June, 2010

Embedded Control System Security: Protecting Against Software Cloning

FTF-ENT-F0234

Geoff Waters

NMG Systems Engineering - Security
This lecture will cover:

- Trends driving the need for secure systems and preventing industrial attack
- Considerations when protecting a system:
  - Requirements
  - Type and level of attack
  - Recent trends
- Cryptography
- Identifying elements in protecting your system
- The Freescale approach to security

**Presenter:** Ken Terry, Systems Solutions Engineer

**Expertise:** Industrial Segment Systems Engineering – Factory Automation

**This lecture will last about 1 hour.**
Session Objectives

At the end of this session, you will be able to:

- Understand why MPU Based Industrial Applications Require Protection
- Understand Basic Cryptographic Concepts
- Understand the Common Mechanisms employed in protecting MPU-Based Systems Against Hacking and Cloning
Agenda
► Introduction: The Need for Security
► What Requires Protection?
► Cryptography
  • Basic Overview and Concepts
► Secure Systems
  • Elements required to provide a secure system
► What can Freescale Offer?
Introduction – the Need For Security
Industrial Trends Driving the Need for Secure Systems

Protecting Industrial Communications Networks

- Historically, industrial controls systems were based on closed networks typically using proprietary communications protocols.
- Increasing levels of sophistication in control systems
- Industrial Automation systems moving from proprietary fieldbus communications to open communications protocols
- Greater level of interconnection with open IT networks and use of the internet making industrial networks more vulnerable to attack from outside
- Increasing requirement to include remote and mobile equipment as part of the industrial network
- Increasing levels of protection on commercial IT systems means that hackers are now focusing attention on “soft” targets such as industrial control systems and SCADA (Supervisory Control and Data Acquisition) systems
- Increasing level of cyber attacks on utilities and “Smart Grid” infrastructure
Industrial Trends Driving the Need for Secure Systems

► Preventing Malicious Attack
  • Greater awareness of need to protect systems against deliberate attack means that effective security measures are a key component of functional safety

► Protecting Intellectual Property
  • Increasing value of software vs. hardware in OEM (Original Equipment Manufacturers) products means that software IP (Intellectual Property) needs to be protected from copying or reverse engineering

► Preventing unauthorized user access to software enabled premium features
Industrial Trends Driving the Need for Secure Systems

► Preventing Product Cloning/Counterfeiting
  • Damages to commercial interests, corporate/product image, functional safety
  • Reverse engineered functional clones
  • Direct blind copy clone – OEMs using contract manufacturers in low cost locations may want to prevent the possibility of design overbuilding by those manufacturers with products being sold back into grey market

► Protecting Industrial Communications Networks
  • Providing secure data storage for master and session encryption keys and other sensitive data

► Networked Medical Devices
  • Pacemakers, Implantable Cardioverter-Defibrillators (ICDs), Bedside Monitors, MRI machines, and portable drug-delivery pumps are now enabled to transmit and receive information making them open to attack with potentially serious and even fatal consequences
Hacking The Industrial SCADA Network

By Frank Dickman, BSMAE, RCDD | NOVEMBER 2009 VOL. 236 NO. 11

It was a Trojan program inserted into SCADA system software that caused a massive natural gas explosion along the Trans-Siberian pipeline in 1982. A newspaper reported the resulting fireball yielded “the most monumental non-nuclear explosion and fire ever seen from space.”

Malicious hackers have discovered supervisory control and data acquisition (SCADA) and distributed control systems (DCS) since reports of successful attacks began to emerge after 2001. A former hacker interviewed by PBS Frontline advised that “Penetrating a SCADA system that is running a Microsoft operating system takes less than two minutes.”

http://pipelineandgasjournal.com/hacking-industrial-scada-network
Smart Meter Security Compromised at Black Hat Conference

- High profile demonstrations of smart meter vulnerabilities by the small security consulting firm IOActive at the 2009 Black Hat conference attracted widespread attention and press coverage. Consultants there were able to take a well-known smart meter, reverse engineer the NAN radio, protocols, and software by gaining access to the debug interface on one of the internal microprocessors, and create a malicious program that used the system’s remote updating functions to propagate the malicious program onto other meters connected to the NAN. This revealed a number of basic security flaws in the overall system design that are now being addressed by all industry participants.

http://www.blackhat.com/presentations/bh-usa-09/MDAVIS/BHUSA09-Davis-AMI-SLIDES.pdf
What Requires Protection?
How Much Security?

►When protecting a system you must consider:
  • What are you trying to protect?
  • What types of attacks do you need to protect against?
  • What are the likely attack points, and methods?
  • How much security do you require?
  • How much are you willing to pay?
  • How will security impact the underlying system?
  • How will you upgrade/maintain the system and security over time?
System Security

► Classic Security Requirements:
  • Confidentiality - prevents eavesdropping
  • Authentication - prevents impersonation
  • Data Integrity - prevents tampering
  • Non-repudiation - prevents denial
  • Trusted Processing - enables trusted platform for authorized access to program and data
  • IP Protection - prevent software/IP theft
Types of Attacks

► Electrical
  • Over/Under voltage
  • Power analysis
  • Frequency analysis
  • Electrostatic discharge
  • Circuit probing

► Software
  • Spy software insertion
  • Flow analysis
  • Trojan horse
  • Virus

► Physical
  • Temperature variation (into extremes)
  • Temperature analysis
  • De-processing
  • System theft
  • Partial destruction
  • Hardware addition/substitution
## Application Security Levels

| Secure System, Server/Client Authentication | Limits access to core system resources to OEM supplied and authorized software and data, and this is periodically authenticated with a secure server. Restricted execution of additional software without authorization is allowed. |
| Secure Local System, External Software supported | Limits access to core system resources to OEM supplied and authorized software and data, but restricted execution of additional software without authorization is allowed. |
| Secure Local System, OEM Software only | Ensures that only OEM supplied and authorized software and data can be used on the system, no other software can be executed. |
| Software (IP) System Protection only | Protection for system software and data IP, prevents software and data from being copied only |
| No System Protection, fully open | No system protection |
Cryptography – Basic Overview and Concepts
Passive Attack: Eavesdropping

Transfer $1000 from acct# 1234567 to acct# 7654321

Attacker gets account information to support identity theft. Other attacks possible; industrial espionage, blackmail…
Encryption Defeats Eavesdropping

Transmission inside encrypted tunnel, combining message with secret key

Transfer $1000 from acct# 1234567 to acct# 7654321

Qw;o9yf4 0hn4m4mc, niofen *#$ hn23n()* 1n

Attacker doesn’t know secret key, can’t eavesdrop.
Active Attack: Data Manipulation

Transfer $1000 from acct# 1234567 to acct# 4321

Transfer $100,000 from acct# 1234567 to acct# 10101010
Transfer $1000 from acct# 1234567 to acct# 7654321 (MAC=3127)  
Transfer $100,000 from acct# 1234567 to acct# 10101010 (MAC=3127)

Alice sends her request along with a Message Authentication Code derived from both the message and a secret key.

Attacker can’t generate the proper MAC for the altered message because he doesn’t know the secret key.

Bob the banker recalculates the Message Authentication Code over the message and compares to the received MAC. The data manipulation is detected, and the fraudulent transaction is not processed.
Asymmetrical (Public) Key Cryptography (RSA*)

Public key cryptography is based on a pair of keys:

► Public key for encryption (open padlock, anyone can lock)
  • Consists of the modulus \(n\), which is the product of two large prime numbers \(p\) and \(q\), which are kept secret, and the public exponent \(e\), typically \(2^{16} + 1 = 65537\)

► Private key for decryption (only the key can unlock the padlock)
  • Consists of the modulus \(n\), and the private exponent \(d\) which is based on the two large prime numbers \(p\) and \(q\)

For more information refer to:
http://en.wikipedia.org/wiki/RSA
The Code Book, by Simon Singh (Anchor)
*RSA - Rivest, Shamir, Adleman
**Cryptography**

**Symmetric Key Cryptography:**
- Same key used to encrypt and decrypt
- Very fast
  - Typically used for bulk of encryption/decryption
- Same key must be at both end points

**Asymmetric (Public) Key Cryptography:**
- 2 related keys are required (known as a public and a private key)
- 1000 times slower than symmetric key
- Typically used for exchange of symmetric keys and sender authentication
- End points need have had no prior contact

**Authentication:**
- Necessary to know who you’re speaking to
- Certificates used to verify identity
Ciphers and Hashing Algorithms

<table>
<thead>
<tr>
<th>Cipher/Algorithm</th>
<th>Type</th>
<th>Block Size</th>
<th>Key Size</th>
<th>Common Modes</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES</td>
<td>Symmetric Block Cipher</td>
<td>64 bit</td>
<td>56 bit</td>
<td>CBC</td>
</tr>
<tr>
<td>3DES</td>
<td>Symmetric Block Cipher</td>
<td>64 bit</td>
<td>168 bit</td>
<td>CBC</td>
</tr>
<tr>
<td>AES</td>
<td>Symmetric Block Cipher</td>
<td>128 bit</td>
<td>128 bit, 192 bit, 256 bit</td>
<td>CBC</td>
</tr>
<tr>
<td>ARC-4</td>
<td>Symmetric Block Cipher</td>
<td>8 bit</td>
<td>40 - 128 bit</td>
<td>-</td>
</tr>
<tr>
<td>RSA</td>
<td>Asymmetric Stream Cipher</td>
<td>NA</td>
<td>Up to 2048 and 4096 bit</td>
<td>-</td>
</tr>
<tr>
<td>MD-5</td>
<td>Hashing Cipher</td>
<td>512 bit</td>
<td>Up to 512 bit</td>
<td>HMAC</td>
</tr>
<tr>
<td>SHA-1/SHA-2</td>
<td>Hashing Cipher</td>
<td>512 bit</td>
<td>Up to 512 bit</td>
<td>HMAC</td>
</tr>
</tbody>
</table>
What Elements are Required to Provide a Secure System?
How are Systems Protected Today?

► Physical security:
  • Secure packaging
  • Secure packaging with tamper detect (e.g., pressure monitoring)
  • Secure packaging with tamper detect and destruction (e.g., dynamite)
  • Obscured part numbers
  • Hidden layers
  • Protected location

► Electronic Security:
  • Security bit protects on-chip non-volatile memory (e.g. Flash) on MCUs
    ▪ Prevent external access to on-chip resources:
      – Locks device into Single Chip mode (disables external parallel bus)
      – Disables Background Debug Mode
      – Disables Test Mode
      – Disables JTAG
      – Disables any (serial) "Bootstrap" functions
    ▪ Memory array bulk erase turns security bit off
  • Secure System (e.g. Freescale PISA)
    ▪ Code signing to prevent software tampering
    ▪ Assurance for stored IP
    ▪ Data stored encrypted in external memory
    ▪ Data decrypted and stored in on-chip private memory at runtime
      – How do you protect software IP?
  • Proprietary (CPU) Design
  • Silicon Obfuscation (e.g., obscuring the metal layer)
  • On-Chip Encryption Acceleration
    ▪ How do you protect the key?
Typical 32-bit Microcontroller System
Microcontroller System Security Considerations

CPU

Integrated FLASH

Peripherals

SRAM

FLASH

DRAM

Optional External Memory System

Protected Flash memory

Encrypted communications

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications

Microcontroller System Security Considerations

LAN PAN WAN

Hardware random number generator

Memory protection unit

Hardware cryptographic acceleration

Unique chip identifier

Secure key storage

Tamper detection

Disable external code accesses

Encrypted communications
Securing Data in External Memory

Secret key in combination with hardware crypto and secure RAM enables secure data storage in external memory.
Protecting a System Against Tampering

- Secure boot and integrity checker ensure that only authorized software will run on the system.
- Memory protection unit enforces restricted access to secure data.
- Tamper detection destroys data and keys when system is threatened.
Data and Program Integrity

Secure Boot & Integrity Monitor

System Reset

Vector to Secure Boot ROM

Determine Boot Mode/Memory

System Integrity Check

Authentication Hash

System Secure? Yes

User Code

User Code

System Integrity Check

Periodic

No

Halt Execution

• Ensures that the system configuration is as expected

• Ensures that the Application Program and Data have not been tampered with

• Only allows a fully verified system to execute application program

• Implemented in hardware, operates continuously in the background
  • Continues to verify system integrity
  • Prevents a Trojan system from being inserted

• Only allows a fully verified system to execute application program

• Ensures that the Application Program and Data have not been tampered with

• Ensures that the system configuration is as expected

• Implemented in hardware, operates continuously in the background
  • Continues to verify system integrity
  • Prevents a Trojan system from being inserted
Program Authentication

OEM System Provisioning

Application Program

Message Digest Hash

Private Key Encryption

Private Key

Public Key

Signature

Note: Program and Signature may also be encrypted for IP protection. Private Key has to be carefully managed and protected.

Secure Boot Authentication

Application Program

Message Digest Hash

Compare Hash Sum

Authentication Result

Public Key

Verify Key

Fuse Box

Public Key

Private Key

Encryption

Note: Program and Signature may also be encrypted for IP protection.
Following Reset:
1. Check initial system integrity
2. Authenticate program code
3. Transfer (decompress, and decrypt) program code from boot flash to DRAM
4. Initialize system and peripherals
5. Transfer control to application program
6. Protect sensitive data with secure RAM
7. Control memory accesses with MMU/MPU
8. Encrypt communications with CAU
9. Continuously hash memory with integrity checker
A secure system’s foundation consists of the hardware platform and the critical code that executes on that platform. This foundation is built with an on-chip tamper-resistant ROM-based process that initiates validation of the platform.

The High Assurance Boot (HAB) process gains control of the system immediately after reset by executing a known boot code resident in on-chip ROM. The HAB process includes:

- **Health Check** - Validating the secure HW
- **Authenticity Check** - Validating that the code image, stored in external memory, originated from a trusted authority
- **Integrity Check** - Verifying that the code is in its original form
- **Versioning Control** - Checks the external code version (code revocation system)

The boot process uses digital signatures to perform the validations.

The boot sequence is flexible because it is controlled by authenticated scripts that reside in off-chip memory.
Run-Time Integrity Checker (RTIC)

► Protecting read only data from modification is one of the basic elements in trusted platforms.

► Write protection can be achieved by using on-chip one time programmable (OTP) elements such as electrical fuses. Though OTP elements are write protected, their data capacity is limited and they are not flexible (once the element is programmed it cannot be modified.)

► The Run-Time Integrity Checker (RTIC) mechanism periodically checks the integrity of code or data sections during normal OS run-time execution without interfering with normal operation.

► The RTIC is an independent module that, once activated, cannot be stopped unless the device is reset followed by a ROM boot sequence. The RTIC is initiated and enabled as part of the high assurance boot sequence.
Debug Port Manipulation

- Debug port manipulation is one known hackers’ way of executing unauthorized program code, getting control over secure applications and running code in privileged modes.

- Debug ports such as the IEEE standard 1149.1 (or, “JTAG”) provides a hacker with all the means required to break the system’s security mechanisms and get control over the operating system (OS).

- Unauthorized debug port usage should be strictly forbidden in order to properly secure the system.

- However, a debug port must be available during platform initial laboratory development, manufacturing tests, and software debugging.

- In order to prevent debug port manipulation while allowing access for manufacturing tests and software debugging, a smartphone System-on-a-Chip (SoC) incorporates a debug port access regulator that provides four different protection levels represented by four fuse modes.
Measures for Effective Clone Protection

► Secure Boot
  • Key to establishing a trusted secure system
  • Prevents unauthorized code being run on a system

► Secure Boot with Device Unique ID
  • Prevents signed code from being used on multiple identical/systems
  • Threats
    ▪ Users with ability to develop hardware and access to unprovisioned (unknown/unconfigured) MPUs can reverse engineer OEM code and generate their own image to circumvent code signing restrictions.

► Secure Boot with Device Unique ID and Encrypted Image
  • Requires
    ▪ Secure Boot – Secure non-volatile storage - Public Key or Public Key Hash storage
    ▪ Encryption/decryption hardware support
    ▪ Secure Key Storage – programmable by OEM – not readable by software
  • Secret key managed by OEM or trusted partner
  • Secure MPU provisioning (including fuse Programming) during manufacture
Board Manufacture and System Provisioning/Flash Programming for Anti-Clone Protection

OEM or Trusted Partner Server

Request Initial Field Update

Provide Device UID

Encrypted Functional Image signed with Device UID

In System Programming Control (ISP)

Encrypted Image Bound to MPU with Unique Device ID and Secret AES OTPMK Key

- Requires OEM-programmed secret key (not accessible by SW)
- OEM provides MPUs directly to manufacturer with programmed PK or PK Hash and secret key and SFP configured for secure operation
- OEM encrypts all or some of functional image with secret key
- Encrypted Image is signed with device UID
- Encrypted and signed image provided to manufacturer
- Encrypted image will work only with MPU containing correct secret key even if signing process is circumvented by new public/private key pair
- Secret key is never made available to Manufacturer
- Countermeasure also protects against reverse engineering of OEM code
- OEM maintains audit trail of all enabled and functional systems

Manufacturer

External Flash

MPU

Device Specific Unique ID (UID)
PK or PK Hash
Secret OTPMK AES Key

External Flash

Blob Key Encrypted with OTPMK
Program Code/Data Encrypted with Blob Key
Message Authentication

External Flash

Encrypted Functional Boot Image Bound to Device UID and Device Secret Key

MPU

Device Specific Unique ID (UID)
PK or PK Hash
Secret OTPMK AES Key

In System Programming Control (ISP)

Request Initial Field Update

Provide Device UID

Encrypted Functional Image signed with Device UID

OEM or Trusted Partner Server

Initial Boot Image with PK Signature

External Flash

Encrypted Functional Boot Image

Manufacturer

External Flash

Encrypted Functional Boot Image

OEM or Trusted Partner Server

Request Initial Field Update

Provide Device UID

Encrypted Functional Image signed with Device UID

In System Programming Control (ISP)

Encrypted Image Bound to MPU with Unique Device ID and Secret AES OTPMK Key

- Requires OEM-programmed secret key (not accessible by SW)
- OEM provides MPUs directly to manufacturer with programmed PK or PK Hash and secret key and SFP configured for secure operation
- OEM encrypts all or some of functional image with secret key
- Encrypted Image is signed with device UID
- Encrypted and signed image provided to manufacturer
- Encrypted image will work only with MPU containing correct secret key even if signing process is circumvented by new public/private key pair
- Secret key is never made available to Manufacturer
- Countermeasure also protects against reverse engineering of OEM code
- OEM maintains audit trail of all enabled and functional systems

OTPMK – One Time Programmable Master Key
AES – Advanced Encryption Standard
UID – Unique Identifier

Freescale, the Freescale logo, AB/Wac, C-5, CodeTEST, CodeWarrior, ColdFire, C-Ware, mobileIQ, PowerQUICC, StarCore, and Symphony are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. BeeKit, BeeStack, CoreNet, the Energy Efficient Solutions logo, Flexis, MWC, Platform in a Package, Processor Expert, CorIQ, QUICC Engine, SMARTMOS, TurboLink and VoHoQs are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2010 Freescale Semiconductor, Inc.
What can Freescale Offer?
# Security Support on Freescale MPUs

<table>
<thead>
<tr>
<th>Categories/ Features</th>
<th>Microprocessor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>i.MX27</td>
</tr>
<tr>
<td>High Assurance Boot</td>
<td>Y</td>
</tr>
<tr>
<td>Secure Storage</td>
<td>Y</td>
</tr>
<tr>
<td>Real Security Monitoring</td>
<td>-</td>
</tr>
<tr>
<td>Hardware Cryptographic Acceleration</td>
<td>Y</td>
</tr>
<tr>
<td>Random Number Generator</td>
<td>Y</td>
</tr>
<tr>
<td>Secure Debug</td>
<td>Y</td>
</tr>
<tr>
<td>Tamper Detection</td>
<td>-</td>
</tr>
</tbody>
</table>
**i.MX25x Multimedia Applications Processor**

**Specifications:**
- **CPU:** ARM926EJ-S, 266-400 MHz
- **Process:** 90 nm LP2
- **Core Voltage:** 266 MHz @ 1.2V – 1.52V
  400 MHz @ 1.38V – 1.52V

**Key i.MX25 Features and Advantages**
- Same proven ARM926 platform as i.MX27
- High performance general purpose processor
- Enhanced security features, including tamper detection
- 10/100 Ethernet MAC with RMII support
- Two on-chip USB ports with PHY
  - High Speed USB OTG with HS PHY
  - High Speed USB Host with FS PHY
- 128 KB on-chip SRAM – ideal for ultra low power LCD refresh
- 3 general purpose 12-bit ADC channels
- Touchscreen controller
- Two CAN interfaces
- Two smart card interfaces
- Enhanced serial audio interface
- 16-bit 1.8V mobile DDR and DDR2
- 16-bit 3.3V SDRAM
- 3.3V I/O

---

**System Control**
- SJTAG/ICEM
- ETM
- Bootstrap
- Clock Mgt.

**CPU Platform**
- ARM926EJ-S
- 266-400 MHz

**Internal Memory**
- 128 KB SRAM
- 32 KB ROM

**External Memory I/F**
- NANDFC
- SLC/MLC (8bit ECC)
- SDRAMC
- mDDR
- DDR2
- SDRAM
- NANDFC
- NOR
- WEIM
- External Peripherals

**User I/F**
- CMOS Sensor I/F
- LCDC
- SLCDC
- 8x8 Keypad

**Security**
- Drv-Ice
- SRTC
- Tamper Det. (V, freq., Temp.)

**Connectivity**
- Smartcard I/F x 2
- HS USB OTG + HS Phy
- HS USB Host + FS Phy
- 10/100 Ethernet
- 12-bit ADC w/ Touchscreen Controller
- MMC+/SD/SDIO x 2
- CE-ATA
- P-ATA
- SSI/I²S x 2
- ESAS
- CAN x 2
- I²C x 3
- CSPI x 3
- GPIO x 4
- CE-ATA
- P-ATA
- SSI/I²S x 2
- ESAS
- CAN x 2
- I²C x 3
- CSPI x 3
- GPIO x 4
- Audio Mux
- 1-Wire

---

Reg. U.S. Pat. & Trm. Off. BeeKit, BeeStack, CoreNNet, the Energy Efficient Solutions logo, Flexis, MXC, Platform in a Package, Processor Expert, QorIQ, QUICC Engine, SMARTMOS, TurboLink and VortiQa are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2010 Freescale Semiconductor, Inc.
i.MX25 Security Subsystem

- RTICv3
- iROM (HAB)
- MMU
- ARM
- Secure Monitor
- KEM
- Secure RAM
- Zeroized RAM
- RNGB
- Key Generator
- Dry-Ice
- JTAG Alarm
- RTIC Alarm
- RNGB Alarm
- SW Alarm
- HAB Param
- JTAG Disable
- SCC
- RNGB Alarm
- Zeroize RAM
- RTICv3
- RTIC Alarm
- RNGB Alarm
- JTAG Disable
- HAB Param
- Secure Monitor
- Electrical Fuse Array (IIM)
  - Fuse Secret Key, Boot Image Version Control, Chip Unique Identification, SRK hash
- Electric Fuse Array (IIM)
  - Fuse Unique Secret Key
  - Volatile Keys

* NIST, Digital Signature Standard, FIPS Publication 186-2

SCC - Security Controller
SRTC - Secure Real Time Clock
KEM - Key Encryption Module

RTIC – Run Time Integrity Checker
RNGB – Random Number Generation Block

Freescale, the Freescale logo, NXP, C5, CODE/TEST, CODE/Warrior, ColdFire, C-Ware, mobileQT, PowerQUICC, StarCore, and Symphony are trademarks of Freescale Semiconductor, Inc.,
Reg. U.S. Pat. & Tm. Off. BeeKit, BeeStack, CoreNet, the Energy Efficient Solutions logo, Flexis, MXC, Platform in a Package, Processor Expert, CoreIQ, QUICC Engine, SMARTMOS, TurboLink and VorIQs are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners. © 2010 Freescale Semiconductor, Inc.
Tamper A and B

DryIce Supply

Power Switch

Programmable De-Bounce

On Board tamper detectors

A

B

buffer

buffer

Tamper Alarm

i.MX25
Dry-Ice Differential PCB Tamper Detection (Wire Mesh)

Dry-Ice module detects Tamper and initiate key erasure whenever:
- C is disconnected (floating)
- D is disconnected (floating)
- C and D are short-circuited
Major Goals for QorIQ Trust Architecture

► Give OEMs the tools they need to create trusted systems without Freescale as part of the chain of trust.
► Provide a unique “untamperable” silicon identifier, and create an “untamperable” binding between hardware and a public key.
► Validate a system image prior to allowing it to execute and allow validated images to use a device-specific secret key.
► Detect and respond to HW and SW security violations that could lead to exposure of secret key or use of secret key by distrusted software.
► Support strong partitioning of system resources.
► Secure Debug interfaces.
► Protect arbitrary numbers of session keys without impacting security acceleration performance.
Chain of Trust

Minimum Implementation: Single Stage Validation

Layered Implementation: Multi-Stage Validation Only

- Internal Secure Boot Code
- Barker Code
- Public Key
- Signature
- Image Pointer
- Main Image Plaintext

Certificate Based Method
- Main Image Plaintext
Session Summary

► We need to consider the threat to Industrial Embedded Systems and develop strategies to counter these threats

► Freescale is Implementing the Secure Systems Capabilities within its MPU Products

► The Secure System capabilities offered on the products described provide an important element in provide robust and secure industrial embedded products.
Thank you for attending, and I hope you achieved the following:

- A better understanding of security requirements
- Knowledge of cryptography methods for the embedded platform
- Knowledge of the security support offered on Freescale MPUs
Further Information

- [FTF-NET-F0695 - QorIQ Platform's Trust Architecture: P4080 Secure Boot](http://pipelineandgasjournal.com/hacking-industrial-scada-network)