TECHNOLOGY WHITE PAPER

Data Protection with the QorIQ™ Platform T2080 Trust Architecture

CURTISS-WRIGHT

DEFENSE SOLUTIONS

Read About

Power Architecture Trust Architecture

Threats to data protection

T2080 System on a Chip (SoC)

High Assurance Computing

Trusted Operating System

Secure Boot

Info

curtisswrightds.com

Email

ds@curtisswright.com

Introduction

With the increased sophistication of embedded applications, systems designers and their customers are expressing heightened concerns regarding the importance of protecting the data residing in their systems, and by extension, their investment in intellectual property.

Curtiss-Wright Defense Solutions has given data protection a very high priority in the design of its Power Architecture® single board computers (SBC). Curtiss-Wright utilizes NXP's (formerly Freescale™) next generation system-on-chip (SoC) QorIQ T2080 Platform for its rugged, SWaP-optimized Power Architecture modules. The QorIQ Platform's Trust Architecture allows for developing systems to achieve higher levels of security, with reductions in cost, size and power.

This paper will refer solely to the T2080 Processor when stating the name of this QorIQ processor family. For a detailed review of the P4080 Processor, please read the following white paper: Embedded High Assurance Computing Using NXP Trust Architecture.

This paper presents an overview of the potential threats to an embedded system, and how the Trust Architecture can effectively defend against these threats.

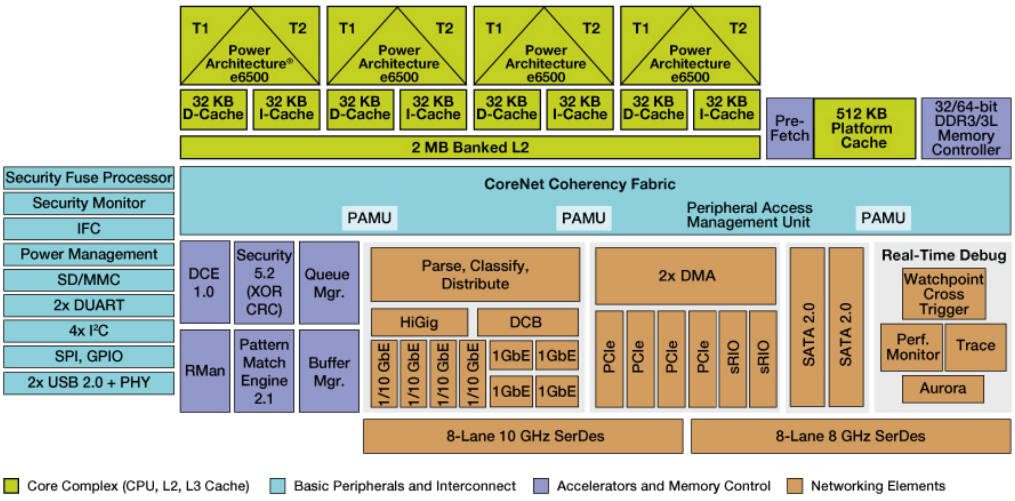

Figure 1: T2080 with QorIQ Trust Architecture

The Threats to Data Protection

In order to support a trusted environment, the security mechanism of an embedded processor needs to provide protection of the system assets against various threats, including the following:

- **Theft of Functionality:** loss of control of the system's functionality, such that the applications enable unauthorized features, or unauthorized parties exploit the system's features to the detriment of the intended application.
- **Theft of Data:** loss of data to an unauthorized party, where the system's users had a reasonable expectation that such a loss of confidential information or intellectual property would not occur.
- **Theft of Uniqueness:** loss of product differentiation through reverse engineering, duplication, and unapproved inter-operability.

In order to mitigate these threats, an embedded processor has to meet certain requirements:

- Unauthorized modifications to application software and system configuration information (such as device trees and certificates) in the supply chain and in deployed systems must be detectable, and such modified software and configurations must be prevented from being executed.
- Confidential code, customer installed private and session keys, and other system secrets must be protected against extraction or exposure by any means short of deprocessing (reverse engineering).
- Session keys negotiated during normal operation of the system must be protected against extraction or exposure.
- In a multi-core device, the processor must enforce strong barriers between partitions, so that the private resources of one partition cannot be accessed by another.

Once authenticated (trusted) software is running on a partition, the processor must implement mechanisms to ensure that this code isn't modified, or quickly discover if such a modification has occurred. It must also be possible to prevent the CPU from executing any instructions from memory reserved for data.

QorIQ T2080 Trust Architecture

T2080 Trusted Architecture has many mechanisms that provide defense against threats. Implementing effective trust into an SoC is a fundamental design aspect of the device's architecture. Without it, the security of confidential data and the system as a whole would be compromised. The QorIQ implementation of its Trust Architecture is quite deliberate.

E6500 processing core

The features incorporated into the processing core play a very important role in protecting data. The e6500 core used in the T2080 provides the following Trust 2.0 features:

- **Secure Boot:** The T2080 can be configured to perform a secure boot. The instructions executed from the internal boot ROM allow the T2080 to determine if code outside the internal boot ROM is safe to execute.
- **No Execute Bit (X-bit):** The Power Architecture Book-III E Translation Look aside Buffer (TLB) includes control bits that CPUs use to determine read, write, execute and caching rules for the memory pages. The 'X' bit in the TLB controls whether the page's contents can be executed as instructions. The ability to define pages as non-executable provides a significant barrier against attacks that overflow data buffers into code memory space.
- **Embedded Hypervisor:** The e6500 embedded hypervisor architecture introduces a third privilege level called "guest state". This allows hypervisor software to run at the most privileged level, with operating systems at a less privileged level, and applications at the least privileged level. The hypervisor software layer can then virtualize the e6500 core and block any guest OS attempts to modify critical security configurations, such as modifications of page table entries. By virtualizing all CPUs in the T2080, as well as the platform as a whole, systems can run with a mixture of trusted and untrusted partitions.

Peripheral Access Management Units (PAMUs)

The e6500 MMU settings determine which memory ranges are accessible by each partition, and the hypervisor prevents these settings from being altered by operating system or application software. In order to prevent system masters other than the e6500 cores from reading or writing sensitive memory regions, the T2080 implements a number of I/O MMUs called PAMUs (Platform MMUs). These PAMUs prevent internal and external masters from accessing memory for which they have not been granted explicit access permission. When secure boot is enabled, PAMUs block all external masters (PCI Express®, Serial RapidIO®) by default. Authenticated software can change the PAMU access permissions to later allow external masters.

Internal Boot ROM

The internal boot ROM contains code known as the ISBC (Internal Secure Boot Code). The ISBC is contained in an internal ROM to ensure that the code cannot be modified by an attacker. The ISBC is deliberately simple, and its only responsibility is to validate a signature over the next code to execute.

Security Fuse Processor

The Security Fuse Processor (SFP) uses provisioned values to enforce security policy in the pre-boot phase, and to securely pass provisioned keys and other secret values to other hardware blocks when the system is in a Trusted/Secure state.

Some of the secret values held in the SFP include the One Time Programmable Master Key (OTPMK), a hash of the Super Root Key (the Public Key used for signature verification), and a debug Challenge/Response value. The T2080 Trust Architecture also allows the OEM to create a battery-backed Zeroizable Master Key (ZMK) that can replace the fused master secret key.

Security Monitor

The Security Monitor (Sec_Mon) senses and controls the security state of the T2080. It receives inputs from hardware and software and uses this information to determine if conditions are safe to allow the SEC 5.2 to use an OEM selected OTPMK and a randomly initialized key encryption key (KEK).

SEC 5.2 with Run Time Integrity Checker

The primary function of the SEC 5.2 is to accelerate cryptographic operations such as IPsec and SSL. However, the SEC also has features that contribute to the Trust Architecture. A few of these features (OTPMK, KEK, and ZMK usage) have already been mentioned.

An additional feature is the Run Time Integrity Checker (RTIC). The RTIC leverages the SEC's cryptographic hashing capability to periodically check the integrity of designated sections of system memory.

The SEC can also be used to encrypt the data and application code which can then be decrypted upon loading and execution.

Is it Safe to Boot?

To ensure safety at boot time, the code should be digitally signed so that accidental or deliberate modifications to the code base will be detected during a secure boot cycle. Then the secure boot cycle should be performed, where the digitally signed code is authenticated before it is booted and executed.

Code Signing

The secure boot process used by the T2080 authenticates the load prior to execution. The customer must provide a digital signature for the code.

The customer does this by calculating a hash over the system code (see Figure 2), defined to include executable instructions and configuration information such as device trees, and a Command Sequence File header. Although the diagram shows the hash being calculated over a plaintext image, it is possible (even advantageous) for portions of the code to be encrypted with the OTPMK (or another key that is protected with the OTPMK). This prevents attackers from stealing the code from flash.

The customer generates an RSA public and private key pair. It is the OEM's responsibility to tightly control access to the RSA private signature key. If this key is ever exposed, attackers would be able to generate alternate images that would pass secure boot. If this key is ever lost, the OEM will be unable to update the image.

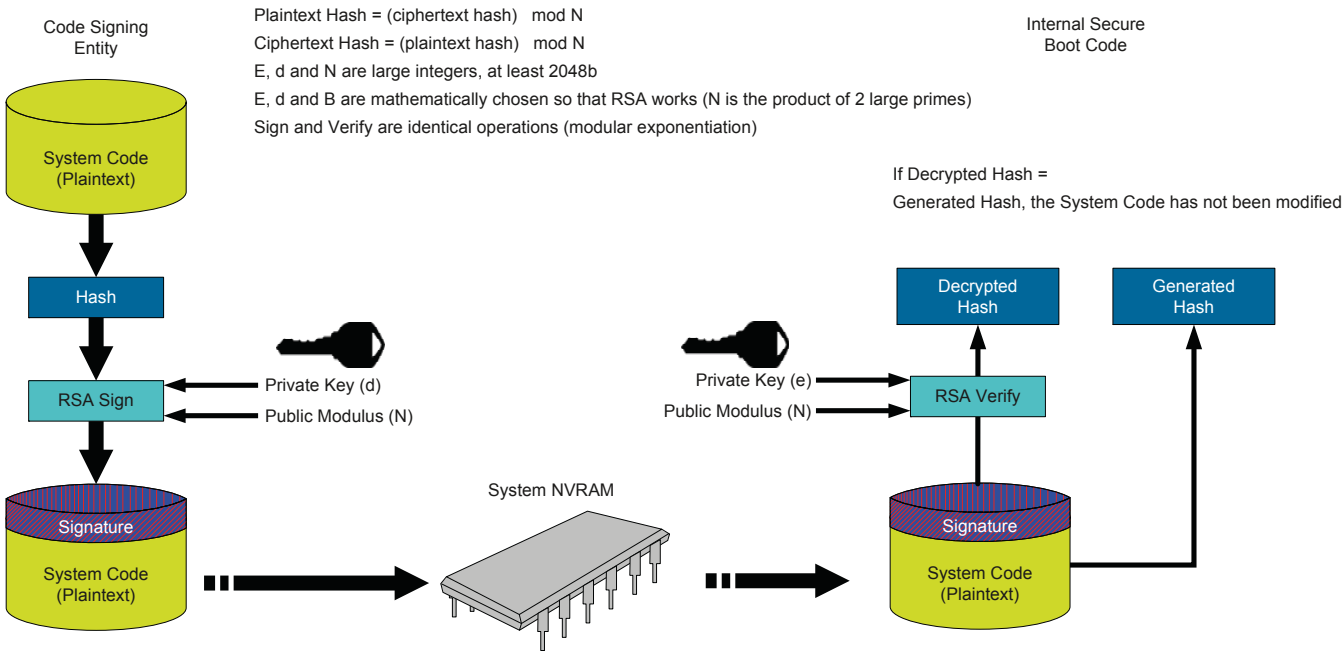

Figure 2: Code Signing and Signature Validation

The hash is signed using an RSA private signature key. This encrypted hash is known as a digital signature, and the digital signature is appended to the code, with both being written to flash (and other system non-volatile memory).

Secure Boot Sequence

The Secure Boot process requires the application to be signed using an RSA private signature key. The digital signature and a hash of the public key is appended to the image and written to flash (or other system nonvolatile memory). When the processor boots, the signature is checked using the RSA public key; the CPU uses the hashed RSA public key to check the signed image and compares the signatures. If the values match, the image is considered authentic and is allowed to boot. QorIQ processors enable portions of the image to be encrypted to prevent attackers from stealing the image from flash memory, and allow for alternate images to be used to add resiliency to the secure boot sequence. Trust Architecture on the T2080 provides extra protective features in that flawed images that were signed by the Super Root Key can be permanently revoked, and a Monotonic Counter prevents roll back to a flawed image without revoking the Super Root Key.

The three main phases of secure boot are as follows:

- **Pre-Boot phase:** In this phase, the T2080 checks to see whether a secure boot is required. The T2080 is configured appropriately and verifies that all external masters are blocked by the PAMUs, and proceeds to the Internal Secure Boot phase.
- **Internal Secure Boot:** In the internal secure boot phase, the T2080 begins executing instructions from the Internal Boot ROM. The instructions inside the Internal BootROM are NXP-developed code known as the Internal Secure Boot Code (ISBC). This set of instructions basically does a signature validation with the validated public key, verifies that the digital signature stored for the boot code is correct, and if so, then proceeds to the external secure boot.
- **External Secure Boot:** In this phase the T2080 starts the actual booting process. NXP provides reference code for a Trusted UBOOT Client, which executes in this phase. Trusted Uboot then performs typical Uboot configuration functions, such as mapping physical memory, initializing the Queue Manager and Frame Manager, and loading next stage software (Trusted Uboot Client) into main memory. The code to be booted into memory is validated using either the same public key as used by the ISBC, or it can be a new public key. If the signature passes, the Trusted Uboot Client begins execution on CPU 0 starting at the Client's First Instruction Pointer.

Trusted Operation

Once the T2080 is up and running, it continues to protect the system from threats in the following ways:

Protection Against System Modification and Theft of Functionality

OEMs and service providers want to prevent their systems from performing functions that the customer did not pay for, and also want to protect their customers from losing the functions they did pay for, due to the malicious actions of others. SBCs using T2080 processors are able to prevent the following types of attacks:

Attack Method #1: Tricking the System into Booting an Alternate Image: An attacker might try to change the board level signals used by the processor to determine the location of its Reset Configuration Word, and trick the system into booting the attacker's code from an alternative interface. If the boot interface uses socketed RAM on DIMMS or NAND flash thumb drives, the attacker's job becomes much easier. Curtiss-Wright boards use solder flash and memory so this is easily defeated.

Defense Method #1: Boot Time Code/Data Authentication: Once the T2080's "Intent to Secure" bit is set, even if an attacker alters the T2080's RCW or replaces the NVRAM, the attacker will only be able to hang the system during the boot cycle. Taking over the system is not possible. If the customer maintains tight control over the private key used to sign the original image, the attacker cannot create an alternate image that will pass the boot time authentication process.

Attack Method #2: Modifying System Code After Boot: If an attacker is unable to cause the T2080 to execute an alternative image at boot time, the obvious next target is modifying the code at run time. An attacker would likely try to exploit a connector on a peripheral interface (i.e. PCI Express) to add a CPU module or "mod chip" to the system. The attacker's code on the CPU module could initiate DMA operations designed to overwrite some or all of the authentic code with the alternate image the attacker wants to run.

Defense Method #2: e6500 Hypervisor MMU, PAMU, and Runtime Code Authentication: The T2080's e6500 core MMU is capable of strongly partitioning memory between executable code and non-executable data using the "X bit" feature in the Page Table Entries. The e6500 also supports a hypervisor privilege level. If the system is properly partitioned, the hypervisor and PAMUs will stop any cross partition modification attempts, and protect against attacks launched from external DMAs. In addition, for added security the SEC 5.2's RTIC can be used to regularly validate a hash over sensitive areas of memory, and trigger a system reset if modified code is detected.

Attack Method #3: Exploiting a Remote Management Interface, Remote Code Update Facility: Updating the system's image in the field from time to time is typically done via a management interface, physically separate from the control and data path interfaces. If this interface is not secured, an attacker could trick the system into accepting his code as a legitimate update.

Defense Method #3: Authenticate the Updater, and Authenticate the Code to be Executed: Communication between management software and a remote management system should always be performed through a secure tunnel (IPSec, SSL, SSH) with mutual authentication, regardless of any physical separation of the management port from control/data path ports.

Protection Against Theft of Data

Code modification attacks are generally a means to an end, and that end is often to extract end user data.

- **Theft of Certificates, Data and Keys by Software Running on the System:** Valuable data is protected against attack by means of strong access control and encryption. The same access control methods that are used to control cross-partition and external DMA write access can be used to control read access. These methods use the hypervisor and PAMU.

  Encryption can also be used to protect data and other information considered secret and make use of the SEC 5.2 acceleration engine. Two types of secrets need to be considered: long term and short term.

  1. Long term secrets are those that must survive from boot cycle to boot cycle and consequently must be held in non-volatile memory. Examples of long term secrets are system private keys, pre-shared session keys, certificates, as well as files in system storage (hard disk drive or solid state disk drive).

2. Short term secrets are not expected to be preserved across boot cycles. Examples of short term secrets are session keys negotiated with a peer system, ephemeral Diffie-Hellman key pairs, and user data such as IP packets flowing through the system.

- **Defense - Encryption to Protect Both Short and Long Term Secrets:** Encryption is a useful defense to protect both short and long term secrets. The T2080 uses a truly secure method for protecting long term keys. Long term keys are encrypted using a key derived from a nonvolatile master secret that is embedded securely in the T2080 hardware. This is the reason for the T2080 One Time Programmable Master Key that the OEM burns into the Security Fuse Processor.

  Keys derived from the OTPMK (referred to as Blob Keys) can be used to protect an unlimited number of arbitrary length long term secrets. Private and symmetric keys protected by block keys can be decrypted directly into the SEC and used without ever being exposed in unencrypted format on an external bus.

  In addition to the OTPMK, the SEC also supports a KEK. The KEK is initialized to a cryptographically secure random value after each boot cycle, and is never exposed outside the SEC. This can be used to protect short term secrets.

Protection Against Theft of Uniqueness

- **Counterfeit Clones:** A counterfeit clone is an exact (or exact as possible) copy of a system.
  - **Defense - Validation and Decryption:** System designers who use the QorIQ Trust Architecture to validate and decrypt their code achieve significant resistance to this type of attack, since driver modification will cause secure boot to fail, and the attacker will not be able to validate the signature, since the keys for these are secured in the T2080 and are not readable.

    Previously described code authentication and data protection methods can also be applied to preventing theft of uniqueness.

T2080 Trust Architecture in Action

Curtiss-Wright employs T2080 processors on the rugged 3U VPX3-133 and 6U VPX6-195 SBCs. These modules benefit from every aspect of NXP's Trust Architecture 2.0, allowing the end user peace of mind that their system and valuable data is protected. Curtiss-Wright's Power Architecture modules enjoy long life availability (15 years) and meet the high reliability requirements that Curtiss-Wright products demand for the long-term success of your programs. Please contact Curtiss-Wright for more information on our QorIQ SBCs.

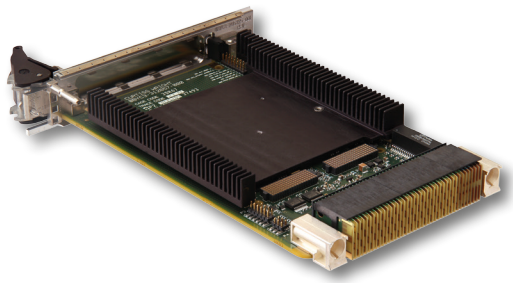

Figure 3: VPX3-133 Rugged 3U T2080 Quad-Core SBC

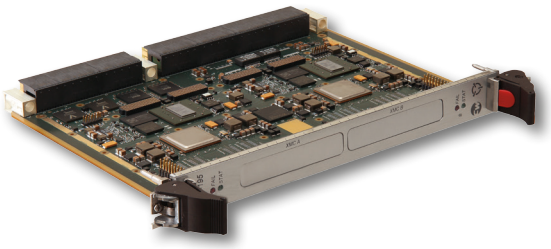

Figure 4: VPX6-195 Rugged 6U Dual T2080 Quad-Core SBC

Author

Mike Slonosky, B.Sc E.E and M.Sc E.E.

Product & Technology Specialist

Curtiss-Wright Defense Solutions

Summary

The QorIQ T2080 Trust Architecture provides all the necessary elements to allow system designs to feature data protection in their board level products. Curtiss-Wright has embraced this technology for its next generation Power Architecture SBCs, allowing our customers to build embedded systems that feature an unprecedented level of security.

If you would like more information on Trust Architecture or on Curtiss-Wright's Trusted COTS initiative, click here or contact us at ds@curtisswright.com.

Learn More

Products:

VPX3-133 - Power Architecture 3U Single Board Computers

VPX6-195 - Power Architecture 6U Single Board Computers

White Paper: The Power Architecture QorIQ Platforms: Architectural Benefits for Small Form Factor Embedded Applications

Article: Trusted Boot in COTS Computing

Technology: Secure Computing with Curtiss-Wright

References

T Series Fact Sheet. Retrieved Nov 25, 2015 from http://www.freescale.com/products/power-architecture-processors/qoriq-power-architecture-processors/qoriq-t2080-and-t2081-multicore-communications-processors:T2080

Freescale Semiconductor. (March 2015) QorIQ Platform's Trust Architecture Overview: Add Trust to Networked and Networking Systems. www.freescale.com.

Freescale Semiconductor (May, 2012). Manufacturing Guidelines for the QorIQ Platform's Trust Architecture. www.freescale.com.

Freescale Semiconductor (2013). Secure boot for QorIQ Communications Processors. www.freescale.com.