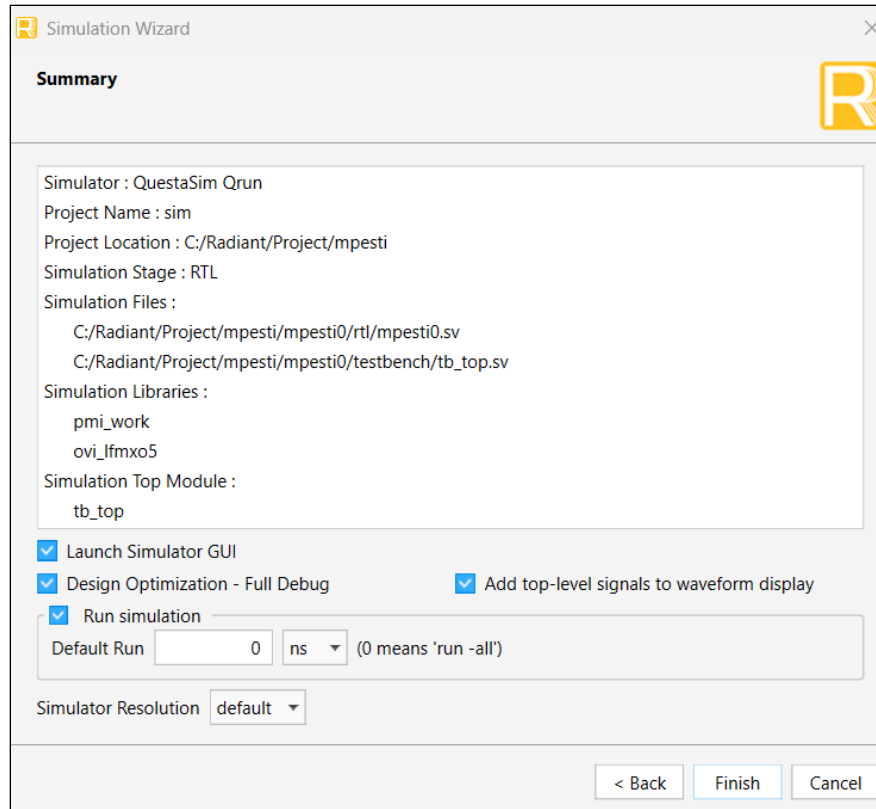


Figure 7.7. Summary Window

4. Click **Finish** to run the simulation.

The following figures show the transcript and waveform of the simulation result.

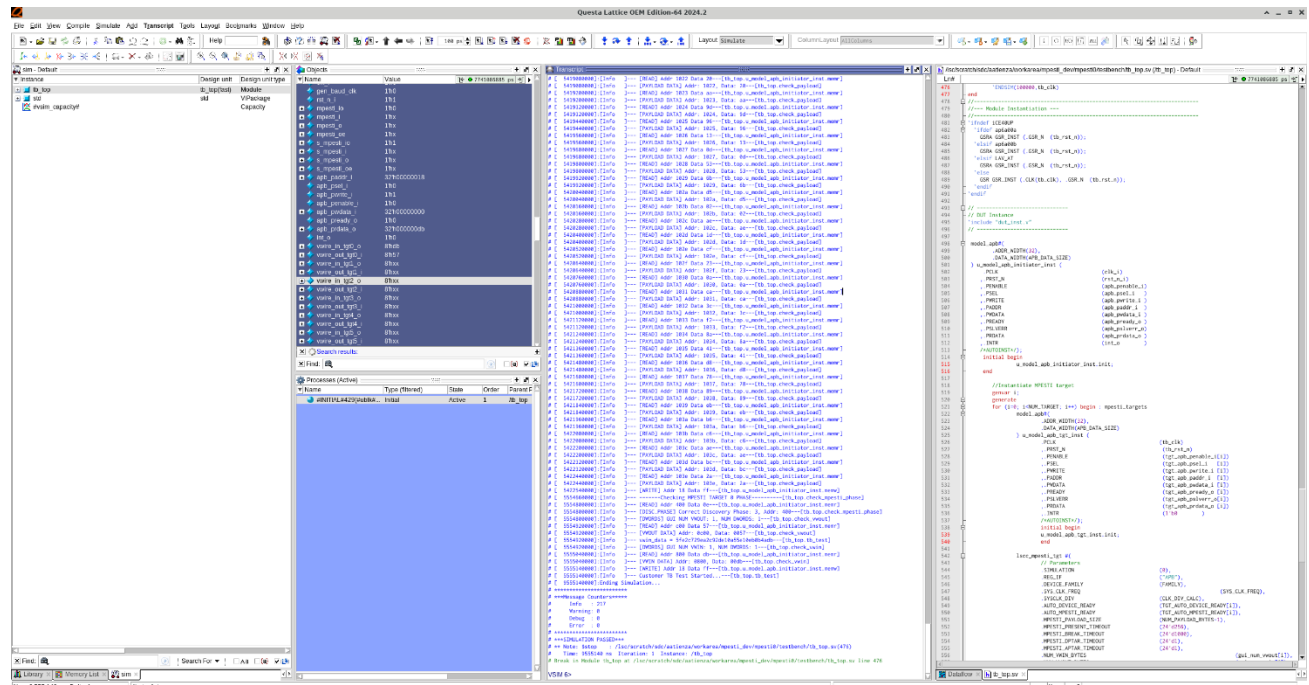


Figure 7.8. Transcript of the Simulation Result

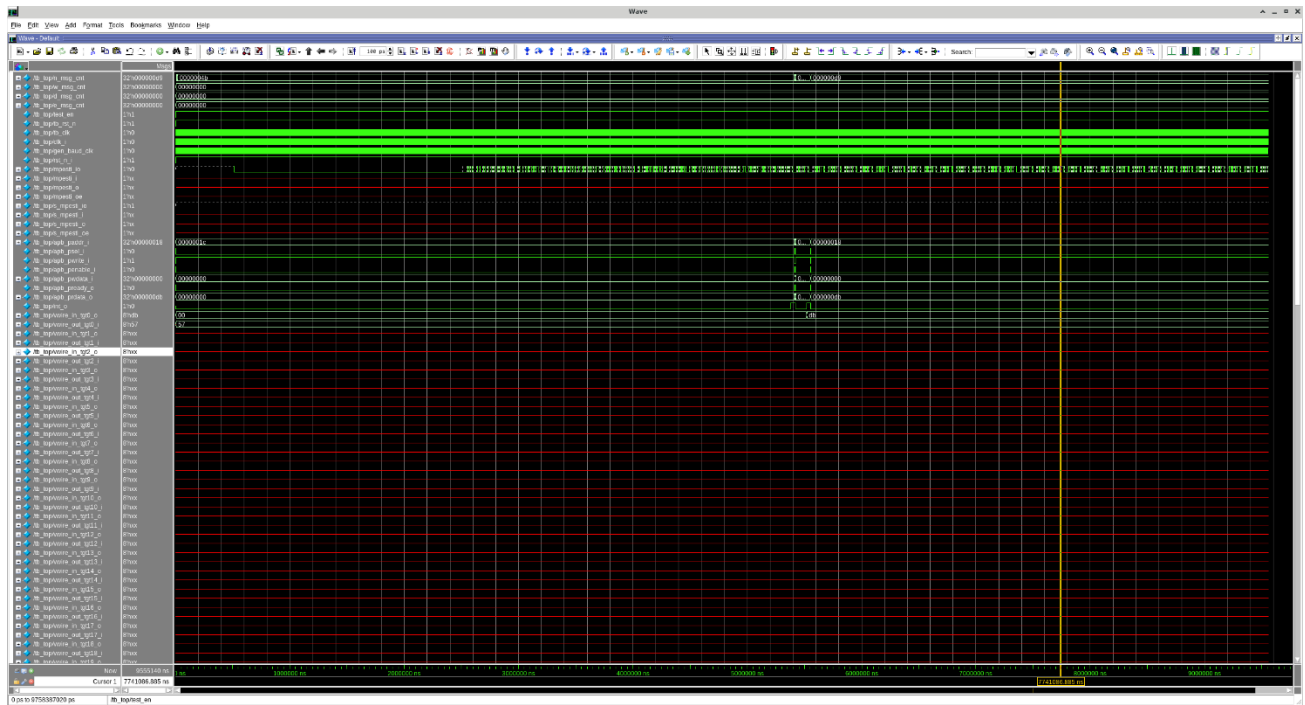


Figure 7.9. Simulation Waveform

Appendix A. Resource Utilization

The following table shows a sample resource utilization of the M-PESTI Initiator IP core v1.1.0 using the LFMX05-25-7BBG400I device with the Synplify Pro synthesis tool in the Lattice Radiant software 2024.2. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.1. Resource Utilization for LFMX05-25-7BBG400I Device (IP v1.1.0³)

IP Configuration	clk_i Fmax (MHz) ¹	Registers	LUTs ²	EBRs	DSPs
Default configuration Number of Payload Bytes = 16	128.584	366	592	1	0
Memory Allocation = Register	128.386	417	629	1	0
Number of Payload Bytes = 64 Number of Targets = 16	130.429	1,419	1,925	1	0
Number of Payload Bytes = 64 Number of Targets = 64	96.927	4,757	6,431	1	0

Notes:

1. Fmax is generated when the FPGA design only contains the M-PESTI Initiator module, and the target frequency is 25 MHz. These values may be reduced when user logic is added to the FPGA design.
2. The distributed RAM utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among logic, and ripple logic.
3. Default configuration: 1 target, maximum static discovery payload of 64 bytes.

The following table shows a sample resource utilization of the M-PESTI Initiator IP core v1.2.0 using the LFMX05-25-7BBG400I device with the Synplify Pro synthesis tool in the Lattice Radiant software 2025.1. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.2. Resource Utilization for LFMX05-25-7BBG400I Device (IP v1.2.0³)

IP Configuration	clk_i Fmax (MHz) ¹	Registers	LUTs ²	EBRs	DSPs
Default configuration	146.477	614	992	2	0
Number of Payload Bytes = 64 Number of Targets = 16	89.008	2,569	3,927	2	0
Number of Payload Bytes = 64 Number of Targets = 64	81.274	8,735	13,260	4	0

Notes:

1. Fmax is generated when the FPGA design only contains the M-PESTI Initiator module, and the target frequency is 25 MHz. These values may be reduced when user logic is added to the FPGA design.
2. The distributed RAM utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among logic, and ripple logic.
3. Default configuration: 1 target, maximum static discovery payload of 64 bytes.

The following table shows a sample resource utilization of the M-PESTI Initiator IP core v1.2.0 using the LAV-AT-E70-1LFG676I device with the Synplify Pro synthesis tool in the Lattice Radiant software 2025.1. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.3. Resource Utilization for LAV-AT-E70-1LFG676I Device (IP v1.2.0³)

IP Configuration	clk_i Fmax (MHz) ¹	Registers	LUTs ²	EBRs	DSPs
Default configuration	191.498	641	1,024	1	0
Number of Payload Bytes = 64 Number of Targets = 16	170.387	2,610	3,967	1	0
Number of Payload Bytes = 64 Number of Targets = 64	123.640	8,755	12,671	2	0

Notes:

1. Fmax is generated when the FPGA design only contains the M-PESTI Initiator module, and the target frequency is 25 MHz. These values may be reduced when user logic is added to the FPGA design.
2. The distributed RAM utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among logic, and ripple logic.

- Default configuration: 1 target, maximum static discovery payload of 64 bytes.

The following table shows a sample resource utilization of the M-PESTI Initiator IP core v1.2.0 using the LN2-CT-20ES-1ASG410I device with the Synplify Pro synthesis tool in the Lattice Radiant software 2025.1. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.4. Resource Utilization for LN2-CT-20ES-1ASG410I Device (IP v1.2.0³)

IP Configuration	clk_i Fmax (MHz) ¹	Registers	LUTs ²	EBRs	DSPs
Default configuration	236.351	641	1,024	1	0
Number of Payload Bytes = 64 Number of Targets = 16	158.378	2,610	3,967	1	0
Number of Payload Bytes = 64 Number of Targets = 64	125.471	8,755	12,671	2	0

Notes:

- Fmax is generated when the FPGA design only contains the M-PESTI Initiator module, and the target frequency is 25 MHz. These values may be reduced when user logic is added to the FPGA design.
- The distributed RAM utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among logic, and ripple logic.
- Default configuration: 1 target, maximum static discovery payload of 64 bytes.

The following table shows a sample resource utilization of the M-PESTI Initiator IP core v2.0.0 using the LFMX05-25-7BBG400I device with the Synplify Pro synthesis tool in the Lattice Radiant software 2025.2. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.5. Resource Utilization for LFMX05-25-7BBG400I Device (IP v2.0.0³)

IP Configuration	clk_i Fmax (MHz) ¹	Registers	LUTs ²	EBRs	DSPs
Default configuration	88.067	946	1,717	4	0
Number of Payload Bytes = 64 Number of Targets = 16	76.272	1,521	2,602	4	0
Number of Payload Bytes = 64 Number of Targets = 64	45.057	4,929	8,946	6	0

Notes:

- Fmax is generated when the FPGA design only contains the M-PESTI Initiator module, and the target frequency is 25 MHz. These values may be reduced when user logic is added to the FPGA design.
- The distributed RAM utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among logic, and ripple logic.
- Default configuration: 8 targets, maximum static discovery payload of 16 bytes.

The following table shows a sample resource utilization of the M-PESTI Initiator IP core v2.0.0 using the LAV-AT-E70-1LFG676I device with the Synplify Pro synthesis tool in the Lattice Radiant software 2025.2. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.6. Resource Utilization for LAV-AT-E70-1LFG676I Device (IP v2.0.0³)

IP Configuration	clk_i Fmax (MHz) ¹	Registers	LUTs ²	EBRs	DSPs
Default configuration	107.956	963	1,670	2	0
Number of Payload Bytes = 64 Number of Targets = 16	121.625	1,541	2,743	2	0
Number of Payload Bytes = 64 Number of Targets = 64	82.210	4,944	8,436	3	0

Notes:

- Fmax is generated when the FPGA design only contains the M-PESTI Initiator module, and the target frequency is 25 MHz. These values may be reduced when user logic is added to the FPGA design.
- The distributed RAM utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among logic, and ripple logic.
- Default configuration: 8 targets, maximum static discovery payload of 16 bytes.

The following table shows a sample resource utilization of the M-PESTI Initiator IP core v2.0.0 using the LN2-CT-20ES-1ASG410I device with the Synplify Pro synthesis tool in the Lattice Radiant software 2025.2. Default configuration is used and some attributes are changed from the default value to show the effect on the resource utilization.

Table A.7. Resource Utilization for LN2-CT-20ES-1ASG410I Device (IP v2.0.0³)

IP Configuration	clk_i Fmax (MHz) ¹	Registers	LUTs ²	EBRs	DSPs
Default configuration	147.754	943	1,595	2	0
Number of Payload Bytes = 64 Number of Targets = 16	140.410	1,541	2,743	2	0
Number of Payload Bytes = 64 Number of Targets = 64	87.627	4,944	8,436	3	0

Notes:

1. Fmax is generated when the FPGA design only contains the M-PESTI Initiator module, and the target frequency is 25 MHz. These values may be reduced when user logic is added to the FPGA design.
2. The distributed RAM utilization is accounted for in the total LUT4s utilization. The actual LUT4 utilization is distribution among logic, and ripple logic.
3. Default configuration: 8 targets, maximum static discovery payload of 16 bytes.

References

- [M-PESTI Initiator IP Release Notes \(FPGA-RN-02005\)](#)
- [M-PESTI Initiator Driver API Reference \(FPGA-TN-02413\)](#)
- [M-PESTI Initiator IP web page](#)
- [Modular-Peripheral Sideband Tunneling Interface \(M-PESTI\) Base Specification Version 1.0 Release Candidate 2](#)
- [Modular-Peripheral Sideband Tunneling Interface \(M-PESTI\) Base Specification Version 1.2 Release Candidate 2](#)
- [Open Compute Project® web page](#)
- [A Step-By-Step Approach to Lattice Propel Application Note \(FPGA-AN-02052\)](#)
- [Lattice Radiant Timing Constraints Methodology Application Note \(FPGA-AN-02059\)](#)
- [Avant-E web page](#)
- [Avant-G web page](#)
- [Avant-X web page](#)
- [MachXO5-NX web page](#)
- [MachXO3D web page](#)
- [MachXO3 web page](#)
- [Mach-NX web page](#)
- [Lattice Radiant Software web page](#)
- [Lattice Propel Design Environment web page](#)
- [Lattice Solutions IP Cores web page](#)
- [Lattice Solutions Reference Designs web page](#)
- [Lattice Solutions Boards web page](#)
- [Lattice Solutions Demonstrations web page](#)
- [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

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 1.4, IP v2.0.0, December 2025

Section	Change Summary
All	<ul style="list-style-type: none"> Added the OCP logo to the cover page. Added a note on the IP version in the <i>Quick Facts</i> and <i>Revision History</i> section. Made minor editorial changes
Abbreviations in This Document	Added <i>Open Compute Project (OCP)</i> and <i>Packet Error Checking (PEC)</i> .
Introduction	<ul style="list-style-type: none"> Added a statement indicating that the IP is OCP Ready and comply with the Modular-Peripheral Sideband Tunneling Interface (M-PESTI) Base Specification Version 1.2 Release Candidate 2. Updated IP core version to 2.0.0 and Lattice Radiant Version to 2025.2 in Table 1.1. Summary of the M-PESTI Initiator IP. Added the following key features of the M-PESTI IP in Features section: <ul style="list-style-type: none"> Support source and destination cable coupling discovery. Support optional discovery phase bypass. Support optional active phase Packet Error Checking (PEC).
Functional Description	<ul style="list-style-type: none"> Added <i>1 baud is 4us</i> statement to section M-PESTI UART Frame. Updated description in Target Presence Detection, Target Setting Reconfiguration, Discovery Phase, Active Phase, and User Command, Broadcast, and Abort Mechanism sections. Added Figure 2.5 Virtual Wire Exchange, Figure 2.6 Virtual Wire Exchange with PEC, sections Optional Source to Destination Detection, Use of I2C GPIO Expander, and Fanout Feature. Removed Figure 2.7. Abort Mechanism. Updated Figure 2.1. Lattice M-PESTI Initiator IP Core Block Diagram and Figure 2.9. User Command Program Flow. Moved section 2.5.2.7 Secondary Wire Capability with Stimulus and Response to section 2.5.3.1. Updated the description of the following in the M-PESTI Target N Error, Status, Control and Configuration 0x00+(N*4) section: <ul style="list-style-type: none"> Active Phase PEC Enable [19] Active Phase Enable [18] Discovery Phase Bypass [17] Discovery Phase Enable [16]
IP Parameter Description	<ul style="list-style-type: none"> Updated Table 3.1. General Attributes: <ul style="list-style-type: none"> Added IP Settings category with the following attribute settings: <i>Specification Version Support</i>, <i>IP Block ID</i>, <i>IP Block Version (Major)</i>, <i>IP Block Version (Minor)</i>, and <i>IP Block Version</i>. Added description for <i>System Clock Period (ns)</i>, <i>M-PESTI Baud Rate (kHz)</i>, <i>M-PESTI Baud Period (ns)</i>, and <i>Calculated Clock Divider Setting</i> in the Clock Settings category. Added General Settings category with the following attribute settings: <i>Presence Detection Timeout Enable</i> and <i>Presence Detection Timeout (us) (multiples of 2us)</i>. Updated the description of <i>Enable CSR for Secondary Wire</i> and <i>Maximum User Receive Byte Size</i> in the General Setting category. Moved the following attribute settings from Target Settings category to General Settings category: <i>Number of Target Devices</i>, <i>Maximum Static Discovery Payload Size (Added 128 to Selectable Values to Maximum Static Discovery)</i>, <i>Total Number of Secondary Wires</i>,

	<p><i>Enable CSR for Secondary Wire, Enable Glitch Filter, Allow Multiple User Receive Bytes, and Maximum User Receive Byte Size.</i></p> <ul style="list-style-type: none"> Removed TDBreak Assertion Time (us) (multiples of 2us) attribute settings from Target Setting category. Added <i>Discovery Phase Enable (in hex)</i> and <i>Active Phase PEC Enable (in hex)</i> in the Target Settings category. Added description of the <i>Payload Memory Allocation</i> in the Memory Settings category. Moved <i>Virtual Wire Settings</i>, <i>Target x</i> categories and table note referencing <i>Target x</i> from <i>Table 3.1</i> to a newly added section Virtual Wire Setting.
Signal Description	<p>Table 4.1. Ports Description:</p> <ul style="list-style-type: none"> Added column titled <i>Width</i> and added values to column <i>Width</i> for each of the listed port in the table. Added table note 3.
Register Description	<ul style="list-style-type: none"> Updated Figure 5.1. Memory Map Allocation and revised the description to reflect the changes in the figure. Updated Table 5.2. Register Address Map, and revised sections 5.1 – 5.6, including section titles and added subsections, to align with the updated information in Figure 5.1. Updated the Field and Width value of Byte Length in Table 5.9. Updated the Access value of Secondary Wire from RW to RO and the Reset value from 0x0 to 0x1 in Table 5.17. Secondary Wire Control Status 0x40. Updated the description of <i>CSR Reset [1]</i> and <i>IP Core Reset [0]</i> in Software Reset 0x04 section to include the following statement: <i>This does not clear the virtual wire data and payload information stored in the memory module.</i> Updated the description of Virtual Wire Input [31:0] in the M-PESTI Target N Virtual Wire Output 0x10+(N*32) section.
Example Design	<ul style="list-style-type: none"> Updated Table 6.1 to reflect changes made in the IP Parameter section. Updated Figure 6.7. Define Instance. Added steps 4 – 6 in Generating the Example Design.
Designing with IP	<ul style="list-style-type: none"> Added the following note in this section: <i>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.</i> Updated Figure 7.1. Module/IP Block Wizard, Figure 7.2. M-PESTI Initiator IP Core Configuration Example, and Figure 7.3. Check Generated Result.
Appendix A. Resource Utilization	<ul style="list-style-type: none"> Updated the value of <i>clk_i Fmax</i>, <i>Registers</i>, <i>LUTs</i>, and <i>EBR</i> in Table A.2, Table A.3, and Table A.4. Added the following table note to Table A.1, Table A.2, Table A.3, and Table A.4: <i>Default configuration: 1 target, maximum static discovery payload of 64 bytes.</i> Added Table A.5, Table A.6, and Table A.7 to show the resource utilization for various devices using IP core v2.0.0 and Lattice Radiant software 2025.2. <ul style="list-style-type: none"> Added the following table note to these tables: <i>Default configuration: 8 targets, maximum static discovery payload of 16 bytes.</i>
References	<p>Added the Open Compute Project web page and the Modular-Peripheral Sideband Tunneling Interface (M-PESTI) Base Specification Version 1.2 Release Candidate 2.</p>

Revision 1.3, IP v1.3.0, July 2025

Section	Change Summary
Introduction	<p>Updated Table 1.1. Summary of the M-PESTI Initiator IP:</p> <ul style="list-style-type: none"> IP Requirements: <ul style="list-style-type: none"> Changed <i>FPGA Families Supported to Supported Devices</i>. Updated list of <i>Supported Devices</i>. Resource Utilization: <ul style="list-style-type: none"> Added Resources and linked to Appendix A. Resource Utilization.

	<ul style="list-style-type: none"> Lattice Implementation: <ul style="list-style-type: none"> Updated IP and Lattice Radiant Software version.
References	Added M-PESTI Initiator Driver API Reference (FPGA-TN-02413).

Revision 1.2, IP v1.2.0, June 2025

Section	Change Summary
Introduction	<ul style="list-style-type: none"> Updated the IP version in Table 1.1. Summary of the M-PESTI Initiator IP. Updated the name of the license type in Table 1.3. Ordering Part Number from Multi-Site Perpetual to Single Seat Perpetual.
Functional Description	<ul style="list-style-type: none"> Updated Figure 2.1. Lattice M-PESTI Initiator IP Core Block Diagram. Updated Table 2.1. User Interfaces and Supported Protocols. Updated the following sentence in the Active Phase section: <i>Current autonomous virtual wire exchange transacts virtual wire input and output data based on the Number of Virtual Wire Input and Output Bytes attribute per target.</i> Added the Target Setting Reconfiguration section. Updated the Active Phase section. Updated Figure 2.7. Abort Mechanism.
IP Parameter Description	Updated Table 3.1. General Attributes.
Signal Description	Updated Table 4.1. Ports Description.
Register Description	<ul style="list-style-type: none"> Updated the following sentence in this section: For example, four targets with 32 bytes payload each = 128 bytes RAM. Added Table 5.1. Register Access Types. Updated Table 5.2. Register Address Map. Updated VWIN to Virtual Wire Input data in the Interrupt Status 0x018 section. Updated the following sentence in the Interrupt Status 0x018 section: <i>1 – Error (this flags active phase receive timeout, parity error and/or framing error) is detected from target(s).</i> Updated Table 5.6. Interrupt Enable 0x01C. Updated Table 5.11. User Command 0x104. Updated the M-PESTI Target N Control Status 0x404+(N*16) section. Added the M-PESTI Target N Virtual Wire Configuration 0x408+(N*16) section. Updated the M-PESTI Target N Virtual Wire Input 0x800+(N*16) section. Updated the M-PESTI Target N Virtual Wire Output 0xC00+(N*16) section.
Example Design	<ul style="list-style-type: none"> Updated Table 6.1. M-PESTI Initiator IP Configuration Supported by the Example Design. Updated Figure 6.6. Define Instance.
Designing with the IP	<ul style="list-style-type: none"> Updated the following figures: <ul style="list-style-type: none"> Figure 7.1. Module/IP Block Wizard. Figure 7.2. M-PESTI Initiator IP Core Configuration Example. Figure 7.3. Check Generated Result. Figure 7.4. Simulation Wizard. Figure 7.9. Simulation Waveform. Added the following figures: <ul style="list-style-type: none"> Figure 7.6. Parse HDL Files for Simulation. Figure 7.7. Summary Window. Figure 7.8. Transcript of the Simulation Result.
Appendix A. Resource Utilization	<ul style="list-style-type: none"> Updated Table A.1. Resource Utilization for LFMX05-25-7BBG400I Device (IP v1.1.0). Added the following tables: <ul style="list-style-type: none"> Table A.2. Resource Utilization for LFMX05-25-7BBG400I Device (IP v1.2.0). Table A.3. Resource Utilization for LAV-AT-E70-1LFG676I Device (IP v1.2.0). Table A.4. Resource Utilization for LN2-CT-20ES-1ASG410I Device (IP v1.2.0).

Revision 1.1, IP v1.1.0, December 2024

Section	Change Summary
All	Added IP version on the cover page.
Introduction	<ul style="list-style-type: none"> Updated Table 1.1. Summary of the M-PESTI Initiator IP. Added the IP Support Summary section. Updated the Features section. Added new ordering part numbers for MachXO3D and Avant devices in Table 1.3. Ordering Part Number. Updated a column title in Table 1.3. Ordering Part Number from Single Machine to Single Seat. Changed the section title from IP Validation Summary to Hardware Support and updated the content in the Hardware Support section.
Functional Description	<ul style="list-style-type: none"> Updated Figure 2.1. Lattice M-PESTI Initiator IP Core Block Diagram Updated the following sections: <ul style="list-style-type: none"> Clocking section Reset section Updated the following sub-sections in the M-PESTI Protocol section: <ul style="list-style-type: none"> M-PESTI UART Frame section M-PESTI Protocol Phase section Discovery Phase section Active Phase section Target Reset section Removed the Multiple Targets section and added the Secondary Wire Capability with Stimulus and Response section. Updated the following figures in the M-PESTI Protocol section: <ul style="list-style-type: none"> Figure 2.2 M-PESTI UART Frame Figure 2.3 M-PESTI Protocol Timing Diagram Figure 2.4 Single Initiator Multiple Targets Transaction Figure 2.5 Manual Multiple Virtual Bytes Program Flow Figure 2.6 Broadcast Command Figure 2.7 Abort Mechanism Figure 2.8 User Command Program Flow Added Figure 2.9 Secondary Wire Capability Program Flow
IP Parameter Description	Updated Table 3.1. General Attributes.
Signal Description	<p>Added the following ports to Table 4.1. Ports Description:</p> <ul style="list-style-type: none"> s_mpesti_io s_mpesti_i s_mpesti_oe s_mpesti_o
Register Description	<ul style="list-style-type: none"> Updated the introductory paragraph and the examples. Updated Table 5.1. Register Address Map. Updated the following sections: <ul style="list-style-type: none"> Configuration 0x004 section. Software Reset 0x008 section Interrupt Status 0x018 section Interrupt Enable 0x01C section Interrupt Set 0x020 section Secondary Wire Control Status 0x040 section Target Select 0x100 section User Command 0x104 section M-PESTI Target 0 Status 0x400+(N*16) section M-PESTI Target 0 Control Status 0x404+(N*16) section M-PESTI Target 0 Virtual Wire Input 0x800+(N*16) section

Section	Change Summary
	<ul style="list-style-type: none"> • M-PESTI Target 0 Virtual Wire Output 0xC00+(N*16) section • Added the following sections: <ul style="list-style-type: none"> • Secondary Wire Select 0x044 section • Target Select 0x100 section • User Command 0x104 section • User Write Data 0x108 section • User Read Data 0x10C section
Example Design	<ul style="list-style-type: none"> • Added a brief introduction to this section and list all the evaluation boards used to run the example design. • Added the following parameters in Table 6.1. M-PESTI Initiator IP Configuration Supported by the Example Design: <ul style="list-style-type: none"> • Total Number of Secondary Wires • Enable CSR for Secondary Wire • Allow Multiple User Receive Bytes • Maximum User Receive Byte Size • Updated the Maximum Static Discovery Payload Size in Table 6.1. M-PESTI Initiator IP Configuration Supported by the Example Design. • Updated the steps to generate example design and the following figures: <ul style="list-style-type: none"> • Figure 6.4 Select Template • Figure 6.5 Create SoC Project • Figure 6.6 Define Instance
Designing with the IP	<p>Updated the following figures:</p> <ul style="list-style-type: none"> • Figure 7.1. Module/IP Block Wizard • Figure 7.2. M-PESTI Initiator IP Core Configuration Example • Figure 7.3. Check Generated Result
Appendix A. Resource Utilization	Updated Table A.1. Resource Utilization for LFMX05-25-7BBG400I Device.
References	<ul style="list-style-type: none"> • Added the following references: <ul style="list-style-type: none"> • Avant-E web page • Avant-G web page • MachX05-NX web page • MachX03D web page • MachX03 web page • Mach-NX web page • Lattice Solutions Reference Designs web page • Lattice Solutions Boards web page • Lattice Solutions Demonstrations web page • Removed the reference to the Lattice Radiant Software User Guide.

Revision 1.0, IP v1.0.0, June 2024

Section	Change Summary
All	Initial release.



www.latticesemi.com