NXP Semiconductors Document Number: AN5338

Application Note Rev. 0, 09/2016

How to use FlexMemory as D-Flash and EEPROM in KE1xF

1. Introduction

The FlexMemory (FlexNVM and FlexRAM) is available on NXP's Kinetis KE1xF family. The FlexMemory allows the user to configure the FlexNVM block as either basic D-Flash (data flash), Enhanced EEPROM(EEE), or a combination of both.

D-Flash memory is useful for applications that need to quickly store large amounts of data or store data that is static. The EEPROM feature is widely used in applications that store small amount of rapidly changing data required to be saved during system power off.

This application note describes the features of FlexMemory and how to implement it as D-flash and Enhanced EEPROM(EEE). The demo example projects of this two usage are included in SDK2.0 KE1xF release packages.

Contents

1. Introduction 1
2. FlexMemory Features 2
   2.1 FlexMemory components 2
   2.2 FlexNVM features 2
   2.3 FlexRAM features 3
   2.4 FlexMemory partition 3
3. D-Flash Implementation 6
4. EEPROM Implementation 6
   4.1 EEE startup 7
   4.2 Read and write the EEE 7
5. Demo Examples 8
   5.1. Flexnvm_dflash demo 9
   5.2. Flexnvm_eeprom demo 12
6. Conclusion 15
7. Revision History 15

2. FlexMemory Features

2.1 FlexMemory components

Figure 1 shows the flash memory blocks and the FlexMemory components on KE1xF family device. Flash memory blocks include Program flash and FlexNVM; FlexMemory blocks are composed of FlexNVM and FlexRAM, and Enhanced EEPROM (EEE) is comprised of EEPROM backup in FlexNVM, FlexRAM and EEE state machine. FlexNVM and FlexRAM are the only memories used for Enhanced EEPROM implementation.

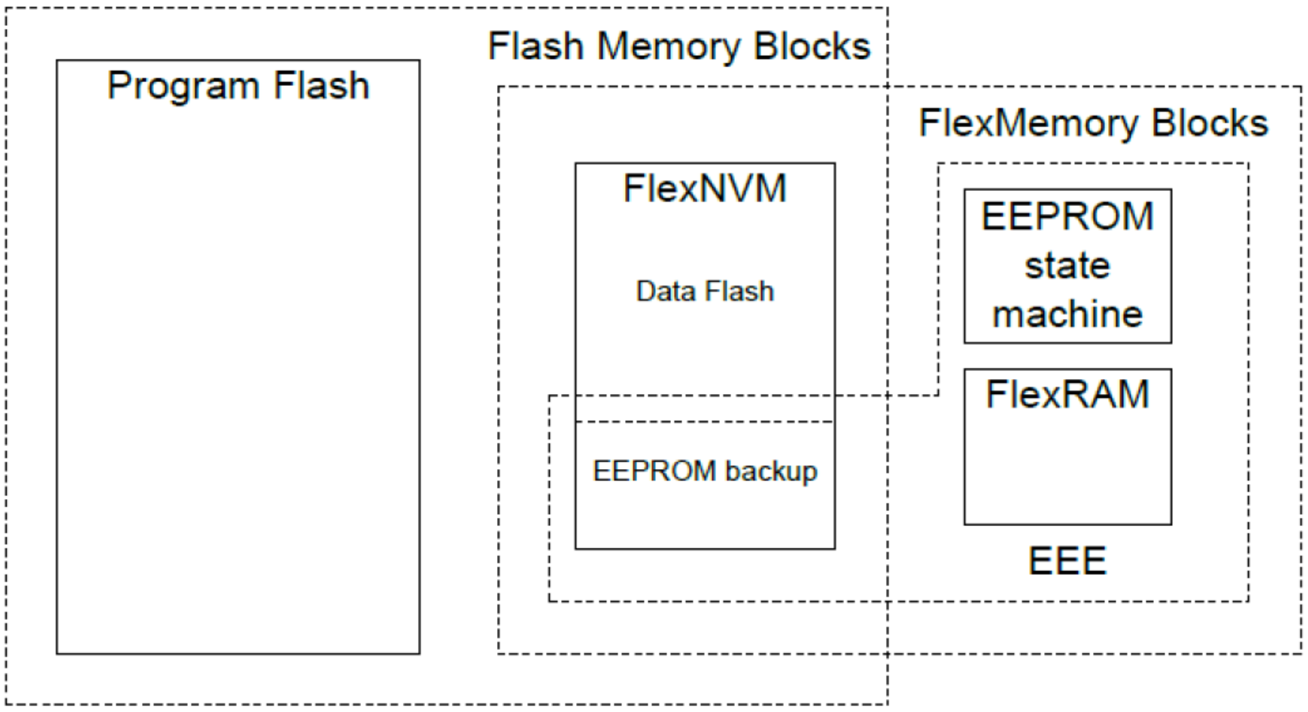


Figure 1. Block diagram of Flash Memory blocks and FlexMemory components

2.2 FlexNVM features

When FlexNVM is partitioned for data flash memory:

- Sector size of 2 KB
- Protection scheme prevents accidental program or erase of stored data
- Automated, built-in program and erase algorithms with verify
- Section programming for faster bulk programming times
- Read access to the data flash block possible while programming or erasing data in the program flash block.

2.3 FlexRAM features

- Memory that can be used as traditional RAM or as high-endurance EEPROM storage
- 4 KB of FlexRAM configured for EEPROM or traditional RAM operations
- When configured for EEPROM:
  - Protection scheme prevents accidental program or erase of data written for EEPROM
  - Built-in hardware emulation scheme to automate EEPROM record maintenance functions
  - Programmable EEPROM data set size and FlexNVM partition code facilitating EEPROM memory endurance trade-offs
  - Supports FlexRAM aligned writes of 1, 2, or 4 bytes at a time
  - Read access to FlexRAM possible while programming or erasing data in the program or data flash memory
- When configured for traditional RAM:
  - Read and write access possible to the FlexRAM while programming or erasing data in the program or data flash memory.

2.4 FlexMemory partition

The user can configure the FlexNVM block as either:

- Basic data flash,
- EEPROM flash records to support the built-in EEPROM feature, or
- A combination of both.

The user's FlexNVM configuration choice is specified using the Program Partition command. The partitioning process tells the EEE state machine how much EEPROM memory will be used and how much the FlexNVM will be used to back up the EEPROM.

To handle varying customer requirements, the FlexRAM and FlexNVM blocks can be split into partitions as shown in the Figure 2 below.

1. EEPROM partition (EEESIZE) — The amount of FlexRAM used for EEPROM can be set from 0 bytes (no EEPROM) to the maximum FlexRAM size (4KB in KE1xF). The remainder of the FlexRAM not used for EEPROM is not accessible while the FlexRAM is configured for EEPROM.The EEPROM partition grows upward from the bottom of the FlexRAM address(0x1400_0000 in KE1xF) space.
2. Data flash partition (DEPART) — The amount of FlexNVM memory used for data flash can be programmed from 0 bytes (all of the FlexNVM block is available for EEPROM backup) to the maximum size of the FlexNVM block (64 KB in KE1xF).
3. FlexNVM EEPROM backup partition — The amount of FlexNVM memory used for EEPROM backup, which is equal to the FlexNVM block total size minus the data flash memory partition size. The EEPROM backup size must be at least 16 times the EEPROM partition size in FlexRAM.

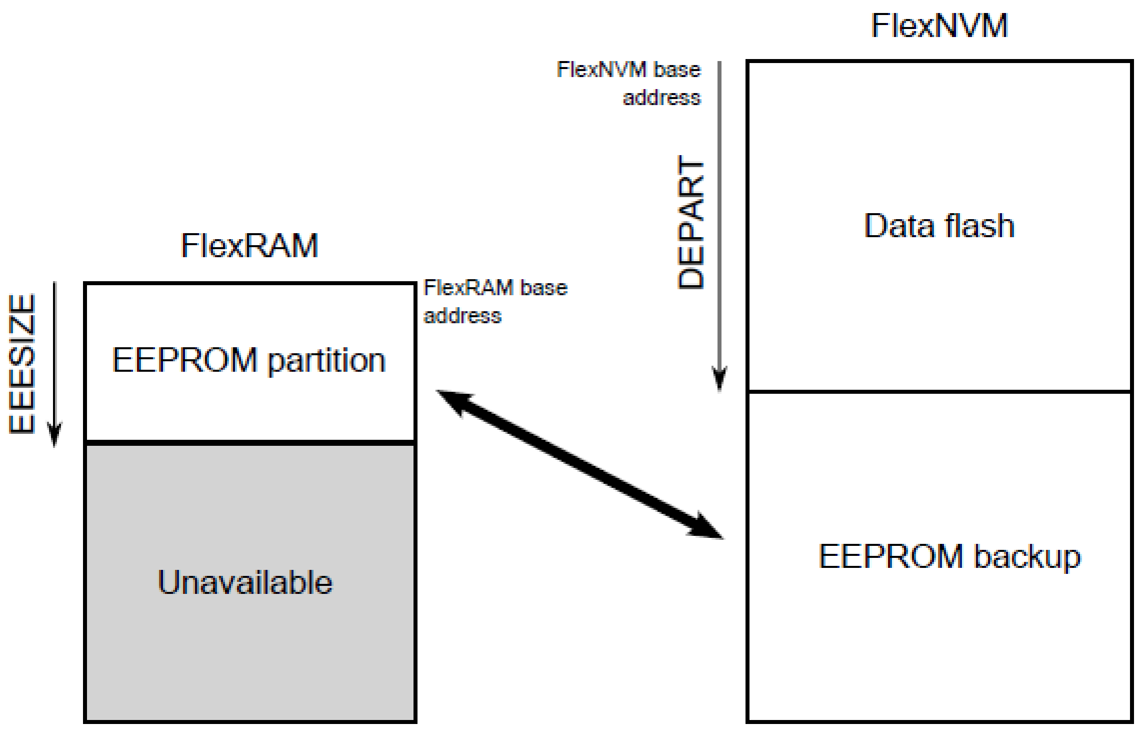


Figure 2. FlexMemory Partition Diagram

2.4.1 Program Partition command

The Program Partition command, as shown in the following table, prepares the FlexNVM block for use as data flash, EEPROM backup, or a combination of both, and initializes the FlexRAM size. The Program Partition command must not be launched from flash memory, since flash memory resources are not accessible during Program Partition command execution.

Table 1. Program Partition Command FCCOB Requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x80 (PGMPART)
1	Not Used
2	Not Used
3	FlexRAM load during reset option (only bit 0 used): 0 - FlexRAM loaded with valid EEPROM data during reset sequence 1 - FlexRAM not loaded during reset sequence
4	EEPROM Data Set Size Code Options: 0,32,64,128,256,512,1024,2048,4096 Bytes
5	FlexNVM Partition Code Options: 0,32,48,64 KBytes

2.4.2 Set FlexRAM Function command

The Set FlexRAM Function command, as shown in the following table, changes the function of the FlexRAM:

- When not partitioned for EEPROM, the FlexRAM is typically used as traditional RAM
- When partitioned for EEPROM, the FlexRAM is typically used to store EEPROM data.

Table 2. Set FlexRAM Function command FCCOB requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x81 (SETRAM)
1	FlexRAM Function Control Code Options: 0xFF: Make FlexRAM available as RAM 0x00: Make FlexRAM available for EEPROM

2.4.3 Read Resource command

The Read Resource command, as shown in the following table, is provided for the user to read data from special-purpose memory resources located within the Flash module. The special-purpose memory resources available include program flash IFR, data flash IFR space, and the Version ID field.

For the FlexMemory usage, the data flash IFR space is needed, as shown in the following table.

Table 3. Read Resource command FCCOB requirements

FCCOB Number	FCCOB Contents [7:0]
0	0x03 (RDRSRC)
1	Flash address [23:16]
2	Flash address [15:8]
3	Flash address [7:0]
4	Resource select code Options: 0x00: Program Flash 0 IFR 0x00: Data Flash 0 IFR 0x01: Version ID
Returned values	
4	Read Data [64:56]
5	Read Data [55:48]
6	Read Data [47:40]
7	Read Data [39:32]
8	Read Data [31:24]
9	Read Data [23:16]
A	Read Data [15:8]
B	Read Data [7:0]

The data flash 0 IFR is a 1 KB nonvolatile information memory that can be read and erased, however the user has limited program capabilities in the data flash 0 IFR (Program Partition command, Erase All Blocks Command and Read Resource Command). The contents of the data flash 0 IFR are summarized in the following table. The data flash 0 IFR is located within the data flash 0 memory block.

Table 4. Data flash IFR

Address Range (offset address)	Size (Bytes)	Field Description
0x00 – 0x3FB, 0x3FE – 0x3FF	1022	Reserved
0x3FD	1	EEPROM Data Set Size Code
0x3FC	1	FlexNVM Partition Code

3. D-Flash Implementation

Figure 3 shows how the memory blocks function, if the Enhanced EEPROM(EEE) functionality is disabled, which means all the FlexNVM is used as D-Flash and FlexRAM is used as traditional RAM.

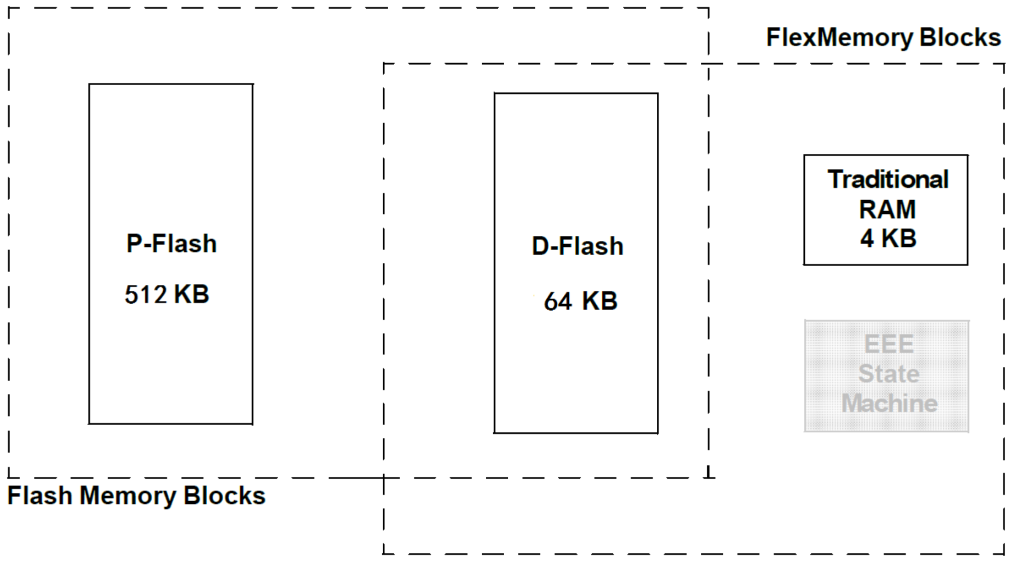


Figure 3. FlexMemory used as D-Flash

The P-Flash block is always a P-Flash. Its functionality does not change for any FlexMemory configuration. Because the EEE functionality is not used in this case, the entire FlexNVM is allocated as D-Flash space, and no E-Flash (EEPROM backup) is needed. The FlexRAM becomes a 4 KB traditional RAM. This means it can be used as extra memory space, however it is important to note that it runs at the flash clock speed and not the core speed. The EEE state machine is present in the device, but not active. When using as D-Flash, the read, write, and program operation is the same as P-Flash block, the FCCOB command sequence is also the same. User can address bit 23 to select between program flash memory (=0) and data flash memory (=1).

D-Flash memory is useful for applications that need to quickly store large amounts of data or store data that is static.

4. EEPROM Implementation

Figure 4 shows how the memory blocks function when the Enhanced EEPROM(EEE) functionality is enabled and the entire FlexNVM is used to back up the EEE data.

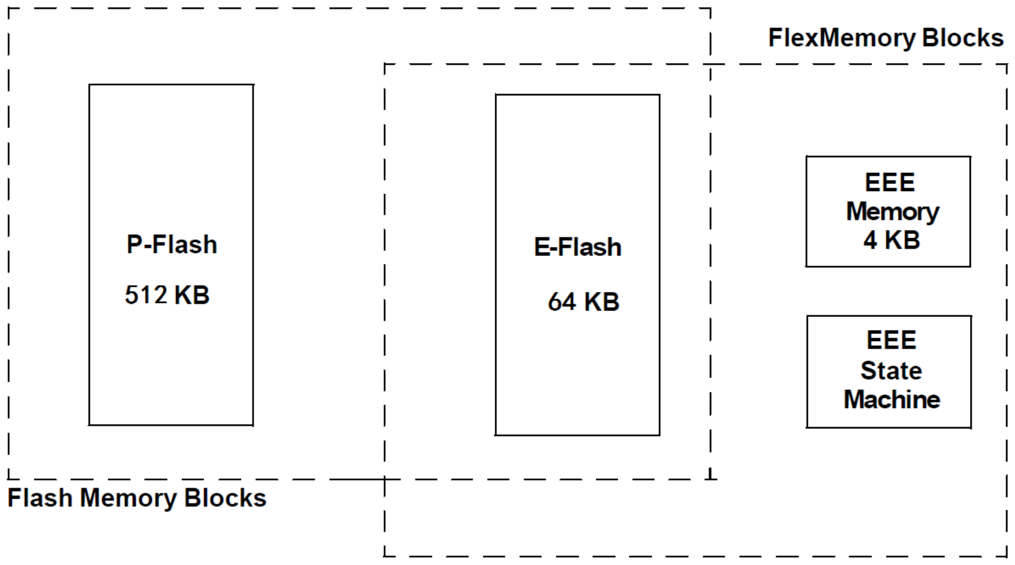


Figure 4. FlexMemory used as EEPROM(Entire FlexNVM used to backup EEE data)

There are a number of configuration options that can be used when the Enhanced EEPROM(EEE) functionality is enabled. The FlexNVM can also be used as a mix of D-Flash and E-Flash (EEPROM backup). Figure 4 shows an example of where the entire FlexNVM is used as E-Flash (EEPROM backup) memory. The FlexRAM becomes the EEE memory space 4 KB. Any reads and writes of the EEE data uses this 4 KB memory space because the E-Flash is not directly accessible. The EEE state machine automatically manages all of the writes to the EEE memory space, and generates flash program and erase operations as needed into the E-Flash.

4.1 EEE startup

After a reset the EEE configuration options written during the partitioning process are automatically loaded. If the EEE is enabled, then the state machine loads the FlexRAM with EEE data from the E-Flash during the system boot. The amount of time needed to copy data from the E-Flash into FlexRAM can vary depending on the configured size of the EEE and the amount of backup E-Flash that needs to be parsed. The FTFE_FCNFG[EEERDY] flag is cleared until the load of the EEE data is complete, therefore software must wait for the EEERDY flag to set before attempting to access the EEE data in the FlexRAM. If an interrupt driven option is needed instead of software polling, then the CCIF interrupt could be used instead of polling EEERDY.

4.2 Read and write the EEE

The EEE data is read and written by accessing the FlexRAM address space. The EEE space is allocated starting at the beginning of the FlexRAM. The addressable space is the FlexRAM base address (0x1400_0000 in KE1xF) up to the programmed EEE size. Any space in the FlexRAM that is not used as EEE must not be accessed while the EEE functionality is enabled. For example, if the EEE size is configured as 32 bytes total, then read or write accesses to any addresses between 0x1400_0000 and 0x1400_001F are allowed, but accesses 0x1400_0020 to 0x17FF_FFFF generates a bus error.

Because the EEE data is accessed through a RAM, the data is readable and writable at any size, byte, word, or longword. Although any access size is possible, the records used to back up the EEE data use a word sized data field. This means that byte writes are possible, but they make less efficient use of the E-Flash.

4.2.1 EEE writes

Writes to the EEE space launch a EEE operation to store the data within the E-Flash memory. Because this is a flash program operation, software must test the CCIF bit to determine if any other flash operations are in progress before writing to the EEE space. Because multiple concurrent writes and read-while-write operations within the same flash block are not allowable, accesses to the EEE or D-Flash space are not allowed until the EEE write is complete.

4.2.2 EEE reads

When the EEE is read, since the data is supplied by the FlexRAM, no flash operations are triggered. However, EEE reads are not allowed while a EEE write is in progress. Software must either test the EEERDY bit before read operations or wait for EEERDY after a write access before allowing software to continue. In many cases it is most efficient for software to test EEERDY (or CCIF) both before and after writes and block other EEE operations until EEERDY sets after the write. This way a special function is needed for EEE writes, but a EEE read does not require any special software. Another advantage to this approach is that no additional delay or flag checking is required if you have multiple EEE read accesses with no EEE write cycles in between.

A special case for a EEE read that must be considered is the first access to the EEE after a reset. For the first read of the EEE after reset, the EEERDY bit may need to be tested to make sure that the state machine has completed the initial load of data from the E-flash to the FlexRAM. If the system start-up time is long, this guarantees that the initial data load has time to complete before the first EEE read, then a test of the EEERDY flag before the first read may not be required. However, it is safer to explicitly test the EEERDY bit before the first read access to the EEE.

5. Demo Examples

There are two demo projects under flash driver example folder included in SDK2.0 KE1xF release package. One is “flexnvm_dflash” demo, it shows how to use flexmemory and SDK flash drivers to operate as D-Flash.The other is “flexnvm_eeprom” demo shows the EEPROM usage. Figure 5 shows the software flowchart of “flexnvm_dflash” project. Figure 8 shows the software flowchart of “flexnvm_eeprom” project. After download and execute code in flash, we can use J-link commander tool to check the memory read, write and programming status.

5.1. Flexnvm_dflash demo

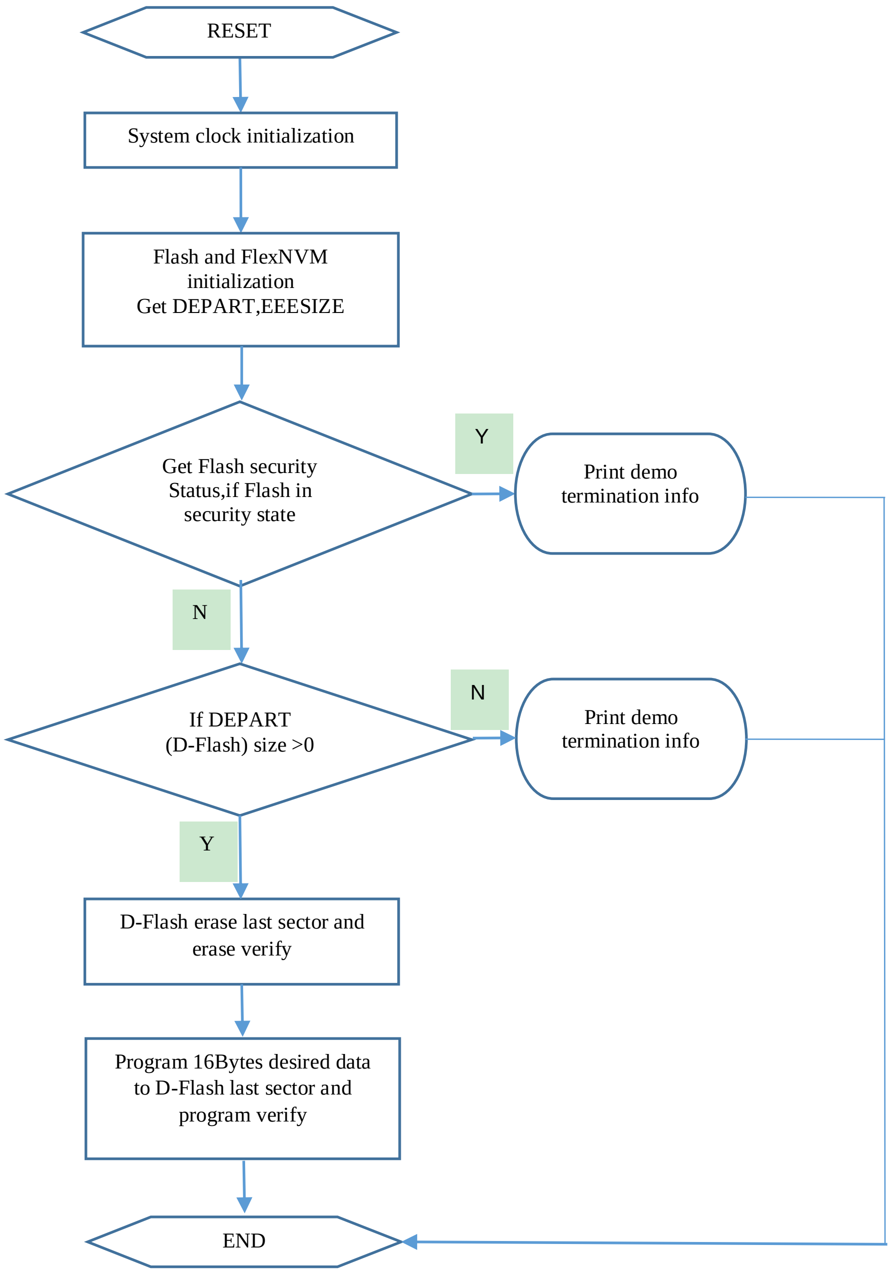
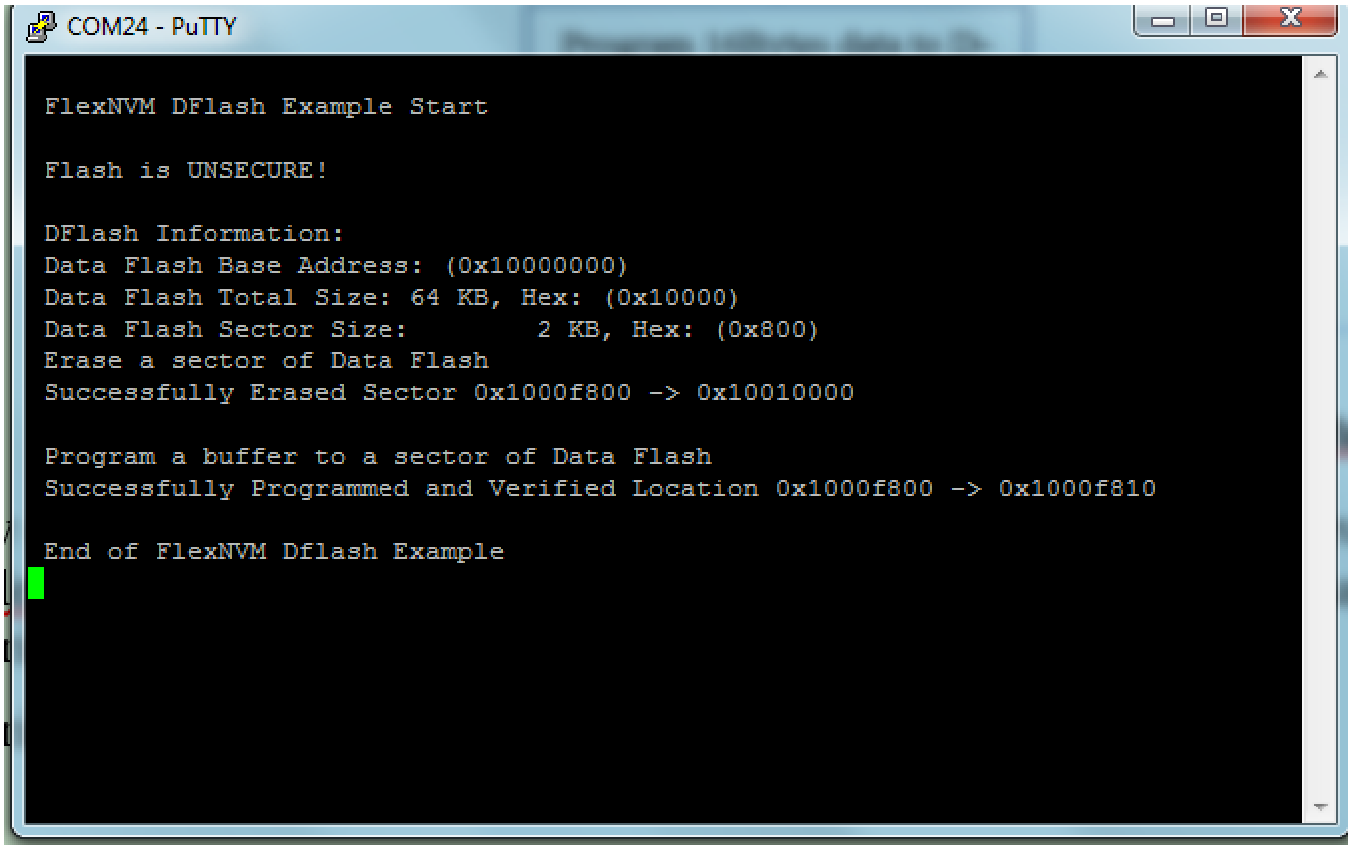

RESET
System clock initialization
Flash and FlexNVM initialization Get DEPART,EEESIZE
Get Flash security Status,if Flash in security state
Y
Print demo termination info
N
If DEPART (D-Flash) size >0
N
Print demo termination info
Y
D-Flash erase last sector and erase verify
Program 16Bytes desired data to D-Flash last sector and program verify
END

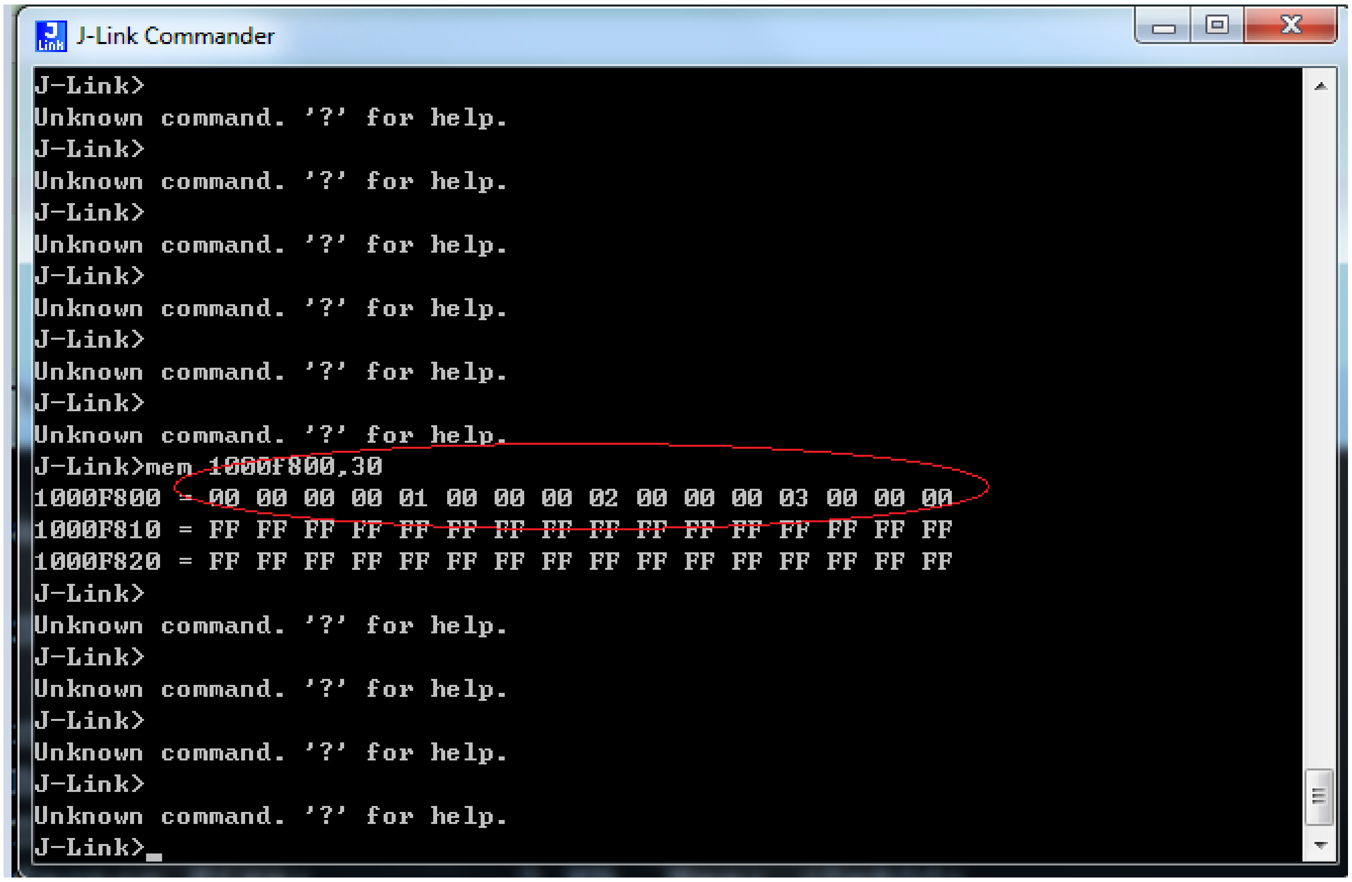
Figure 5. Software Flowchart of “flexnvm_dflash” Demo

By default, the KE1xF 64KB FlexNVM is not partitioned, and is used entirely as D-Flash. Users can do the partition to configure the necessary EEPROM size. After “flexnvm_dflash” demo has downloaded and executed, log messages will be printed on OpenSDA serial terminal as shown in Figure 6. When using J-link commander tool to check the memory status, the pre-defined 16 Bytes data is programmed to the last sector of D-Flash as shown in Figure 7.



```
FlexNVM DFlash Example Start

Flash is UNSECURE!

DFlash Information:
Data Flash Base Address: (0x10000000)
Data Flash Total Size: 64 KB, Hex: (0x10000)
Data Flash Sector Size:        2 KB, Hex: (0x800)
Erase a sector of Data Flash
Successfully Erased Sector 0x1000f800 -> 0x10010000

Program a buffer to a sector of Data Flash
Successfully Programmed and Verified Location 0x1000f800 -> 0x1000f810

End of FlexNVM Dflash Example
```

Figure 6. Logs print on terminal of “flexnvm_dflash” demo



```
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>mem 1000f800,30
1000F800 = 00 00 00 00 01 00 00 00 02 00 00 00 03 00 00 00
1000F810 = FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
1000F820 = FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>
Unknown command. '?' for help.
J-Link>_
```

Figure 7. Memory status checking with J-link Commander of “flexnvm_dflash” demo

5.2. Flexnvm_eeprom demo

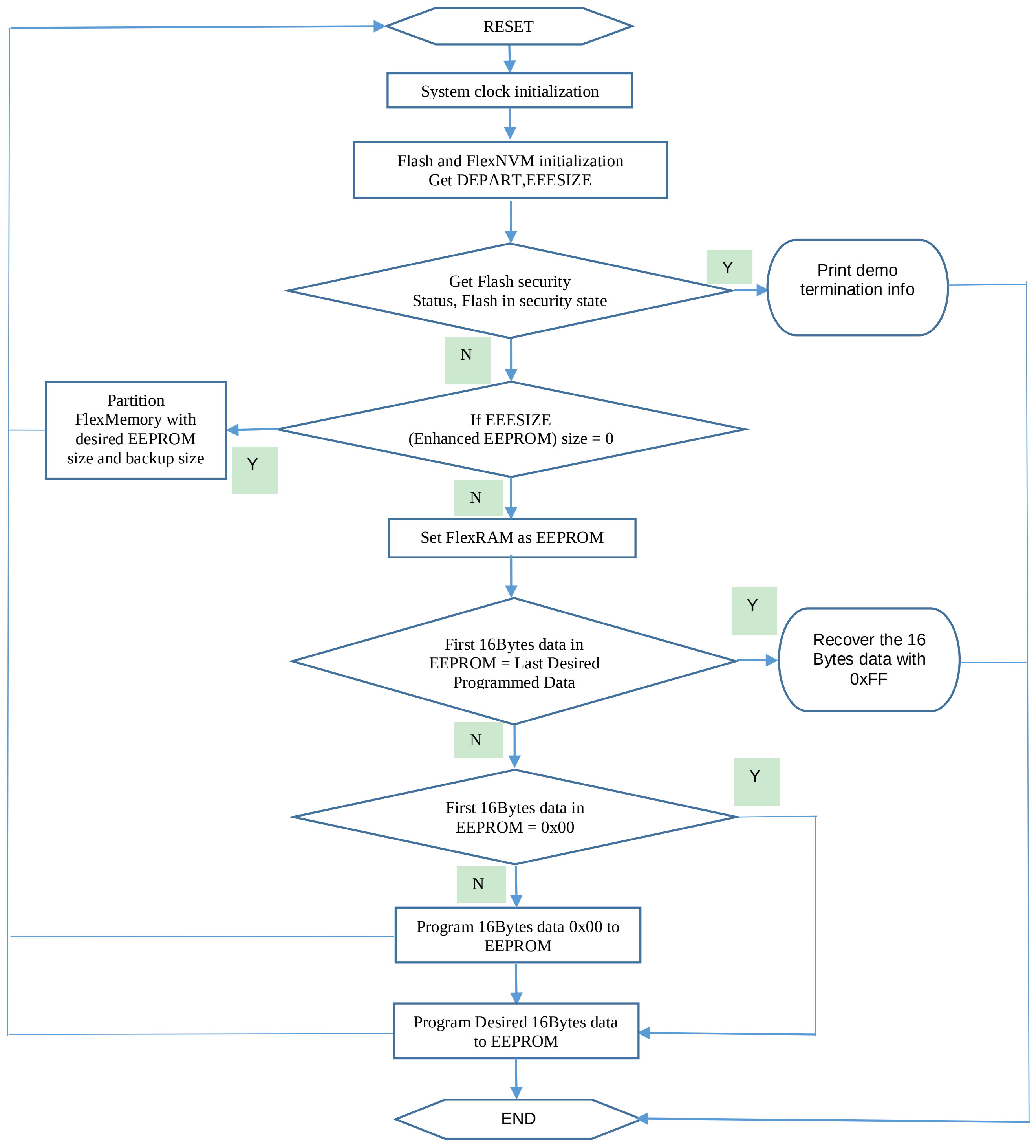
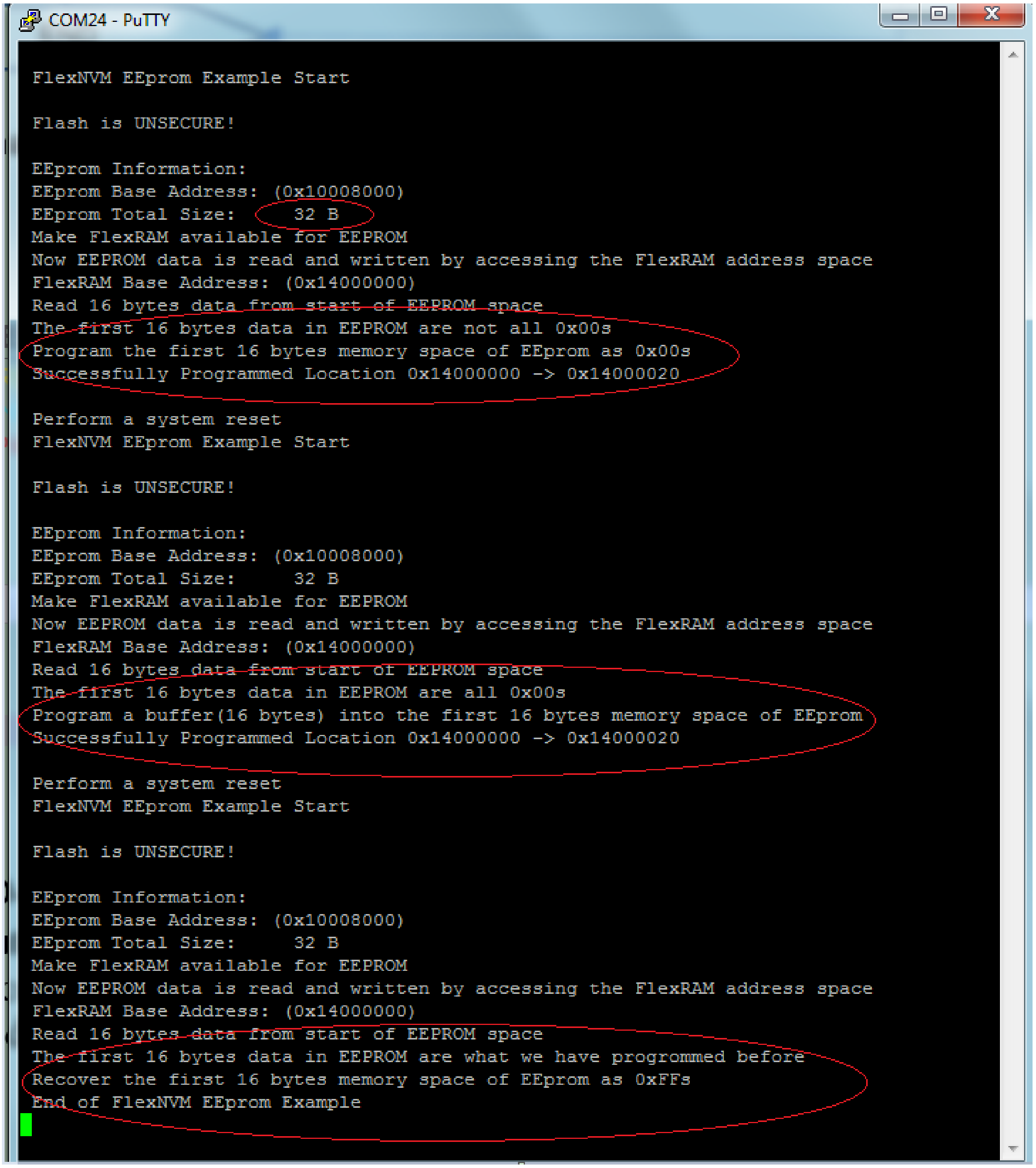


Figure 8. Software Flowchart of “flexnvm_eeprom” Demo

By default, the KE1xF 64KB FlexNVM is not partitioned, and is used entirely as D-Flash. Users can do the partition to configure the necessary EEPROM size. After “flexnvm_eeprom” demo downloaded and executed, log messages will be printed on OpenSDA serial terminal as shown in Figure 9. FlexNVM is configured to be used as a mix of D-Flash and E-Flash (EEPROM backup). The size of both is 32KB. The EEPROM data size in FlexRAM is configured as 32 Bytes. If we mask the recover the data code in source and using J-link commander tool to check the memory status, the pre-defined 16 Bytes data could be seen programmed to the start address of FlexRAM as shown in Figure 10.

NOTE

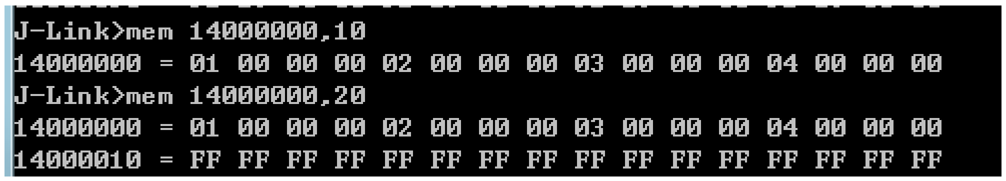
While different partitions of the FlexNVM are available, the intention is that a single partition choice is used throughout the entire lifetime of a given application. The FlexNVM Partition Code choices affect the endurance and data retention characteristics of the device. So partitioning must only be done once. If the flash is re-partitioned for a different configuration, then recorded data is lost and you are not guaranteed to get the expected endurance. If users want to modify the parameter in “flexnvm_eeprom”demo to do another configuration for EEPROM size, they must issue a mass erase command first.



COM24 - PuTTY

```
FlexNVM EEprom Example Start

Flash is UNSECURE!

EEprom Information:
EEprom Base Address: (0x10008000)
EEprom Total Size:      32 B
Make FlexRAM available for EEPROM
Now EEPROM data is read and written by accessing the FlexRAM address space
FlexRAM Base Address: (0x14000000)
Read 16 bytes data from start of EEPROM space
The first 16 bytes data in EEPROM are not all 0x00s
Program the first 16 bytes memory space of EEprom as 0x00s
Successfully Programmed Location 0x14000000 -> 0x14000020

Perform a system reset
FlexNVM EEprom Example Start

Flash is UNSECURE!

EEprom Information:
EEprom Base Address: (0x10008000)
EEprom Total Size:      32 B
Make FlexRAM available for EEPROM
Now EEPROM data is read and written by accessing the FlexRAM address space
FlexRAM Base Address: (0x14000000)
Read 16 bytes data from start of EEPROM space
The first 16 bytes data in EEPROM are all 0x00s
Program a buffer(16 bytes) into the first 16 bytes memory space of EEprom
Successfully Programmed Location 0x14000000 -> 0x14000020

Perform a system reset
FlexNVM EEprom Example Start

Flash is UNSECURE!

EEprom Information:
EEprom Base Address: (0x10008000)
EEprom Total Size:      32 B
Make FlexRAM available for EEPROM
Now EEPROM data is read and written by accessing the FlexRAM address space
FlexRAM Base Address: (0x14000000)
Read 16 bytes data from start of EEPROM space
The first 16 bytes data in EEPROM are what we have progrommed before
Recover the first 16 bytes memory space of EEprom as 0xFFs
End of FlexNVM EEprom Example
```

Figure 9. Logs print on terminal of “flexnvm_eeprom” demo



```
J-Link>mem 14000000,10
14000000 = 01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00
J-Link>mem 14000000,20
14000000 = 01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00
14000010 = FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
```

Figure 10. Memory checking with J-link commander of “flexnvm_eeprom” demo

6. Conclusion

This application note summarizes the features, user perspective and how to implement FlexMemory as D-Flash and EEPROM on the Kinetis KE1xF device. Two demo projects (included in SDK2.0 KE1xF release packages associated with this application note) are introduced to help better understand the FlexMemory functional implementation, and then assist users with the features in their applications.

7. Revision History

Revision number	Date	Substantive changes
0	09/2016	Initial release

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Document Number: AN5338
Rev. 0
09/2016