



Using TraceID

Technical Note

FPGA-TN-02084-2.8

April 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..... | 5 |
| 1. Introduction | 6 |
| 2. Why is TraceID Important? | 6 |
| 3. How Does TraceID Work? | 6 |
| 4. How to Program the TraceID User-Defined Code..... | 7 |
| 5. Accessing the TraceID Register | 7 |
| 5.1. TraceID Access Through the JTAG Port | 9 |
| 5.2. TraceID Access Through the sysCONFIG WISHBONE Target Interface (MachXO2 and MachXO3 Devices Only) | 10 |
| 5.3. TraceID Access through the sysCONFIG Target SPI Port (MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX Devices Only) | 10 |
| 5.4. TraceID Access through the sysCONFIG Target SPI Port (Avant and Nexus 2 Devices Only) | 11 |
| 5.5. TraceID Access through the I2C Port (MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX Only) | 13 |
| 5.6. TraceID Access through sysCONFIG LMMI (CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, Avant, and Nexus 2 Only)..... | 14 |
| 6. Example Uses of TraceID | 15 |
| References | 16 |
| Technical Support Assistance | 17 |
| Revision History | 18 |

Figures

| | |
|--|----|
| Figure 5.1. Read TraceID through Lattice Diamond Programming Tool | 7 |
| Figure 5.2. Program TraceID through Lattice Radiant Programmer Tool | 8 |
| Figure 5.3. Feature Row Editor | 8 |
| Figure 5.4. Avant – Program Feature Row through Lattice Radiant Programmer Tool | 9 |
| Figure 5.5. TraceID Read through SPI | 11 |
| Figure 5.6. TraceID Read through SPI, Continued | 11 |
| Figure 5.7. TraceID Read through SPI (Avant and Nexus 2 Devices only) | 12 |
| Figure 5.8. TraceID Read through SPI, Continued (Avant and Nexus 2 Devices only) | 12 |
| Figure 5.8. TraceID Read through I2C | 14 |
| Figure 6.1. TraceID Read through I2C | 15 |

Tables

| | |
|---|----|
| Table 3.1. TraceID Register | 6 |
| Table 5.1. Methods to Read the TraceID Values | 7 |
| Table 5.2. sysCONFIG WISHBONE Register | 10 |

Abbreviations in This Document

A list of abbreviations used in this document.

| Abbreviation | Definition |
|------------------|---|
| EFB | Embedded Function Block |
| I ² C | Inter-integrated circuit |
| IP | Intellectual property |
| JTAG | Joint Test Action Group |
| JED | Abbreviation of JEDEC |
| JEDEC | Joint Electron Device Engineering Council |
| LMMI | Lattice Memory Mapped Interface |
| OEM | Original equipment manufacturer |
| PLD | Programmable Logic Device |
| SPI | Serial Peripheral Interface |
| SSPI | Security Support Provider Interface |

1. Introduction

Design theft has caused many companies to explore methods to insure that their designs and intellectual property (IP) are protected or less prone to blatant copying.

Design theft occurs when a design is copied in part or as a whole and then designed into a cheaper competing product.

The MachXO2™, MachXO3™, ECP5™, ECP5-5G™, CrossLink™-NX, CrossLink-NX-33, CrossLinkU™-NX, Certus™-NX, CertusPro™-NX, MachXO5™-NX, Avant™, and Nexus™ 2 FPGA families have a feature called TraceID™ for securing original design and IP. TraceID is a unique, 64-bit code that is programmed during the manufacturing of the device, thus linking a specific design to a specific device. This ensures that only the original product manufacturer who ordered a specific device has access to it.

2. Why is TraceID Important?

TraceID can be used to prevent overbuilding and cloning of user designs. Overbuilding occurs when a contract manufacturer builds more products than the original company has approved. These extra products are, in turn, sold through other channels for profit without the knowledge or consent of the original company. Cloning is the act of making exact copies of a product and selling them under a different name at a lower price (thus reducing the OEM profit). These practices cause OEMs to lose money not only from lost sales and lower margins, but also from unseen support costs such as failure analysis.

The TraceID feature can be used to prevent both overbuilding and cloning.

3. How Does TraceID Work?

TraceID is a unique, 64-bit device identification tracking number which is stored in the feature row of the device.

The 64 bits contain the following unique information:

- Lattice Traceability Code
- User-defined design-specific code

The TraceID register format is shown in [Table 3.1](#).

Table 3.1. TraceID Register

| | |
|---------|---------------------------|
| [63:56] | 8 bits, User-Defined Code |
| [55:0] | Lattice Traceability Code |

The most significant eight bits are the user-defined design-specific code. These eight bits are read and write accessible. The remaining 56 bits are read-only and are programmed at the time the device is manufactured by Lattice. The 8-bit user-defined code plus the 56-bit factory-programmed portion together guarantee that every device has a unique TraceID. This uniqueness provides OEMs greater control over how many of their products are introduced in the market and the ability to detect false products.

4. How to Program the TracelD User-Defined Code

Lattice design software can be used to set a specific 8-bit user-defined code in the TracelD register. The TRACE_ID_BINARY preference must be set to a chosen value in the LPF file.

In the LPF file, set the TracelD value using the following format:

```
TRACEID "<8-bit value>"
```

Below is an example:

```
TRACEID "01101001"
```

When a programming file is created, the 8-bit TracelD value is embedded in the JED file feature row. During programming, the TracelD registers in the device are updated with the one in the JED. The default value for the user defined code is 00000000.

5. Accessing the TracelD Register

You can read the TracelD value for the devices using the methods listed in [Table 5.1](#).

Table 5.1. Methods to Read the TracelD Values

| Device | Method to Read TracelD Value |
|---|---|
| MachXO2 and MachXO3 | Internal sysCONFIG WISHBONE port or externally through the JTAG, SSPI, or I2C ports |
| ECP5 and ECP5-5G | JTAG or SSPI port |
| MachXO5-NX | JTAG, SSPI, or I2C ports |
| CrossLink-NX, CrossLink-NX-33, CrossLinkU-NX, Certus-NX, and CertusPro-NX | JTAG, SSPI, I2C, or sysCONFIG Lattice Memory Mapped Interface (LMMI) ports |
| Avant and Nexus 2 | JTAG, SSPI, or sysCONFIG LMMI ports |

For Avant and Nexus 2 devices, the READ_UIDCODE_PUB_L (0x01030000) and READ_UIDCODE_PUB_H (0x01040000) commands are used to read the TracelD registers through SSPI or sysCONFIG LMMI ports, or the UIDCODE_PUB JTAG opcode (0x19) is used.

For all other device families, the UIDCODE_PUB command is used to read the TracelD register. The OPCODE for the UIDCODE_PUB command is 00011001 (0x19).

The TracelD value can also be read using either the Lattice Diamond® Programmer tool or the Lattice Radiant® Programmer tool, depending on the device family. [Figure 5.1](#) and [Figure 5.2](#) show the settings required to read the TracelD through the Diamond Programmer and Lattice Radiant Programmer tools, respectively.

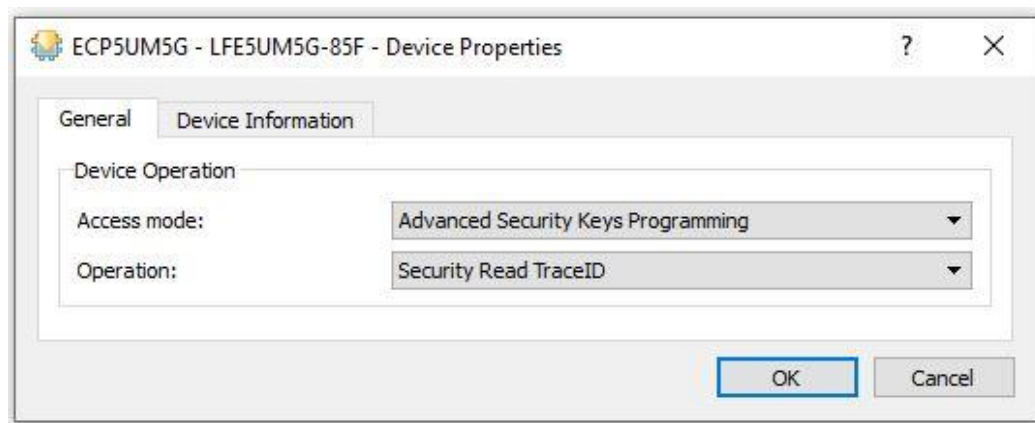


Figure 5.1. Read TracelD through Lattice Diamond Programming Tool

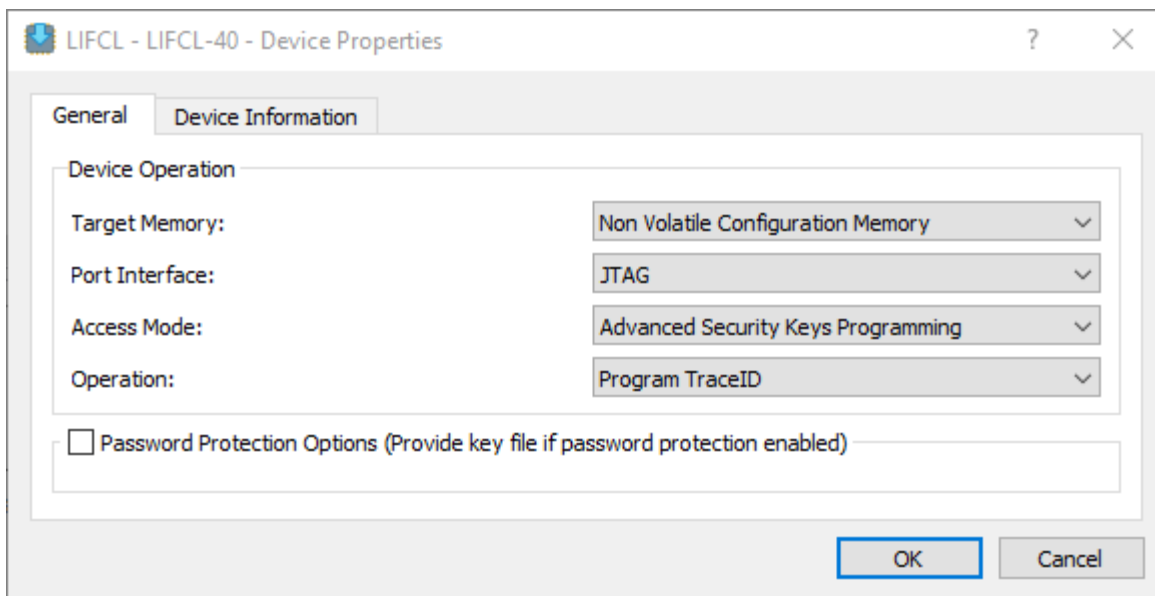


Figure 5.2. Program TraceID through Lattice Radiant Programmer Tool

For MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-NX, Certus-NX, CertusPro-NX, and MachXO5-NX devices, bits [63:56] of the TraceID can be written through the Feature Row. In Diamond Programmer, select **Advanced Security Keys Programming** in **Access Mode** and **Security Program Feature Rows** in **Operation**. In Radiant Programmer, select **Advanced Security Keys Programming** in **Access Mode** and **Program TraceID** in **Operation**. When the operation is run, a window similar to Figure 5.3 opens. For Avant and Nexus 2 devices, bits [63:56] of the TraceID can be programmed through Radiant Programmer by selecting **Feature Rows Programming** in **Access Mode** and **Program Feature Row** in **Operation** (Figure 5.4).

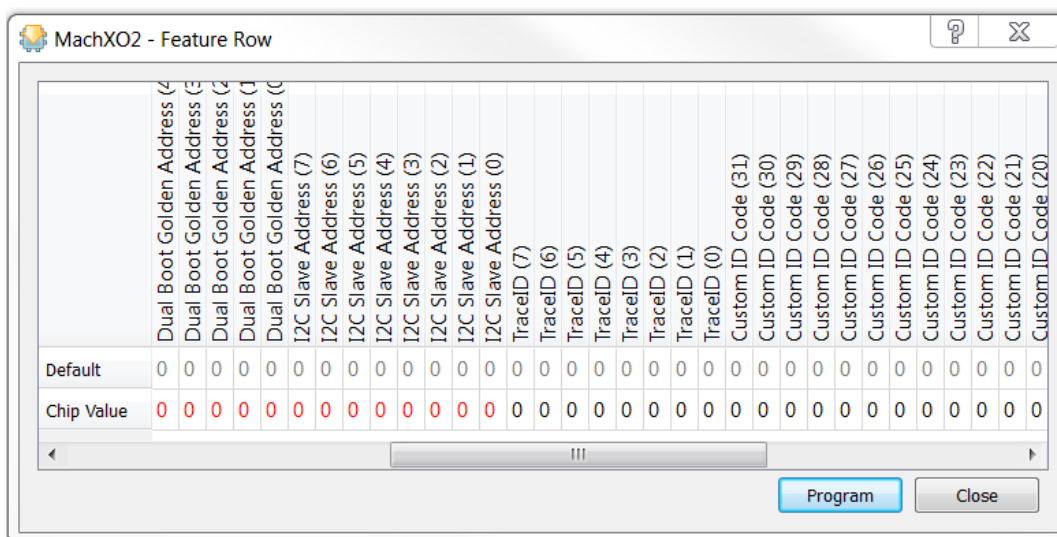


Figure 5.3. Feature Row Editor

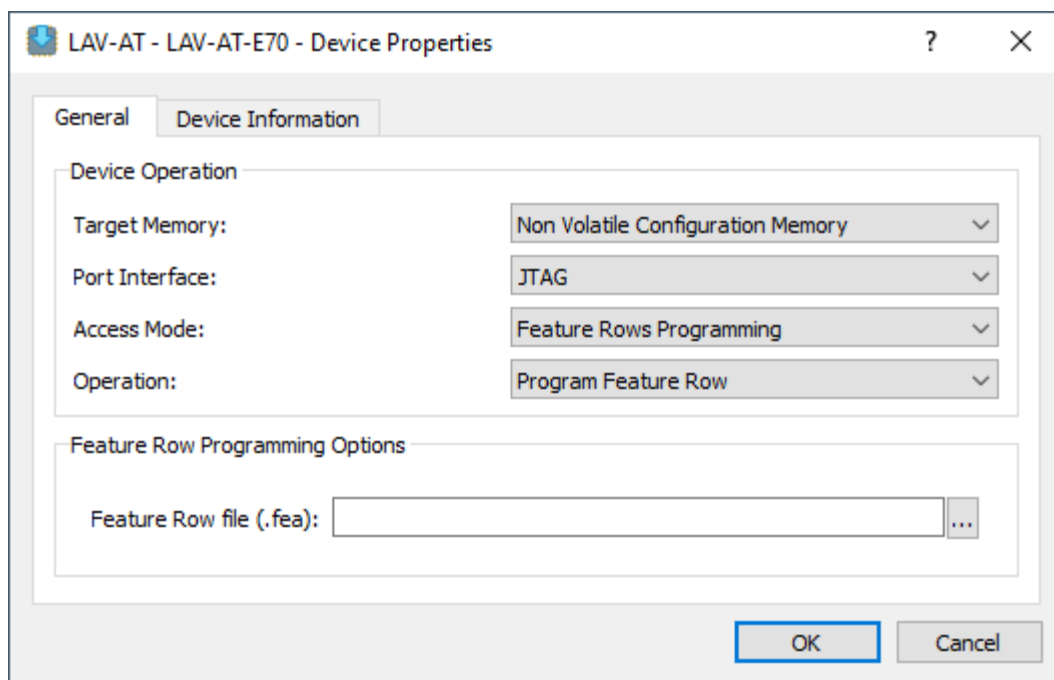


Figure 5.4. Avant – Program Feature Row through Lattice Radiant Programmer Tool

5.1. TraceID Access Through the JTAG Port

The JTAG port has access to configuration logic resources including the TraceID. To read the TraceID using JTAG, shift the UIDCODE_PUB command (0x19) into the instruction register then read the 64-bit TraceID out of the data register.

5.2. TraceID Access Through the sysCONFIG WISHBONE Target Interface (MachXO2 and MachXO3 Devices Only)

The sysCONFIG WISHBONE target interface of the EFB module enables designers to access the TraceID directly from the PLD core logic. The sysCONFIG WISHBONE bus signals are utilized by a sysCONFIG WISHBONE host that designers can implement using the general purpose PLD resources. In addition to the sysCONFIG WISHBONE bus signals, an interrupt request output signal is brought to the PLD fabric.

The sysCONFIG WISHBONE interface communicates to the configuration logic through a set of data, control and status registers. Table 5.2 shows the register names and their functions. These registers are the subset of the EFB register map. The details of the sysCONFIG WISHBONE target interface pins, EFB register map, and sysCONFIG WISHBONE register definition can be found in [Using User Flash Memory and Hardened Control Functions in MachXO2 Devices \(FPGA-TN-02162\)](#) and [Using Hardened Control Functions in MachXO3 Devices \(FPGA-TN-02063\)](#).

Table 5.2. sysCONFIG WISHBONE Register

| sysCONFIG WISHBONE to CFG Register Name | Register Function | Address | Access |
|---|--------------------------|---------|------------|
| CFGCR | Control | 0x70 | Read/Write |
| CFGTXDR | Transmit Data | 0x71 | Write |
| CFGSR | Status | 0x72 | Read |
| CFGRXDR | Receive Data | 0x73 | Read |
| CFGIRQ | Interrupt Request | 0x74 | Read/Write |
| CFGIRQEN | Interrupt Request Enable | 0x75 | Read/Write |

When using the sysCONFIG WISHBONE bus interface, the opcodes, operand, and data are written to the CFGTXDR register. This is required only when communicating with the configuration logic inside the MachXO2 and MachXO3 devices. The TraceID can be accessed through the sysCONFIG WISHBONE interface by writing the opcode and operand into the CFGTXDR register. The TraceID information can then be read from the CFGRXDR register.

The opcode to access the TraceID is 0x19h and the operand is 0x000000h.

5.3. TraceID Access through the sysCONFIG Target SPI Port (MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX Devices Only)

The target SPI port can be used to perform read operations to the TraceID. The configuration SPI port is shared with the hardened SPI core of the EFB module. Asserting the configuration SN (select) pin causes the SPI port to transition its service from user mode to configuration mode. The TraceID can be accessed using the SPI port by following the command sequence described below.

To access the TraceID:

1. Pull down the configuration SN select pin (SPI Target Select).
2. Send UIDCODE_PUB (0x19) command from the external SPI controller.
3. Send 24-bit operand 0x000000.
4. Receive TraceID[63:0] from SPI target in next 64 SCLK cycle.
5. Pull up configuration SN select pin.

The complete sequence is shown in [Figure 5.5](#) and [Figure 5.6](#).

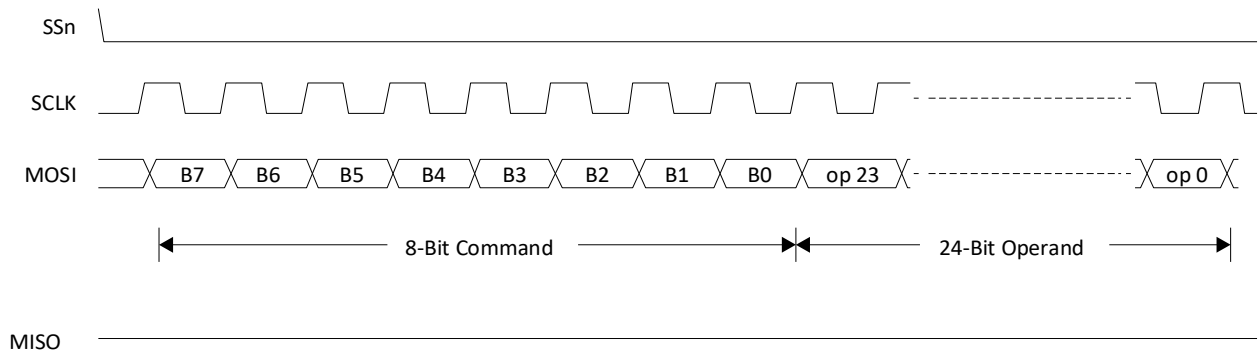


Figure 5.5. TraceID Read through SPI

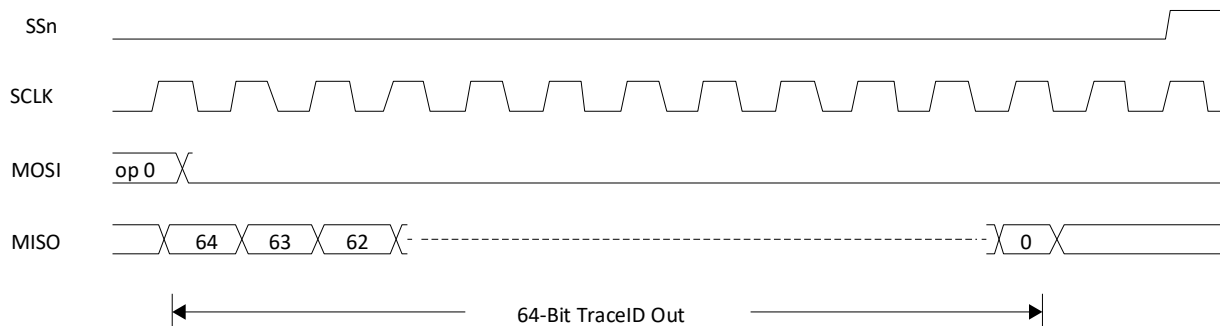


Figure 5.6. TraceID Read through SPI, Continued

5.4. TraceID Access through the sysCONFIG Target SPI Port (Avant and Nexus 2 Devices Only)

The target SPI port can be used to perform read operations to the TraceID by following the command sequence described below.

To access the TraceID:

1. Pull down the configuration SN select pin (SPI Target Select).
2. Send UIDCODE_PUB_L command (0x01030000) from the external SPI controller.
3. Receive TraceID[31:0] from SPI target in next 32 SCLK cycle.
4. Pull up configuration SN select pin.
5. Pull down the configuration SN select pin (SPI Target Select).
6. Send UIDCODE_PUB_H command (0x01040000) from the external SPI controller.
7. Receive TraceID[63:31] from SPI target in next 32 SCLK cycle.
8. Pull up configuration SN select pin. The complete sequence is shown in [Figure 5.7](#) and [Figure 5.9](#).

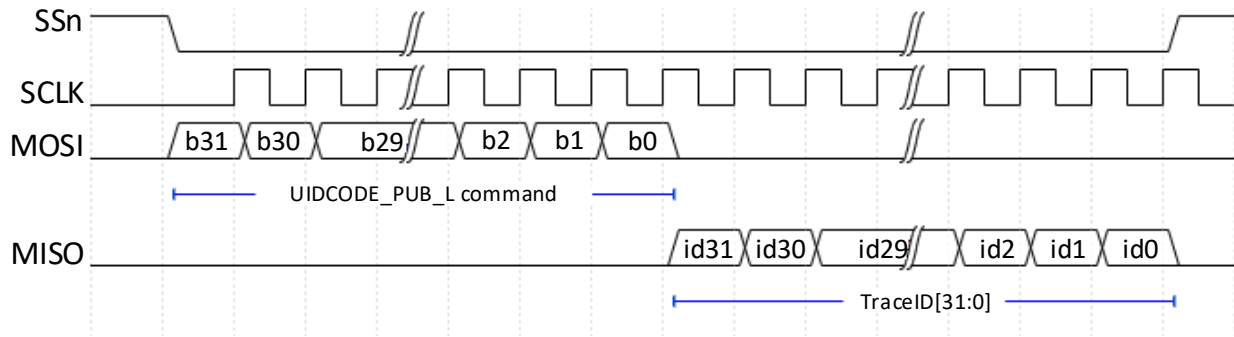


Figure 5.7. TraceID Read through SPI (Avant and Nexus 2 Devices only)

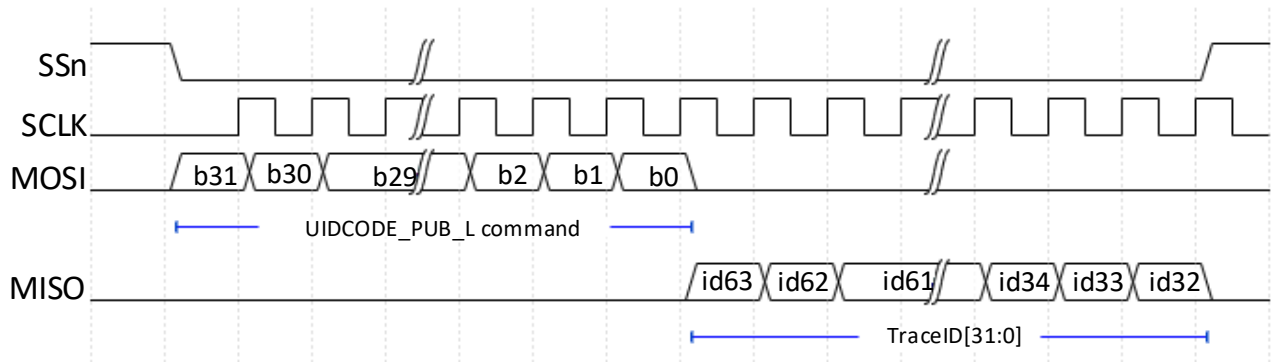


Figure 5.8. TraceID Read through SPI, Continued (Avant and Nexus 2 Devices only)

5.5. TraceID Access through the I2C Port (MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX Only)

All MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX devices have an I2C port, which can be used to perform read operations to the TraceID.

For MachXO2 and MachXO3, the configuration I2C port is shared with the hardened I2C primary core of the EFB module. Addressing the I2C primary port with the configuration address changes the port service from user mode with the sysCONFIG WISHBONE interface to configuration mode. The pin locations of the configuration I2C port are pre-assigned in all MachXO2 and MachXO3 devices.

For CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX devices, the configuration I2C port is separate from the hard I2C block accessible from user logic, and has pre-defined pins. Sending the I2C address followed by the correct preamble activates the port. The I2C configuration port is a shared interface to the configuration logic with the configuration SPI port, and only one can be active at a time.

There is one address byte required since 7-bit addresses are used. The last bit of the address byte is the read/write bit and should always be set according to the required operation. This 7-bit I2C address is 1000000 (80h) which is the default address. The read sequence uses a repeated start condition during the sequence to avoid bus release during communication. For 10-bit addressing the I2C target address is 10'b1111000000.

To read the TraceID through the I2C bus:

1. Send start condition.
2. Send default target address (0x80) and write command.
3. Send the UIDCODE_PUB command (0x19).
4. Send the 24-bit operand 0x000000 in three single-byte transfers.
5. Send repeated start.
6. Send the target address and read command.
7. Read the first through seventh bytes of the TraceID and send ack for each byte read.
8. Read the last TraceID byte and send nack.
9. Send the stop command.

Figure 5.9 shows the TraceID read through I2C.

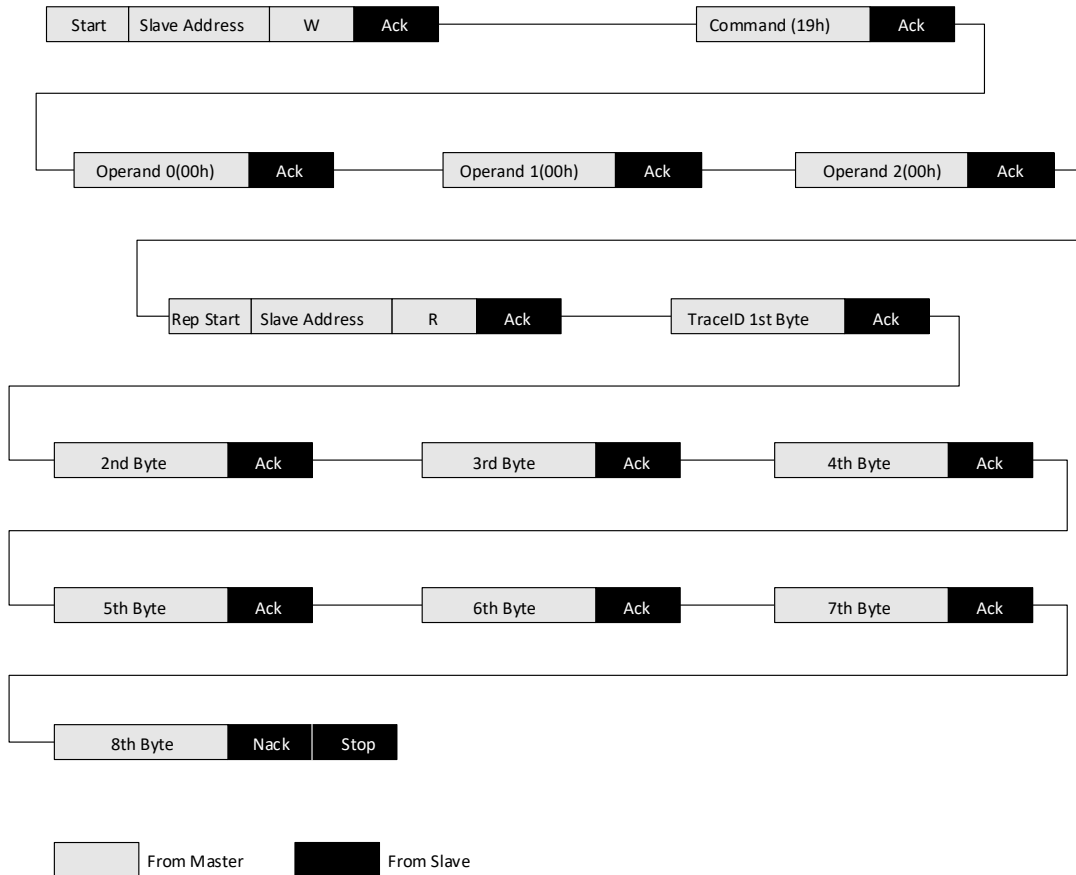


Figure 5.9. TraceID Read through I2C

5.6. TraceID Access through sysCONFIG LMMI (CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, Avant, and Nexus 2 Only)

The sysCONFIG LMMI is a memory-mapped address/data interface. It defines a standard set of interface signals for register/memory access and supports both single and burst transactions. sysCONFIG LMMI allows the designer to access the TraceID directly from the PLD core logic. For Avant and Nexus 2 devices, the command to read the TraceID lower bits is READ_UIDCODE_PUB_L (0x01030000) and the command to read the upper bits is READ_UIDCODE_PUB_H (0x01040000). For all other device families, the opcode to read the full 64-bit TraceID is 0x19. Details on sysCONFIG LMMI can be found in [Lattice Memory Mapped Interface and Lattice Interrupt Interface User Guide \(FPGA-UG-02039\)](#).

6. Example Uses of TraceID

TraceID can be used to validate that the device or the system in general, is authorized by the OEM.

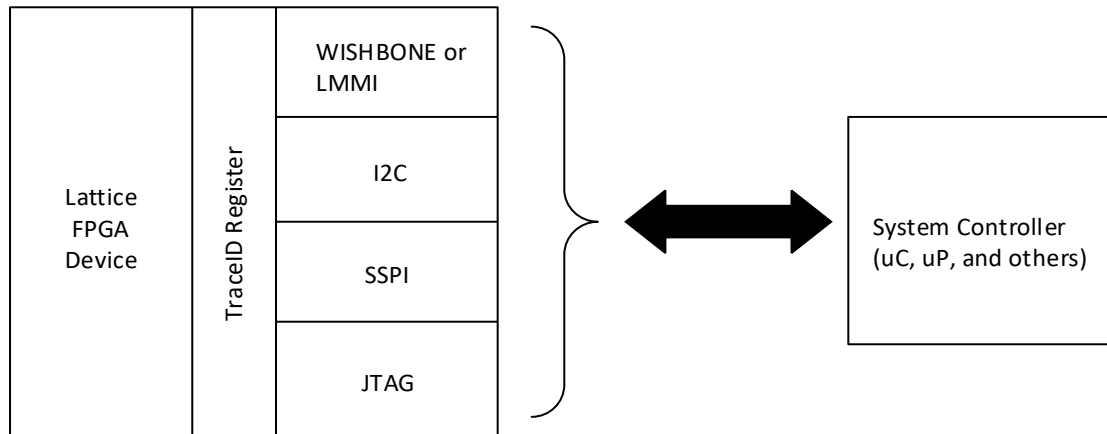


Figure 6.1. TraceID Read through I2C

One way of implementing this is to use the system controller, either a microcontroller or microprocessor, to access the TraceID register (through sysCONFIG WISHBONE, sysCONFIG LMMI, I2C, SPI or JTAG interfaces) and compare it against a list of approved TraceID device tables. If the TraceID matches the approved device list, the system can continue to function as intended.

In cases where the read TraceID does not match with the approved device list, the system controller can choose to log the event and take one of the following actions:

- Stall – Stop working
- Continue with limited functionality – Partial operation of system
- Erase or destroy integral data in the system – Erase the boot ROM, Flash memory, register tables, etc.

References

- [CrossLink-NX](#) web page
- [Certus-NX](#) web page
- [CertusPro-NX](#) web page
- [MachXO2](#) web page
- [MachXO3](#) web page
- [MachXO5-NX](#) web page
- [Avant-E](#) web page
- [Avant-G](#) web page
- [Avant-X](#) web page
- [Certus-N2](#) web page
- [ECP5/ECP5-5G](#) web page
- [Lattice Radiant Software](#) web page
- [Lattice Diamond Software](#) 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

Revision 2.8, April 2025

| Section | Change Summary |
|--------------------------------|---|
| All | <ul style="list-style-type: none"> Replaced WISHBONE with <i>sysCONFIG WISHBONE</i>. Replaced LMMI with <i>sysCONFIG LMMI</i>. Changed I²C to I2C. |
| Abbreviations in This Document | <ul style="list-style-type: none"> Updated the section title to the current. Newly added JED and JEDEC. |
| Accessing the TraceID Register | <ul style="list-style-type: none"> Added description about Avant and Nexus 2 devices support under Table 5.1. Methods to Read the TraceID Values. Updated the OPCODE for UIDCODE_PUB command to 00011001 (0x19); Newly added the TraceID Access through the sysCONFIG Target SPI Port (Avant and Nexus 2 Devices Only) section. Unified the numeric representation in the TraceID Access through the I2C Port (MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX Only) section. |

Revision 2.7, November 2024

| Section | Change Summary |
|--------------------------------|--|
| Introduction | Added Nexus™ 2 to the description. |
| Accessing the TraceID Register | <p>Added Nexus 2 to Table 5.1. Methods to Read the TraceID Values and the introductory description.</p> <p>Added Nexus 2 to the TraceID Access through sysCONFIG LMMI (CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, Avant, and Nexus 2) section.</p> |
| References | Added Certus-N2 web page. |

Revision 2.6, May 2024

| Section | Change Summary |
|--|---|
| All | Changed <i>master</i> to <i>controller</i> and <i>slave</i> to <i>target</i> . |
| Disclaimers | Updated disclaimers. |
| Inclusive Language | Added inclusive language boilerplate. |
| Introduction | Added support for CrossLink-NX-33 and CrossLinkU-NX devices. |
| How Does TraceID Work? | <ul style="list-style-type: none"> Updated the unique information in TraceID. Updated Table 3.1. TraceID Register. |
| How to Program the TraceID User-Defined Code | Changed the section title from <i>How to Program User-Defined Code of the TraceID for MachXO2</i> to <i>How to Program the TraceID User-Defined Code</i> . |
| Accessing the TraceID Register | <ul style="list-style-type: none"> Updated the methods to read TraceID values for devices in Table 5.1. Methods to Read the TraceID Values. Updated Figure 5.4. Avant - Program Feature Row through Lattice Radiant Programmer Tool. Added support for CrossLink-NX-33 and CrossLinkU-NX devices in the TraceID Access through the I2C Port (MachXO2, MachXO3, CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, and MachXO5-NX Only) section. Added support for CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX devices in the TraceID Access through sysCONFIG LMMI (CrossLink-NX, CrossLink-NX-33, CrossLinkU-33, Certus-NX, CertusPro-NX, Avant, and Nexus 2) section. |
| References | Added this section. |

Revision 2.5, January 2023

| Section | Change Summary |
|--------------------------------|---|
| Accessing the TraceID Register | Fixed the figure reference error for Figure 5.1. Read TraceID through Diamond Programming Tool and Figure 5.2. Program TraceID through Lattice Radiant Programmer Tool. |
| Technical Support Assistance | Removed https: from the Technical Support Assistance Section. |

Revision 2.4, November 2022

| Section | Change Summary |
|--------------------------------|--|
| All | Updated formatting styles. |
| Accessing the TraceID Register | Added link to FPGA-UG-02039 in Section 5.5 TraceID Access Through the LMMI (Avant Only). |

Revision 2.3, May 2022

| Section | Change Summary |
|--------------------------------|---|
| All | Added MachXO5-NX device family. |
| Accessing the TraceID Register | Updated the document numbers of referenced technical notes. |

Revision 2.2, June 2021

| Section | Change Summary |
|---------|-----------------------------------|
| All | Added CertusPro-NX device family. |

Revision 2.1, October 2020

| Section | Change Summary |
|--------------------------------|--|
| Accessing the TraceID Register | <ul style="list-style-type: none"> Updated process of writing TraceID through Feature Row. Updated Figure 5.2. Read TraceID through Lattice Radiant Programmer Tool. Corrected reference to Figure 5.6. TraceID Read through I2C. |

Revision 2.0, June 2020

| Section | Change Summary |
|---------|--|
| All | <ul style="list-style-type: none"> Added Certus-NX device family. Minor changes in formatting and style. |

Revision 1.9, December 2019

| Section | Change Summary |
|--------------------------------|--|
| All | <ul style="list-style-type: none"> Changed document number from TN1207 to FPGA-TN-02084. Updated document template. Updated document links. |
| Disclaimers | Added this section. |
| Introduction | Added CrossLink-NX device family. |
| Accessing the TraceID Register | <ul style="list-style-type: none"> Added CrossLink-NX device family. Added Figure 5.2. Read TraceID through Lattice Radiant Programmer Tool. |

Revision 1.8, October 2015

| Section | Change Summary |
|--------------------------------|------------------------------|
| Introduction | Added ECP5-5G device family. |
| Accessing the TraceID Register | Added ECP5-5G device family. |
| Technical Support Assistance | Updated this section. |

Revision 1.7, March 2015

| Section | Change Summary |
|---------|---|
| All | Product name adjustment. Included MachXO3LF device. |

Revision 1.6, June 2014

| Section | Change Summary |
|---------|---|
| All | Corrected typographical errors on product name. |

Revision 1.5, April 2014

| Section | Change Summary |
|---------|---|
| All | <ul style="list-style-type: none"> Changed document title to Using TraceID. Added support for MachXO3L device family. |

Revision 1.4, April 2014

| Section | Change Summary |
|--------------------------------------|------------------------------------|
| TraceID Access Through the JTAG Port | Updated Table 1, TraceID Register. |

Revision 1.3, March 2014

| Section | Change Summary |
|---------|---|
| All | Updated Technical Support Assistance information. |

Revision 1.2, March 2014

| Section | Change Summary |
|---------|---|
| All | <ul style="list-style-type: none"> Changed document title to Using TraceID in MachXO2 and ECP5 Devices. Added support for ECP5 device family. |

Revision 1.1, February 2012

| Section | Change Summary |
|--------------------------------|--|
| All | <ul style="list-style-type: none"> Document status changed from advance to final. Updated document with new corporate logo. |
| Accessing the TraceID Register | Added the following sections: <ul style="list-style-type: none"> TraceID Access Through the JTAG Port TraceID Access Through the WISHBONE Slave Interface TraceID Access Through the Slave SPI Port TraceID Access Through the I²C Port |

Revision 1.0, November 2010

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



www.latticesemi.com