freescale i.MX

White Paper

i.MX Technology

Security Solutions for a Connected World

Introduction

As the world evolves toward wireless connectivity, product manufacturers must find ways to secure and protect their own intellectual property (IP). This white paper covers the important role of security within embedded systems and how Freescale's i.MX platforms can help customers manage this new era of uncertainty. Although the consumer space drives the adoption of security solutions, this white paper also details market segments such as automotive and industrial where the need for robust security is growing quickly. The document starts with a brief history of security implementation and details Freescale's legacy of developing IP solutions to satisfy emerging security challenges. The second section covers common problems encountered by the product designer, while the third section highlights the benefits of Freescale's i.MX applications processors-based solutions. The focus of this document is on security (the prevention of malicious manipulation) and not safety (the protection against technical failures).

Table of Contents

2 Freescale's Security Legacy

In the late 1990s, with the fast-growing demands of cellular technology and over-the-air data deployment, the lack of security began to negatively impact the industry. Cellular companies addressed the lack of security by defining specific security IP blocks. As the former Motorola Semiconductor Sector, Freescale developed specific technologies to satisfy the growing security demands of cellular devices and network infrastructure. As a combined effort, an IC design group was created to share IP and infrastructure to create advanced capabilities and implementations for the cellular and networking marketing segments. More than 10 years later, Freescale's IC security team continues to deploy strong expertise and advance the company's legacy of security solutions. With the deployment of security solutions in the automotive and industrial markets, Freescale is in the position to lead in this sector well into the future.

3 Overcoming Security Challenges

The list of potential security challenges is a long one. Wireless technology, open source software and Internet protocol simplify the possibility of modifying information. Three kinds of threats are potentially problematic:

- **Eavesdropping:** Someone accesses information without modifying it. The end user might make a decision without knowing that someone else has seen sensitive information, resulting in serious financial and personal damages for the end user.
- **Tamper:** Someone modifies information without the approval of the manufacturer or the end user.
- **Impersonation:** Someone pretends to be another person to obtain an end user's confidence or a manufacturer's approval.

While cellular—and more broadly, consumer—spaces require more security to protect sensitive data as smart phones, tablets and other smart devices gain popularity, the automotive and industrial segments also have similar protection needs. Here are just some of the factors creating new security concerns across all segments:

- **Proliferation of wireless end products:** By moving from a wired world towards a wireless one, the possibility of malicious attacks that download new software—altering the manufacturer's original intent—become easier. Also, wireless networking removes the need for physical access to mount an attack. The hacker can easily and remotely attack any system without being physically close to the end product.
- **Field programmability:** Combined with the adoption of rich operating systems that require larger memory capacity, external flash memory becomes mandatory to allow both the addition and upgrade of new services. Upgrades performed by product manufacturers must be encapsulated into a defined process and methodology to ensure the integrity of the new program. This process and methodology must also reject any malware code.
- **Adoption of new business models:** The cellular segment has shown how new software applications can be deployed quickly with a multitude of system solutions. In line with this evolution, the automotive segment can use the cellular model to ease the adoption of downloaded services and applications in the next generation of cars. A prepay model of downloading movies, music and other services needs to secure financial transactions without adding additional risk to critical automotive systems.

- **Protection of private information:** Cellular end users can easily understand the problem of hackers entering wireless consumer products to obtain private information such as calendars, contacts, biometrics and other sensitive information. This is a serious problem for an individual or a family, but poses minimal system impact. However, when it comes to corporate intellectual property, many more people might be affected by such a malicious attack. Stolen ideas, patents or secret business information such as merger and acquisition activity can cause widespread damage and must be rigorously protected.
- **Personalized settings and information:** Although it can be difficult to implement, one trend is to personalize end products. End users can personalize the user interface and specific applications for a smartphone. By applying a secure code, the smartphone can then boot with all the specific profile information for each user. This concept might also be applied to cars where the car might have different seat, wheel and mirror positions, navigation and infotainment information depending on the driver's profile. Today's technology allows driver recognition thanks to biometrics data. However, a hacker could switch profiles between people or change profiles if the biometric reading system is not secure. As a consequence, this personalized concept requires secure identification of the end user.
- **Increasingly connected infrastructure:** To improve efficiency and safety, governments are increasingly developing "connected" infrastructure across a number of functions, including public safety, energy management, medical and municipal services. Consequences of hacking or manipulation of these systems range from problematic to be potentially disastrous. For example, tampering with energy grids could range from illegal siphoning of electricity to full grid failure for a region or country.

To prevent hackers from modifying the content of information and services, technical solutions must comply with the following basic concepts:

- **Integrity:** To assure the receiver of the information that no one has altered the content.
- **Authentication:** To ensure that the information originates from the expected sender.
- **Confidentiality:** To protect active information by using a concealing mechanism.
- **Non-repudiation:** To ensure that the sender cannot deny that the information was sent.

It is important to note that a secure system can never be 100 percent secured. The level of security must correspond to the time and money a hacker can spend to break a device. The device designer must solve this simple equation: a system becomes secured when the time and money required to break the product is greater than the benefit derived from breaking the product.

4 Required Technical Implementation to Secure End System

This section covers technical solutions to security challenges. Chapter 5, "Security Implementation on i.MX," covers how Freescale has implemented a fully secured solution to protect the security and investments of end users and manufacturers.

4.1 Secure Boot

Flash memory used to be embedded in the main CPU, allowing for easy code integrity and authentication. Due to rich OS adoption, external flash memories are now required. This turns the reads and writes of the manufacturer code into a weakness for the entire system. The mechanism explained below represents a secure path to ensure that the code residing in the flash corresponds to the code embedded during the manufacturing phase. This meets necessary integrity and authentication criteria.

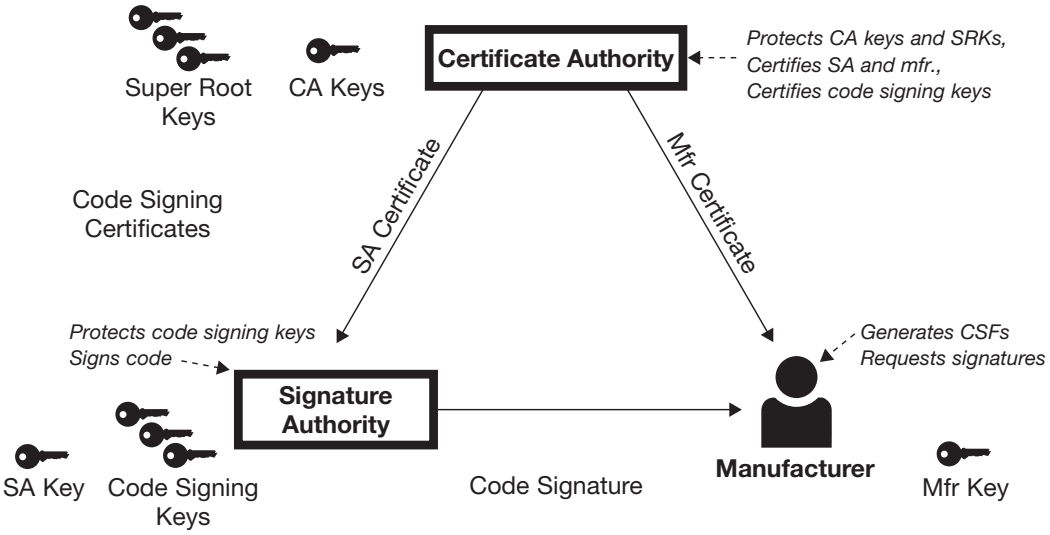
To maintain the security of the primary functions of the end product, the manufacturer must ensure that code cannot be modified by a malicious hacker. Also, in the case of software upgrades, the manufacturer must assure that no one can access the system while the newest code downloads. Another layer of security during the boot process consists of decrypting the code downloaded from the external flash.

4.1.1 Required Infrastructure

To understand the overall concept of integrity and authentication, trust becomes the first criteria among different parties involved in the signing process. The basis of this concept begins with complete trust in the Certificate Authority.

- **Certificate Authority (CA):** This entity owns the top-level keys (CA keys) used for generating SA Cert, Mfg Cert and Code Signing certificates. The different certificates represent the proof that all parties can trust each other. The Super Root Keys (SRK) can generate a key for code signing. CA represents the reliable entity that other parties involved in this process must rely on during the certificate exchange process to simply address the authentication process.
- **Signature Authority (SA):** This entity creates the signature of the manufacturer's software product. SA keeps the code-signing private key and can generate the code-signing public key that a customer can use for the secured boot. The SA must rely on CA to obtain the SA certificate for authenticating the manufacturer. CA and SA could be the same company.
- **Manufacturer:** Owns the software license and must rely on the CA to provide their IP to the SA.

Figure 1: Secured Boot Infrastructure



Relationships and responsibilities within the secured boot infrastructure

No transaction can start before the CA provides the required certification to the other parties. The signature process occurs as follows:

1. **SA must prove its trustworthiness to other parties**

   a. SA sends the CA its SA public key

   b. The CA generates the SA certificate by signing the SA public key with the CA private key

c. The SA then publishes its SA certificates publicly

2. Manufacturer must prove its loyalty to other parties

a. The Manufacturer can create a Manufacturer key and send the public part to the CA

b. The CA signs the Manufacturer key to create a Manufacturer certificate with its CA private key

c. The Manufacturer can publish the Manufacturer certificate publicly

Once the process starts, SA can verify the Manufacturer certificate thanks to the CA certificate, and the Manufacturer can also verify the SA certificate.

4.1.2 Code Signing Process Initiation

Once the mutual loyalty checks have been performed, the next step consists of generating the code signing keys between the SA and the manufacturer. Again, this process relies on the CA. The code signing keys depends on the SRK. This code signing key creation is the responsibility of the CA. Specifically, the CA generates the public and private SRK pairs and signs the SRK public keys with the CA key (the most secure key). Once the recognition process is performed, the SA can create code-signing keys and send the public key to the CA. The CA signs the keys, creating code-signing certificates which are sent back to the SA and then forwarded to the manufacturer.

Note: This acknowledgement process is similar to what happens when a bank customer desires to access his/her bank account. There is a need to obtain a certificate from a third party that the user relies on (CA) while the bank (SA) must verify the identity of the account holder. Initially, the user has been recognized to the CA through a password and username. When an access is performed, the bank asks for this username/password, while the user also must make sure the Web page is the one created by the bank and not by a hacker. In this case, the public key is the bank user name while the private key is the account information.

4.1.3 Code Signature Process

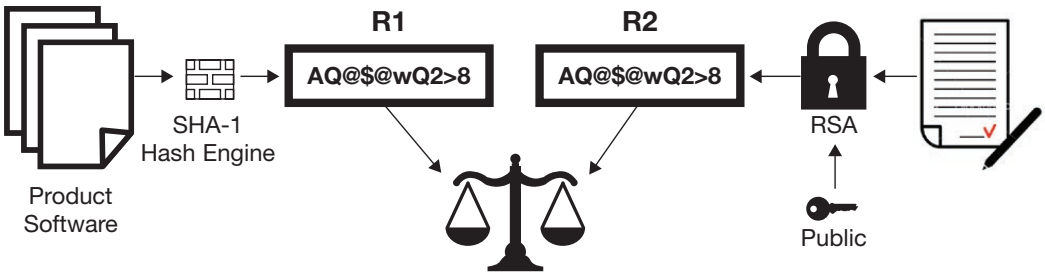
Once the mutual loyalty and code signing process is finished, the SA product code signature can start. A secured hardware link between the manufacturer and the SA has been already installed. The manufacturer enters the SA environment using a password-protected account (to prove to the SA that the user is who he says he is and can be compared with the CA certificate). Once both pieces of information are correlated, the SA can accept the code through an uploading process. Figure 2 explains the hash process. Hash is the function of mapping large data sets to smaller data sets called keys.

Figure 2: Signature Generation

RSA/SHA-1 signature generation applied on Manufacturer code

The product software is hashed using either SHA-1 or SHA-256 to obtain a 160-bit (for SHA-1) and 256-bit (for SHA-256) word. The signature is generated through a Rivest-Shamir-Adleman (RSA) algorithm with the code-signing private key of the manufacturer combined with the hash result. This signature goes back to the manufacturer through the secured link. Along with the product software, these two components are downloaded to an external flash memory.

Figure 3: Signature Verification



A comparison between the on-board software component and the embedded signature takes place during the boot sequence

4.1.4 On-Board Computing

With the same process used during the signature obtainment, the product software embedded in the flash is hashed to obtain an encrypted result called R1. On the other hand, the embedded signature residing in the flash will be RSA'ed with the code-signing public key located in the end product. A result R2 is then calculated. This public key is paired with the secretly stored private key, meaning the R1 and R2 must have a mathematical relationship (R1 must be equal to R2). If they are equal, this means that the code residing in the embedded flash was not modified and corresponds to the primary code downloaded during the manufacturing phase. If they are different, this means that the product software experienced a change and the boot must stop to reflect this unwanted change.

4.1.5 Required Protection to Ensure a Safe Secured Boot

For the concept explained above, some precautions must be set.

4.1.5.1 Code Signing Public Key Protection

If a hacker could change the private-public key, new product software could be downloaded with the new signature hashed with the hacker's private key. The end product could then boot and apply the same comparison process with a new paired key linked to the malicious product software. The secure boot mechanism cannot detect the new software upgrade, because the comparison was applied with the public key paired with the hacker's private key. That implies that the public key must be well protected to avoid such attacks. This is why public keys are located in the ROM or in OTP, depending on the flexibility vs. space tradeoff.

4.1.5.2 Security on Secure Boot

The manufacturer must ensure that there is no attempt to access the product during the boot up phase in order to modify the sequence or work around the security aspects of the authenticated software. An on-chip ROM-based implementation can help prevent this.

In summary, integrity and authentication drives the secure boot process, which plays an important role in the system. The secure boot represents a complex implementation to satisfy the IP ownership, independence and financial requirements of the manufacturer. This includes the reliability

and trust of the CA, which is the clear basis of this concept combined with the protection of assets such as keys. Without this trust, the secure boot process can easily fail, jeopardizing entire business, financial and technical models. The reader can easily understand the importance of secure boot to a company's success.

4.2 Cryptography

This 2,000-year-old technique was used by Julius Caesar to provide instructions to his army that the enemy could not read. Although both civilization and cryptography techniques have evolved over time, the goal has always been the same: making sensitive data obscure to unauthorized users. The essential concept consists of safely maintaining a key that only a few people know in order to encrypt or decrypt the message. Two techniques available today are symmetric or asymmetric cryptography.

4.2.1 Symmetric Key Cryptography

Symmetric cryptography is used for data encryption and data integrity due to its high-performance characteristics. Exchanging big files is feasible with this concept. However, decrypting the message requires that the keys are received safely and securely. The foundations of this concept are based on ways to send the keys and prevent an attack from an unauthorized person. If this unauthorized person is able to obtain the keys through malicious ways, the encrypted message becomes easily readable. Well known symmetric algorithms include AES, DES and 3DES.

4.2.2 Asymmetric Key Cryptography

This concept relies on mathematically related keys where one message encrypted with one key can only be decrypted by the other keys. Private and public keys represent those above paired keys. The private key must be protected and is provided only to trusted people, while the public key can be deployed in the field. Asymmetric cryptography can be used for short messages with slow decryption mechanisms. This is why asymmetric cryptography is commonly used for key exchange, attestation and authentication. Well known asymmetric algorithms include RSA, DSA and ECDSA.

4.3 Run-Time Integrity Checking

After all boot stages are complete and the rich OS and applications are running on a platform, this software becomes vulnerable to run-time attacks. For example, an attacker may attempt to modify the OS kernel to change the operating behavior of the system. This is possible because the attack occurs at run time after the secure boot has authenticated the OS. Run-time integrity (RTIC) checking can protect against such attacks and occurs in two stages. First, a reference cryptographic hash is computed during secure boot as part of the authentication process. This reference hash is then stored. In the second stage, periodic cryptographic hashes are computed over the same software that was authenticated during the secure boot phase. The computed hashes are compared with the stored reference hash. Any difference between the two hash values indicates the software has been modified post authentication.

Protecting Secured Information and Encryption

Signing the code implies the authentication of the code. However, this does not prevent a hacker from accessing code residing in the external flash memory. Some precautions and protections must be set up to avoid cloning and reverse engineering. Cloning allows a hacker to duplicate a product, which can cause the legitimate manufacturer to lose some market share versus the copied product. Reverse engineering can cause more financial impact if a manufacturer's entire product line is duplicated. As a result, encryption of the code ensures that the manufacturer can protect its IP and profits.

The encryption requires secured and protected keys stored in the end product, preferably in the protected CPU memory, with appropriate access policies in place. The code encryption key and the software image can be downloaded during the manufacturing phase in a secure way that the manufacturer must protect (called provisioning). During the boot process, a decryption must take place to get the system up and running. The decryption phase operates on the fly as to not delay the program and to avoid adding stalls between the external flash and in-RAM reading. The decryption uses the encryption key located in the secured environment.

4.4 Task Separation

Task separation is critical in a secure system as it prevents entrusted tasks from interfering with any other task. Providing a memory management unit (MMU) can easily enforce such restrictions.

4.5 Unique Identification

Protecting the end-product IP becomes an interesting and mandatory requirement for product manufacturers. One of the first steps consists of having a unique ID for each product which can be programmed by burning an electrical fuse.

4.6 Tamper Detection Mechanism

The purpose of the tamper detection mechanism is to provide evidence of any physical attempt to remove the device cover. The specific embedded mechanism drives to zeroing the sensitive date stored in the embedded memory. This information can be a secret key or any other personal data. Specific market segments such as payment and financial companies require detailed and accurate implementation where the tamper pins have to meet specific payment card industry PIN transaction security (PCI PTS) requirements.

4.7 Random Number Generator Accelerator

This block provides random data created by clocking shift registers with clocks derived from ring oscillators. The random numbers generated by the random number generator are intended for direct use as secret keys, per-message secrets, random challenges and other similar quantities used in cryptographic algorithms.

4.8 Secure Debug Phase

The next opportunity to hack the code is by reading the on-chip RAM where the code displays in plain text. The JTAG port represents the link to access the RAM. While protecting the JTAG requires specific implementation, the port must stay accessible for different reasons:

- Technical bring-up
- Manufacturing, test and troubleshooting
- Off-site debugging to either upgrade the software or perform maintenance

Different security modes provide a certain level of flexibility the programmer can enable:

- **No Debug:** Provides the maximum security to the device. All security-sensitive JTAG features are permanently blocked.
- **Secure JTAG:** High security. JTAG use is regulated by challenge-response-based authentication mechanism.
- **JTAG Enabled:** Low security. JTAG port is always enabled.
- **No Security:** The Secure State Machine is forced to remain in its secure state during JTAG operation.

5 Security Implementation on i.MX Applications Processors

5.1 i.MX Security at a Glance

Secure boot, real-time integrity checking, secure JTAG and encryption represent the key assets the manufacturer has to understand and develop to secure the company's IP ownership and financial balance sheet. This represents heavy investment in resources, expertise, time and money. In the past, most companies declined the investment. Today, declining the security investment may worsen the financial bottom line and perhaps cause a company's IP roadmap to fail. Freescale's i.MX solutions portfolio implements security technology to address the complex requirements for security applications. This helps to simplify development, enabling faster time to market and lower system costs by facilitating consumer development.

Table 1: Security Blocks Implemented in i.MX Applications Processors

	i.MX Applications Processor Families						
Categories/Features	**i.MX233**	**i.MX25x**	**i.MX28x**	**i.MX35x**	**i.MX50x**	**i.MX51x**	**i.MX53x**
ARM® Core	926EJ-S	926EJ-S	926EJ-S	1136FJ-S	Cortex-A8	Cortex-A8	Cortex-A8
ARM TrustZone® Technology Support	N	N	N	N	Y	Y	Y
High Assurance Boot (Digital Signature Based)	N	HABv3	HABv4	HABv3	HABv4	HABv3	HABv4
Encrypted Boot	Y	N	Y	N	N	N	N
Secure Storage	DCP**	SCC/DRYICE	DCP**	SCC	SCC	SCC	SCC
On-Chip Secure RAM	N	2 KB	N	2 KB	N	16x8 KB	4x4 KB
Real-Time Security Monitoring	N	SCC/DRYICE	N	SCC	SCC	SCC	SCC
Hardware Cryptographic Acceleration	AES-128	N	AES-128	N	N	AES-128, DES, 3DES, ARC4,SHA-1, SHA-256	AES-128, DES, 3DES, ARC4,SHA-1, SHA-256
True Random Number Generator	N	Y	N	Y	Y	Y	Y
Run-Time Integrity Check	N	Y	N	Y	Y	Y	Y
Tamper Resistant Real-Time Clock and Monotonic Counter	N	DRYICE	N	N	SRTC	SRTC	SRTC
Secure Debugging	N*	Y	N*	Y	Y	Y	Y
On-Chip OTP Space	Y	Y	Y	Y	Y	Y	Y
Physical Tamper Detection	N	SCC/DRYICE	N	N	N	SCC***	SCC***

*JTAG support can be disabled via e-fuse
** On-chip storage only
*** Works with main power applied only

Table 1 illustrates the high-level security blocks required to protect manufacturers' products and the corresponding Freescale i.MX applications processors.

5.2 Specific Example: ePOS Segment with DryIce Block

Freescale has developed a highly optimized implementation of electronic point of sale (ePOS) to meet the PCI PTS certification. The i.MX258 processor embeds both the application management and the secure processor to protect the keys and other sensitive data protected by a specific tamper pin mechanism. The secure processor embeds the DryIce security module which provides a trusted and certifiable time source for digital rights management (DRM). DryIce also provides volatile storage of encryption keys together with robust tamper detection and secure key erase. This implementation was PCI PTS 3.0-certified in September 2011.

5.3 i.MX Resources

5.3.1 i.MX Security Reference Manuals*

The following security user manuals are recommended for a deeper understanding of each block and its embedded functionalities.

* Please contact your sales person or FAE for information on accessing these manuals.

- **i.MX51**, **i.MX53** and **i.MX25** security reference manuals
- i.MX28 and i.MX50x security reference manuals are not required for these devices as the security feature set is a subset of those available on the i.MX51, i.MX53, i.MX35 and i.MX25.

  The information in the i.MX28 and i.MX50x manuals is also covered in the IC-specific reference manual (DCP chapter covers encryption and hashing. Boot chapter introduces secure boot.) as well as in the documentation for the code signing tool package.

5.3.2 Code Signing Resources

Embedding the high-assurance boot IP block allows customers to prevent hackers from potentially modifying, cloning, changing and stealing code. In line with the system solution, Freescale provides an environment to sign the code:

- Freescale provides code-signing tools to emulate CA and SA entities. Customers must provide a host computer to perform the code signing. This host computer must be protected, as it contains all of the signing keys. The **i.MX code-signing tools package** is available at no charge.
- Third-party partners with security implementation experience are available to customers who desire additional services. A separate business model would be required between the third party partner and the customer. Please contact your local Freescale representative for more information.

6 Conclusion

The vision of a converged environment drives technology adoption in almost every segment. Security follows this trend, and the need to deploy a system solution becomes crucial for our customers. The security expertise and knowledge the i.MX team developed in the consumer segment can be deployed and leveraged in automotive and industrial segments. i.MX technology delivers leading security solutions for our increasingly connected world.

Contributor

Jean-Louis Dolmeta

Appendix A: Glossary of Terms

Advanced Encryption Standard (AES): A symmetric algorithm for public-key cryptography

Data Encryption Standard (DES/3DES): A symmetric algorithm for public-key cryptography

DryIce: i.MX module that provides volatile key storage, tamper protection and secure real-time clock services

Hash: The function of mapping large data sets to smaller data sets called keys

High Assurance Boot (HAB): Freescale's implementation for secure boot

Joint Test Action Group (JTAG): A common debug interface used for testing printed circuit boards

Rivest, Shamir and Adleman (RSA): An asymmetric algorithm for public-key cryptography

Secure Hash Algorithm (SHA): A cryptographic hash function

How to Reach Us:

Homepage:

freescale.com

i.MX Information:

freescale.com/iMX

Email:

support@freescale.com
USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
1-800-521-6274
480-768-2130
support@freescale.com

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate,
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For more information, visit freescale.com/iMX

Document Number: IMXSCRTYWP REV 1