WP468 (v1.0) October 20, 2015

# Leveraging Asymmetric Authentication to Enhance Security-Critical Applications Using Zynq-7000 All Programmable SoCs

*By: Ed Peterson*

*In demanding security environments, the Zynq®-7000 SoC Programmable Logic can support user-defined functions that leverage its built-in secure boot and run-time processes.*

**ABSTRACT**

The need for security in electronic silicon devices becomes more important every day due to their increasingly popular deployment in critical embedded applications including industrial control, automotive, and military systems. These devices are often subject to adversarial attacks such as reverse engineering, IP theft, and unauthorized modifications. As the tools of the adversary improve, electronic devices must be continually updated with enhanced security countermeasures at the board or system level. These additions can consume valuable real estate, add cost to the system, and expose additional signals to a potential adversary.

The security of a Zynq-7000 AP SoC based design can be enhanced by including security features (e.g., side channel attack countermeasures) in the programmable sections of the Zynq device. One key enabler, the RSA-2048 asymmetric authentication capability, is built into all Zynq devices.

By leveraging asymmetric authentication, user-defined soft features can be "trusted" —in this context, referring, to the capability of restricting a Zynq device to accept only authorized code images and bitstreams. The functions contained within the authorized image/bitstream are then trusted to be free from any type of malicious insertions, such as hardware or software Trojans.

This white paper describes the trade-offs associated with these methods, as well as the advantages that a soft, user-defined security function has over built-in security functions.

# Introduction

To combat the increasing capabilities of adversaries, improvements in security evolve with each new Xilinx® family of FPGAs and SoCs. Ideally, countermeasures against an adversarial attack are designed-in at the silicon level to provide the most robust security at the lowest cost. Otherwise, more costly countermeasures must be implemented at the board and/or system level. Silicon integration of security also helps to minimize parts count, real estate, and power.

Although security features built into the silicon are designed to mitigate against today's known attacks, they might not be sufficient for tomorrow's unknown attacks. Security standards and capabilities of adversaries change over time, so security functions built into the silicon can become outdated.

As an example of an electronic device attack, there is a class of non-intrusive attacks that come under the heading of a side-channel attack (SCA). An SCA does not attempt to attack the security function directly but exploits a "side-channel" where information can be extracted from the function via a pathway that was never intended to exist by the initial designer. Two important types of SCAs are known as simple power analysis (SPA) and differential power analysis (DPA). [Ref 1] Basically, by monitoring and recording power supply fluctuations while the electronic device is in normal operation, it is possible to:

- Identify when a particular operation is occurring by visually observing the power traces using SPA. Depending on the level of side-channel leakage, secret or hidden information can be extracted directly with SPA.
- Extract secret or hidden information from the device by performing signal processing and statistical analysis of the identified operation over a large number of power data samples using DPA. (However, as the adversary's techniques improve, the number of power samples required to perform DPA are reduced.)

An adversary that is able to gain close physical access can employ SPA and DPA attacks on an electronic device. For example, defeating a cryptographic function such as the Data Encryption Standard (DES) and/or the Advanced Encryption Standard (AES) can be accomplished by extracting the secret key from the electronic device. In DPA-Resistant Secure Boot, an example shows how to implement a soft security enhancement that can defend against a DPA attack on a Zynq-7000 device.

The Zynq-7000 family integrates a feature-rich dual-core ARM® Cortex™-A9 based processing system (PS) and 28 nm Xilinx programmable logic (PL) in a single device. The ARM Cortex-A9 CPUs are the heart of the PS and also include on-chip memory, external memory interfaces, and a rich set of peripheral connectivity interfaces.

In the context of Xilinx All Programmable devices, the obvious question is: How can security functions be implemented programmatically in a soft manner? Before describing how this can be accomplished, here are some advantages of soft security features:

- As new vulnerabilities and attacks are reported, the security features can be enhanced/improved accordingly.
- As security standards change, the security features can be updated accordingly.

- The customer has control over the performance characteristics of the security features. For instance, security functions can be "parallelized" (at the expense of resources) to increase performance.
- Since the customer instantiates the security feature, they can ensure it is trusted and error free.
- Soft security features allow customers to create their own customized or non-standard security approaches as they see fit.
- Partial Reconfiguration (PR) can dynamically bring in security features as needed and remove them when done. This means that fewer overall resources are required if some security functions are needed only at certain times and not simultaneously.

One extremely important attribute of any security function, whether dedicated or programmed, is that it must be trusted (i.e., free from any sort of modifications, such as a Trojan). For example, a true random number generator (TRNG) function that is used to create cryptographic keys (for internal use only) is useless if an adversary is able to modify the design to also send the keys out of a device pin. In the case of Zynq-7000 devices, a method to ensure this level of trust is enabled by its capability to authenticate the first stage boot loader (FSBL) via asymmetric authentication using RSA-2048, a proven cryptographically strong method. (The number "2048" refers to the key length of the public and private keys.) [Ref 2]

By taking advantage of this asymmetric authentication security feature (also known as a digital signature), it can be assured that only authorized and unmodified data can be loaded into the Zynq-7000 device—which includes any programmable security functions.

In Methods of Enhancing Security, enhancing the secure boot process itself is also described. It can be argued that asymmetric authentication is the most important security feature of Xilinx's All Programmable devices. For additional technical details, refer to the *Zynq-7000 All Programmable SoC Technical Reference Manual*. [Ref 3]

# Background Information

## Asymmetric Authentication

Asymmetric authentication (or digital signature) using RSA is a cryptographic function where the encryption key is public and differs from the decryption key, which is kept private (secret). In RSA, the strength of its security is based on the practical difficulty of factoring the product of two large prime numbers. The RSA private key is used to compute a digital signature and the RSA public key is used to verify a digital signature.

For Zynq-7000 devices, the digital signature computation is performed by the bootgen software [Ref 4] using a private (secret) key, with the digital signature verification being performed within the Zynq-7000 device itself using a public (not secret) key. The RSA-2048 signature verification functionality resides in the PS BootROM, a mask-programmed, immutable memory. The private key is not stored on the Zynq-7000 device, nor is the public key. A 256-bit hash of the public key is permanently programmed into the eFUSE array (nonvolatile memory storage) on the device by the user (using SHA-256 to reduce the number of eFUSE bits required). [Ref 5] The Xilinx customer is responsible for generating the public/private RSA key pairs as well as enabling the RSA feature and programming the public key hash into the Zynq-7000 device. [Ref 6]

When the RSA feature is enabled (via eFUSE) and the hash of the public key is programmed into the eFUSE array, the entire FSBL is RSA authenticated prior to execution by the PS. Since the FSBL is the first piece of user code that gets loaded from an external source into the Zynq-7000 device, it is extremely important at this point to ensure that it is authorized to be executed and has not been modified in any way in order to initiate a "root of trust." The asymmetric authentication in Zynq-7000 AP SoCs does this.

Figure 1 summarizes the asymmetric authentication process:

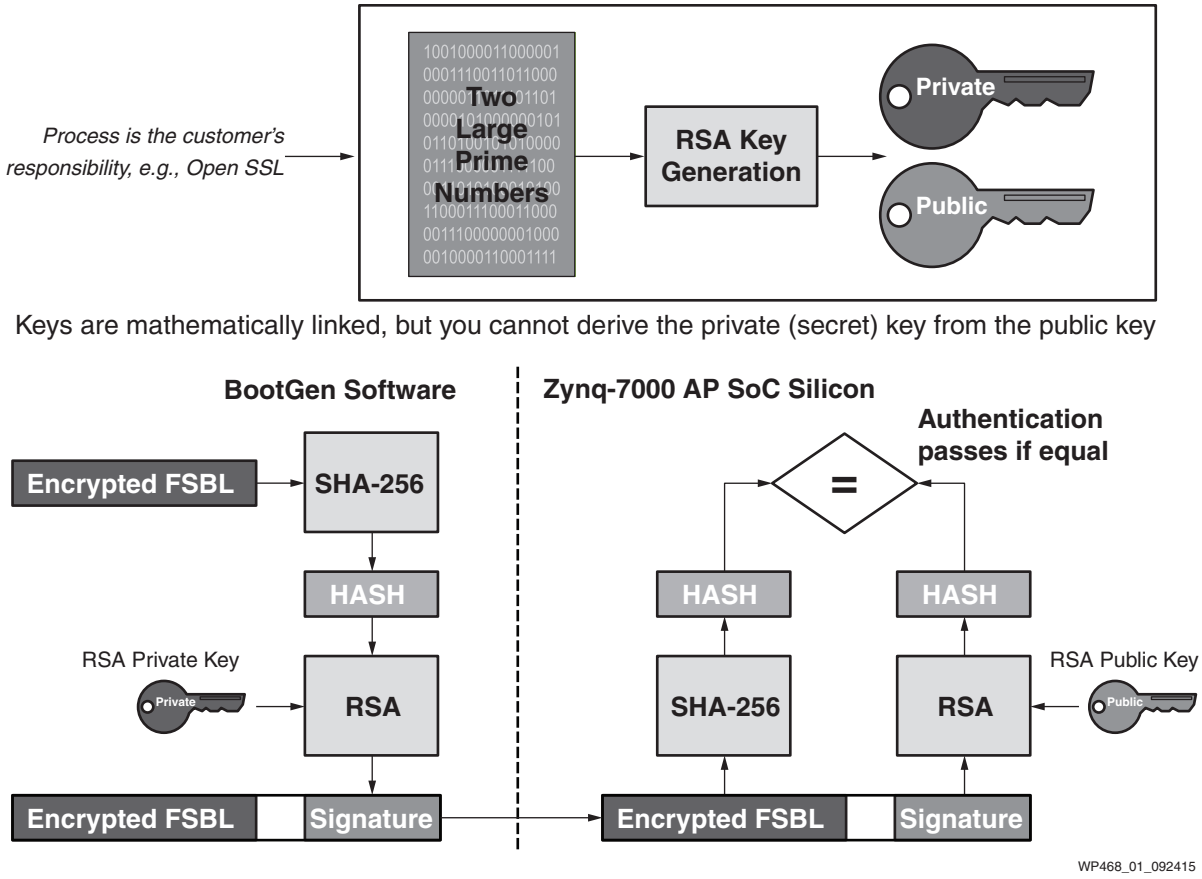


*Figure 1:* **Asymmetric Authentication Process**

The FSBL is responsible for loading the user's PS image and PL configuration. To do that securely, the FSBL must ensure those subsequent loads are also authenticated and decrypted. Since the BootROM RSA function authenticates only the FSBL, an authentication algorithm needs to be included as part of the FSBL code itself so that subsequent loads (via the FSBL) are also authenticated. If a customer wants to use the same RSA function that is in the BootROM, Xilinx provides this code in a library. [Ref 4][Ref 6] To protect the privacy of the PS image and PL configuration, the FSBL is sent through the AES-GCM decryptor in the PL after authentication. That decryptor uses the key stored in battery-backed RAM (BBRAM) or nonvolatile eFUSE. [Ref 7]

## Partial Reconfiguration

The capability of performing PR is also an important element that helps to enable soft security features in Zynq devices. Basically, FPGA and SoC technologies provide the flexibility of onsite programming and reprogramming without going through refabrication with a modified design. PR takes this flexibility one step further, allowing the modification of an operating PL design in a Zynq-7000 device by loading a partial configuration file, usually a partial BIT file. After a full BIT file configures the PL, partial BIT files can be downloaded to modify reconfigurable regions in the PL without compromising the integrity of the applications already running on those parts of the device that are not being reconfigured.

Figure 2 illustrates the basic premise of PR:

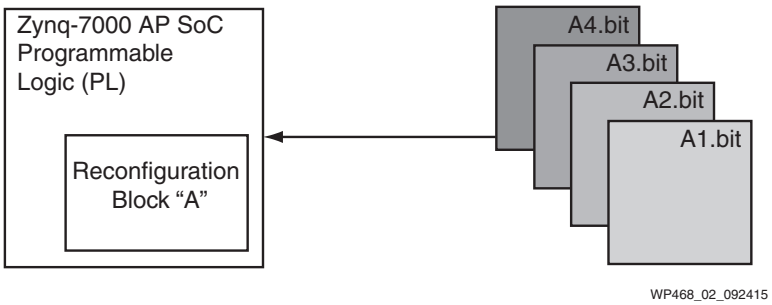


*Figure 2:* **Partial Reconfiguration (PR) Basic Concept**

In the context of soft security features, the PR capability of Zynq-7000 devices is used in two ways:

- As part of the secure boot process itself, to enhance the security of the process
- Post secure boot, to dynamically enable soft security functions as needed by the application at run-time, which limits the exposure (i.e., the "attack window") of the security function

In the next section, enhancing security using PR is explored in greater detail. For additional details on PR, refer to the *Vivado® Design Suite User Guide: Partial Reconfiguration*. [Ref 8]

# Methods of Enhancing Security

In this section, details are provided on how Zynq-7000 device security can be enhanced during and after secure boot (i.e., during run-time). It must be emphasized that the following techniques *all rest upon having a secure "root of trust" enabled by RSA asymmetric authentication*.

## DPA-Resistant Secure Boot

In addition to having cryptographically strong authentication, it is also important that a secure device provides confidentiality (i.e., encryption and decryption) of the user software image(s) and bitstream(s). [Ref 9] This is to prevent adversarial attacks such as IP theft, cloning, reverse engineering, and sensitive data extraction. As part of the Zynq-7000 device, an AES-256 decryptor implemented in silicon (located in the PL portion of the device) can be used to decrypt the FSBL, software image(s), and PL bitstream(s).

The software and bitstream data contains most, if not all, of the critical IP and sensitive data for the Zynq-7000 application, so it is important that it is protected by AES (i.e., confidentiality). The FSBL might not contain very much sensitive IP or data (if any), so its confidentiality is usually not much of a concern. Nevertheless, RSA authentication is vitally important for the FSBL, as it proves that it is authorized for a particular device and has not been modified in any way. However, when booting a Zynq-7000 device in "secure mode"—not to be confused with the ARM® TrustZone® "secure world" model [Ref 10]—the FSBL must pass through the AES decryptor in the PL after RSA authentication by the BootROM. [Ref 11]

Depending on the requirements of a system (security, cost, size, etc.), it might be necessary to protect against adversarial attacks at the device level—for example, DPA. As previously described, a DPA attack (which requires close physical access) attempts to noninvasively extract a key from a cryptographic function operating on an electronic device.

A designer might be faced with a need to provide even stronger protection against SCAs at some point in the future for an existing device, such as a Zynq-7000 SoC. To accomplish this, a soft cryptographic function can be instantiated in the programmable logic (or possibly coded in software). Designers have the flexibility to design their own, or purchase a core from a third party. (Various Xilinx partners that provide this type of IP are listed in a subsequent section).

Careful attention is required when providing a key source for the soft decryptor. Only the hardened AES-256 decryptor in the silicon has access to the BBRAM key and the eFUSE key, so neither of them can be used as a key source. Two possible means of providing a key are:

- Generate a key inside the Zynq-7000 device using a soft Physically Unclonable Function (PUF). [Ref 12] This is then used as a Key Encryption Key (KEK) generator for wrapping the red (secret) key that was used to encrypt the software and bitstream data. This requires a "registration" phase, where the red key is initially encrypted with the PUF's KEK (performed at a secure facility). It can then be stored externally along with the encrypted software and bitstream data. Subsequently, a "boot" phase is entered, where the PUF's KEK is reproduced on-the-fly during secure boot to decrypt the software and bitstream data key using the soft DPA-resistant decryptor.

  As an alternative, the PUF output can be used as a key split instead of a KEK.

- Instantiate a soft secure key exchange function (e.g., Diffie-Hellman) [Ref 13] so that the secret key can be securely transferred into the PL from an external source during secure boot, then decrypted for use by the soft DPA-resistant decryptor.

The basic idea of this secure boot enhancement method is that only the Zynq-7000 AP SoC's integrated silicon AES block is used for decryption of the FSBL, the soft DPA-resistant decryption core, and the soft key generator (where confidentiality is not much of a concern, but authenticity definitely is). All other software and bitstream data is decrypted with the soft DPA-resistant decryption core residing in the Zynq device's PL. Partial reconfiguration is then used to populate the remainder of the PL application design.

Since the hard silicon AES block must be used in this method, the user must store a decryption key either in BBRAM or eFUSE. If confidentiality of the FSBL, the soft DPA-resistant decryption core, and the soft key generator is not required, then a simple solution is to use an all-zeros eFUSE-based key (which is the default state of an unprogrammed eFUSE key).

Since the secure boot process in this method has additional phases, it is important to realize that this has some additional impact upon the overall boot time. Refer to Xilinx Answer Record #55572 [Ref 14] or consult with a Xilinx FAE to help quantify this impact.

The flow diagrams in Figure 3 and Figure 4 illustrate the REG and BOOT phases for a PUF-based key generation solution:

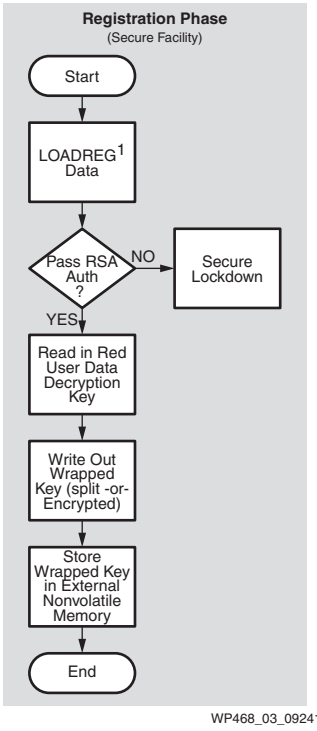


*Figure 3:* **Registration (REG) Phase**

## *Notes on Figure 3:*

1. REG data contains the FSBL, a soft PUF, and optionally an encryptor. The output of the PUF is either a KEK—or, where desired, it can be used as a key split (XOR with red key). The FSBL is RSA-authenticated by the BootROM code; the PUF and encryptor are RSA-authenticated by the FSBL code. The REG design has a read path to bring the red key into the device and a write path for the wrapped key (either split or encrypted) so that it can be stored external to the device. *The REG phase must be done in a secure facility, because the red key is being transferred into the device.*

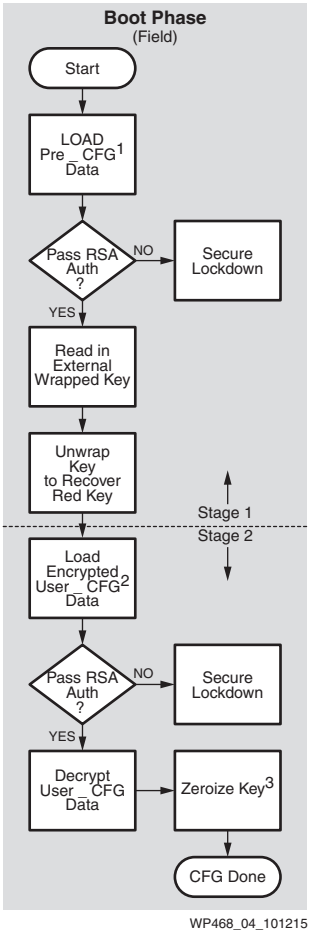


*Figure 4:* **BOOT Phase (Field)**

## *Notes on Figure 4:*

1. PRE_CFG data contains the FSBL, a soft PUF (must be the same exact PUF as the REG bitstream design) and a DPA-resistant decryptor. The output of the PUF is either a KEK or can be used as a key split (XOR with black key to recover red key). The FSBL is RSA authenticated by the BootROM code. The PUF and encryptor are authenticated by RSA code contained within the FSBL. The PRE_CFG data is used in the field during secure boot and it has no external output paths. It writes the unwrapped red key to key memory via an internal-only path. The unwrapping function is either a decryption using the PUF- generated KEK or an "un-split" function using the PUF-generated internal key split.
2. USER_CFG data is encrypted on the red user key and contains the end user application (software image(s) and bitstream(s)). Partial reconfiguration is used to populate the rest of the PL design.
3. The key can be optionally zeroized at this point, or it can be retained for future secure PR operations that bring in encrypted security functions at run-time.

### Run-Time Security Enhancements

After the Zynq-7000 AP SoC has booted securely, then soft and trusted security functions can become operational. These soft security functions can be part of the initial bitstream or loaded in on the fly (only as needed) using PR. Depending on the needs of the application, a number of different types of soft security functions can be implemented:

- TRNG
- Inline crypto engine to secure off-chip DDR memory contents
- Data crypto engine to secure communication channels (e.g., network)
- Public key crypto function (RSA or ECC)
- Customer-defined / proprietary security function

Additionally, Xilinx offers the soft Security Monitor (SecMon) IP core on Zynq-7000 devices, which can provide for a number of anti-tamper (AT) functions post secure boot. [Ref 15] If the DPA-resistant secure boot method previously described is being used, then SecMon is loaded along with the DPA-resistant decryptor; any subsequent PR operations goes through the SecMon secure PR port.

As mentioned earlier (as an advantage of a soft security feature), trade-offs between performance and resource utilization must be made. A very high performance soft security function can be implemented, but this can impact the resources available to the rest of the application. If the soft security functions are only needed at certain times, using PR can help alleviate resource conflicts by time sharing the reconfigurable regions amongst different functions.

## Third-Party Solutions

In addition to the SecMon IP core available from Xilinx, a rich ecosystem of third-party developers that provide security-based soft cores for use on Xilinx devices, such as DPA-resistant crypto cores, PUFs, TRNGs, etc. Here are some soft security IP developers that might be considered:

1. Helion Technology Limited (http://www.heliontech.com)
2. Verayo (http://www.verayo.com)
3. The Athena Group (http://www.athena-group.com)
4. Lewis Innovative Technologies (http://lewisinnovative.com)
5. Intrinsic ID (https://www.intrinsic-id.com)
6. Cryptography Research, Inc. (http://www.cryptography.com)
7. FMS Secure Solutions, LLC (http://www.fmssecuresolutions.com)

In addition to the above list, the Xilinx Alliance Program Members website can be used to find potential soft security IP developers. [Ref 16]

Regardless of the third-party solution chosen, as with any application implemented on an FPGA or SoC, the application developer is advised to verify the intellectual property status of the chosen implementation.

# Conclusion

In this white paper, various methods are presented to enhance the security of Zynq-7000 AP SoC-based applications by implementing soft security features such as a DPA-resistant decryptor. Both the security of secure boot and run-time processes can be enhanced with these methods. The value of Zynq-7000 devices in many embedded applications is well known, and the maturity of the tool set and rich ecosystem of software and IP continue to make the Zynq-7000 family a very attractive solution, now and into the future. Having the ability to enhance security further adds to the value of Zynq-7000 devices.

The reader is encouraged to refer to the various documents and websites referenced in this document and to contact a local Xilinx FAE for more in-depth discussions.

# References

1. Rambus Cryptography Research: Kocher, Paul; Jae, Joshua; Jun, Benjamin; *Differential Power Analysis (DPA)*, retrieved 13 September 2015.
2. National Institute of Standards and Technology (NIST), Information Technology Laboratory, Computer Security Resource Center: *Digital Signature Standard (DSS)*, FIPS PUB 186-4, July 2013, retrieved 13 September 2015.
3. Xilinx User Guide UG585, *Zynq-7000 AP SoC Technical Reference Manual*, v1.10, 2 February 2013, retrieved 13 September 2015.
4. Xilinx User Guide UG821, *Zynq-7000 All Programmable SoC Software Developers Guide,* v11.0, retrieved 13 September 2015.
5. National Institute of Standards and Technology (NIST), Information Technology Laboratory, Computer Security Resource Center: *Secure Hash Standard (SHS)*, FIPS PUB 180-4, August 2015, retrieved 13 September 2015.
6. Xilinx Application Note XAPP1175, *Secure Boot of Zynq-7000 All Programmable SoC*, v2.0, 3 April 2015, retrieved 13 September 2015. Design files accessed separately.
7. Xilinx User Guide UG1025, *Zynq-7000 All Programmable SoC Secure Boot,* v1.0.1, 18 March 2014, retrieved 13 September 2015. Design files accessed separately.
8. Xilinx User Guide UG909, *Vivado Design Suite User Guide: Partial Reconfiguration,* v2015.2, 24 June 2015, retrieved 13 September 2015.
9. Xilinx Application Note XAPP1084, *Developing Tamper Resistant Designs with Xilinx Virtex-6 and 7 Series*, v2.0, 3 April 2015, retrieved 13 September 2015.
10. Xilinx White Paper WP429, *TrustZone Technology Support in Zynq-7000 All Programmable SoC,* v1.0, 20 May 2014, retrieved 13 September 2015.
11. Xilinx White Paper WP426, *Secure Boot in the Zynq-7000 All Programmable SoC,* v1.0, 5 April 2013, retrieved 13 September 2015.
12. Google search soft puf fpga, re: Physically Unclonable Function.
13. Google search diffie-hellman, re: Diffie-Hellman soft secure-key exchange.
14. Xilinx Answer Record 55572, retrieved 13 September 2015.
15. Xilinx Product Brief SecMon, Security Monitor, retrieved 13 September 2015.
16. Xilinx Landing Page, IP Developers Member Locator, verified 13 September 2015.

# Revision History

The following table shows the revision history for this document:

| Date | Version | Description of Revisions |
|---|---|---|
| 10/20/2015 | 1.0 | Initial Xilinx release. |

# Disclaimer



# Automotive Applications Disclaimer