The EEPROM Emulation package contains the header and source files in the C programming language that enables the code from AN2302 to be used for the MC9S12XS and MC9S12P families members that have only D-flash memory for non-volatile purposes, instead of a real EEPROM.

You must read AN2302 to better understand the considerations stated in this application note. The AN3961SW implements the S12XS/P low level drivers for handling the D-Flash commands.

1 Introduction

Many applications require Electrically Erasable Programmable Read-Only Memory (EEPROM) for non-volatile data storage. EEPROM is typically characterized by the ability to erase and write individual bytes of memory many times over, with programmed locations retaining their data over an extended period when the power supply is removed.
**Introduction**

Most of MC9S12X family microcontrollers include an on-chip EEPROM, but the MC9S12XS and MC9S12P families microcontrollers include an on-board D-flash for non-volatile data storage purposes. As compared to EEPROM, the D-Flash memory operations (data storage, erase and so on) are performed using the FTM Common Command Object (CCOB) and this algorithm already stands the CCOB handling.

Another issue concerns the number of D-flash Erase and Program cycles, since some applications specify a number of program/erase cycles bigger than the D-flash reliability parameter. As MC9S12XS/P families guarantee a minimum of 50,000 program/erase cycles (consult the latest Data Sheet for the current electrical characteristics), if for some reason, an application requires a minimum guarantee of 100,000 program/erase cycles for non-volatile data, this demands an efficient algorithm to manage.

This application note provides a method to emulate an EEPROM for the MC9S12XS/P D-flash memory, based on AN2302, which implements the method to the MC9S12C32 family. This application note describes the main differences from the original implementation, such as: devices characteristics, function calls, addressing model, and flash memory commands.

### 1.1 Address Notation

All address references in this document use the S12X global address map (indicated by the G suffix). This is a 23-bit linear address format and all flash commands require an address in global format.

### 1.2 Memory

Table 1 gives an overview of the D-flash memory size available for MC9S12XS and MC9S12P families microcontrollers. Local and global memory maps are compatible across each family, but not across MC9S12XS and MC9S12P families. For devices with small memory, the unimplemented locations in the map cause a reset if accessed.

<table>
<thead>
<tr>
<th>Device</th>
<th>P-flash</th>
<th>D-flash</th>
</tr>
</thead>
<tbody>
<tr>
<td>9S12XS256</td>
<td>256 KB</td>
<td>8 KB</td>
</tr>
<tr>
<td>9S12XS128</td>
<td>128 KB</td>
<td>8 KB</td>
</tr>
<tr>
<td>9S12XS64</td>
<td>64 KB</td>
<td>4 KB</td>
</tr>
<tr>
<td>9S12P128</td>
<td>128 KB</td>
<td>4 KB</td>
</tr>
</tbody>
</table>

General features for both P-flash and D-flash are:

- No external high-voltage power supply required for flash memory program and erase operations
- Interrupt generation on flash command completion and flash error detection
- Security mechanism to prevent unauthorized access to the flash memory

### 1.3 P-Flash

The P-flash memory constitutes the main nonvolatile memory for applications. The P-flash memory is compatible for all flash devices in the MC9S12XS family for all memory sizes in terms of functionality for program, erase, security, and protection setup. All family members have a single physical flash block.
that is equivalent to the defined flash size. The P-flash is a flash block with Error Correction Coding (ECC) intended for storing NVM data.

The MC9S12XS/P P-flash features include:

- Automated program and erase algorithm with verification and generation of ECC parity bits
- Fast sector erase and phrase program operation
- Flexible protection scheme to prevent accidental program or erase of P-flash memory
- Ability to read the P-flash memory while programming a word in the D-flash memory
- Single bit fault correction and double bit fault detection within a 64-bit phrase for MC9S12XS and 32-bit double word for MC9S12P, during read operations
- P-flash sector of 1024 bytes for S12XS and 512 bytes for S12P, which is the smallest portion of the P-flash memory that can be erased
- Flash memory block composition is dependent on the MCU (S12XS or S12P) device and the memory size. See Table 2.

<table>
<thead>
<tr>
<th>Device</th>
<th>P-flash</th>
<th>Flash block</th>
</tr>
</thead>
<tbody>
<tr>
<td>9S12XS256</td>
<td>256 KB</td>
<td>256 sectors of 1024 bytes</td>
</tr>
<tr>
<td>9S12XS128</td>
<td>128 KB</td>
<td>128 sectors of 1024 bytes</td>
</tr>
<tr>
<td>9S12XS64</td>
<td>64 KB</td>
<td>64 sectors of 1024 bytes</td>
</tr>
<tr>
<td>9S12P128</td>
<td>128 KB</td>
<td>256 sectors of 512 bytes</td>
</tr>
</tbody>
</table>

1.4 D-Flash

D-flash will be compatible in terms of flash commands for all MC9S12XS/P families. S12XS256 and S12XS128 have an equivalent implemented amount of 8 KB and 9S12XS64 and S12P have an equivalent amount of 4 KB D-flash. ROM devices do not implement D-flash. The global memory map starts from a lower order address to a higher order address and shows the difference between MC9S12XS/P and S9S12P families. See Table 3. The D-flash is a separate flash block with ECC intended for storing NVM data.

The MC9S12XS and MC9S12P D-flash features include:

- 4 KB of D-flash memory is composed of one 4 KB Flash block divided into 16 sectors of 256 bytes
- Single bit fault correction and double bit fault detection within a word during read operations
- Automated program and erase algorithm with verification and generation of ECC parity bits
- Fast sector erase and word program operation
- Protection scheme to prevent accidental program or erase of D-glash memory
- Ability to program up to four words in a burst sequence
The two main differences from the proposed method in AN2302 are:

- Addressing model. The AN2302 implementation uses a local addressing model whilst the MC9S12XS/P uses a global addressing model, to get the whole D-flash memory size available for emulation.
- Flash programming code execution, since the AN2302 implements the P-Flash programming code from RAM, due to the fact that the MC9S12C device does not have a Data Flash as MC9S12XS/P.

### 2.1 List of Differences

The following items identify the whole set of differences and the same document sequence from AN2302.

### 2.2 Introduction

- As the interrupt vectors normally reside in flash memory and it is not possible to read from a flash block while any command is executing on that specific flash, an alternative approach is required for S12C32, since this device features only one flash block. As compared to S12C32, the S9S12XS/P devices feature two separated flash blocks: P-flash where the interrupt vectors and

---

### Table 3. D-flash addressing

<table>
<thead>
<tr>
<th>EPAGE</th>
<th>Global Address</th>
<th>9S12XS256 9S12XS128</th>
<th>9S12XS256 9S12XS128</th>
<th>9S12P</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>0x10_0000</td>
<td>8 KB</td>
<td>4 KB</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_03FF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x01</td>
<td>0x10_0400</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_07FF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x02</td>
<td>0x10_0800</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_0BFF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x03</td>
<td>0x10_0C00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_0FFF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x04</td>
<td>0x10_1000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_13FF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x05</td>
<td>0x10_1400</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_17FF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x06</td>
<td>0x10_1800</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_1BFF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x07</td>
<td>0x10_1C00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0x10_1FFF</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NA</td>
<td>0x00_4400</td>
<td></td>
<td></td>
<td>4 KB</td>
</tr>
<tr>
<td></td>
<td>0x00_53FF</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The Flash implementation on the MC9S12XS/P is programmed one word (two bytes) at a time and erased in 256 byte sectors. Refer to the S12XS Flash 64K, 128K, or 256K Block Guide (S12XFTMR64K1V1, S12XFTMR128K1V1, and S12XFTMR256K1V1, respectively), and for S12P Flash 128K Block Guide (S12FTMRC128K1V1) for further details.

There are no constraints regarding the flash programming code being executed from the P-flash, since the emulated EEPROM is located in the D-flash, which eliminates the usage of code execution in RAM as in AN2302.

The MC9S12XS32 electrical characteristics at the time of writing, guarantee a minimum of 50,000 program/erase cycles per sector of D-flash. Refer to the latest Data Sheet for the current electrical characteristics at www.freescale.com.

Valid bank sizes are 256, 128, 64, 32, 16, 8, 4, and 2 bytes.

Program/Erase Time: The time base for all NVM program or erase operations is derived from the oscillator. A minimum oscillator frequency ($f_{NVMOSC}$) is required for performing program or erase operations. The value of the flash clock prescaler (by configuring the FCLKDIV register) must be chosen. Thus, the frequency of the flash clock ($f_{NVMOOP}$) shall be within the range of 800 kHz to 1050 kHz. The $f_{NVMBUS}$ is defined as the bus frequency for programming or erase operations. $N_w$ parameter defines the number of words to be programmed or erased at D-flash. The maximum programming time can be calculated using Equation 1 for S12XS and Equation 2 for S12P.

$$t_{dpem} = \left(15 \times \left(56 \times N_w\right) + \left(16 \times BC\right) \times \frac{1}{f_{NVMOOP}}\right) + \left(460 \times \left(840 \times N_w\right) + \left(500 \times BC\right) \times \frac{1}{f_{NVMBUS}}\right)$$

Equation 1
Main Differences From AN2302 (S12C32)

$$t_{dpgm} \approx \left(14 + (54 \times N_w) + (14 \times BC) \times \frac{1}{f_{NVMOP}}\right) + \left(500 + (750 \times N_w) + (100 \times BC) \times \frac{1}{f_{NVMBUS}}\right)$$  \hspace{1cm} Eqn. 2

- Maximum values for $t_{dpgm}$ are slightly under 104 μs, depending on the selected values for the oscillator frequency and the MCU internal bus frequency. Whenever a non-volatile data variable is updated, an entire bank is programmed. Therefore, if a bank is defined to be 32 words, the programming time will be approximately $104 \times 32 = 3.3$ ms.

- The time required by the flash state machine to erase a sector is defined by Equation 3 for S12XS and Equation 4 for S12P:

$$t_{eradf} \approx 20100 \times \frac{1}{f_{NVMOP}} + 3300 \times \frac{1}{f_{NVMBUS}}$$  \hspace{1cm} Eqn. 3

$$t_{eradf} \approx 20100 \times \frac{1}{f_{NVMOP}} + 3400 \times \frac{1}{f_{NVMBUS}}$$  \hspace{1cm} Eqn. 4

- The D-flash sector erase time on a new device is ~5 ms and can be extended to 20 ms as the flash is cycled.

### 2.2.2 Interrupts

- Differently from the AN2302 implementation, the emulated EEPROM for MC9S12XS/P does not require Interrupt vectors re-mapping to RAM, since it is possible to read from a flash block while a command is executed on a different flash block.
2.2.3 Callback Function

- The MC9S12XS/P low level functions provide a callback function, which enables, for the time critical events, the possibility to be serviced during the D-flash operations. Servicing watchdog timers is one such critical events. If it is not necessary to provide the callback service, you can disable it by setting the macro CALLBACK_ENABLE to FALSE.
- The callback function during program/erase is located inside D-flash low level driver (EED_MiddleLevel.c module).

2.2.4 Memory Map

- Figure 1 illustrates the default memory map of the MC9S12XS with a typical allocation of resources for a non-volatile data storage in the D-flash memory.

![Memory Map Diagram]

**NOTE**

Figure 1 illustrates an example with two sectors of the D-flash allocated for EEPROM Emulation, with 8 banks of 64 bytes of non-volatile data storage.

All program code and interrupts run in the P-flash memory, in other words, an interrupt may be serviced meanwhile a D-flash command is executed.
2.2.5 Software Description

The code was written in the C language using the Freescale Codewarrior for HCS12 tools, v4.7, but it is also compatible to Cosmic Compiler, selected by the COMPILER_SELECT preprocessor definition inside SSD_SGF18.h module.

This section states the differences from AN2302 to the implementation for MC9S12XS/P, considering the reference module changes and new functions.

2.2.5.1 Reference Changes from AN2302

- **API's Calls in main.c:**
  - Non-Volatile Data Variables: Same as AN2302, however the addresses are different between HC9S12C and MC9S12XS/P. To get the whole D-flash range available for EEPROM Emulation purposes, the implementation for MC9S12XS/P uses global addressing.
  - **EE_Emulation.h**: Modified the user’s application header name from the reference EE_Emulation.h to EE_AN_Emulation.h. Created a new ee_emulation.h file that corresponds to the MC9S12XS low level driver header.
    - **IRQ_DURING_PROG**: Removed from AN2302 reference, because it is no longer necessary to concern about enabling an interrupt services option, to re-allocate the interrupt routines to RAM during the programming and erasing of flash. This condition is enhanced in MC9S12XS/P, since it is possible to service an interrupt meanwhile a D-flash command is being executed.
    - **EEPROM_SIZE_BYTES**: As the MC9S12XS/P D-flash sector is reduced to 256 bytes from 512 of HC9S12C, the valid values for EEPROM_SIZE_BYTES are: 2, 6, 14, 30, 62, 126, 254 and \((m \times 512) - 2\), where \(m = 1, 2, 3...\)
    - **EEPROM_BANKS**: Due to the same reason as stated before, if EEPROM_SIZE_BYTES is smaller than 254, the total amount of flash allocated must be two or more complete flash sectors, for example \(\text{EPROM_BANKS} \times (\text{EEPROM_SIZE_BYTES} + 2) = n \times 256\), where permitted values of \(n\) are 2, 3, 4...
    - If EEPROM_SIZE_BYTES is greater or equal to 254, the total amount of flash allocated must be two or more banks, in other words \(\text{EEPROM_BANKS} = n\), where permitted values of \(n\) are 2, 3, 4...
    - **EEPROM_START**: loaded with EED_FLASH_START_ADDRESS definition from ee_emulation.h, since the D-flash start address is different for S12XS/P.
    - **FLASH_COPY_START**: Removed.
    - **RAM_FUNCS_START**: Removed.
    - **RAM_FUNCS_SIZE**: Removed.
    - **VT_START**: Removed.
    - **VT_SIZE**: Removed.
- **EE_Emulation.c**: The file EE_Emulation.c contains the functions required for EEPROM emulation of non-volatile data variables that are executed from flash. All functions in this file are located in the DEFAULT code segment. The user should not modify this file.
— ReadEeprom: Modified the addressing mode of the srcAddr parameter.
  – Parameters: The srcAddr pointer to the non-volatile data variable to be read. Included the __far keyword to enable the global addressing of a variable.

— WriteEeprom: The elapsed time to program a new bank will be (EEPROM_SIZE_BYTES + 2) × 52 μs. Modified the addressing mode of the destAddr parameter.
  – Parameters: The destAddr pointer to the non-volatile data variable to be updated. Included the __far keyword to enable the global addressing of a variable.

— InitEeprom: The flash prescaler initialization is performed by calling the low-level driver function FlashInit.

— InitRAMFuncs: Removed.

— CopyToRAM: Removed.

— ProgEepromWord: Modified the addressing mode of the progAddr parameter.
  – Parameters: The progAddr pointer to a pointer to the flash location to be programmed. Included the __far keyword to enable the global addressing of a pointer.

— EraseEepromBank: Modified the addressing mode of the eepromAddr parameter.
  – Parameters: eepromAddr address of the (first) flash sector(s) to be erased. Included the __far keyword to enable the global addressing.

— MaskInterrupts: Removed.

— RestoreCCR: Removed.

• EE_RAMFuncs.h: Removed.
• EE_RAMFuncs.c: Removed.
• VectorTable.c: Removed.
• Start12.c: Not necessary to re-map the RAM to the top of the memory map.
• mcucfg.h: Removed. The OSCCLK_FREQ_KHZ and BUS CLOCK FREQUENCY are defined in SSD_SGF18.h.
• mottypes.h: Removed. The type definitions are defined in SSD_Types.h.
• s12_fectl.h: Removed.
• s12_stdtimer.h: Removed.

2.2.5.2 New Modules

The listed modules and functions below represent a sort from Freescale S12XS and S12P EEPROM Emulation Driver, which provides a set of high-speed read/write operations for emulated EEPROM.

• FlashInit.c: The function FlashInit() checks and initializes the flash module for EEPROM emulation. It will check the enabling state of the flash module, initialize the flash clock, and initialize the flash status register. SGF_OK will be returned if everything is OK. For other cases, a bit-mapped return code combined with the flash status will be returned to user application.
  — Function: FlashInit
  — Prototype: UINT16 FlashInit(void)
Main Differences From AN2302 (S12C32)

— Parameters: void.
— Return:
  – Successful completion: SGF_OK
  – Error value: SGF_ERR_INVALIDCLK

• **DFlashErase.c**: The function DFlashErase() is used to perform sector erase operation on D-flash.
  — Function: DFlashErase
  — Prototype: UINT16 DFlashErase(UINT32 destination, UINT32 size)
  — Parameters:
    – destination: Start address for the D-flash erase operation.
    – size: Size to be erased in bytes.
  — Return:
    – Successful completion: SGF_OK
    – Error value: SGF_ERR_ACCERR, SGF_ERR_SIZE, SGF_ERR_RANGE, SGF_ERR_MGSTAT0, SGF_ERR_MGSTAT1

• **DFlashProgram.c**: The function DFlashProgram() is used to perform program operation on D-flash.
  — Function: DFlashProgram
  — Prototype: UINT16 DFlashProgram(UINT32 destination, UINT32 size, UINT32 source)
  — Parameters:
    – destination: Start address for the D-flash program operation.
    – size: Size to be programmed in bytes.
    – source: Address of the data to be programmed.
  — Return:
    – Successful completion: SGF_OK
    – Error value: SGF_ERR_ACCERR, SGF_ERR_SIZE, SGF_ERR_RANGE, SGF_ERR_MGSTAT0, SGF_ERR_MGSTAT1

• **FlashCommandSequence.c**: The function FlashCommandSequence() is used to perform command write sequence on the flash blocks.
  — Function: FlashCommandSequence
  — Prototype: UINT16 FlashCommandSequence(UINT8 index)
  — Parameters:
    – index: Maximum index value for the FCCOBIIX register for a particular command.
  — Return:
    – Successful completion: SGF_OK
    – Error value: SGF_ERR_ACCERR, SGF_ERR_PVIOL, SGF_ERR_MGSTAT0, SGF_ERR_MGSTAT1

• **EED_MiddleLevel.c**: This module provides an individual functionality to support the operation of the EEPROM Emulation Driver. This module implements the FSL_Program function, which will
program specified data to the destination address in flash and verify the data. It encapsulates two low-level SSD functions: 'DFlashProgram()' and FlashProgramVerify()'.

- Input parameters are used directly without any check.
  — Function: FSL_Program
  — Prototype: UINT16 FSL_Program (UINT32 destination, UINT16 size, UINT16 sourceData)
  — Parameters:
    - destination: Start address for the D-flash Program operation.
    - size: Size to be programmed in bytes.
    - source: The source data.
  — Return:
    - Successful initialization: SGF_OK
    - Source data and the corresponding destination data do not match: EED_ERROR_VERIFY

- SSD_SGF18.h: The HCS12XS/P EEPROM Emulation Driver has some configurations macros. These macros are placed in the SSD_SGF18.h. Depending upon the macro selected, there will be change in some internal macros and source code.
  — Derivative Selection: Based on the value assigned to the macro 'SGF18_SELECT,' the other macros will be given the corresponding values and some parts of the code will be conditionally compiled:
    - If the code is used for S12XS64 family then, define 'SGF18_SELECT' as 'S12XS_64.'
    - If the code is used for S12XS128 or S12XS256 family then, define 'SGF18_SELECT' as 'S12XS_128_256.'
    - If the code is used for S12P family then, define 'SGF18_SELECT' as 'S12P.'
  — Other Configuration Macros: Table 4 gives an overview of configuration macros used by the low level functions.

<table>
<thead>
<tr>
<th>Macro Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSD_MCU_REGISTER_BASE</td>
<td>Base address of MCU register block</td>
</tr>
<tr>
<td>SSD_BUS_CLOCK</td>
<td>Bus clock. The unit is 10 KHz</td>
</tr>
<tr>
<td>SSD_OSCILLATOR_CLOCK</td>
<td>Oscillator clock. The unit is 10 KHz</td>
</tr>
</tbody>
</table>

Table 5 describes all error returns used in the modules above.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EED_OK</td>
<td>0x0000</td>
<td>The requested operation was successful</td>
</tr>
<tr>
<td>EED_ERROR_NOT_BLANK</td>
<td>0x0001</td>
<td>The flash memories are not blank</td>
</tr>
<tr>
<td>EED_ERROR_SIZE</td>
<td>0x0002</td>
<td>Size is not word aligned</td>
</tr>
<tr>
<td>EED_ERROR_NOFND</td>
<td>0x0003</td>
<td>Record not found</td>
</tr>
<tr>
<td>EED_ERROR_RANGE</td>
<td>0x0004</td>
<td>Size or destination or end address for program exceeds the valid range</td>
</tr>
</tbody>
</table>

EEPROM Emulation for the MC9S12XS and MC9S12P Families Using AN2302 as a Reference, Rev. 0
Table 5. D-flash addressing (continued)

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EED_ERROR_SSTAT</td>
<td>0x0005</td>
<td>Sector status error</td>
</tr>
<tr>
<td>EED_ERROR_VERIFY</td>
<td>0x0006</td>
<td>Corresponding source data and content of destination location mismatch.</td>
</tr>
<tr>
<td>EED_ERROR_IDRNG</td>
<td>0x0007</td>
<td>Record identifier exceeds the valid range</td>
</tr>
<tr>
<td>EED_ERROR_ACCERR</td>
<td>0x0008</td>
<td>Access error flag is set while operating the flash.</td>
</tr>
<tr>
<td>EED_ERROR_PVIOL</td>
<td>0x0010</td>
<td>Protection violation flag is set while operating the flash.</td>
</tr>
<tr>
<td>EED_ERROR_MGSTAT0</td>
<td>0x0020</td>
<td>Non-correctable errors have been encountered during the verify operation. (Hardware Error)</td>
</tr>
<tr>
<td>EED_ERROR_MGSTAT1</td>
<td>0x0040</td>
<td>Errors have been encountered during the verify operation. (Hardware Error)</td>
</tr>
<tr>
<td>EED_ERROR_INVALIDCLK</td>
<td>0x0800</td>
<td>The clock divider value is not correct.</td>
</tr>
</tbody>
</table>

2.2.6 Software Example

A Freescale CodeWarrior example project is available for download with this document, modify the code to run with MC9S12XS/P and implement the whole list of changes through this document.

3 References

- Application note titled *EEPROM Emulation for the MC9S12C32* (document AN2302)
- S12XS and S12P EEPROM Emulation Driver User Manual