Using TraceID

June 2014 Technical Note TN1207

Introduction

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

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

The MachXO2™ PLD family, MachXO3L and ECP5™ family have a feature called TraceID for securing original design and IP. TraceID is a unique, 64-bit code that is programmed during 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.

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 will 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.

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:

- Wafer lot number
- Wafer number within the lot
- Die X location
- Die Y location
- User-defined design-specific code

The TraceID register format is shown in Figure 15-1.

Figure15-1. TraceID Register

[63 : 56]	8 bits, User-Defined Code
[55 : 24]	32 bits, Wafer Lot Number
[23 : 19]	5 bits, Wafer Number (within the lot)
[18 : 12]	7 bits, Wafer Die X Location
[11 : 5]	7 bits, Wafer Die Y Location
[4 : 0]	5 bits, Extra Spare bits

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 will have 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.

How to Program User-Defined Code of the TraceID for MachXO2

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

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

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

Below is an example:

```
TRACEID "01101001"
```

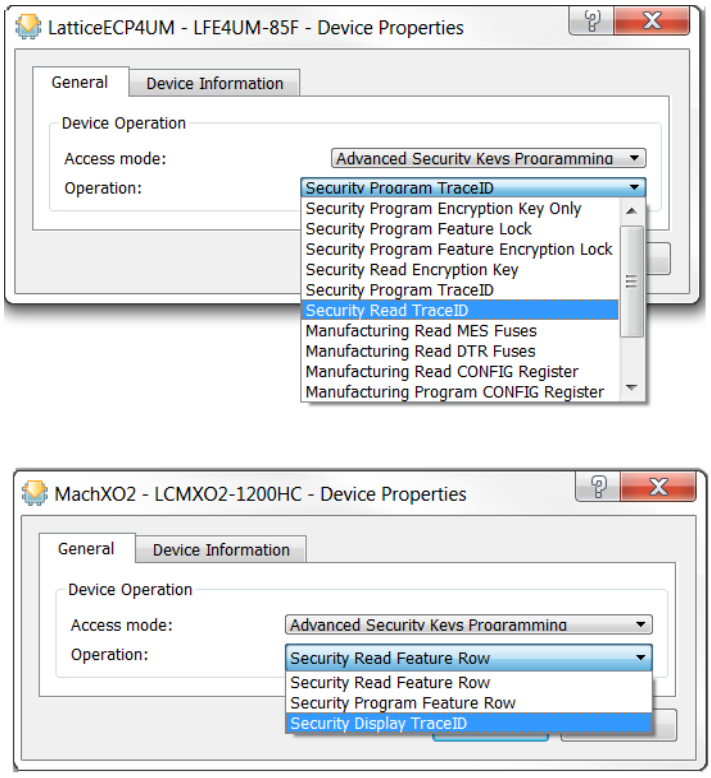
When a programming file is created, the 8-bit TraceID value is embedded in the JED file feature row. During programming, the TraceID registers in the device are updated with the one in the JED. The default value for the user defined code is “00000000”.

Accessing the TraceID Register

The TraceID value in the MachXO2 and MachXO3L devices can be read using the internal WISHBONE port or externally through the JTAG, SSPI or I²C ports. The TraceID value in ECP5 device can be read through JTAG or SSPI port since ECP5 does not have I²C port nor internal WISHBONE. For SSPI, I²C and JTAG interfaces, the UIDCODE_PUB command must be used to read the TraceID register. The OPCODE for the UIDCODE_PUB command is “00011001”.

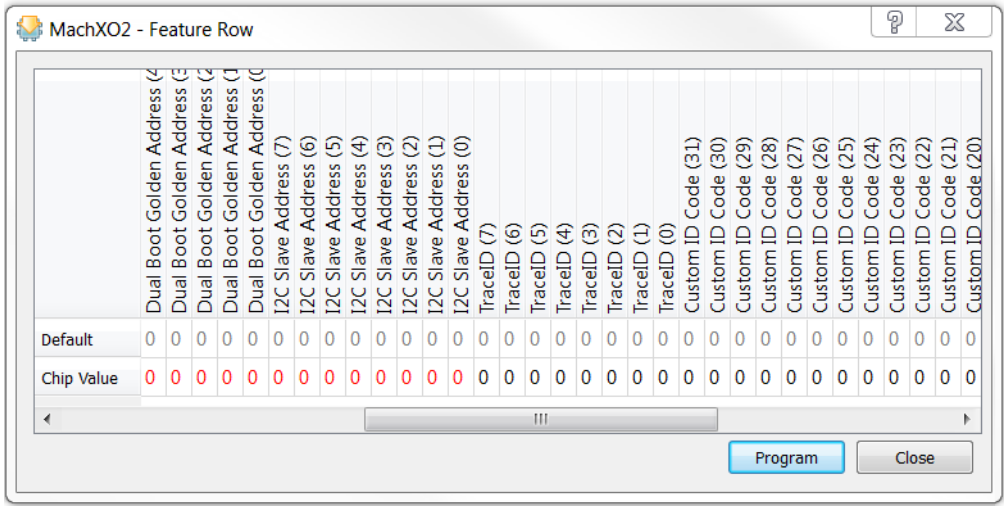
The TraceID value can also be read using the Diamond® Programmer tool.

Figure15-2. Programmer Tool



For MachXO2 and MachXO3L, the TraceID can be written through the Feature Row. Choose **Security Program Feature Row** and the window shown in Figure 15-3 will open. From here, the Feature Row which includes the TraceID, can be read from and written to.

Figure15-3. Feature Row Editor



TraceID Access Through the JTAG Port

The JTAG port has access to all configuration logic resources including the TraceID. To read the TraceID using JTAG, shift the command (0x19h) into the instruction register then read the 64-bit TraceID out of the data register.

TraceID Access Through the WISHBONE Slave Interface (MachXO2 and MachXO3L Only)

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

The WISHBONE interface communicates to the configuration logic through a set of data, control and status registers. Table 15-1 shows the register names and their functions. These registers are a subset of the EFB register map. The detail of the WISHBONE slave interface pins, EFB register map, and WISHBONE register definition can be found in TN1205, Using User Flash Memory and Hardened Control Functions in MachXO2 Devices and TN1293, Using Hardened Control Functions in MachXO3L Devices.

Table15-1. WISHBONE Registers

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 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 MachXO3L devices. The TraceID can be accessed via the 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.

TraceID Access Through the Slave SPI Port

The Slave 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 will cause 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.

1. Pull down the configuration SN select pin (SPI Slave Select)
2. Send 8'h19 command from the external SPI master
3. Send 24-bit operand 24'h000000
4. Receive TraceID from SPI slave in next 64 SCLK cycle
5. Pull up configuration SN select pin

The complete sequence is shown in Figures 15-4 and 15-5.

Figure15-4. TraceID Read via SPI

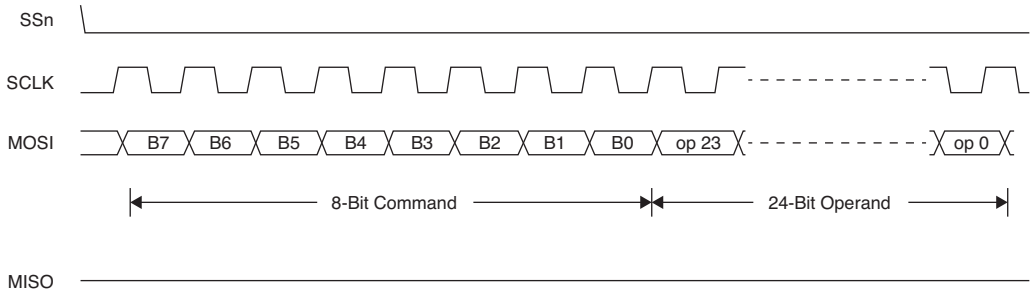
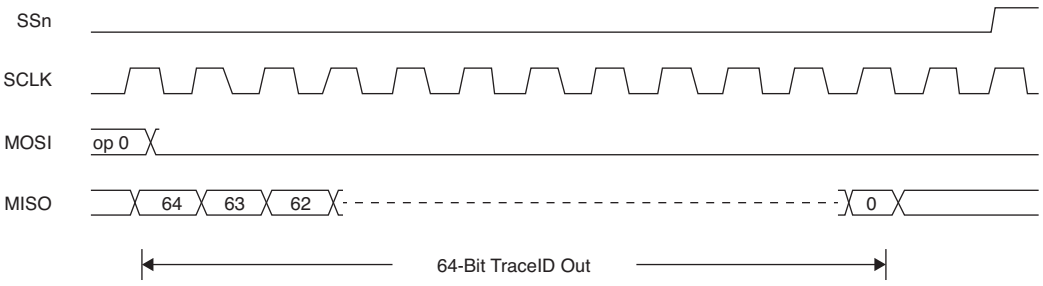


Figure15-5. TraceID Read via SPI, Continued



TraceID Access Through the I²C Port (MachXO2 and MachXO3L Only)

All MachXO2 devices have an I²C port, which can be used to perform read operations to the TraceID. 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 will change the port service from user mode with the WISHBONE interface to configuration mode. The pin locations of the configuration I²C port are pre-assigned in all MachXO2 and MachXO3L devices.

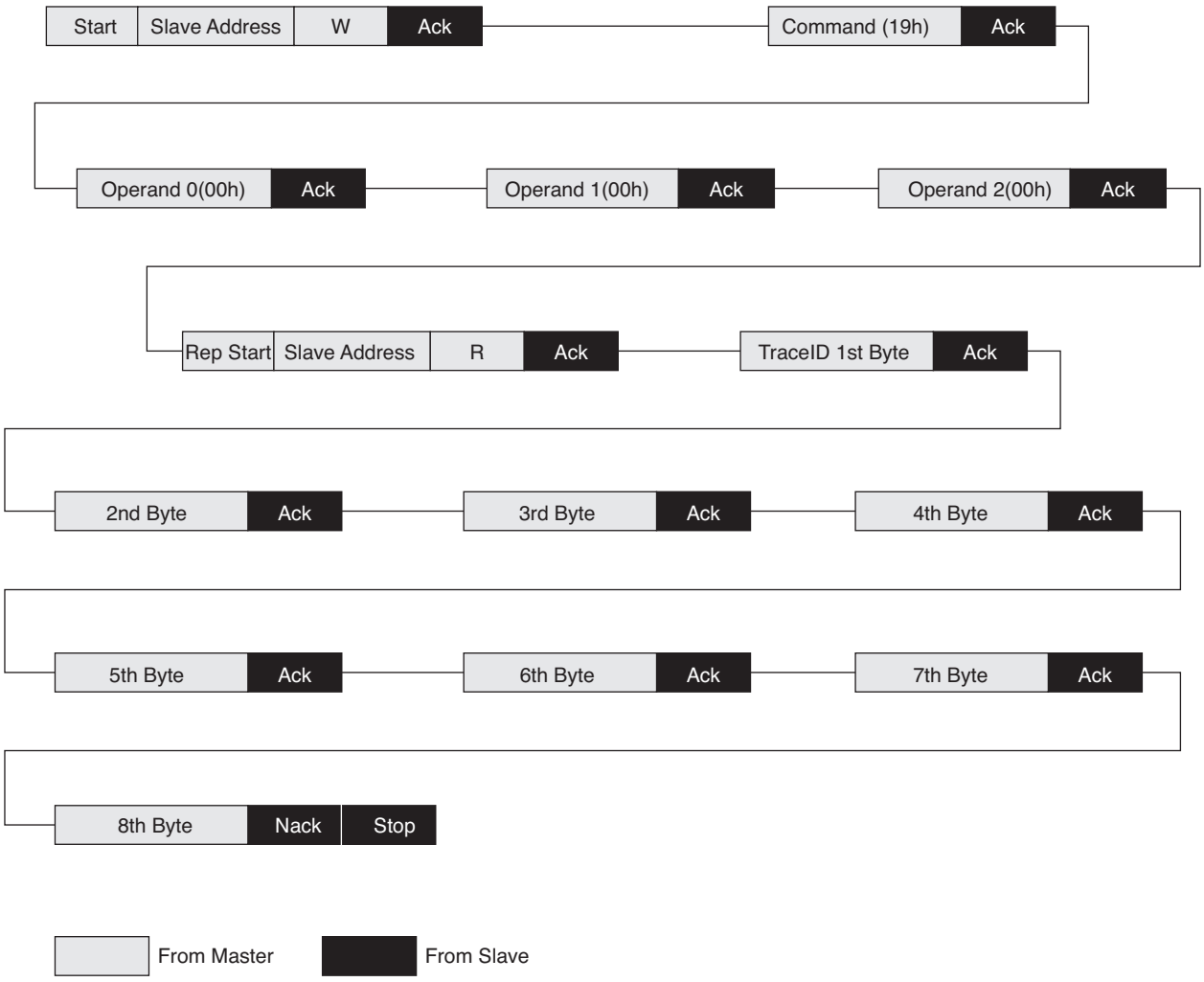
There is one address byte required since 7-bit addresses are used. The last bit of the address byte is the read/write bit and should always be set according to the required operation. This 7-bit I²C address is 1000000 (80h) which is the default address. The read sequence uses a repeated start condition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

1. Send start condition.
2. Send default slave address (8'h80) and write command.
3. Send the 8-bit command 8'h19.
4. Send the 24-bit operand 24'h000000 in three single-byte transfers.

5. Send repeated start.
6. Send the slave address and read command.
7. Read the first byte of the TraceID and send ack.
8. Read the second to seventh bytes of the TraceID and send ack for each byte read.
9. Read the last TraceID byte and send nack.
10. Send the stop command.

Table 15-6 shows the TraceID read via I^2C.

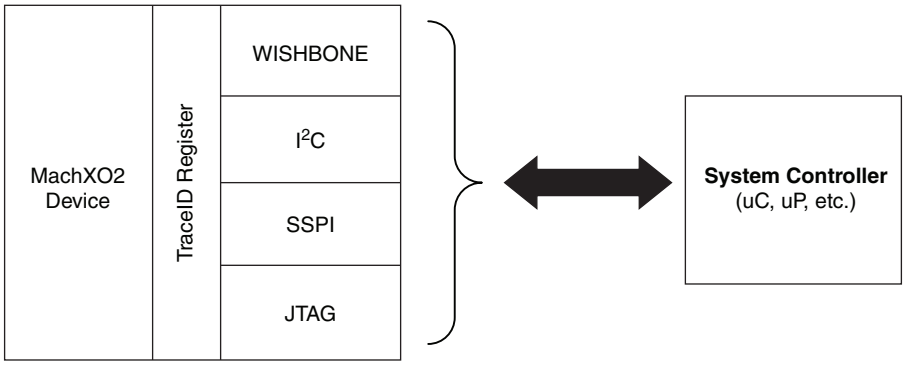
Figure15-6. TraceID Read via I^2C



Example Uses of TraceID

TraceID can used to validate that the MachXO2 device or the system in general, as authorized by the OEM.

Figure15-7. Example System-level Use of TraceID



One way of implementing this is to use the system controller, either a microcontroller or microprocessor, to access the TraceID register (via WISHBONE, I^2C, 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.

Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
November 2010	01.0	Initial release.
February 2012	01.1	Document status changed from advance to final.
		Updated document with new corporate logo.
		Added the following new sections: • TraceID Access Through the JTAG Port • TraceID Access Through the WISHBONE Slave Interface • TraceID Access Through the Slave SPI Port • TraceID Access Through the I^2C Port
March 2014	01.2	Changed document title to Using TraceID in MachXO2 and ECP5 Devices
		Added support for ECP5.
		Update Technical Support Assistance information.
	01.3	Changed document title to Using TraceID in MachXO2 and ECP5 Devices
		Added support for ECP5 device family.
April 2014	01.4	Updated Table 15-1, TraceID Register.
	01.5	Changed document title to Using Trace ID.
		Added support for MachXO3L device family.
June 2014	1.6	Corrected typographical errors on product name.