WP266 (v1.1) April 22, 2008

Security Solutions Using Spartan-3 Generation FPGAs

By: Maureen Smerdon

In today's world, security is a huge concern for our global society. Whether boarding a plane, closing the front door, or beginning your next generation circuit design, security has become a significant issue. In our homes, we try to build in the right amount of security to protect ourselves against theft. Security is rapidly becoming a necessity in the electronics industry as well. It is important to understand why security issues have escalated to the forefront in the electronics design field. One reason is the alarming amount of counterfeited goods that are the result of theft. These goods threaten the economy and have a significant effect worldwide in the consumer markets according to the Anti-counterfeiting Coalition. This white paper identifies the top design security threats, explores the basic levels of security, and describes how new, low-cost Spartan®-3A, Spartan-3AN, and Spartan-3A DSP FPGAs from Xilinx can help protect your products and profits.

What are the Financial Impacts of Counterfeiting?

Counterfeiting not only causes a substantial and unrecoverable loss in revenue, it also can tarnish a company's reputation, overload customer support with false products in the field, and impact the bottom line with return material authorizations (RMAs) that need to be validated and processed. Companies whose product lines have been stolen face having to locate the false and potentially unreliable products in order to maintain their reputation and image. Future sales are at risk as are the companies' abilities to stay in business.

The estimated dollar exchange associated with counterfeiting throughout the United States during 2003 was $287 billion, which is 63% of the total $456 billion annual worldwide figure. In 2004, the World Customs Organization estimated that counterfeiting accounted for 5% to 7% of global merchandise trade. This threat continues to grow by 12% to 15% a year, resulting in lost revenues due to counterfeit products. All industries are affected, including consumer electronics, semiconductor devices, batteries, automotive parts, currency, pharmaceuticals, and sporting goods. In this white paper, you will learn what's new from Xilinx to help protect you from the top three security threats that designs face today when using low cost FPGAs.

What are the Top Security Breaches?

The top security breaches that designs face today are *reverse engineering*, *overbuilding*, and *cloning*.

Reverse engineering occurs when a thief takes your design with the intent of recreating or rebuilding a competitive product and selling it on the open market. The effects of reverse engineering are that the perpetrator can build the design much faster, and minimize Research and Development costs. This has been the most common threat since the genesis of the electronics industry.

Today, as companies have moved to outsource manufacturing, they are subject to new security breaches, *overbuilding* and *cloning*. Let's take a look at what those are:

Overbuilding is a potential concern in an outsourcing business model. In this situation, what can occur is unauthorized overbuilding of product that is then sold through other channels without the permission of the original equipment manufacturer. The obvious challenge here is that this can have very adverse ramifications once this product hits the market. Usually, the "overbuilt" products are sold at a lower cost with a much faster time-to-market.

Cloning is when a thief creates a duplicate of your design, IP or product under the same or different label. The obvious benefit to the cloner is that they incur no Research and Development costs and have a drastically reduced time-to-market for the cloned product.

How Much Security is Enough?

What's a designer to do? First, it is important to realize that there is no such thing as unbreakable security. Ultimately, there is nothing you can do to completely stop a determined attacker from breaking a system. If someone wants your data or design, they can use brute force to get what they want. This is not the casual hacker, but possibly a well-funded government or a well-funded competitor. So with that in mind, you will not be creating a solution that can never be broken, but rather one that adequately protects you from the threats that you commonly face from cloners, overbuilders, and reverse engineers.

When you think about security, what you need to consider is what is appropriate for your needs. If your product cost is $10, there's a certain amount of security that you can afford for a system in this price range versus a system that might cost $10,000. This is an evaluation that you need to do. Once you have gone through that evaluation you can determine which set of products and which pieces of the security that you might wish to implement based on that evaluation. There are a variety of solutions available from Xilinx that you can explore to solve your problem from simple to much more complex. Solutions that are considered to be the more basic solutions for security implementation within the Spartan-3 Generation are addressed in this white paper.

If you wish to explore more advanced techniques, you can refer to our document on Advanced Security Techniques using Spartan-3A and Spartan-3AN FPGAs. Beyond the Spartan products Xilinx offers an even more advanced solution with our Virtex® FPGA products.

Spartan FPGAs Enable Flexible, Low Cost Security

On-Chip Flash Memory and Hidden Bitstream in Spartan-3AN FPGAs

Spartan-3AN devices provide on-chip Flash memory that can be used to store configuration data. If your design does not connect the Flash to the outside world, then the Flash cannot be read from the I/O pins.

The Spartan-3AN device bitstream is hidden during configuration because the Flash is inside the FPGA. This configuration provides a starting point for security in a design, where it cannot directly be copied from the Flash.

Configuration Security

The simplest way to secure the Spartan-3 device from loading of an unknown configuration is to hardwire the mode pins to allow Flash auto-configuration only and tie off the data pins. In addition, it is extremely difficult for anyone to directly access the pins from a BGA or CS package when all the circuit connections are under the package. If the pins are hardwired, a direct assault on the PCB will be required in order to load a different configuration.

Bitstream Generator Security Levels

During the test and debug phase of a design, designers can decide to leave the Internal Configuration Access Port (ICAP) or the ChipScope™ Pro analyzer cores in the design for possible maintenance or for random check-ups after the design goes into production. Some of the software utilities, such as the ChipScope Pro analyzer, require these macros for reading the state of internal logic. While this is handy for the designer, this can leave a potential security hole. To eliminate this potential hole when you go to production, remove the ChipScope Pro core.

The Bitstream Generator creates the configuration **`.bit`** file based on the contents of a physical implementation file called the NCD file. The **`.bit`** file defines the behavior of the programmed FPGA. The Bitstream Generator includes many options; one of these options is the Security Level settings. The Bitstream Generator has four security level settings; the first one is the default, and the remaining three options provide additional security. As shown in the following table, Readback operations can be disabled completely, or disabled except for limited access options. Table 1 shows the security level settings and their functionality.

Table 1: **Bitstream Generator Security Level Settings**

Security Level	Description
None	Default. Unrestricted access to all configuration and Readback functions.
Level 1	Disable all Readback functions from both the SelectMAP or JTAG ports (external pins). Readback via the ICAP allowed.
Level 2	Disable all Readback operations on all ports.
Level 3	Disable all configuration and Readback functions from all configuration and JTAG ports. The only command (in terms of Readback and configuration) that can be issued and executed in Level3 is REBOOT. This erases the configuration of the device. This has the same function as enabling the PROG_B pin on the device, except it is done from within the device.

For a detailed explanation of all the Bitstream Generator options; refer to the *Spartan-3 Generation Configuration User Guide* (UG332).

Device DNA Security

Xilinx offers Device DNA Security in the Spartan-3A/3AN/3A DSP FPGA platforms to protect your design, IP, embedded code, and more. The Device DNA is a 57-bit ID, unique to every Spartan-3A/3AN/3A DSP FPGA. This ID can be used to tie a design to a specific FPGA. The designer's personalized algorithm, also stored on the FPGA, is an arithmetic equation that defines how to take the unique Device DNA and create a result. The ID is combined using the designer's personalized algorithm, and the result is then stored wherever the designer chooses, such as in the external memory or internal Flash (for Spartan-3AN FPGA devices only). The algorithm is the secret to the security because it is known by the designer only.

Device DNA Operation

Before exploring how security works in each family, it is important to understand what is at the core of the solution. Device DNA, unique to Xilinx FPGAs and specifically the Spartan-3A/3AN/3A DSP FPGAs, is used for design security. This section describes how Device DNA works and how you can protect your future designs using our patented new approach.

What is the Device DNA?

The Device DNA is a unique 57-bit identifier that is entered into the Spartan-3A/3AN/3A DSP FPGA in the manufacturing process at Xilinx. Every FPGA has a unique ID, allowing you to associate your design to one specific FPGA device. This security or licensing process is designed with complete flexibility in mind. You can easily change your security or licensing process from model to model, thus increasing your design security. The read-only Device DNA is accessible through either the external JTAG port or the internal DNA port for easy connection to the Security Algorithm.

If a cloner or overbuilder copies the bitstream and places it into another FPGA, the Device DNA of the new FPGA will be different. After using the algorithm to check the Device DNA, the design will return an unauthorized or a failed result, allowing the user or designer to determine how to respond to the security breach.

Device DNA Security Basics

The Device DNA security process is like an ATM transaction. To withdraw money from an ATM, you insert your ATM card and enter your PIN on the touch pad. If your card and associated PIN match the ID stored at the bank, your transaction is approved and your money is available. If the match fails, your transaction is rejected and you do not get your money. Figure 1 shows the security flow.

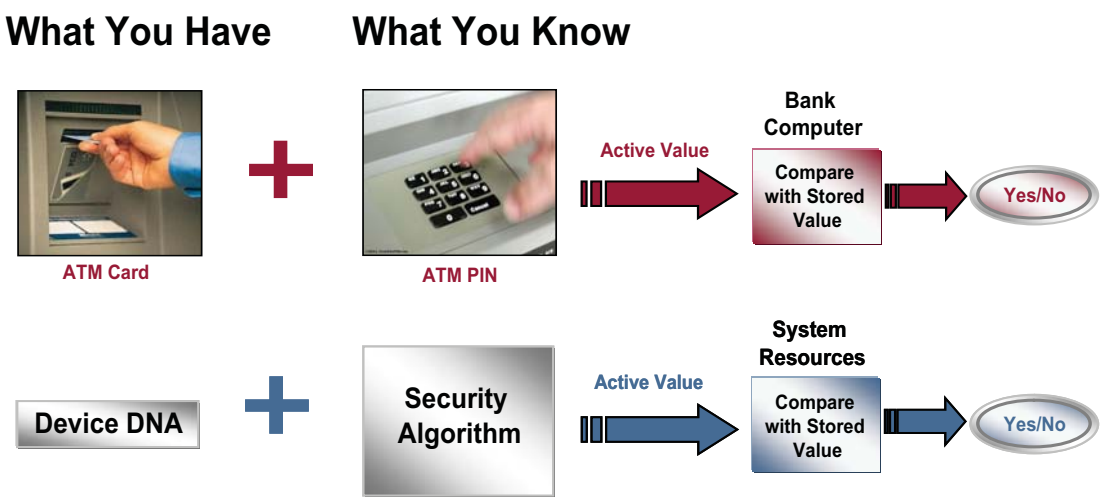

Figure 1: **Security Flow**

Both the Security Algorithm and the Device DNA are contained in the Spartan-3A/3AN/3A DSP device. Using the Device DNA, the Security Algorithm generates the Check Value. Even though Device DNA is 57-bits by default, you can use additional bits for increased security. Furthermore, the 64-byte Factory Flash ID available on the Spartan-3AN can also be used in the algorithm for increased security. The Check Value is then stored anywhere in the system resources (for example, configuration memory, peripheral memory, system memory). In the case of a Spartan-3AN FPGA, the Check Value can also be stored in the one-time programmable 64-byte User Defined Field of the Security Register. This register allows the complete security system to be self-contained with no need for external interfaces or storage.

Unauthorized Operation

During normal operation, the device is powered up, and the bitstream is loaded for configuration of the FPGA. The Security Algorithm reads the Device DNA and generates an Active Value. It then compares the Active Value with the Check Value, stored during the initial setup. If the Check Value is equal to the Active Value, the normal operation can occur. You can design your product to respond in one of the following ways when the two values do not match:

- No functionality

  The design completely stops functioning. This can be easily implemented in a Spartan FPGA by using global control signals like 3-state, Gated Clocks, Flip-flop clock enable and so on.

- Limited functionality

  The design has partial or basic operation. This is where the key functionality is disabled or bypassed. This response allows a third-party test house or contract manufacturer to build and test while preventing overbuilding. It also allows the system to be run in evaluation or demo mode.

- Time bomb

  The design operates with full functionality for a predetermined amount of time before shutting off. This response allows a third-party test house or contract manufacturer to build and test. It also allows the system to operate in demo mode or for IP evaluation.

- Self-destruction (Spartan-3AN devices only)

  Uses Flash sector erase and lockdown protection to erase all sectors and permanently lock Flash memory to all zeros. This response prevents repeated unauthorized access attempts.

Implementing Security in Spartan-3A FPGA Using Device DNA

This is just one *possible* scenario for setting up security in your design. We say *possible* because this is similar to deciding on a security system for your home. If there were only one *possible* lock and key in the world, there would be no security. During the initial one-time setup process, the Device DNA of the Spartan-3A/3AN/3A DSP FPGA can be read via the JTAG port, or from within the fabric of the FPGA. The Check Code can then be generated. This code (Check Value) is then stored somewhere in the system, such as in configuration or system memory. This can be seen in purple in Figure 2, illustrating this possible implementation.

Next we see the Device DNA is blue, and the "secret" Security Algorithm and the key/seed code (if that is what you are using in your design) are green. Finally, there is a comparator and options for an authorized or for an unauthorized result.

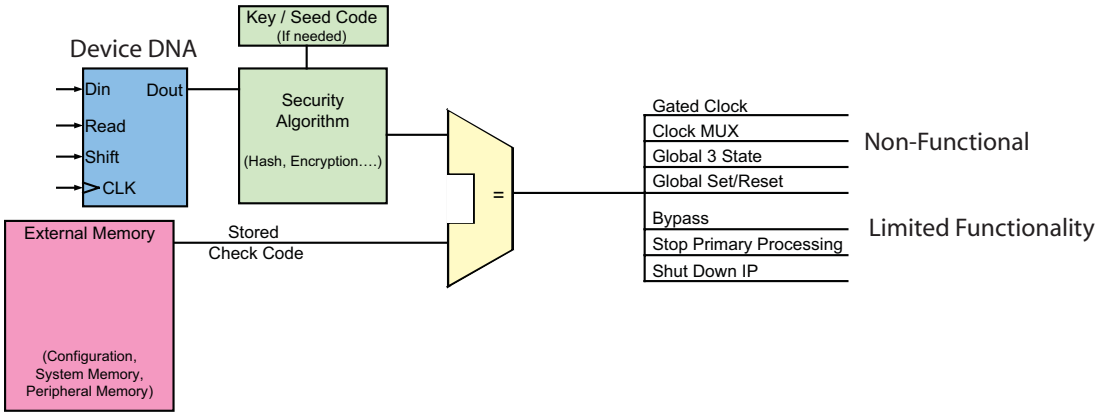

Figure 2: **Device DNA Security Example**

In this scenario, if this is your setup, the following sequence will take place:

1. The device is powered up, and the bitstream is loaded for configuration. Both the Security Algorithm and the Device DNA are contained in the Spartan-3A device
2. The Device DNA is read and sent to the Security Algorithm
3. The Security Algorithm generates the Active Code (result)
4. A compare is done of the result (Active Code) and the Stored Check Code
5. If the Stored Check Code is equal to the Calculated Active Code, then the design is authorized
6. If the two codes do not match then the design is not authorized and the design will respond as set by the designer. Multiple responses can be set for an unauthorized design such as no functionality, limited functionality, and time bomb.

This again is just one simple possibility. More complex security can be accomplished just as easy.

Security with Spartan-3AN Device DNA and Factory Flash ID

In the Spartan-3AN platform, our non-volatile FPGA, the process is almost the same with Spartan-3A devices with a few enhancements. The first security enhancement is that the bitstream is hidden inside the FPGA. This makes it more difficult for someone to monitor.

The second security enhancement that the Spartan-3AN FPGA has is two unique serial numbers, the Device DNA and the Factory Flash ID located in the Flash memory. The two unique IDs give more than 70-bytes of serial numbers resulting in a much larger number of algorithmic possibilities and, therefore, dramatically increases the time needed to breach the Security Algorithm. The design is now specifically tied to both the FPGA and the Flash IDs.

Having two unique IDs is like requiring two different ATM cards to get your money. If you want to get the cash you must have both cards. If one of the cards is lost then your money cannot be taken and is still secure.

The third improvement is in the Stored Check Code. On the Spartan-3AN platform the Security Code can be stored on-chip in a special one-time programmable 64-byte User Defined Field of the Security Register. This allows the complete security system to be self-contained with no need for external interfaces or storage. This feature increases the overall security and makes it more difficult for someone to reverse engineer your product.

The Security Algorithm is user defined to allow the designer to implement the right level of security at the correct system cost. The Security Algorithm is also the primary secret in the security system. Something in the security process must be a secret so that security is not breached. Because the algorithm is unknown, it is the key to the design level security. The algorithm is implemented in the fabric of the FPGA, therefore, it becomes a handful of bits in the millions of configuration bits in the FPGA. Unless you know how the bits fit together, or what the algorithm does, it's just a mass of numbers

to an onlooker or cloner. Figure 3 below outlines one possible flow using the Spartan-3AN devices.

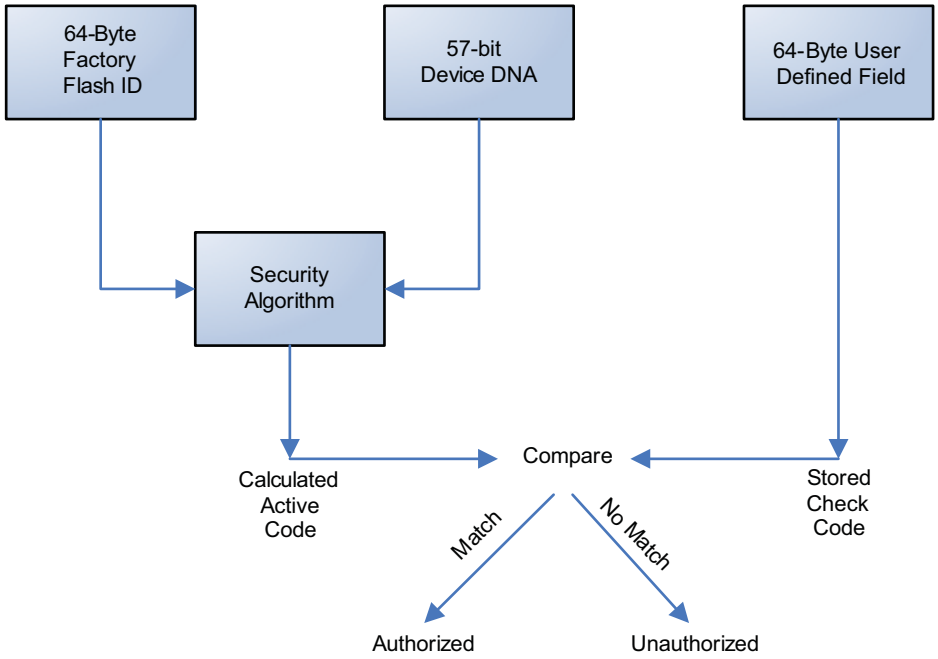

Figure 3: **Spartan-3AN Security**

The Spartan-3AN design level security in the Figure 4 below is a completely self-contained security solution. The Flash contains both the FPGA configuration bitstream and a previously generated Check Code. This code is stored in the one time programmable Flash user field by a trusted/secured manufacturer or registration process.

At power-up, the FPGA configures normally. Once configured, the FPGA application includes circuitry that validates that the design is authorized to operate on the associated Spartan-3AN FPGA. The Device DNA and Factory Flash ID are read by the Security Algorithm which, in turn, generates the Active Code that is compared to the previously generated Check Code stored in the Flash User Defined field. If both codes are equal, the device is authorized. Otherwise, the device is illegitimate and unauthorized.

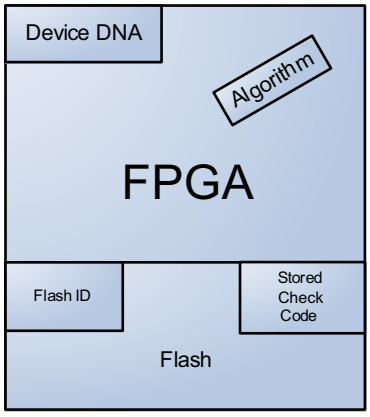

Figure 4: **Spartan-3AN Security**

The handling of failed authorization is another one of the strengths of the Device DNA design level approach. The additional advantage of design level security is that it can be completely integrated into the design. Multiple responses can result from an unauthorized design just as in the Spartan-3A platform.

The design level security in the Spartan-3AN platform provides many ways to protect from overbuilding, cloning and reverse engineering. In addition to this article, you can also learn more about how to secure your low-cost FPGA designs. To learn more about designing in security with our three families, please refer to our Spartan-3 Generation Configuration User Guide at **http://www.xilinx.com/bvdocs/userguides/ug332.pdf** for more on protecting your designs. For advanced techniques, please refer to **White Paper 267**.

Other Security Solutions from Xilinx

Virtex-4 and Virtex-5 FPGAs use AES encryption to accomplish their security. Both families store the key bits in volatile memory that is kept active with a battery, which has very long lifetime. It's interesting that the FIPS 140 standard requires 'key zeroing,' which automatically occurs with this technology whenever power is disconnected. Hence, the parts become unusable with the encrypted bitstream if the power is interrupted.

The Virtex-5 family supports 256-bit AES encryption/decryption technology to achieve a very high degree of design security. With 1.1 X 10^{77} possible key combinations, an externally intercepted bitstream is highly unlikely to be cloned without knowledge of the correct encryption key.

Conclusion

Security attacks through reverse engineering, overbuilding, or cloning result in substantial revenue loss for companies in unrealized sales, returns, and technical support. The costs and losses are permanent and unrecoverable.
Spartan-3A/3AN/3A DSP platforms can help protect your designs from these threats. Device DNA security allows you to associate your design to one specific device, making security threats less likely. The additional design-level security features in the Spartan-3AN platform, including on-chip Flash and hidden bitstream configuration, provide additional protection from any security threats.

References

Xilinx Documentation

The following documents from Xilinx provide supplemental information on low-cost FPGAs and/or security:

Spartan Security Application Documentation

1. UG332, *Spartan-3 Generation Configuration User Guide* (for more on the concepts of how to protect your designs)
2. WP267, *Advanced Security Techniques* (for advanced security techniques using Spartan-3A and Spartan-3AN FPGAs)
3. DS529, *Spartan-3A FPGA Family Data Sheet*

4. DS557, *Spartan-3AN FPGA Family Data Sheet*
5. DS610, *Spartan-3A DSP FPGA Family Data Sheet*
6. WP261, *IP Security in FPGAs*
7. XAPP780, *FPGA IFF Copy Protection Using Dallas Semiconductor/Maxim DS2432 Secure EEPROMs*

Virtex Family Security White Papers

1. **WP155**, *Triple DES Encryption in Selected Virtex-II Devices*
2. **WP261**, *IP Security in FPGAs*

CoolRunner-II Security Application Notes and White Paper

1. **XAPP371**, *CoolRunner-II Galois Field GF (2^M) Multiplier*
2. **XAPP374**, *CryptoBlaze, 8-Bit Security Microcontroller*
3. **WP170**, *CoolRunner-II CPLDs in Secure Applications*

Starter Kits/Design Boards

http://www.xilinx.com/products/devboards/index.htm

Related References

The following documents and links provide additional material useful to security issues:

1. Anderson, Ross. *Security Engineering: A Guide to Building Dependable Distributed Systems*:

   http://www.cl.cam.ac.uk/~rja14/book.html
2. Anderson, Ross, Mike Bond, Jolyon Clulow, and Sergei Skorobogatov. *Cryptographic Processors—A Survey*:

   http://www.cl.cam.ac.uk/~mkb23/research/Survey.pdf
3. Schneier, Bruce. *Applied Cryptography*, John Wiley Sons, 1996
4. Dolan, D. G. Abraham, G. Double, and J. Stevens. *Transaction Security System*, IBM Systems Journal v. 30, no. 2 (1991), pp. 206-229
5. Health Information Privacy:

   http://www.hhs.gov/ocr/hipaa/
6. Other government agencies:

   http://www.nist.gov/

   http://csrc.nist.gov/CryptoToolkit/aes/ (AES)

   http://www.itl.nist.gov/fipspubs/fip180-1.htm (SHA)

   http://csrc.nist.gov/cryptval/ (FIPS 140-1, FIPS 140-2)
7. Certicom

   http://www.certicom.com
8. RSA

   http://www.rsa.com/

9. Menezes, A.J., P.C. van Oorschot, and S.A. Vanstone, *Handbook of Applied Cryptography*, 1996, CRC Press. Also on the Internet at: http://www.cacr.math.uwaterloo.ca/hac/
10. Kerchoffs, Auguste. *La cryptographie militaire*, Journal des sciences militaires, vol. IX, pp. 5-83, Jan. 1883, pp. 161-191, Feb. 1883.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
07/24/07	1.0	Initial Xilinx release.
04/22/08	1.1	Remove references and links to obsolete CoolRunner-II White Paper.