1 Introduction

i.MX RT10xx, except i.MX1010, provides an on-the-fly encryption engine called Bus Encryption Engine (BEE), which is dedicated for FlexSPI. This application note explains how to use BEE to decrypt data in application layer. The purpose is that data is encrypted in FlexSPI to avoid getting plain text and code is encrypted and can be read and on-the-fly executed. This is especially useful for a second secure boot.

NOTE
BEE can only decrypt data.

2 Abbreviations

Table 1 provides an overview of the abbreviations as used in this document.

Table 1. Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEE</td>
<td>Bus Encryption Engine</td>
</tr>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>ECB</td>
<td>Electronic Cookbook Mode</td>
</tr>
<tr>
<td>CTR</td>
<td>Counter Mode</td>
</tr>
<tr>
<td>SW_GP2</td>
<td>Software general purpose key from efuse</td>
</tr>
<tr>
<td>SNVS</td>
<td>Secure Non-Volatile Storage</td>
</tr>
<tr>
<td>DCP</td>
<td>Data Co-Processor. This module provides general encryption and hashing functions</td>
</tr>
</tbody>
</table>

3 Basic knowledge of BEE

3.1 Introduction of BEE

The BEE module is implemented as an on-the-fly decryption engine. Main features of the BEE module are:

- Standard AXI interconnection.
- On-the-fly AES-128 decryption, supporting ECB and CTR mode.
- Aliased memory space support. Address remapping for up to two individual regions.
- Independent AES Key management for those two individual regions.
• Bus access pattern optimization with the aid of local store and forward buffer.
• Non-secured access filtering based on security label of the access.
• Illegal access check and filtering.

The known hardware limitations of the BEE module are as follows:
• Only supports 128 bits data width AXI interconnection.
• Only supports 16-byte burst access size. For a single transaction, the minimum supported access size is limited to 4-byte.
• Granularity of the address bias is 128 KB per step.
• Maximum supported burst length is limited to 4.

3.2 Mechanism of BEE

As described in Security Reference Manual for the i.MX RT1050 Processor (document IMXRT1050SRM), a simple block diagram is abstracted away for users, as shown in Figure 1.

![Figure 1. BEE block diagram](image)

BEE key is from three ways, including SNVS Master Key, SW_GP2 and Software key. SW_GP2 is from eFuse which can be remapped at OCOTP Bank5. After the system resets, SW_GP2 be copied to shadow registers automatically. If SNVS Master Key or SW_GW2 is selected as BEE decryption key, and it can be loaded by BEE engine. Users can also use self-defined key via setting BEE key registers. Table 2 gives a comparison for three keys.

<table>
<thead>
<tr>
<th>Key</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNVS Master Key</td>
<td>User cannot set it, and cannot get the value.</td>
</tr>
<tr>
<td>SW_GP2</td>
<td>User can burn and lock it. Once burned lock bits, SW_GP2 cannot be read/wrote by software.</td>
</tr>
<tr>
<td>Software key</td>
<td>User can set register as BEE key.</td>
</tr>
</tbody>
</table>

BEE supports two AES decryption modes: ECB and CTR. For ECB mode, the plain data is divided into blocks, and each block is encrypted separately. The disadvantage of ECB mode is a lack of diffusion. Because ECB encrypts identical plaintext blocks into identical ciphertext blocks, it does not hide data patterns well. In some senses, it doesn’t provide serious message confidentiality, and it is not recommended for use in cryptographic protocols at all. Figure 2 shows a picture is encrypted using ECB mode (from https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#Electronic_Codebook_(ECB)). However, there are some corner cases where you might consider ECB mode. If you want the message to be exactly one block long and there is no statistical profile in the input, then ECB might be an acceptable choice.
CTR mode is a simple counter-based block cipher implementation. Every time a counter initiated value is encrypted and given as input to XOR with plaintext which results in ciphertext block. The counter is 128-bit for BEE engine to use, the value of counter [127:32] can be set by user counter [31:0] is the physical address.

\[ \text{The value of physical address need to be shift right 4-bit.} \]

Two individual regions can be defined for BEE engine. Independent AES key and decryption mode can be used for each region. And each region can be remapped to another address space by setting the BEE_ADDR_OFFSETx register. The remapped address \( \text{addr}_o_x \) is calculated as below.

\[
\begin{align*}
\text{addr}_o_0 &= (\text{addr}_i_0 + \text{addr}_{\text{offset}0} \ll 16) \mod (2^{32}) \\
\text{addr}_o_1 &= (\text{addr}_i_1 + \text{addr}_{\text{offset}1} \ll 16) \mod (2^{32})
\end{align*}
\]

Where, \( \text{addr}_i_x \) is the physical address and \( \text{addr}_{\text{offset}x} \) is the BEE_ADDR_OFFSETx register.

## 4 Application example

### 4.1 Introduction of example project

As shown in Figure 3, plaintext is encrypted by DCP, and the encrypted text is written to FlexSPI Flash. BEE is used for decryption on FlexSPI to get the plaintext.
The example code is based on SDK_2.7.0_EVK-MIMXRT1060 can be downloaded on https://mcuxpresso.nxp.com/. Download AN12852SW and then unzip it in the SDK project folder, \boards\evkmimxrt1060\demo_apps, as shown in Figure 4.

![Example code folder](image)

Figure 4. Example code folder

The example project can only be opened by MDK IDE. And as FlexSPI Flash is used to store encrypted text, the example code runs in RAM in debug mode.

Two regions of BEE are used. Region0 works in the AES CTR mode and Region1 works in the AES ECB mode. The `init_bee()` function in the project performs this.

![Region layout in flash](image)

Figure 5. Region layout in flash

The macro definition, `DEF_SELECT_BEE_KEY`, is defined to select different BEE key in the project. If `DEF_SELECT_BEE_KEY` is 3, running the project will program the SW_GP2 key.

```c
#define DEF_SELECT_BEE_KEY 2 // 0 -- Software key, 1-- OTPMK, 2-- SW_GP2, 3-- override SW_GP2
```

The AES key for BEE must be programmed using big-endian mode. For example, the AES key is `{0x00, 0x01, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xaa, 0xbb, 0xcc, 0xdd, 0xee, 0xff}`, and it should be programmed in eFuse as SW_GP2 following...
this order: 0xcdddeeff, 0x8899aabb, 0x44556677, 0x00112233. In this example, the programmed SW_GP2 key is (0x2b, 0x7e, 0x15, 0x16, 0x28, 0xae, 0x09, 0xf7, 0x7e, 0x02, 0x15, 0x88, 0x09, 0xcf, 0x4f, 0x3c).

```c
if(DEF_SELECT_BEE_KEY == 3){
  PRINTF("Program key to SW_GP2.\r\n");
  OCOTP_Init(OCOTP, CLOCK_GetFreq(kCLOCK_IpgClk));
  // SW_GP20
  (keyAes128[14]  <<  8) |(keyAes128[15] <<  0) );
  // SW_GP21
  (keyAes128[10]  <<  8) |(keyAes128[11] <<  0) );
  // SW_GP22
  (keyAes128[6]   <<  8) |(keyAes128[7]  <<  0) );
  // SW_GP23
  OCOTP_WriteFuseShadowRegister(OCOTP, 0x2C, (keyAes128[0]  <<  24) | (keyAes128[1]  <<  16) |
  (keyAes128[2]   <<  8) |(keyAes128[3]  <<  0) );
  while(1);
}
```

NOTE
The programmed SW_GP2 key will be valid after the system resets.

DCP is used to encrypt plain text with same key as BEE key. DCP key setting need be configured by the SetDcpAesKey() function before initialing DCP module.

4.2 Running example project

For running the example project, perform the following steps.

1. Connect a USB cable between the host PC and the OpenSDA USB port (J14) on the MIMXRT1060-EVK board.

   NOTE
   Set the boot switch (SW7) to 0b'0000 to avoid to run codes in FlexSPI Flash.

2. Open a serial terminal with the settings:

   • 115200 baud rate
   • 8-data bits
   • No parity
   • One stop bit
   • No flow control

3. Open and compile the example project with MDK IDE, and then press the DEBUG button to load the image file to RAM.

4. Press F5 on MDK IDE to run the project. The serial terminal will show the text as Figure 6.
By default, the BEE key is a software key.

Set `DEF_SELECT_BEE_KEY` to 1. The BEE key is an SNVS Master key. In this project, the SNVS Master key is not enable, and BEE gets SNVS Master key is all 0. For about how to enable SNVS Master key, see *How to use HAB secure boot in i.MXRT10xx* (document AN12681).

Set `DEF_SELECT_BEE_KEY` to 3. SW_GP2 is burned as same as software key. After the power cycle, set `DEF_SELECT_BEE_KEY` to 2. BEE key is from SW_GP2.

## 5 Conclusion

This application note introduces BEE basic knowledge and gives an example project to use BEE function. The project shows BEE function, as well as how to use DCP module to encrypt data. It is useful to protect encrypted data or code in FlexSPI Flash to avoid to get plain data.

## 6 References

- *i.MX RT1050 Processor Reference Manual* (document IMXRT1050RM)
- *Security Reference Manual for the i.MX RT1050 Processor* (document IMXRT1050SRM)
Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLK, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, Altivec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QoriQ, QoriQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, UMEMS, EdgeScale, EdgeLock, eIQ, and Immersive3D are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, μVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

© NXP B.V. 2020.

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 07/2020
Document identifier: AN12852