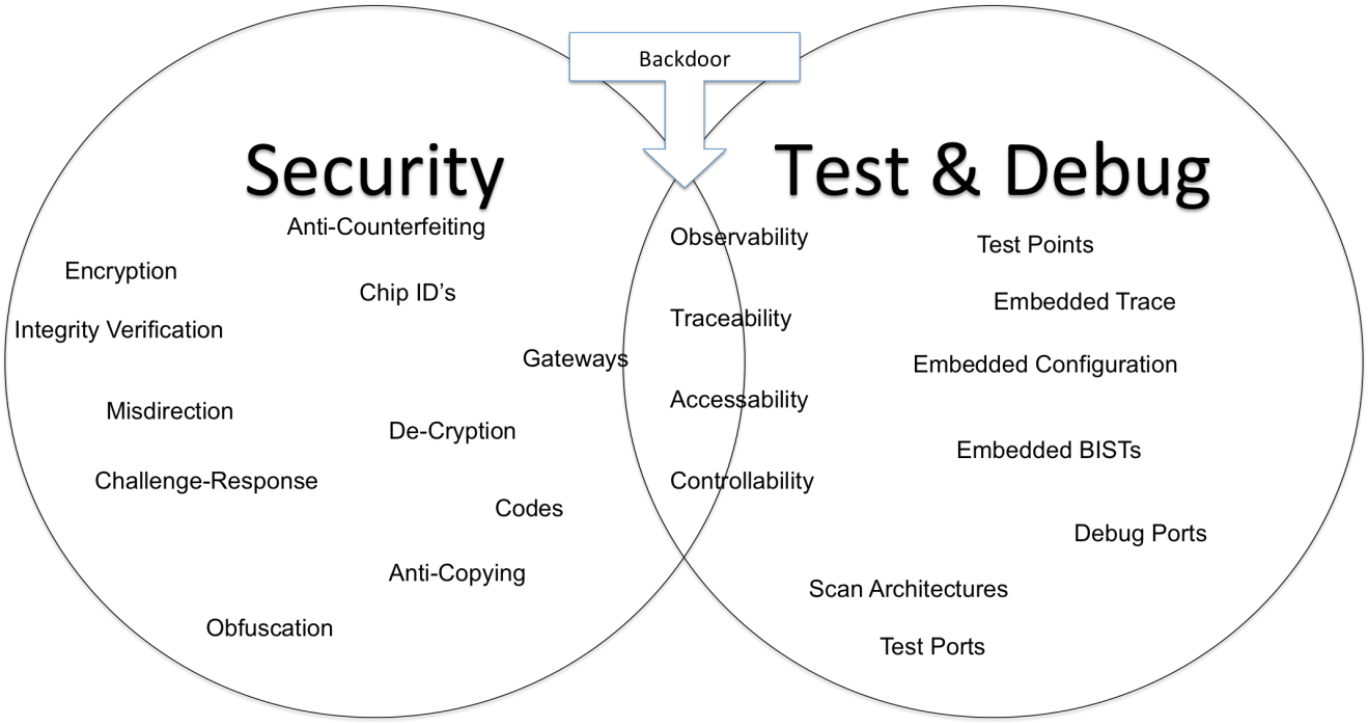
JTAG | IJTAG

SEMICONDUCTOR AND BOARD

TEST SECURITY



EBOOK

BY

AL CROUCH

JENNIFER DWORAK

Al Crouch – Chief Technologist, Embedded Instrumentation Methodologies and IJTAG – ASSET InterTech, Inc.

Al investigates the use of embedded instruments for IC test and debug, board test and debug, and software debug. He is a Senior Member of the IEEE and serves as the vice chairman of the IEEE P1687 IJTAG working group that is developing this standard for embedded instruments. He has contributed significantly to its hardware architecture definition. Al is also a member of the P1838 Working Group on 3D test and debug, and co-chair of the iNEMI BIST group, which is defining the use of embedded instruments for board test. Al's previous experience includes design-for-test and debug at various semiconductor companies, including TI, DEC and Motorola, as well as chief scientist at startup companies DAFCA and INOVYS.

JenniferDworak – Southern Methodist University (SMU)

Jennifer Dworak is an assistant professor at Southern Methodist University. She has published approximately 40 papers in conferences and journals in semiconductor test, fault tolerance, and hardware security. She was a co-author on a paper that won the Best Paper Award at the 1999 VLSI Test Symposium and was a recipient of a TTTC Naveena Nagi Award presented at the 2004 VLSI Test Symposium. Her research interests include test set optimization, on-chip test, built-in-self-repair, hardware Trojan detection, and DFT security. She is a member of IEEE and ACM.

Table of Contents

Table of Figures

Executive Summary

Typically, the normal operating functions and data contained on many chips and circuit boards are secured by encrypting the data passing through Ethernet, USB or other ports, and the data stored in on-board memory. However, there are no inherent security measures associated with the IEEE 1149.1 Boundary-Scan (often referred to as JTAG) ports found on chips and boards. Since a wealth of embedded instruments and other intellectual property (IP) is now being inserted into chips, the JTAG port represents a security weakness or back-door vulnerability through which hackers can reverse engineer, extract sensitive data or disrupt operations.

There are no industry standards for chip and board test security. Any security on a chip's JTAG port today is a custom method, developed specifically for a certain device by the chip's designer. Examples include challenge-response mechanisms, encryption-decryption, data locks and others. As a result, JTAG security measures will vary by chip supplier. Board/system developers are faced with the daunting task of integrating a variety of different security protocols, each associated with a certain chip vendor. Exacerbating the problem is the fact that some chips have no test security at all. Ultimately, chip, board and system developers, as well as manufacturing engineers, become concerned that this piecemeal approach to security will not protect chip or board resources and any security measures that are present only add undue complexity and time to manufacturing test.

This eBook on board test security is introductory. It addresses security issues currently associated with circuit boards and chips on boards, and discusses several potential test and debug security countermeasures that could be included into a board-level security standard in the future.

Introduction

Several recent events have brought electronic security into the limelight. These include the counterfeiting of chips and entire printed circuit boards, disrupting normal system operations and embedding malicious Trojan circuits within semiconductors and firmware. As a result of these events and hacker attacks, laws concerning the sale and distribution of counterfeit chips have been changed. In addition, many semiconductor and electronics companies are making greater commitments to ensuring the security of their products. In specific, many companies have assigned individuals or groups of employees to address security concerns.

Although many security initiatives are focused on the integrated circuit (IC) to prevent attacks aimed at reverse engineering or copying the device and to protect the data stored in the chip's embedded memory, circuit board developers and manufacturers have similar concerns at the board level. In reality, some would say the board is more susceptible than a chip since a board has test points, connectors, sockets and other entryways that can provide easy access via a common probe device. In addition, some boards now include remote network access via a test or debug port that normally provides remote configuration and service. Cases have been reported where large expensive circuit boards have been returned to the manufacturer for service or repair, only for the manufacturer to discover that some of the returned boards were counterfeits. Some refer to this is a 'tip of the iceberg' problem. The manufacturer can calculate the a portion of the cost of lost sales based on the returned counterfeits, but this is likely only a small fraction of all counterfeit boards in circulation. The manufacturer only sees 'the tip of the iceberg' of the problem.

Because of this and other factors, many companies are incorporating design-for-security (DFS) groups into their research and development deprtments to work hand-in-hand with design-for-test (DFT) and design-for-debug (DFD) personnel. One of the biggest concerns of security personnel is the fact that the IEEE 1149.1 JTAG Test Access Port (TAP) is not inherently secure, and yet the TAP port has extensive access to items that are considered sensitive or absolutely critical. At the board level, the main goals of security would be to prevent the disruption of normal operations and to protect firmware from being counterfeited or stolen.

In a sense, test and debug is exactly the opposite of security (Figure 1). The goals of test and debug are all about observability, controllability, traceability and accessibility; whereas security goals involve limiting these items to protect embedded logic, embedded instruments, data, memory, configuration settings and codes. When concerns about both test/debug and security are present, the access needed for test/debug could represent a breach in security or backdoor access for hackers.

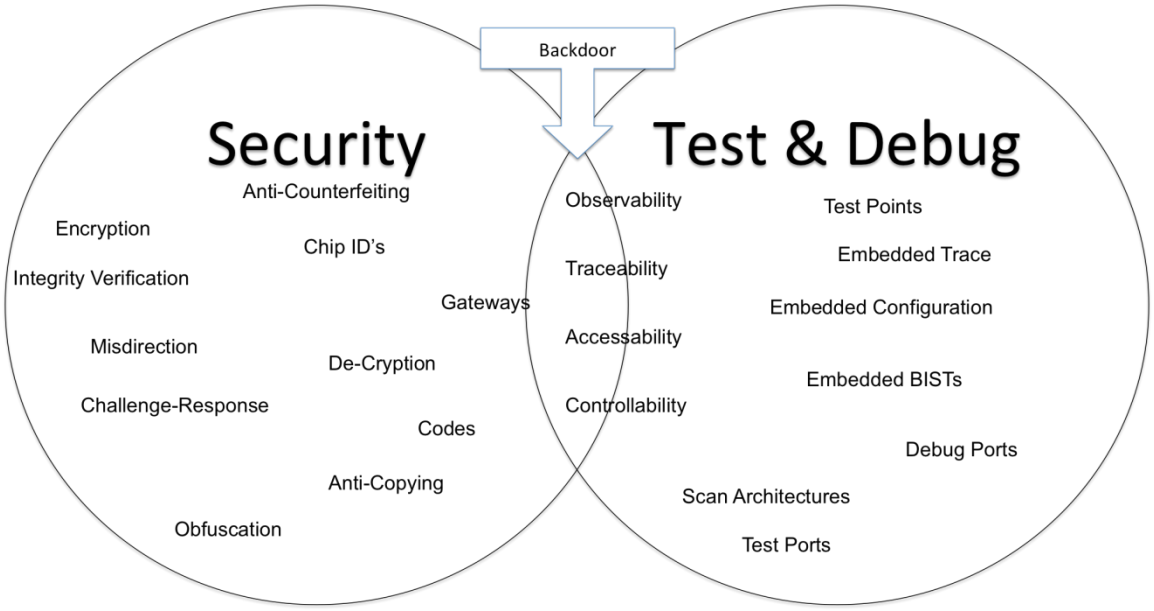


Figure 1: The security and test overlap showing the backdoor access test could provide

Goals of Board Test Security

At the chip level, the contents of a chip may be hidden, but it is accessible through its device pins via electrical snooping equipment like logic analyzers, oscilloscopes, the chip's scan test architecture and other internal probes. Serious and malicious hackers may bring to bear side-channel attacks such as operation-vs-temperature mapping, power analysis, emission analysis, chip freezing with liquid nitrogen and other direct on-the-bench types of spying. Extreme spying requires chip deconstruction, delayering and layer-by-layer photography.

The main security goals at the chip-level are aimed at protecting the device's internal logic from investigation and unauthorized operation. Items such as encryption codes, chip configuration settings, chip identification codes, and test/debug functions must be secure because they could

provide information on the chip's internal architecture or their illicit operation could disrupt the functional operations of the chip.

At the board level, however, security goals are similar to chip security, but a bit different. Generally, the main board-level security goals are:

1. prevent disruption of the board's functionality during normal operation
2. prevent copying and reverse engineering of the board (counterfeit prevention)
3. protect access to board operations, settings, firmware and codes

The two main environments where the security of a board could be breached would be when the board is in an actual operational setting and when it is in a hacker's lab and subject to on-bench investigations. On-bench hacking could include electrical investigations via probes, homemade or illegal software tools, legal licensed tools and various tampering methods, such as removing chips from the board, drilling into the board to create test points or physically probing accessible contacts.

These high-level concern resolve at the board level to the following security concerns:

1. protect the board's memory from unauthorized access
2. protect the board's firmware from unauthorized access
3. prevent operation of on-board chip features that would disrupt board-level functional operations
4. prevent access to secure chip data
5. provide anti-counterfeit and counterfeit detection functionality

The questions that board security measures must address are:

1. Can the board be made secure by simply implementing and including chip-level security measures during board development, integration and operation? Or, are specific board-level security features needed?
2. Can board security be supported without negatively impacting the board's test time and test cost?

3. Can security be quantified through some sort of 'goodness' metric that indicates how strong or weak the security measures are?
4. What will be the cost of security in terms of the silicon area in the chips on a board?

Investigating the Attacker and the Attack Setup

Understanding the Attacker

Before examining security issues, consider how an attacker or hacker would use the IEEE 1149.1 JTAG port to investigate a chip or hijack instruments embedded in a chip to carry out an attack such as a denial-of-service (DoS) attack. To begin with, the JTAG port is a set of pins on a chip's package. The functionality of these pins conforms to the IEEE 1149.1 Boundary-Scan Standard (JTAG) and are generally identified in the chip's documentation. So, a normal user, such as an engineer performing validation, test or debug functions, would identify these pins and their public documented features, and then connect them to a JTAG tester, which typically consists of hardware and software to operate the JTAG test resources in order to exercise some specific designed-in public function. After processing the chip's Boundary Scan Description Language (BSDL) file, the JTAG tester can conduct interconnect tests, verify the chip's identification code or operate other test, validation and debug functions.

However, a nefarious individual may employ this same sort of legal JTAG tester to criminal ends, such as investigating the chip beyond its public operations or using knowledge of the chip's JTAG-accessible operations to place the circuit board on which the device has been installed into a mode that hinders the board's functional operation.

Investigating the JTAG Test Architecture

In a JTAG architecture that complies with the IEEE 1149.1 Boundary-Scan Standard, each embedded instrument, test function or adjustable configuration is represented by an instruction encoding in a chip's JTAG Instruction Register (IR). Figure 2 is an example of an IEEE 1149.1 JTAG architecture where IR instruction encodings represent embedded test instruments. Even though these instruments may be documented as private, they can still be explored by scanning in undocumented encodings.

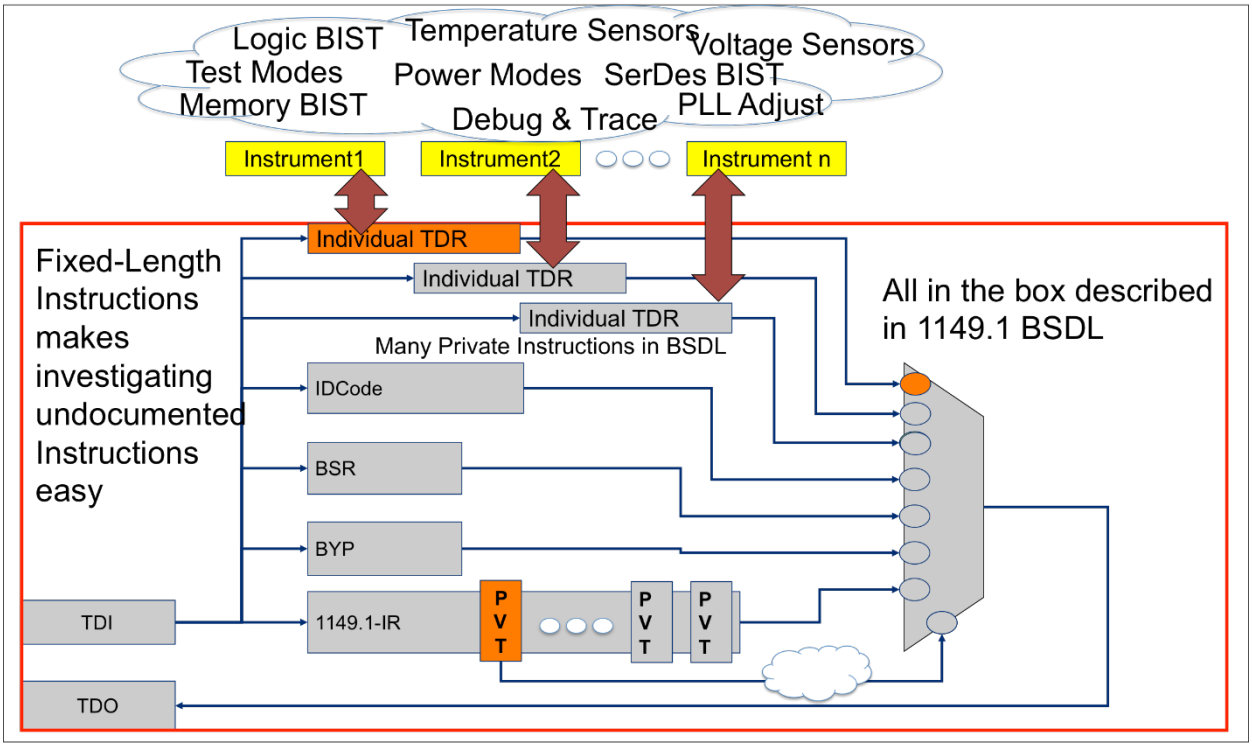


Figure 2: Example of an IEEE 1149.1 JTAG architecture where IR instruction encodings represent embedded test instruments

The length of the JTAG IR is publicly documented in the chip's BSDL file, which is a documentation or descriptive file used by JTAG tools to operate the JTAG functions integrated into a JTAG-compliant chip. To investigate the internal resources of a chip, most hackers start by performing a JTAG IR-Scan via the board's active JTAG scan path in order to scan undocumented encodings into the chip's IR. Next, they would apply a JTAG UpdateIR state and observe which register is connected to the device's JTAG TDI-TDO internal scan chain. The scan registers may then be investigated by performing a DR-Scan of the active register connected to the TDI-TDO to scan in random or directed data values followed by an UpdateDR to determine what occurs. In many cases, embedded instruments such as memory built-in-self-test (BIST) instruments, logic BIST, phase lock loop (PLL) BIST, serdes BIST and others will begin operating. The hacker is looking for insight into the chip's internal registers. For example, the hacker might hijack debug registers to capture normally hidden information on the chip's internal states. Or, he might be able to control features that could disrupt the normal operations of the circuit board. Note that some of the hacker's investigation may occur while the board is in a normal functional mode since JTAG operations are possible during functional operations.

The practice of scanning-in undocumented IR encodings has become so prevalent that many chip suppliers provide a warning along with their devices, such as: "Operation of undocumented instructions could lead to the destruction of this device."

The Black Box Problem

Investigating the internal resources and operations of a chip through a port like the JTAG port is known as the 'Black Box Problem.' For serial interfaces, data is scanned-in, causing data to be scanned-out. For the JTAG port, this relationship is 1-to-1. For every bit pushed in, a bit is pushed out. The investigator or hacker is looking at the data exiting the serial output to gain insight into what has happened inside the chip. He analyzes this data in conjunction with other visible or measureable indications such as power, temperature, activity on other pins, etc. He must determine which activities or data represent feedback that is helpful for his purposes. The hacker then repeats sequences or operations that provide additional useful feedback until he has achieved his goals.

The Time Value of Hacked Information

Note that the hacker may continue to investigate a chip for an extended period of time, but the knowledge he extracts has a limited useful life. For example, if a chip reaches its natural end-of-life when it is no longer being deployed in new products, it is no longer available in inventory and it is in a relatively few products in the marketplace, then the hacker would gain little or no value other than curiosity to continue his investigations of a chip. This fact provides the basis for a 'goodness' metric regarding security methods.

It is important to note at this point that no security method is foolproof. Theoretically, all security methods can be broken. The measure of the goodness of a security method will depend on how long the hacker is delayed from breaching the security method or how long the hacker will need to break through the device's security measures. If the investigation leading to the breaking of a device's security exceeds the time that the information would be valuable, then the security is more than adequate. For example, a chip for mobile phone applications may be in production for eight to 12 months. Mobile phones that have deployed this chip may be in use in the marketplace

for three years. So, a security methodology that would require five years or more to break would be considered adequate.

Board Test and Debug Security Considerations

Disabling the JTAG Test Port

The most common security technique with regards to the JTAG port and the easiest one for integrated circuit (IC) providers to adopt is to permanently disable the JTAG TAP following chip test. This is usually accomplished (Figure 3) by electrically fusing off the TMS signal on the JTAG TAP with an internal pull-up. This keeps the JTAG state machine in Test-Logic-Reset (TLR) permanently. The IC provider generally implements this solution after final IC test. Unfortunately, this action makes further access to the chip's JTAG functionality impossible. The port is no longer available for chip debug, board test, board debug, software development and debug, and other chip life-cycle tasks. When development and manufacturing engineers are questioned on this topic in industry surveys, the typical response is that disabling the JTAG port for security purposes is a bad idea.

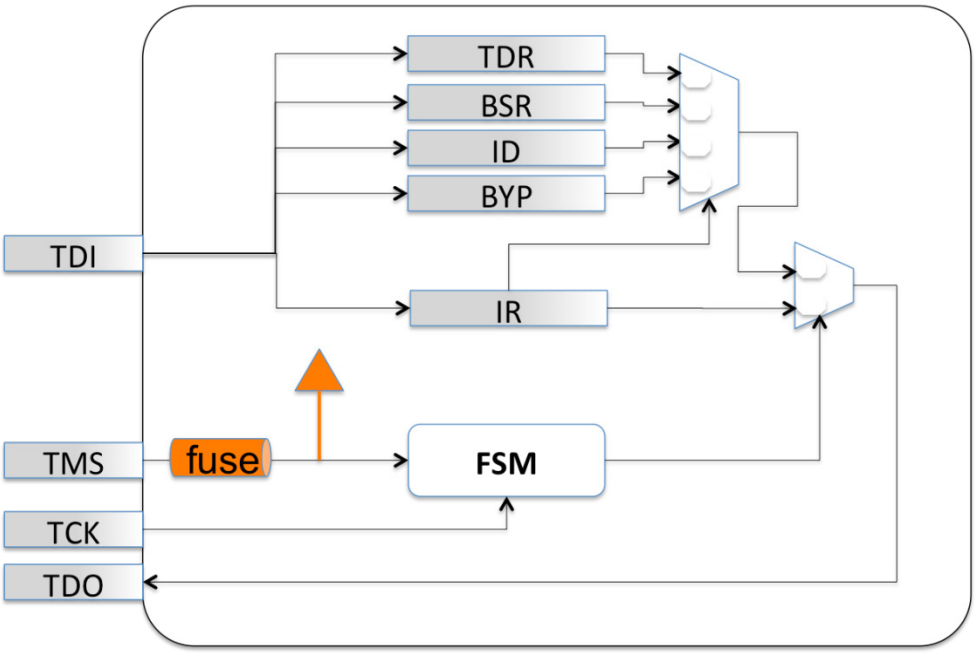


Figure 3: Disabling the 1149.1 JTAG TAP controller with a fuse

Before examining alternative security techniques that retain the JTAG port's usefulness, engineers should consider the features on the circuit board that should be secured.

Protecting On-Board Memory

One key concern that many board developers share is how to prevent snooping into the contents of on-board memory. The main environments when this could happen would be when the board is being manufactured and tested, and later when it is operational. Whether the data in certain memories is encrypted or not will not deter a hacker who wants to obtain the data stored in the memories as it was originally stored. An extreme example of how data is sometimes extracted from on-board memories involves the freezing the memory devices with liquid nitrogen. The devices are then removed from the board and their contents are read out by a memory reader. Anti-tampering methods are needed to prevent this.

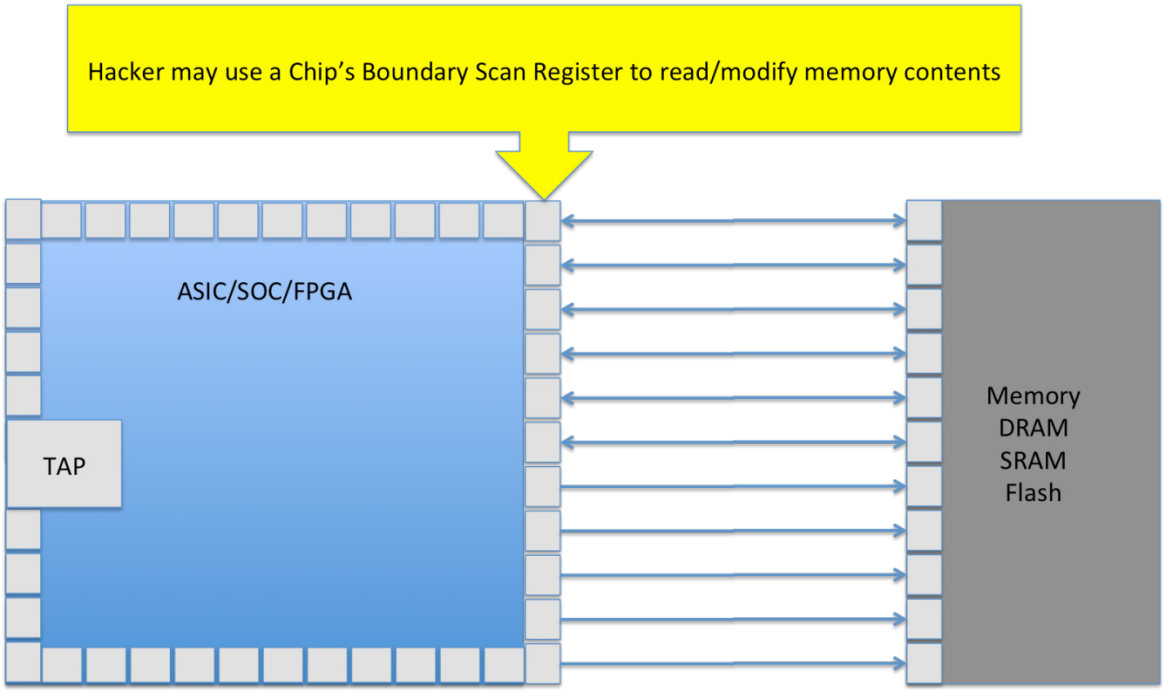


Figure 4: Boundary-scan access from an IC to external memory on the same board.

One of the more common investigation methods for hackers would be to operate the memory on the board and observe the 'reads' that result. This can be done fairly easily with a common JTAG interconnect test tool (Figure 4) by placing the boundary scan register that interfaces to memory into an EXTEST state. The hacker would then perform DR-Scans by operating the JTAG finite

state machine (FSM) from Run-Test-Idle through the Data Register side of the FSM and returning to Run-Test-Idle. This is commonly referred to as 'bit-banging' the memory. The boundary scan register can issue memory addresses and operational commands such as a 'read' or a 'capture' to observe the values produced on the data bus. Furthermore, the hacker may also issue 'write' commands, which modify the contents of memory.

Clearly, one strategy for protecting memory from illicit snooping would be to disable or limit access to the boundary scan register for all purposes except interconnect tests or other legal and authorized actions. Of course, other anti-tampering strategies such as providing no test points that might allow monitoring of the memory interface should also be supported.

Protecting On-Board Firmware

Another key concern of board developers is preventing hackers from reading or modifying on-board firmware. Firmware is defined as non-hardened operation and configuration data and directives stored in on-board flash memory or boot ROM(s) for any on-board processors. Firmware also refers to the configuration and operational data for programmable devices such as programmable logic devices (PLD) and field programmable gate arrays (FPGA). (Figure 5)

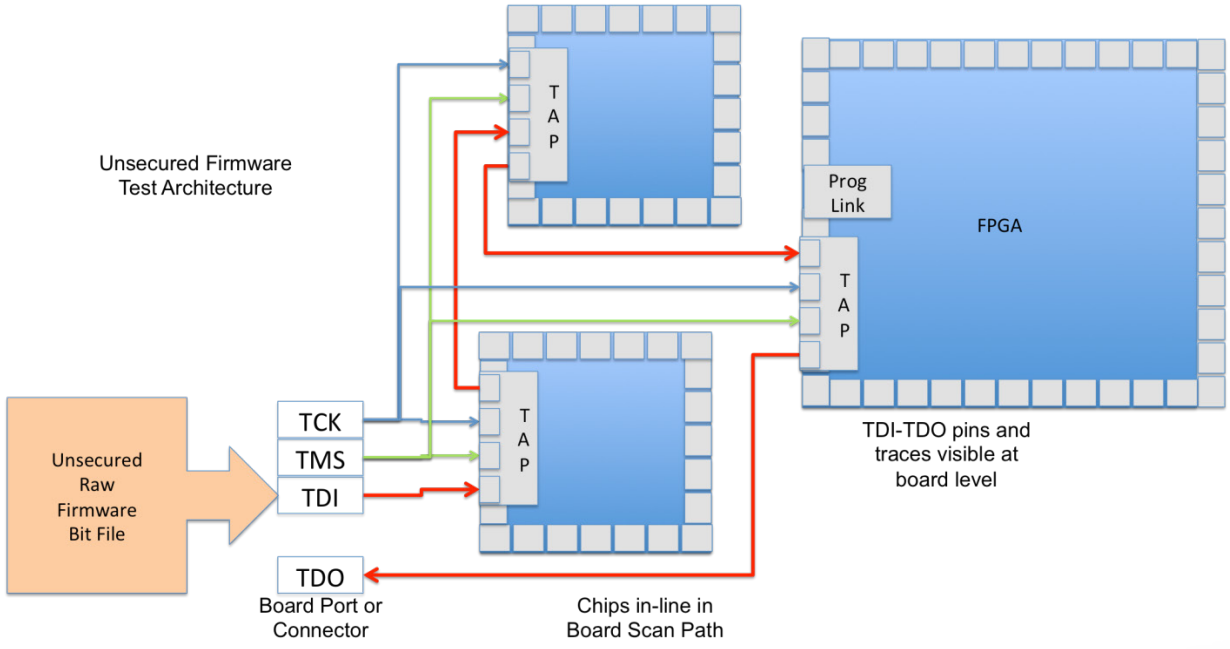


Figure 5: Access to an FPGA's JTAG port through the board's JTAG port and scan path

Some developers believe that firmware is the most important aspect of the board to protect to prevent counterfeiting. For example, the bill of materials (BOM) and the schematic are typically available to contract manufacturers (CM). If these fall into the wrong hands, an exact copy of the board's hardware could be made by a counterfeiter. However, some would say that operational settings, the content of boot ROMs and flash memory, and FPGA firmware make the board unique and create its value. Preventing this information from falling into the wrong hands is a critical strategy for thwarting counterfeiters.

In many cases, the JTAG port is used to initially program on-board flash memory devices and programmable devices such as FPGAs. To accomplish this programming step, the ICs that interface to memories (most memory devices do not have JTAG ports) and the programmable devices themselves (which have JTAG ports) must be connected into the board's JTAG scan path. This makes the devices to be programmed visible to the software-based JTAG tools that will do the programming, as well as testing and debugging. These software-based JTAG tools can also be used by hackers to investigate and reverse engineer the firmware.

Two possible strategies can be adopted to prevent hackers from stealing firmware as it enters the board's JTAG Test Data In (TDI) port to eventually program the targeted devices or verify the program contained in pre-programmed devices. One strategy would be to physically hide the JTAG TAP of a programmable device by not directly including it on the board's JTAG scan path. This is done by inserting a scan path linker with security features between the programmable device and the scan path.

The second strategy would be to include some form of decryption on-board so that the encrypted firmware is decrypted as it is stored in on-board memory. This would convert the encrypted scanned-in bit-stream into executable decrypted values, programming code or instructions. The decryption should take place somewhere along the scan path and at a location such that the board's TDI pin can't be probed and monitored following the decryption. Ideally, this would occur within the memory or FPGA, but this functionality has been proven to be very weak.

Another way to hack firmware would be to conduct 'reads' on storage devices and push the already programmed data out through the scan path to the JTAG port's Test Data Out (TDO) pin. In this case, a similar encryption function is needed to prevent data from being observed by a

probe at a TDO pin on the scan path on the board or at the board connector. Similarly, the encryption should happen on the scan path before the data can be observed on the board.

To facilitate encryption and decryption, non-tampering strategies may be required, including burying the TDI-TDO scan path from one chip to the next in the middle layers of the board. In addition, the TDI and TDO pins for chips should be solder balls that are underneath the silicon die in a chip package so they are not an accessible at an edge of a device where they can be probed. Note that both the JTAG port and any encryption/decryption of the data stream could be concealed and take place within a secure scan path linker device on the board. Such secure scan path linker devices should require an input code to access the JTAG port on a programmable device. Without this access, the TDI-to-TDO serial data stream going into and coming out of a programmable device could not be easily manipulated or viewed by a hacker.

Preventing Functional Disruption

Many system operators fear that malicious individuals or organizations will disrupt board operations. This is often played out in denial-of-service (DoS) attacks that block access and thereby render useless a website, for example. A DoS attack can be launched from outside the system under attack by flooding its communication subsystem with traffic. Often, hackers hijack thousands or millions of computers to access one website simultaneously. Besides websites, cell towers, engine controllers, pacemakers or any number of systems can be brought down through DoS attacks. This is a real concern for military and government systems, as well as medical devices and automotive controllers. A user of one of these systems could be placed in a life threatening situation if the system is hacked.

Out of necessity, these types of applications must secure their functional operations, but in many cases the weaknesses of the JTAG port is either not recognized or a draconian solution is adopted; namely, disabling the JTAG port completely. But disabling the JTAG port has negative effects on yield analysis, debug and diagnosis of field returns, and test coverage during board and system manufacturing. Additionally, the JTAG port has been widely deployed throughout the electronics industry. Ever since the ratification in 1991 of the first version of the IEEE 1149.1 Boundary-Scan Standard, the JTAG port has become more than just a board test function for interconnect testing. It has been coopted by designers to access, configure, and operate

embedded test and debug logic for IC test. Moreover, the JTAG port is also used as an access mechanism by other IEEE standard architectures for test and debug. For example, the IEEE 1500 Embedded Core Test standard, which was ratified in 2005, and, more recently, the IEEE 1687 Embedded Instrumentation Standard, which passed balloting in 2014, both rely on JTAG for access to chips and boards. IEEE 1500 facilitates the testing of cores embedded in chips while IEEE 1687, which is also referred to as Internal JTAG or simply IJTAG, standardizes the operation of embedded instruments for test, debug, environmental monitoring, process monitoring and functional configuration.

One common scenario hackers employ to disrupt operations is to connect to a board's JTAG port either directly at the physical connector on the board or remotely through a computer on a network. This would give the hacker access to certain IC test functions. The hacker could then operate embedded instruments like memory BIST or serdes BIST instruments in the chip. These types of test functions might fully consume the operational functionality of the targeted chips, disrupting the operations of the board and system. Figure 6 shows an example of a chip's memory BIST. When this type of memory BIST is activated, the system's functional memory management unit (MMU) is disconnected from the memory interface while the BIST conducts algorithmic read-and-write operations to test the connected memory. Effectively, this removes the memory from the microprocessor's memory map as long as the BIST is activated. Without processor access to memory, operations would likely come to a halt.

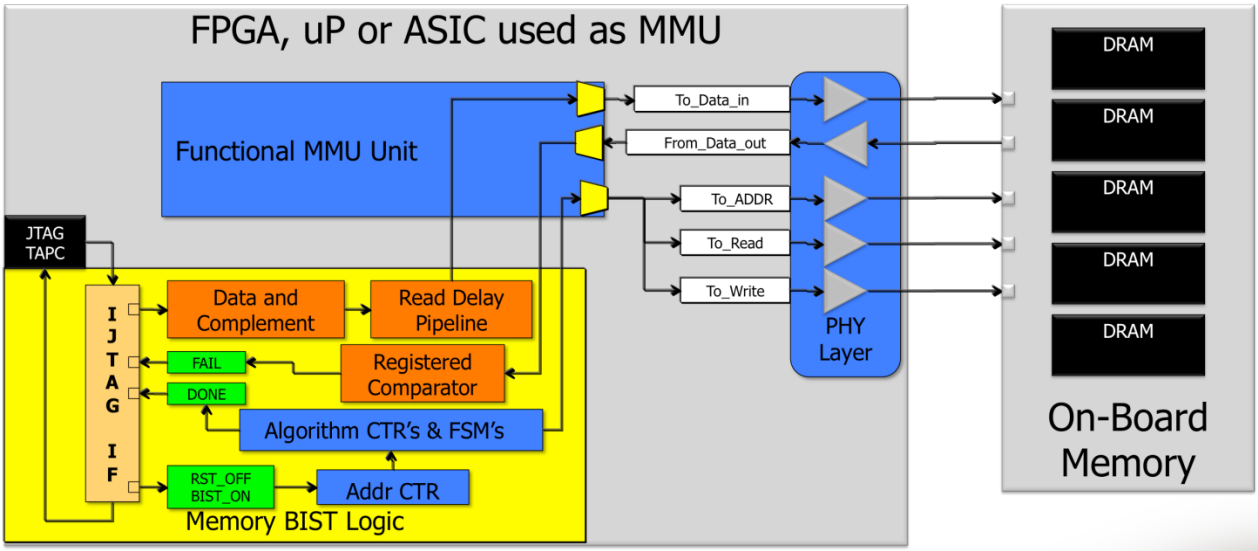


Figure 6: A memory BIST in a system-on-a-chip (SoC) or application-specific IC (ASIC)

Of course, security experts fear that hackers could buy or license a perfectly legal software-based test tool that could operate these test features and then apply such a tool in a malicious way to disrupt a board's operation. Multiple protection strategies may be required to defeat this type of incursion. One strategy could block or limit access – even legal and authorized access – to an embedded test instrument. For example, a memory BIST might only be allowed to run a total of three or four times over the lifetime of the chip. Another method could require a security code to operate the test function such as a memory BIST. This security code could change algorithmically after each use. Operating a disruptive function would then require an authorized access code that might only be available by contacting the IC provider directly.

Preventing Access to Secure Chip Data

Similarly to preventing functional disruption, other features and data stored within a chip are likely to require security protection. One example would be the Digital Versatile Disc (DVD) codes stored in a microprocessor or graphics processing unit (GPU). Another would be the functional security measures, such as challenge-response or encryption-decryption procedures. These security functions often involve security codes and are considered high-value targets by hackers. Consequently, they may merit dedicated security measures that would require much more effort for hackers to defeat.

Although some of this high value data may or may not be accessible through the JTAG port, certain test and debug features embedded in a chip give hackers the ability to dump memory or capture register values. It is conceivable that some hacking strategy could be developed to extract high value information through the JTAG port.

One security strategy for high value data would be to always hide this data and the registers or memories where it is stored behind hardware locks and encryption functions. Some form of time-varying encryption would also be effective since this would require synchronization with a counter or clock. That is, access to this data would require analysis of or access to an algorithm that is not contained within the targeted chip. Like the preventative measures against disrupting a chip's functionality, access codes that require contacting the IC provider would be optimal.

Identifying Counterfeit Boards

One of the best ways to identify that a circuit board is not a counterfeit is to integrate something on the board that cannot be copied easily or to include a unique identifier on each board, such as documented chip identification codes. To functionally identify an authentic board, chips could communicate with each other and pass secure or hidden data or codes to each other to verify that the board consists of a proper configuration. If this sort of authentication procedure did not complete, then the board could be prevented from booting or its operations limited. This procedure would require that the board is able to verify itself through a security code or identifier stored in a protected location somewhere on the board. For example, a description of the chip identifier codes for this particular board could be stored as part of the firmware. It is possible that a counterfeit board could be a complete clone which seems to function as expected.

Consider this scenario. A circuit boards ends up in a certified repair shop. First, it must be validated as an authentic board and not a counterfeit. If the board does not boot or operate functionally, it cannot be validated through functional means. The alternative would be to assess the configuration through the board's test and debug infrastructure. This may mean that the JTAG port and scan path must be able to access the on-board chip identifiers and also access the secure configuration verification data.

If the board's JTAG infrastructure is able to authenticate the board, then a hacker may be able to extract this information and clone the board. So clearly, this configuration information must be protected with strong security measures and access restrictions. Possibly, this information could be encrypted and only read through some form of decryption.

The Board Test/Debug Security Solution Space

Summarizing the Solution Space

This eBook introduces the test security solution space without delving into the details to a great degree. A more comprehensive evaluation of solution architectures and techniques will be dealt with in a future eBook. Some of the more prominent concepts that affect how security solutions are designed into circuit boards are explained in the following sections.

As mentioned in previous sections, securing a circuit board's test and debug infrastructure requires methods designed into the chips on the board and into the board itself. These two layers of security must prevent disruption of the board's functionality during normal operation, prevent copying or reverse engineering of the board or chips on the board (counterfeit prevention), and protect the board's operational settings, firmware and codes from unauthorized snooping. More specifically, these goals are met by protecting the on-board memory and firmware from unauthorized access, by preventing the illicit operation of chip features that could disrupt functional operation, by preventing access to secure chip data and by providing anti-counterfeiting and counterfeit detection functions.

Potential solution spaces were discussed in the sections covering each of these areas. The techniques discussed can be summarized as follows:

1. limit access to the boundary scan register on chips that interface to any on-board memory (SRAM, DRAM, ROM, etc.) to protect the values held in those memories from being read or modified
2. limit access to the JTAG TAP on programmable devices by deploying a secure scan path linker to protect these devices' firmware from being read or modified
3. provide encryption/decryption for the writing or reading of programming data such as FPGA firmware
4. limit access to embedded instruments within chips to prevent unauthorized operation and functional disruption
5. hide the access to and apply obfuscation to IC data so that more hacking effort is required to access data of higher value
6. hide identifiers, codes or data values on a board and in chips to provide a secure method for authenticating the configuration of a board, minimizing or preventing counterfeiting, and providing a method for verifying authenticity

Introducing Locks and Keys

Embedded hardware locks provide a standardized, repeatable and possibly portable security method. One form of a hardware lock, which was presented and published at the International Test Conference (ITC) in 2013 and Design Automation and Test in Europe (DATE) in 2014, is

the Locking Segment-Insertion-Bit (LSIB). The LSIB is a JTAG-like cell based on the IEEE 1687 IJTAG standard's definition of a Segment Insertion Bit (SIB). SIBs support the JTAG Shift-Side and Update-Side, which allows a segment of a scan path to be added or subtracted from the active scan path within a chip. When updated with an 'assert' value, a SIB opens a port that contains a second TDI/TDO and activates a SELECT signal. This SELECT signal is used to unblock the ShiftEn/CaptureEn/UpdateEn that connects a Test Data Register (TDR) to the active scan path. A SIB can be 'locked' by requiring that other scan path bits provide signals from their Update-Side. These signals can be referred to as keys or Key-Bits (which support both logic 1 and 0 values). They gate the UpdateEn signal to the SIB, making it a LSIB. (Figure 7)

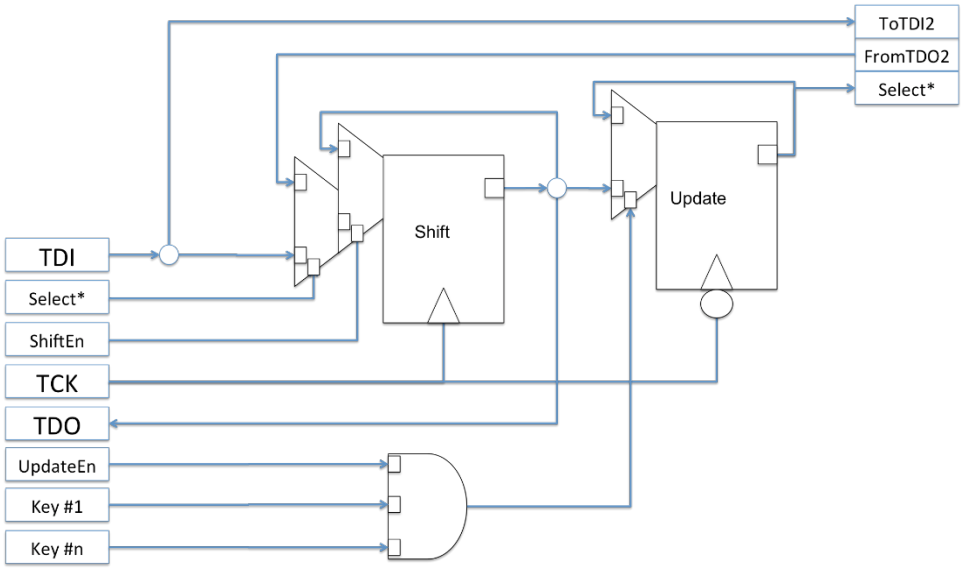


Figure 7: The Locking Segment Insertion Bit (LSIB)

Calculating the Goodness or Strength of a Security Solution

The LSIB and associated keys in different configurations can provide security solutions for all of the board test and debug concerns discussed in this eBook. Different configurations result in different measureable degrees of strength or weakness. These measureables make up the 'goodness metrics' which quantify: 1) how much silicon area is required when adding security? 2) How much time is added to test processes by including security? And 3) how long would it likely take a hacker to analyze and break through the security architecture? Research conducted at Southern Methodist University (SMU) in Dallas, TX, and in conjunction with ASSET

InterTech, has produced mathematical equations to predict the goodness of security methods based on the LSIB.

Hacking or illegally opening just one LSIB within an IEEE 1687 IJTAG on-chip architecture requires a number of clock cycles that is equivalent to:

$$\text{Equation 1: Applied TCK Clock Cycles} = (5+n+d) \times 2^{(K+1)}$$

The '5' refers to the number of Test Clock (TCK) cycles associated with operating the JTAG FSM to conduct a DR-Scan; 'n' is the length of the active scan chain; 'd' represents the unique overshift required for the hacker to identify when a scan vector has fully filled a scan chain in order to determine the length of the scan chain; 'K' represents the number of Key-Bits required to unlock the LSIB; and '+1' represents the fact that the LSIB must also have the correct 'assert' value to be opened. The first portion of this equation, '(5+n+d)', represents the clocking cost in terms of time spent conducting one DR-Scan, such as taking one guess at a set of key values and identifying the location of the LSIB, while '$2^{(K+1)}$' represents the number of possible encodings that must be applied to find the Key+LSIB encoding that will unlock the LSIB. Multiplying these two factors calculates the number of TCK cycles needed to have a high probability of opening the LSIB. Note that this calculation does not factor in luck. That is, the hacker may get lucky and one of the first random values scanned-in could unlock the LSIB.

Table 1: The effects of increasing the number of Key-Bits for a fixed base register length

Base Register Length	1024	1024	1024	1024	1024
Number of Keys	40	42	44	46	48
Operation Frequency of the Test Clock (MHz)	10	10	10	10	10
Analysis Time (Years)	7.21	28.84	115.36	461.45	1845.79
Unlock Time with Legal Key (ms)	0.207	0.207	0.207	0.207	0.207

To turn Equation 1 into a usable goodness rating, it must be evaluated as a function of time. So, the product of the equation must be divided by a frequency and then converted into a time period.

Equation 2: (TCK Clock Cycles) ÷ Frequency in Hz = Investigation time in seconds to unlock a LSIB

For example, Table 1 shows that distributing 40 keys across a scan path 1024 bits long, with a uniqueness vector of 5-bits and an applied TCK frequency of 10MHz would result in a maximum investigation time for the hacker of 7.21 years. Increasing the number of Key-Bits to 42 increases the time to 28.84 years. Furthermore, increasing the keys to 44 increases investigation time to 115.36 years and, finally, supporting 48 Key-Bits would result in an investigation time of 1,845.79 years. These calculations provide a relative indication of the level of goodness or strength of the security. Table 1 shows that adjusting the number of keys will increase the goodness level and does not require a significant amount of physical overhead since this type of security configuration can be operated for manufacturing test purposes in only 207 microseconds (the time it takes one scan to apply the correct key encoding and a second scan to unlock the LSIB).

Changing the length of the base register (Table 2) can also impact the time a hacker will need to explore the circuit and the time to operate a test. The hacker's investigation time is not increased to the extent that it is by increasing the number of keys, but it is impacted. The length of the base register is represented by "n" in (5+n+d) and grows linearly as 'n' increases; whereas, the number of Key-Bits (K) grows exponentially as (2^{K+1}) increases. The difference between changing 'n' versus changing 'K' can be seen by comparing Tables 1 and 2. Table 2 shows that changing the length of the base register from 1024 to 16384 bits with the number of Key-Bits held constant at 40 results in increasing the hacker's analysis time from 7.21 years to 114.32 years. However, a longer base register does impact the normal operation time to unlock a lock with a legal key, adding to test times. Table 2 shows that test time increases from 207 microseconds at 1024 bits long to 3.28 milliseconds at 16,384 bits long.

Table 2: The effects of changing the base register length while keeping Key-Bits fixed

Base Register Length	1024	2048	4096	8192	16384
Number of Keys	40	40	40	40	40
Operation Frequency of the Test Clock (MHz)	10	10	10	10	10
Analysis Time (Years)	7.21	14.35	28.63	57.19	114.32
Unlock Time with Legal Key (ms)	0.207	0.412	0.821	1.640	3.279

Note that another important factor in the analysis is the applied frequency of TCK. Generally, TCK is dictated by the design and is consistent across the entire board. It is determined by the slowest JTAG device on the active scan chain. The TCK on most designs today is limited to less than 50MHz.

A follow-on companion eBook will delve more deeply into security issues, such as lock-and-key architectures and configurations that can be applied to the specific problems identified in this eBook. This companion eBook will provide further analysis concerning goodness metrics and the impacts security measures will have on design area, test times and the strength of security measures relative to the time it will take a hacker to circumvent them.

Summary and Conclusion

Circuit board security is a particularly critical topic because board designers must protect boards from reverse engineering and a variety of malicious attacks such as DoS and others. To safeguard circuit boards, the designer must address the vulnerabilities inherent in on-board test and debug ports such as the IEEE 1149.1 JTAG Test Access Port (TAP). No security methods are specified in the JTAG standard, so board developers must compensate by designing their own. In addition, the security found in the chips that populate circuit boards is inconsistent and, in the end, inconsequential to board security. Some chips will have embedded security protections while others will have none. Plus, security provided by a chip maker is not intended to protect anything beyond the device itself, let alone the circuit board.

Although a board's test ports, such as the JTAG port, may provide an opportunity for hackers and other unauthorized users, security measures that protect a number of board resources can thwart many attacks. The board-level resources that must typically be secured include: on-board memory and firmware; access to on-chip embedded instruments, codes and data; and the chip identifiers associated with a unique board identifier.

Even though securing a board's test ports and test infrastructure cannot be cobbled together by relying on chip security, low-cost and standardized test-related security solutions can be designed into boards through a series of hardware locks and keys contained within chips or on boards within scan-path linkers. Locks and keys can be portable insofar as complete security subsystems may be contained within cores and embedded IP, which in turn can be integrated into chips and, subsequently, deployed on boards. The resulting board-level security system is flexible to the requirements of a particular board design and it would have minimal impact on board debug and manufacturing test times.

Learn More

A tutorial on the IEEE 1687 IJTAG standard which enables many of the security measures mentioned in this book is available here.