17/04/2025

Driver v5.12

Ivan Liberotti

Interfacing FlashRunner 2.0 with S32K devices

Summary

S32K Standard Commands

CONNECT

This command is used to connect to the device. It might print information on the status of the debug interfaces.

MASSERASE <memory_type>

This command is used to erase the specified memory.

BLANKCHECK <memory_type> <[startAddress]><[size]>

This command is used to check if the specified memory is blank or a portion of it. Start address and size are optional parameters.

PROGRAM <memory_type>

This command is used to flash the specified memory with a customer’s firmware which fits into this memory.

VERIFY <memory_type> <verify_method>

This command is used to compare the content of the memory with a customer's firmware.

R – Readout

S – CRC32

READ <memory_type> <start_address> <size>

This command is used to read the specified memory or a portion of it and print it out in the GUI terminal.

DUMP <memory_type> <start_address> <size>

This command is used to read the specified memory or a portion of it and save it into a binary file stored inside the programming system SD-CARD.

DISCONNECT

This command is used to disconnect from the device.

S32K1xx Additional Commands

SECTOR_ERASE <memory_type> <startAddress> <size>

This command is used to erase a portion of the specified memory area.

PARTITION <CSEc_key_size> <SFE> <loadRam> <EEprom data set size code> <FlexNVM partition code>

This command is used to partition the flexNVM (D) memory. To check more about it, see chapter S32K1xx Partition.

SETFLEXRAM <function> <quick_write_size>

This command is used to change flexRAM mode. To check more about it, see chapter S32K1xx Set FlexRAM.

UNSECURE

This command is used to unlock the device debug interface. It also removes any partition in the device in standard circumstances (CSEc disbaled), deletes FLASH, FlexNVM and FlexRAM content and restore the FSEC register (0x40C) in the FLASH memory to 0xFE. It removes also other active protections on flash regions/FlexNVM/FlexRAM.

RUN <time [ms]>

This command is used to execute the customer's firmware for.the specified amount of time in ms

READ_REGISTER <address> <size [8|16|32|-bit]>

This command is used to read the specified register in the memory space.

READ_REGISTER_COMPARE <address> <size [8|16|32|-bit]><compareValue>

This command is used to read the specified register in the memory space and compare its value with the specified one.

CHECK_PROTECTION

This command is used to read the the status of the debug interface. Use it alone only, between TPSTART and TPCMD DISCONNECT: it serves the purpose to print the information to be parsed by customer's high level application to choose what to do after.

READ_CHECKSUM <memType><[address]> <[size]>

This command is used to compute the checksum16 on the specified memory area or a portion of it

COMPUTE_CRC32 <memType><[address]> <[size]>

This command is used to compute the checksum32 on the specified memory area or a portion of it.

S32K3xx Additional Commands

SECTOR_ERASE <memory_type> <startAddress> <size>

This command is used to erase a portion of the specified memory area.

WRITE_UTEST <address> <size>

This command is used to program the DCFs UTEST area as an internal storage. To check more about it, see chapter DCFs. Data can be set in an FRB file or dynamic memory.

READ_UTEST <address> <size>

This command is used to verify the DCFs UTEST area as an internal storage. To check more about it, see chapter DCFs. Data can be set in an FRB file or dynamic memory.

FUNCTIONAL_RESET <time [ms]>

This command is used to execute a functional reset. Mostly used to install HSE firmware. To check more about it, see chapter HSE.

RUN <time [ms]>

This command is used to execute the customer's firmware for the specified amount of time in ms.

READ_REGISTER <address> <size [8|16|32|-bit]>

This command is used to read the specified register in the memory space.

READ_REGISTER_COMPARE <address> <size [8|16|32|-bit]><compareValue>

This command is used to read the specified register in the memory space and compare its value with the specified one.

CHECK_PROTECTION

This command is used to read the the status of the debug interface. Use it alone only, between TPSTART and TPCMD DISCONNECT: it serves the purpose to print the information to be parsed by customer's high level application to choose what to do after.

READ_CHECKSUM <memType><[address]> <[size]>

This command is used to compute the checksum16 on the specified memory area or a portion of it. Start address and size are optional parameters.

COMPUTE_CRC32 <memType><[address]> <[size]>

This command is used to compute the checksum32 on the specified memory area or a portion of it. Start address and size are optional parameters.

S32K3xx Additional Parameters

#TCSETPAR PWD_ADDR **<address>**

Parameter used to indicate the FRB or dynamic memory address of the password that needs to be used to unsecure the device during the CONNECT command.

#TCSETPAR CORE_M7 **<core>**

Parameter used to specify which core M7 should be used to flash the target non-volatile memories. To check more about it, see chapter S32K3xx cores.

#TCSETPAR HSE_DELAY **<time[ms]>**

Parameter used to introduce a delay in the CONNECT command before interacting with the core when the HSE is installed for the first time. Used in very few specific cases.

S32K Supported protocols

S32K flashing algorithm supports only SWD protocol. JTAG does not give access to any additional unique features for these devices and the JTAG physical connections allow the user to anyway flash the device through the SWD protocol.

Furthermore the SWD protocol is more efficient and robust.

#TCSETPAR CMODE **<SWD>**

S32K memory maps and commands

S32K1xx supported memories and commands

1) [F] – Flash
2) [D] – FlexNVM
3) [R] – FlexRAM

CONNECT

MASSERASE F|D

BLANKCHECK F|D <[startAddress]><[size]>

PROGRAM F|D|R

VERIFY F|D|R R|S

READ F|D|R <start_address> <size>

DUMP F|D|R <start_address> <size>

SECTOR_ERASE F|D <startAddress> <size>

PARTITION <CSEc_key_size> <SFE> <loadRam> <EEprom data set size code> <FlexNVM partition code>

SETFLEXRAM <function> <quick_write_size>

UNSECURE

RUN <time [ms]>

READ_REGISTER <address> <size [8|16|32|-bit]>

READ_REGISTER_COMPARE <address> <size [8|16|32|-bit]><compareValue>

CHECK_PROTECTION

READ_CHECKSUM F|D<[address]> <[size]>

COMPUTE_CRC32 F|D <[address]> <[size]>

DISCONNECT

S32K3xx supported memories and commands

1) [F] – Flash
2) [D] – DataFlash
3) [O] – UTEST OTP
4) [U] – UTEST DCFs

CONNECT

MASSERASE F|D

BLANKCHECK F|D <[startAddress]><[size]>

PROGRAM F|D|O|U

VERIFY F|D|O|U R|S (S method available only for F and D memory)

READ F|D|O|U <start_address> <size>

DUMP F|D|O|U <start_address> <size>

SECTOR_ERASE F|D <startAddress> <size>

WRITE_UTEST U <address> <size>

READ_UTEST U <address> <size>

FUNCTIONAL_RESET <time [ms]>

RUN <time [ms]>

READ_REGISTER <address> <size [8|16|32|-bit]>

READ_REGISTER_COMPARE <address> <size [8|16|32|-bit]><compareValue>

CHECK_PROTECTION

READ_CHECKSUM F|D <[address]> <[size]>

COMPUTE_CRC32 F|D <[address]> <[size]>

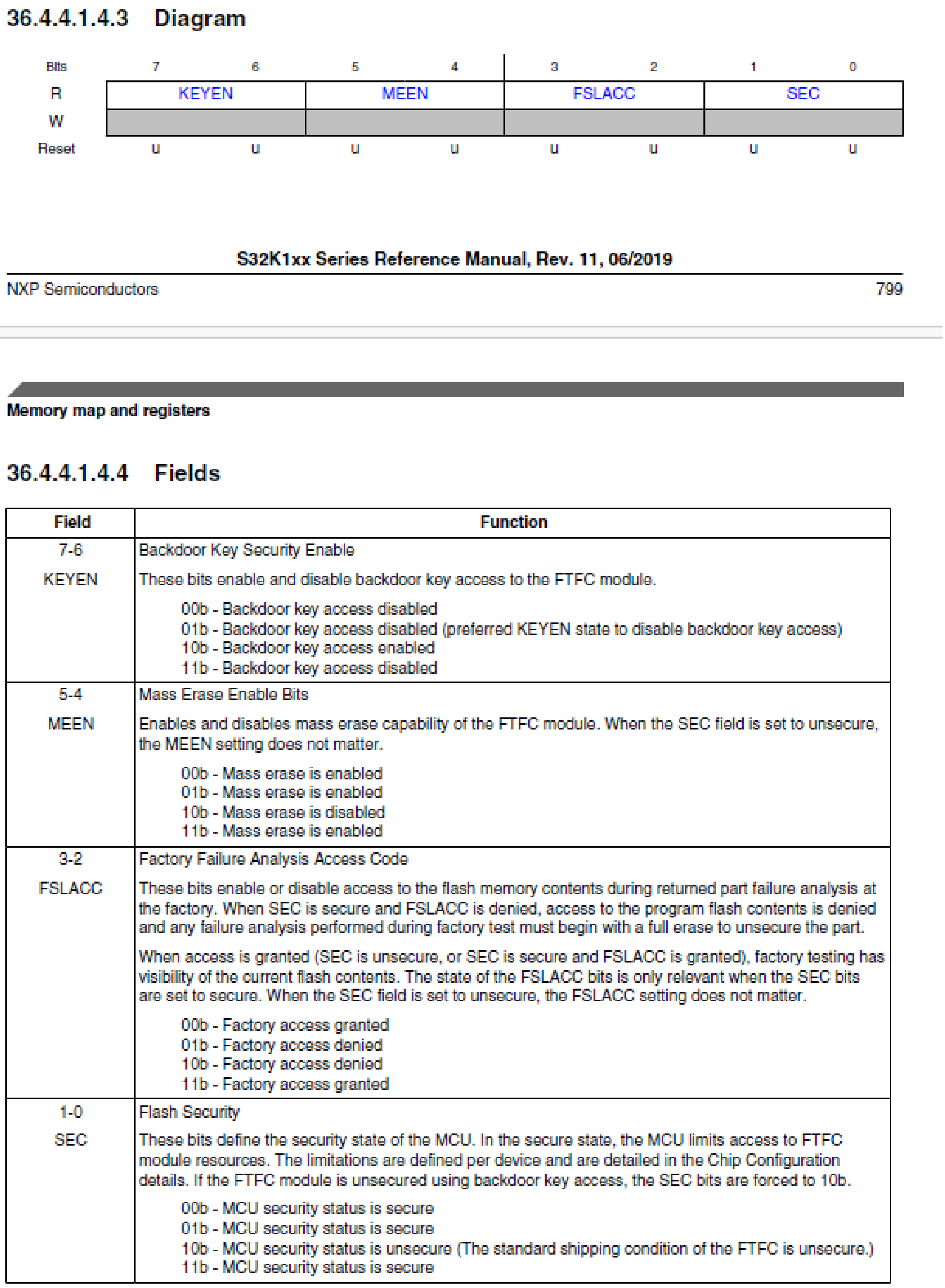
DISCONNECT

S32K1xx [0x400-0x40F] flash area

The value stored at these addresses are very delicate and could influence the reflashing of the MCUs.

1) [0x400 – 0x407]: backdoor key
2) [0x408 – 0x40B]: Flash regions protection
3) [0x40C]: FSEC
4) [0x40D]: FOPT [0x40E]: FEPROT [0x40F]: FDPROT

FSEC explains itself in the image below:



36.4.4.1.4.3 Diagram

Bits	7	6	5	4	3	2	1	0
R	KEYEN		MEEN		FSLACC		SEC	
W								
Reset	u	u	u	u	u	u	u	u

S32K1xx Series Reference Manual, Rev. 11, 06/2019

NXP Semiconductors 799

Memory map and registers

36.4.4.1.4.4 Fields

Field	Function
7-6 KEYEN	Backdoor Key Security Enable These bits enable and disable backdoor key access to the FTFC module. 00b - Backdoor key access disabled 01b - Backdoor key access disabled (preferred KEYEN state to disable backdoor key access) 10b - Backdoor key access enabled 11b - Backdoor key access disabled
5-4 MEEN	Mass Erase Enable Bits Enables and disables mass erase capability of the FTFC module. When the SEC field is set to unsecure, the MEEN setting does not matter. 00b - Mass erase is enabled 01b - Mass erase is enabled 10b - Mass erase is disabled 11b - Mass erase is enabled
3-2 FSLACC	Factory Failure Analysis Access Code These bits enable or disable access to the flash memory contents during returned part failure analysis at the factory. When SEC is secure and FSLACC is denied, access to the program flash contents is denied and any failure analysis performed during factory test must begin with a full erase to unsecure the part. When access is granted (SEC is unsecure, or SEC is secure and FSLACC is granted), factory testing has visibility of the current flash contents. The state of the FSLACC bits is only relevant when the SEC bits are set to secure. When the SEC field is set to unsecure, the FSLACC setting does not matter. 00b - Factory access granted 01b - Factory access denied 10b - Factory access denied 11b - Factory access granted
1-0 SEC	Flash Security These bits define the security state of the MCU. In the secure state, the MCU limits access to FTFC module resources. The limitations are defined per device and are detailed in the Chip Configuration details. If the FTFC module is unsecured using backdoor key access, the SEC bits are forced to 10b. 00b - MCU security status is secure 01b - MCU security status is secure 10b - MCU security status is unsecure (The standard shipping condition of the FTFC is unsecure.) 11b - MCU security status is secure

The backdoor key is a sequence of 8 bytes to be used in certain conditions to unlock access to the device:

1) KEYEN field of FSEC register should be set to 0b10
2) MEEN field of FSEC set to 0b10 (disable UNSECURE command)
3) SEC field of FSEC set to 0b00 or 0b01 or 0b11

In this condition, the device cannot be unlocked through SWD, but only through the verify backdoor key command that should be sent through a serial port and should be read from an already running FW inside the MCU that will then unlock the debug interface. From an SWD programmer/debugger point of view, this is an irreversible condition. S32K algorithm could still use the UART to send the backdoor key, but this would require an implementation activity and collaboration with the FW design team.

If the KEYEN is disabled, but again point 2) and 3) are in the way explained before, from a programmer/debugger point of view, this is a completely irreversible condition.

As soon as the MEEN is enabled, the device can be reflashed through an UNSECURE command in standard circumstances.

Flash regions protection allow the user to disable any program/erase operation through FTFC peripheral (MASSERASE F, PROGRAM F…). The UNSECURE will remove these protections.

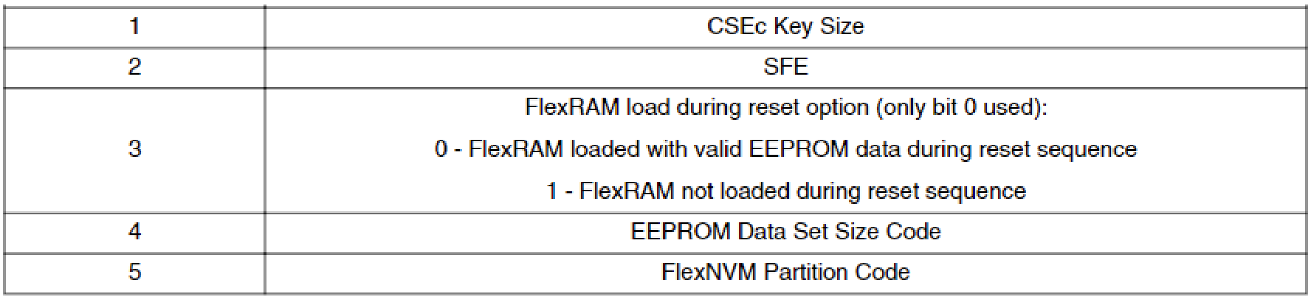
FOPT customizes MCU functionality.

FEPROT and FDPROT enable protections on EEPROM/FlexNVM area (MASSERASE D, PROGRAM D, PROGRAM R…). The UNSECURE will remove these protections.

S32K1xx Partition

The partition command is mainly used to divide the FlexNVM area into two parts:

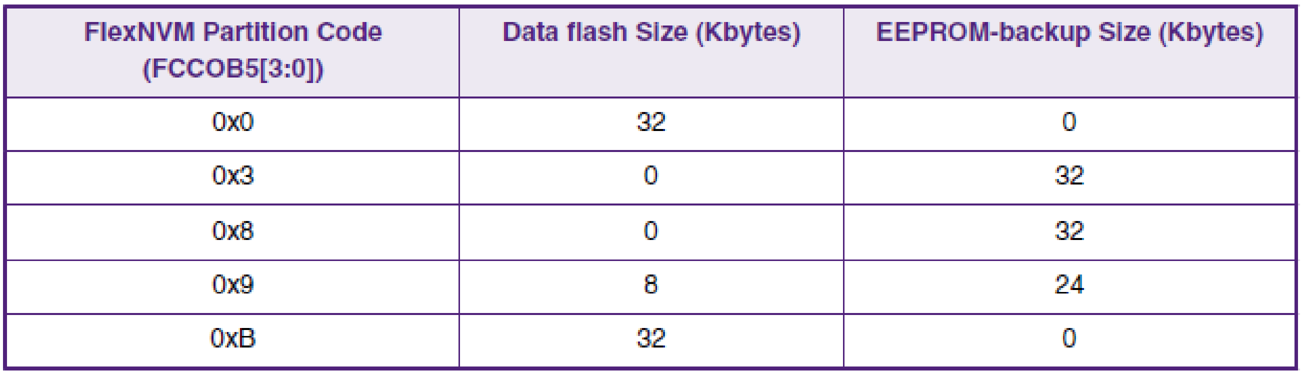
1) DataFlash
2) EEPROM backup



1	CSEc Key Size
2	SFE
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
5	FlexNVM Partition Code

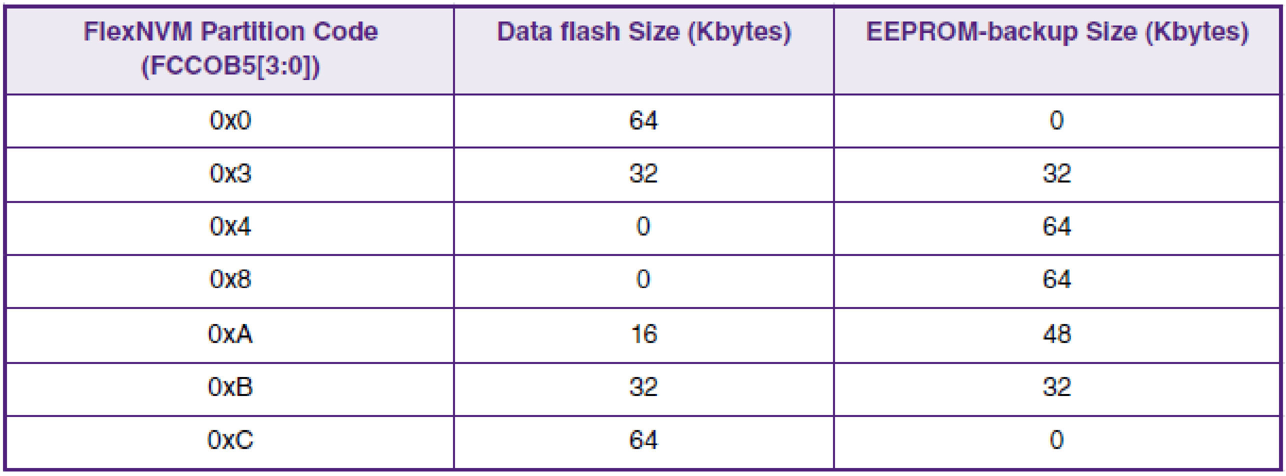
Based on the last parameter of the command (<FlexNVM partition code>), the following images contain the configuration for every S32K1xx supported by this algorithm:

Table 3. FlexNVM partition codes for 32 kB FlexNVM devices (S32K11x)



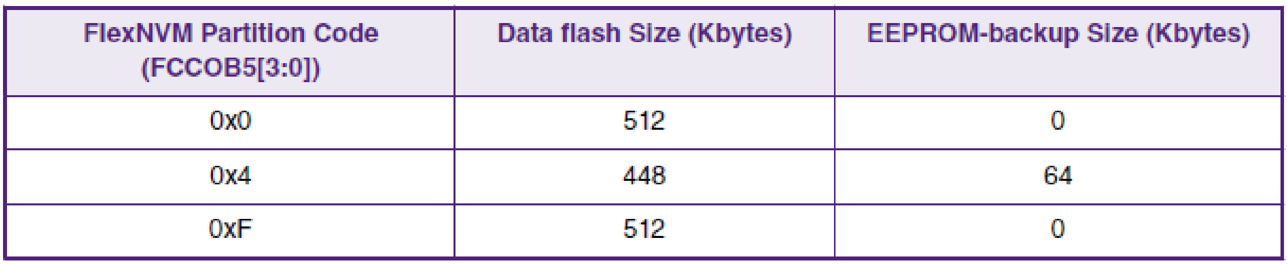
FlexNVM Partition Code (FCCOB5[3:0])	Data flash Size (Kbytes)	EEPROM-backup Size (Kbytes)
0x0	32	0
0x3	0	32
0x8	0	32
0x9	8	24
0xB	32	0

Table 4. FlexNVM partition codes for 64 kB FlexNVM devices (S32K142, S32K144, S32K146)



FlexNVM Partition Code (FCCOB5[3:0])	Data flash Size (Kbytes)	EEPROM-backup Size (Kbytes)
0x0	64	0
0x3	32	32
0x4	0	64
0x8	0	64
0xA	16	48
0xB	32	32
0xC	64	0

Table 5. FlexNVM partition codes for 64 kB FlexNVM devices (S32K148)



FlexNVM Partition Code (FCCOB5[3:0])	Data flash Size (Kbytes)	EEPROM-backup Size (Kbytes)
0x0	512	0
0x4	448	64
0xF	512	0

The DataFlash size can still be accessed by FlexNVM flashing commands (MASSERASE D, PROGRAM D), while the EEPROM backup area cannot be accessed by any tool. It is just important to not have a FW whose data are in the reserved area: that would make absolutely no sense, it means something is wrong in customer flow.
Furthermore, the partition command deletes the content from the FlexNVM memory: it is useless to program data here

and partition it: the content will just be deleted, that would again make no sense, it means something is wrong in customer flow.

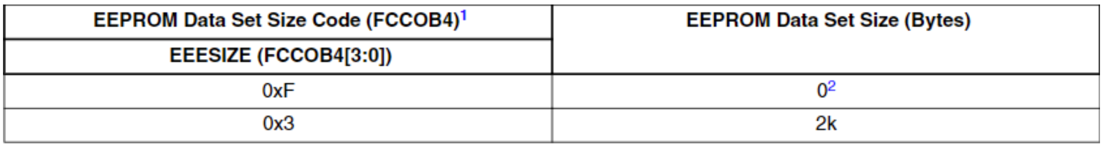
To make partitioning take effect, a reset/reconnect is needed and this is done automatically by the algorithm command: if the FLASH of the device has been erased or for any reasons the FSEC register is set to a value that enables the protection, after a PARTITION command, the device will be locked. This is the reason why usually the PARTITION is launched right after the CONNECT or UNSECURE command before executing flashing operations on the device.

The EEPROM backup part is strictly connected to the FlexRAM memory: in fact, this FlexRAM memory, once written, save data on this EEPROM backup area and at every MCU power cycle it is again loaded with the EEPROM bacup data.

An UNSECURE command can remove the partition applied to a device under standard circumstances (no CSEc peripheral in place).

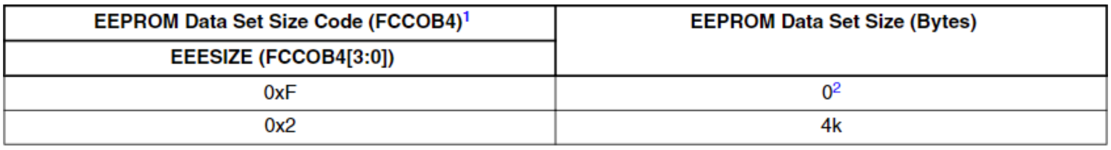
The fourth parameter (<EEprom data set size code>) can be:

1) S32K11x



EEPROM Data Set Size Code (FCCOB4)[1]	EEPROM Data Set Size (Bytes)
EEESIZE (FCCOB4[3:0])	
0xF	0[2]
0x3	2k

2) S32K14x



EEPROM Data Set Size Code (FCCOB4)[1]	EEPROM Data Set Size (Bytes)
EEESIZE (FCCOB4[3:0])	
0xF	0[2]
0x2	4k

The first two parameters are not relevant and should be set to 0 for standard applications (this would require an implementation activity and collaboration with the FW design team if CSEc peripheral, keys and SFE are needed).

The third parameter is used to choose if the content of the FlexRAM should be loaded with valid data at the next device startup or not.

S32K1xx Set FlexRAM

The SETFLEXRAM command is mainly used to change the default FlexRAM usage as RAM into EEPROM to then store data inside this area (PROGRAM R, VERIFY F R). Below the table of functionalities:

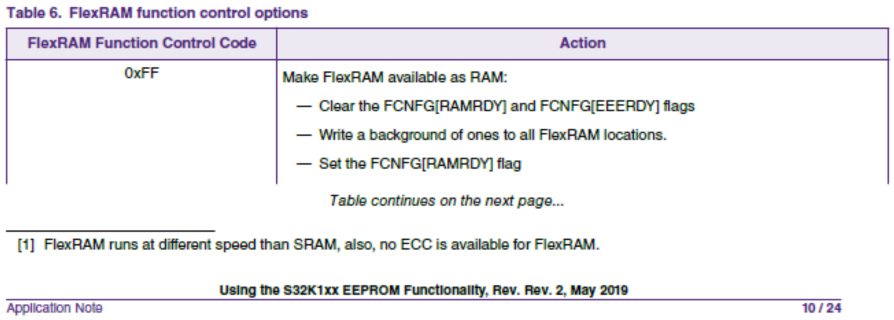


Table 6. FlexRAM function control options

FlexRAM Function Control Code	Action
0xFF	Make FlexRAM available as RAM: — Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags — Write a background of ones to all FlexRAM locations. — Set the FCNFG[RAMRDY] flag

Table continues on the next page...

[1] FlexRAM runs at different speed than SRAM, also, no ECC is available for FlexRAM.

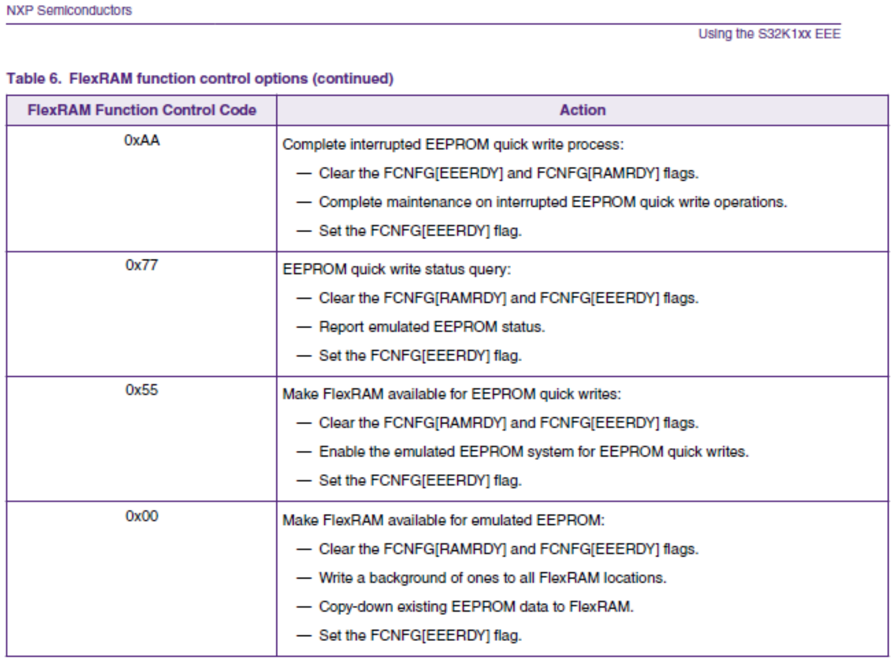


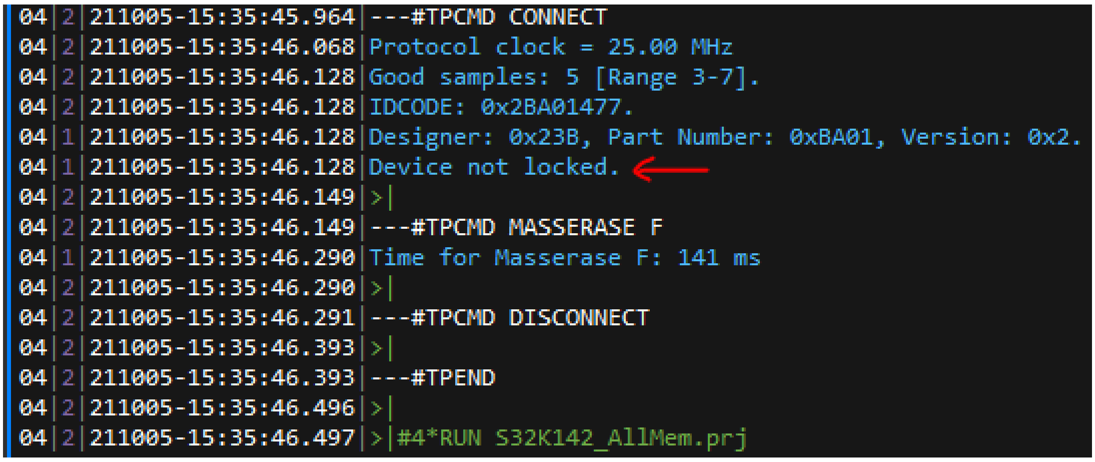
Table 6. FlexRAM function control options (continued)

FlexRAM Function Control Code	Action
0xAA	Complete interrupted EEPROM quick write process: — Clear the FCNFG[EEERDY] and FCNFG[RAMRDY] flags. — Complete maintenance on interrupted EEPROM quick write operations. — Set the FCNFG[EEERDY] flag.
0x77	EEPROM quick write status query: — Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags. — Report emulated EEPROM status. — Set the FCNFG[EEERDY] flag.
0x55	Make FlexRAM available for EEPROM quick writes: — Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags. — Enable the emulated EEPROM system for EEPROM quick writes. — Set the FCNFG[EEERDY] flag.
0x00	Make FlexRAM available for emulated EEPROM: — Clear the FCNFG[RAMRDY] and FCNFG[EEERDY] flags. — Write a background of ones to all FlexRAM locations. — Copy-down existing EEPROM data to FlexRAM. — Set the FCNFG[EEERDY] flag.

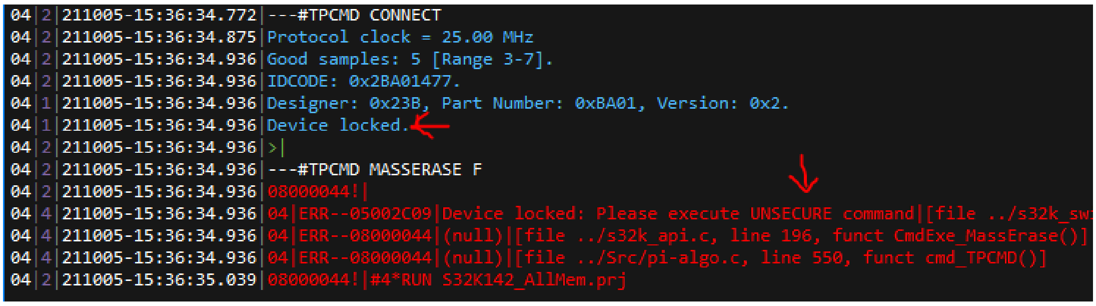
This command must be executed after PARTITION and FLASH flashing operations (PROGRAM F). The FTFC program command to flash this non-volatile memory requires the FlexRAM in RAM mode: if this is changed to EEPROM before, the algorithm will change it back again and the previous step that changed FlexRAM in EEPROM mode would be useless.

S32K1xx User Cases

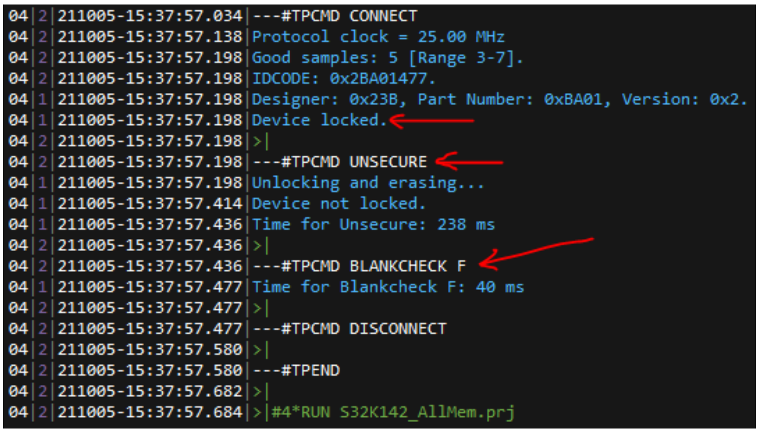
1) **MASSERASE F + RESET**: this combination will result in the **FSEC** byte put to secure state. To connect to the board again, an “**UNSECURE**” command needs to be used.



```
04|2|211005-15:35:45.964|---#TPCMD CONNECT
04|2|211005-15:35:46.068|Protocol clock = 25.00 MHz
04|2|211005-15:35:46.128|Good samples: 5 [Range 3-7].
04|2|211005-15:35:46.128|IDCODE: 0x2BA01477.
04|1|211005-15:35:46.128|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-15:35:46.128|Device not locked.
04|2|211005-15:35:46.149|>|
04|2|211005-15:35:46.149|---#TPCMD MASSERASE F
04|1|211005-15:35:46.290|Time for Masserase F: 141 ms
04|2|211005-15:35:46.290|>|
04|2|211005-15:35:46.291|---#TPCMD DISCONNECT
04|2|211005-15:35:46.393|>|
04|2|211005-15:35:46.393|---#TPEND
04|2|211005-15:35:46.496|>|
04|2|211005-15:35:46.497|>|#4*RUN S32K142_AllMem.prj
```

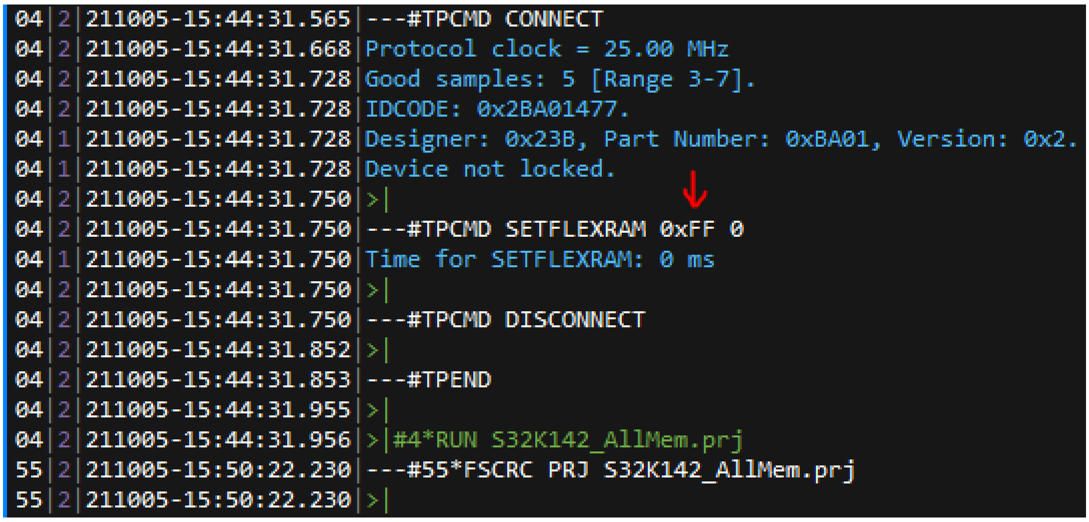


```
04|2|211005-15:36:34.772|---#TPCMD CONNECT
04|2|211005-15:36:34.875|Protocol clock = 25.00 MHz
04|2|211005-15:36:34.936|Good samples: 5 [Range 3-7].
04|2|211005-15:36:34.936|IDCODE: 0x2BA01477.
04|1|211005-15:36:34.936|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-15:36:34.936|Device locked.
04|2|211005-15:36:34.936|>|
04|2|211005-15:36:34.936|---#TPCMD MASSERASE F
04|2|211005-15:36:34.936|08000044!|
04|4|211005-15:36:34.936|04|ERR--05002C09|Device locked: Please execute UNSECURE command|[file ../s32k_sw
04|4|211005-15:36:34.936|04|ERR--08000044|(null)|[file ../s32k_api.c, line 196, funct CmdExe_MassErase()]
04|4|211005-15:36:34.936|04|ERR--08000044|(null)|[file ../Src/pi-algo.c, line 550, funct cmd_TPCMD()]
04|2|211005-15:36:35.039|08000044!|#4*RUN S32K142_AllMem.prj
```



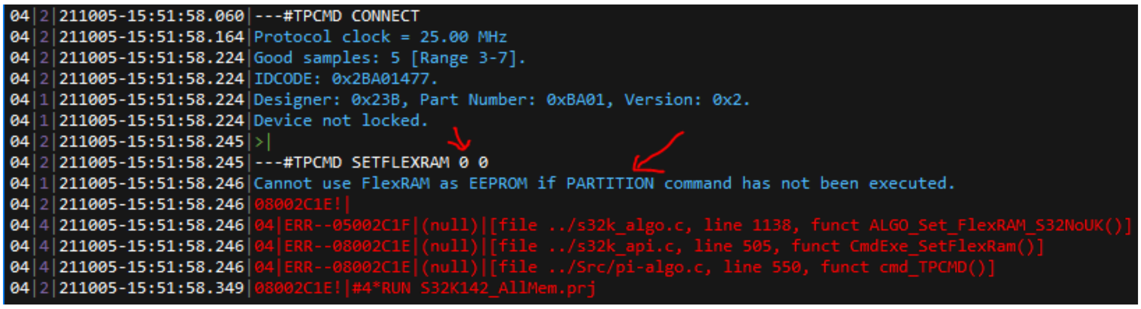
```
04|2|211005-15:37:57.034|---#TPCMD CONNECT
04|2|211005-15:37:57.138|Protocol clock = 25.00 MHz
04|2|211005-15:37:57.198|Good samples: 5 [Range 3-7].
04|2|211005-15:37:57.198|IDCODE: 0x2BA01477.
04|1|211005-15:37:57.198|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-15:37:57.198|Device locked.
04|2|211005-15:37:57.198|>|
04|2|211005-15:37:57.198|---#TPCMD UNSECURE
04|1|211005-15:37:57.198|Unlocking and erasing...
04|1|211005-15:37:57.414|Device not locked.
04|1|211005-15:37:57.436|Time for Unsecure: 238 ms
04|2|211005-15:37:57.436|>|
04|2|211005-15:37:57.436|---#TPCMD BLANKCHECK F
04|1|211005-15:37:57.477|Time for Blankcheck F: 40 ms
04|2|211005-15:37:57.477|>|
04|2|211005-15:37:57.477|---#TPCMD DISCONNECT
04|2|211005-15:37:57.580|>|
04|2|211005-15:37:57.580|---#TPEND
04|2|211005-15:37:57.682|>|
04|2|211005-15:37:57.684|>|#4*RUN S32K142_AllMem.prj
```

2) **PARTITION + SET_FLEXRAM:** Using the **SET_FLEXRAM** to set the memory as **RAM** can be done, but it is kind of useless



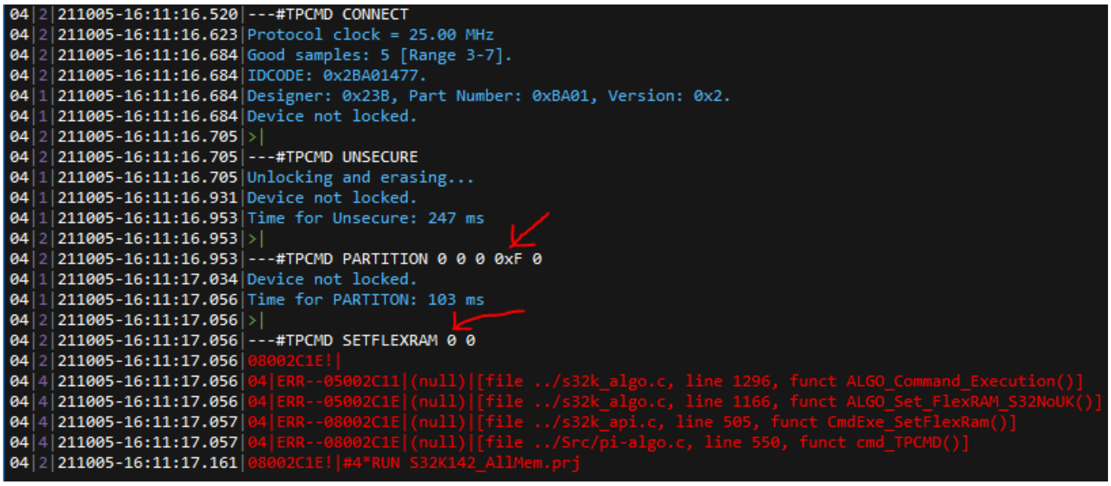
```
04|2|211005-15:44:31.565|---#TPCMD CONNECT
04|2|211005-15:44:31.668|Protocol clock = 25.00 MHz
04|2|211005-15:44:31.728|Good samples: 5 [Range 3-7].
04|2|211005-15:44:31.728|IDCODE: 0x2BA01477.
04|1|211005-15:44:31.728|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-15:44:31.728|Device not locked.
04|2|211005-15:44:31.750|>|
04|2|211005-15:44:31.750|---#TPCMD SETFLEXRAM 0xFF 0
04|1|211005-15:44:31.750|Time for SETFLEXRAM: 0 ms
04|2|211005-15:44:31.750|>|
04|2|211005-15:44:31.750|---#TPCMD DISCONNECT
04|2|211005-15:44:31.852|>|
04|2|211005-15:44:31.853|---#TPEND
04|2|211005-15:44:31.955|>|
04|2|211005-15:44:31.956|>|#4*RUN S32K142_AllMem.prj
55|2|211005-15:50:22.230|---#55*FSCRC PRJ S32K142_AllMem.prj
55|2|211005-15:50:22.230|>|
```

Using it to set the **FlexRAM** as **EEPROM** needs a **PARTITION** command first.



```
04|2|211005-15:51:58.060|---#TPCMD CONNECT
04|2|211005-15:51:58.164|Protocol clock = 25.00 MHz
04|2|211005-15:51:58.224|Good samples: 5 [Range 3-7].
04|2|211005-15:51:58.224|IDCODE: 0x2BA01477.
04|1|211005-15:51:58.224|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-15:51:58.224|Device not locked.
04|2|211005-15:51:58.245|>|
04|2|211005-15:51:58.245|---#TPCMD SETFLEXRAM 0 0
04|1|211005-15:51:58.246|Cannot use FlexRAM as EEPROM if PARTITION command has not been executed.
04|2|211005-15:51:58.246|08002C1E!|
04|4|211005-15:51:58.246|04|ERR--05002C1F|(null)|[file ../s32k_algo.c, line 1138, funct ALGO_Set_FlexRAM_S32NoUK()]
04|4|211005-15:51:58.246|04|ERR--08002C1E|(null)|[file ../s32k_api.c, line 505, funct CmdExe_SetFlexRam()]
04|4|211005-15:51:58.246|04|ERR--08002C1E|(null)|[file ../Src/pi-algo.c, line 550, funct cmd_TPCMD()]
04|2|211005-15:51:58.349|08002C1E!|#4*RUN S32K142_AllMem.prj
```

Trying to set the **FlexRAM** as **EEPROM** without reserving size for it (fourth parameter) in the **PARTITION** command will fail

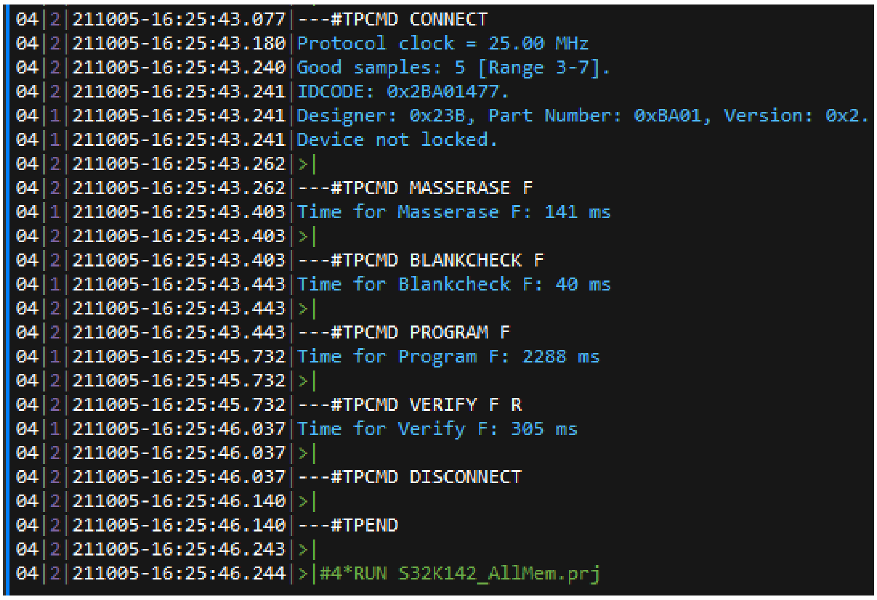


```
04|2|211005-16:11:16.520|---#TPCMD CONNECT
04|2|211005-16:11:16.623|Protocol clock = 25.00 MHz
04|2|211005-16:11:16.684|Good samples: 5 [Range 3-7].
04|2|211005-16:11:16.684|IDCODE: 0x2BA01477.
04|1|211005-16:11:16.684|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-16:11:16.684|Device not locked.
04|2|211005-16:11:16.705|>|
04|2|211005-16:11:16.705|---#TPCMD UNSECURE
04|1|211005-16:11:16.705|Unlocking and erasing...
04|1|211005-16:11:16.931|Device not locked.
04|1|211005-16:11:16.953|Time for Unsecure: 247 ms
04|2|211005-16:11:16.953|>|
04|2|211005-16:11:16.953|---#TPCMD PARTITION 0 0 0 0xF 0
04|1|211005-16:11:17.034|Device not locked.
04|1|211005-16:11:17.056|Time for PARTITON: 103 ms
04|2|211005-16:11:17.056|>|
04|2|211005-16:11:17.056|---#TPCMD SETFLEXRAM 0 0
04|2|211005-16:11:17.056|08002C1E!|
04|4|211005-16:11:17.056|04|ERR--05002C11|(null)|[file ../s32k_algo.c, line 1296, funct ALGO_Command_Execution()]
04|4|211005-16:11:17.056|04|ERR--05002C1E|(null)|[file ../s32k_algo.c, line 1166, funct ALGO_Set_FlexRAM_S32NoUK()]
04|4|211005-16:11:17.057|04|ERR--08002C1E|(null)|[file ../s32k_api.c, line 505, funct CmdExe_SetFlexRam()]
04|4|211005-16:11:17.057|04|ERR--08002C1E|(null)|[file ../Src/pi-algo.c, line 550, funct cmd_TPCMD()]
04|2|211005-16:11:17.161|08002C1E!|#4*RUN S32K142_AllMem.prj
```

3) PARTITION COMMAND AFTER ANYTHING THAT LOCKS THE DEVICE:

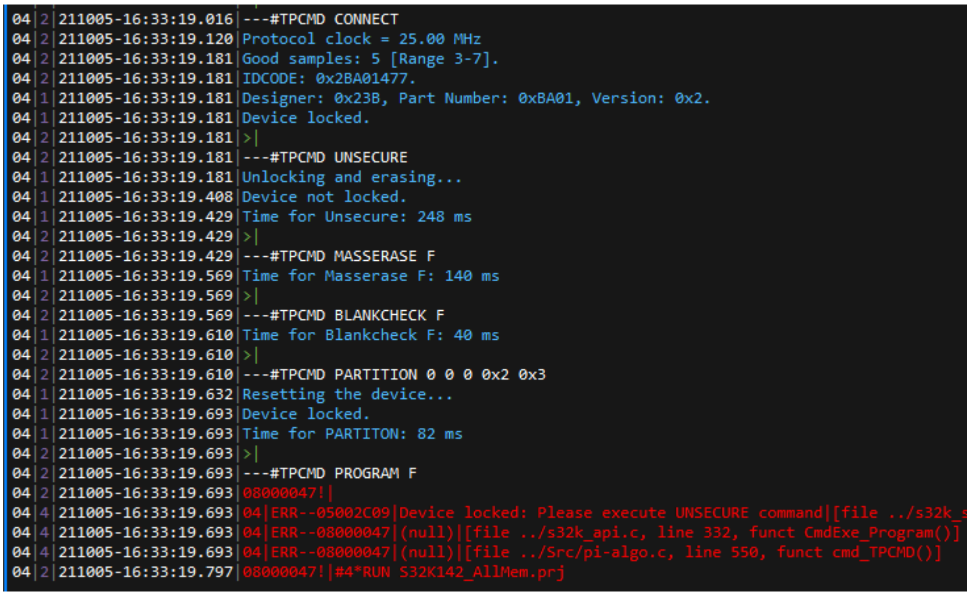
Performing a **PARTITION** command will reset the device, because it is needed for the partition to take effect; if something that locks the device is executed before, the result after the **PARTITION** command (+**RESET**) will be device locked.

In this log everything is passing because there is no reset among the **MASSERASE F** and the rest of the flashing operations



```
04|2|211005-16:25:43.077|---#TPCMD CONNECT
04|2|211005-16:25:43.180|Protocol clock = 25.00 MHz
04|2|211005-16:25:43.240|Good samples: 5 [Range 3-7].
04|2|211005-16:25:43.241|IDCODE: 0x2BA01477.
04|1|211005-16:25:43.241|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-16:25:43.241|Device not locked.
04|2|211005-16:25:43.262|>|
04|2|211005-16:25:43.262|---#TPCMD MASSERASE F
04|1|211005-16:25:43.403|Time for Masserase F: 141 ms
04|2|211005-16:25:43.403|>|
04|2|211005-16:25:43.403|---#TPCMD BLANKCHECK F
04|1|211005-16:25:43.443|Time for Blankcheck F: 40 ms
04|2|211005-16:25:43.443|>|
04|2|211005-16:25:43.443|---#TPCMD PROGRAM F
04|1|211005-16:25:45.732|Time for Program F: 2288 ms
04|2|211005-16:25:45.732|>|
04|2|211005-16:25:45.732|---#TPCMD VERIFY F R
04|1|211005-16:25:46.037|Time for Verify F: 305 ms
04|2|211005-16:25:46.037|>|
04|2|211005-16:25:46.037|---#TPCMD DISCONNECT
04|2|211005-16:25:46.140|>|
04|2|211005-16:25:46.140|---#TPEND
04|2|211005-16:25:46.243|>|
04|2|211005-16:25:46.244|>|#4*RUN S32K142_AllMem.prj
```

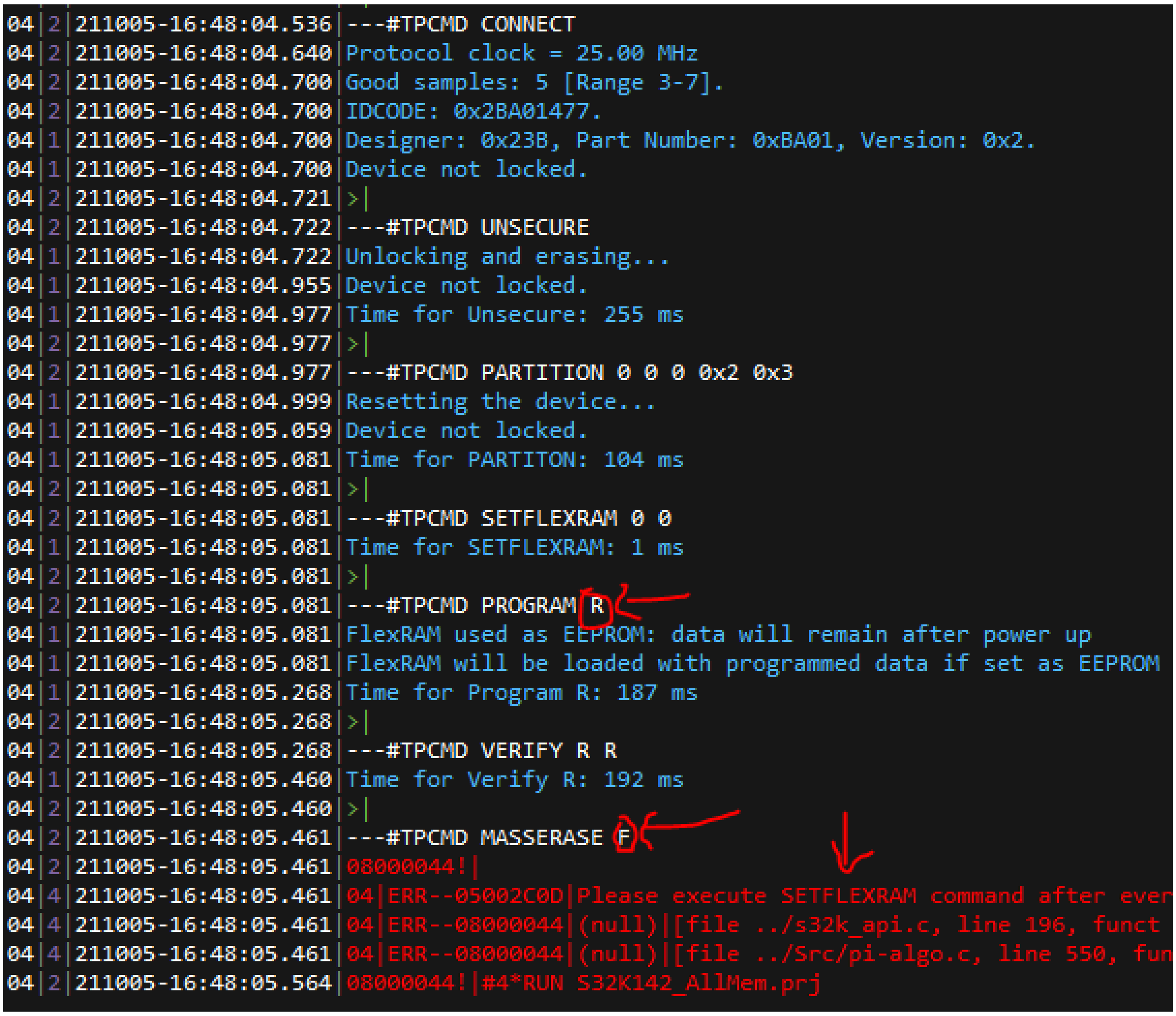
If a **PARTITION** is put in the middle, reset happens, and device is locked



```
04|2|211005-16:33:19.016|---#TPCMD CONNECT
04|2|211005-16:33:19.120|Protocol clock = 25.00 MHz
04|2|211005-16:33:19.181|Good samples: 5 [Range 3-7].
04|2|211005-16:33:19.181|IDCODE: 0x2BA01477.
04|1|211005-16:33:19.181|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-16:33:19.181|Device locked.
04|2|211005-16:33:19.181|>|
04|2|211005-16:33:19.181|---#TPCMD UNSECURE
04|1|211005-16:33:19.181|Unlocking and erasing...
04|1|211005-16:33:19.408|Device not locked.
04|1|211005-16:33:19.429|Time for Unsecure: 248 ms
04|2|211005-16:33:19.429|>|
04|2|211005-16:33:19.429|---#TPCMD MASSERASE F
04|1|211005-16:33:19.569|Time for Masserase F: 140 ms
04|2|211005-16:33:19.569|>|
04|2|211005-16:33:19.569|---#TPCMD BLANKCHECK F
04|1|211005-16:33:19.610|Time for Blankcheck F: 40 ms
04|2|211005-16:33:19.610|>|
04|2|211005-16:33:19.610|---#TPCMD PARTITION 0 0 0 0x2 0x3
04|1|211005-16:33:19.632|Resetting the device...
04|1|211005-16:33:19.693|Device locked.
04|1|211005-16:33:19.693|Time for PARTITON: 82 ms
04|2|211005-16:33:19.693|>|
04|2|211005-16:33:19.693|---#TPCMD PROGRAM F
04|2|211005-16:33:19.693|08000047!|
04|4|211005-16:33:19.693|04|ERR--05002C09|Device locked: Please execute UNSECURE command|[file ../s32k_s
04|4|211005-16:33:19.693|04|ERR--08000047|(null)|[file ../s32k_api.c, line 332, funct CmdExe_Program()]
04|4|211005-16:33:19.693|04|ERR--08000047|(null)|[file ../Src/pi-algo.c, line 550, funct cmd_TPCMD()]
04|2|211005-16:33:19.797|08000047!|#4*RUN S32K142_AllMem.prj
```

4) **PERFORMING FlexRAM OPERATIONS BEFORE FLASH OR FlexNVM FLASHING:**

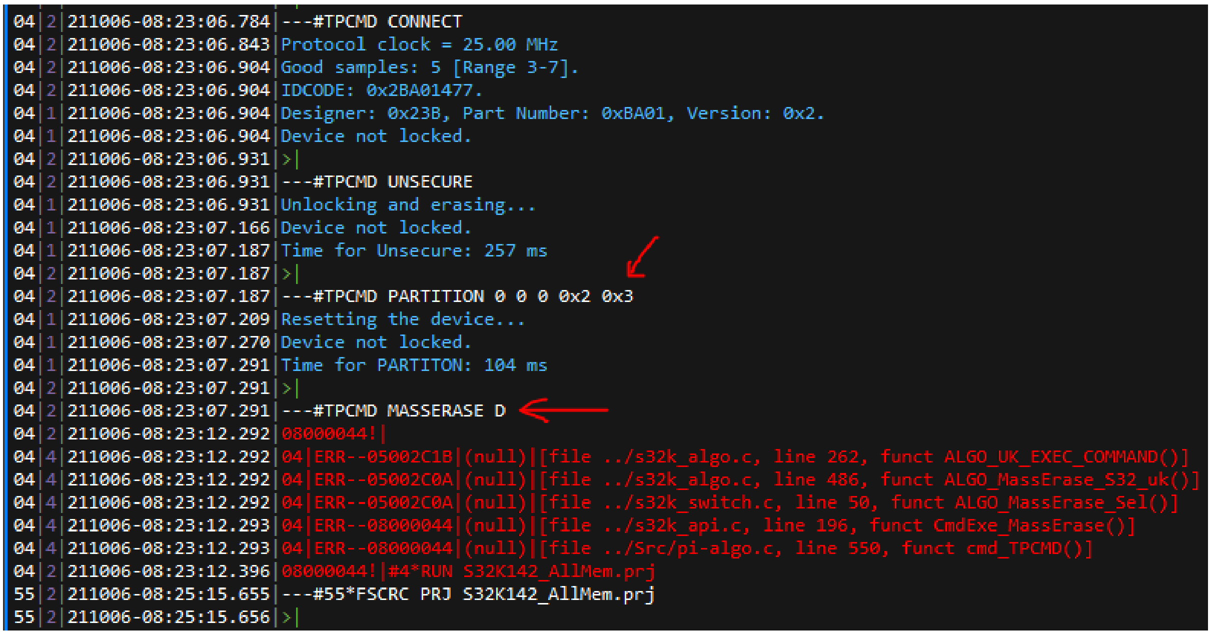
It is not allowed to flash the **FlexRAM** before flashing the **FLASH** or **FlexNVM** simply because these two area use **FlexRAM** as **RAM** and doing so, they will delete all the content that was previously stored in the **FlexRAM** used as **EEPROM**



```
04|2|211005-16:48:04.536|---#TPCMD CONNECT
04|2|211005-16:48:04.640|Protocol clock = 25.00 MHz
04|2|211005-16:48:04.700|Good samples: 5 [Range 3-7].
04|2|211005-16:48:04.700|IDCODE: 0x2BA01477.
04|1|211005-16:48:04.700|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211005-16:48:04.700|Device not locked.
04|2|211005-16:48:04.721|>|
04|2|211005-16:48:04.722|---#TPCMD UNSECURE
04|1|211005-16:48:04.722|Unlocking and erasing...
04|1|211005-16:48:04.955|Device not locked.
04|1|211005-16:48:04.977|Time for Unsecure: 255 ms
04|2|211005-16:48:04.977|>|
04|2|211005-16:48:04.977|---#TPCMD PARTITION 0 0 0 0x2 0x3
04|1|211005-16:48:04.999|Resetting the device...
04|1|211005-16:48:05.059|Device not locked.
04|1|211005-16:48:05.081|Time for PARTITON: 104 ms
04|2|211005-16:48:05.081|>|
04|2|211005-16:48:05.081|---#TPCMD SETFLEXRAM 0 0
04|1|211005-16:48:05.081|Time for SETFLEXRAM: 1 ms
04|2|211005-16:48:05.081|>|
04|2|211005-16:48:05.081|---#TPCMD PROGRAM R
04|1|211005-16:48:05.081|FlexRAM used as EEPROM: data will remain after power up
04|1|211005-16:48:05.081|FlexRAM will be loaded with programmed data if set as EEPROM
04|1|211005-16:48:05.268|Time for Program R: 187 ms
04|2|211005-16:48:05.268|>|
04|2|211005-16:48:05.268|---#TPCMD VERIFY R R
04|1|211005-16:48:05.460|Time for Verify R: 192 ms
04|2|211005-16:48:05.460|>|
04|2|211005-16:48:05.461|---#TPCMD MASSERASE F
04|2|211005-16:48:05.461|08000044!|
04|4|211005-16:48:05.461|04|ERR--05002C0D|Please execute SETFLEXRAM command after ever
04|4|211005-16:48:05.461|04|ERR--08000044|(null)|[file ../s32k_api.c, line 196, funct
04|4|211005-16:48:05.461|04|ERR--08000044|(null)|[file ../Src/pi-algo.c, line 550, fun
04|2|211005-16:48:05.564|08000044!|#4*RUN S32K142_AllMem.prj
```

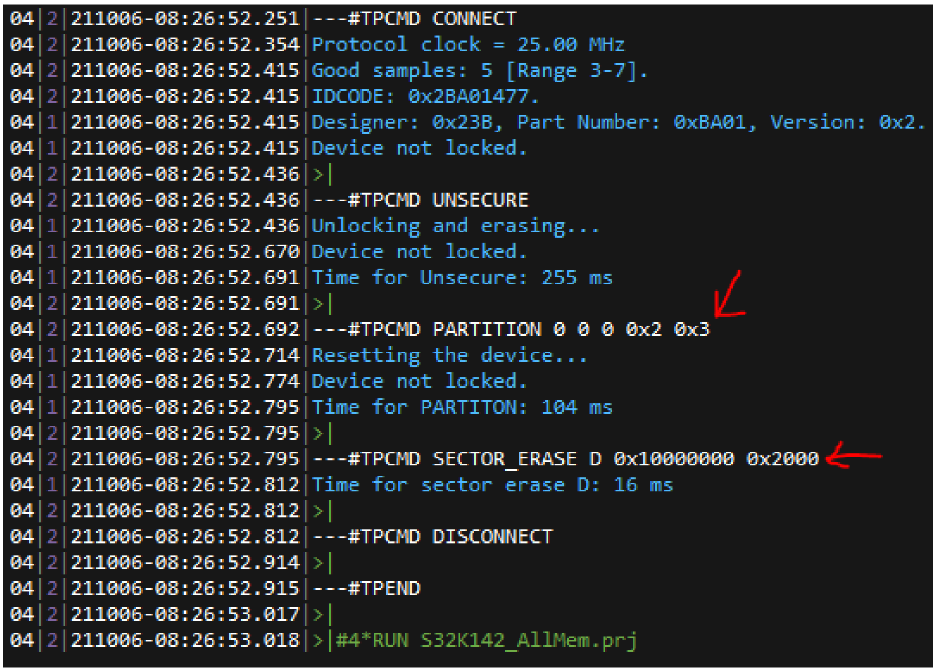
5) FlexNVM OPERATIONS vs PARTITION COMMAND:

If the **PARTITION** is performed to reserve size for **EEPROM** backup, then every flashing operation that will try to access this reserved area will fail



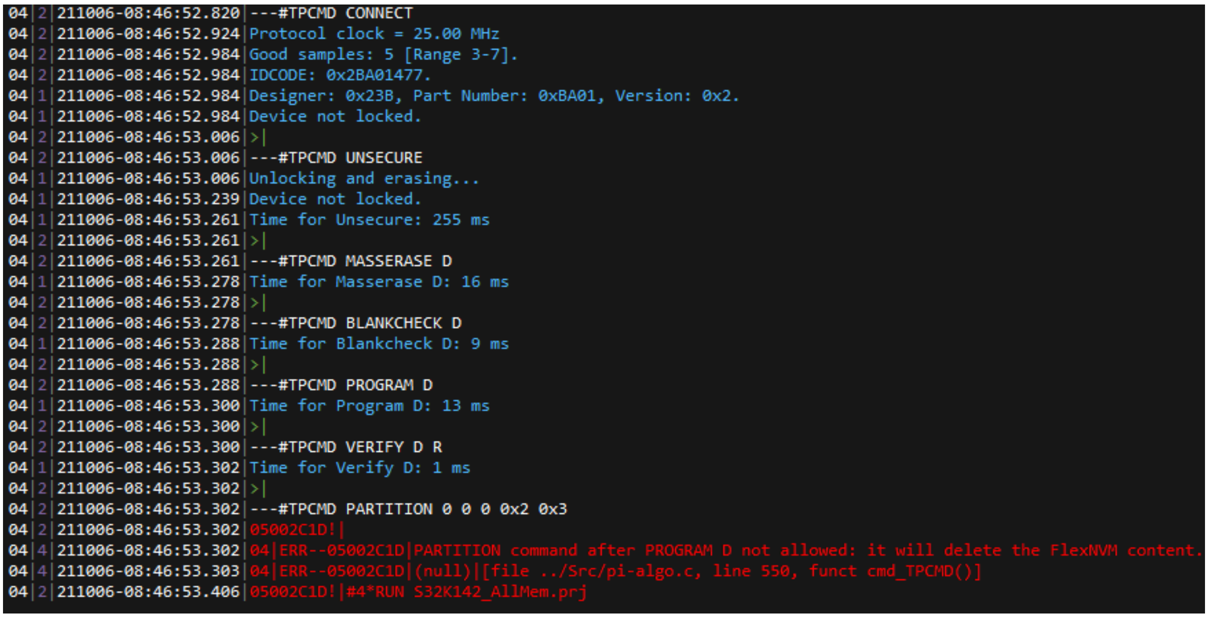
```
04|2|211006-08:23:06.784|---#TPCMD CONNECT
04|2|211006-08:23:06.843|Protocol clock = 25.00 MHz
04|2|211006-08:23:06.904|Good samples: 5 [Range 3-7].
04|2|211006-08:23:06.904|IDCODE: 0x2BA01477.
04|1|211006-08:23:06.904|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211006-08:23:06.904|Device not locked.
04|2|211006-08:23:06.931|>|
04|2|211006-08:23:06.931|---#TPCMD UNSECURE
04|1|211006-08:23:06.931|Unlocking and erasing...
04|1|211006-08:23:07.166|Device not locked.
04|1|211006-08:23:07.187|Time for Unsecure: 257 ms
04|2|211006-08:23:07.187|>|
04|2|211006-08:23:07.187|---#TPCMD PARTITION 0 0 0 0x2 0x3
04|1|211006-08:23:07.209|Resetting the device...
04|1|211006-08:23:07.270|Device not locked.
04|1|211006-08:23:07.291|Time for PARTITON: 104 ms
04|2|211006-08:23:07.291|>|
04|2|211006-08:23:07.291|---#TPCMD MASSERASE D
04|2|211006-08:23:12.292|08000044!|
04|4|211006-08:23:12.292|04|ERR--05002C1B|(null)|[file ../s32k_algo.c, line 262, funct ALGO_UK_EXEC_COMMAND()]
04|4|211006-08:23:12.292|04|ERR--05002C0A|(null)|[file ../s32k_algo.c, line 486, funct ALGO_MassErase_S32_uk()]
04|4|211006-08:23:12.292|04|ERR--05002C0A|(null)|[file ../s32k_switch.c, line 50, funct ALGO_MassErase_Sel()]
04|4|211006-08:23:12.293|04|ERR--08000044|(null)|[file ../s32k_api.c, line 196, funct CmdExe_MassErase()]
04|4|211006-08:23:12.293|04|ERR--08000044|(null)|[file ../Src/pi-algo.c, line 550, funct cmd_TPCMD()]
04|2|211006-08:23:12.396|08000044!|#4*RUN S32K142_AllMem.prj
55|2|211006-08:25:15.655|---#55*FSCRC PRJ S32K142_AllMem.prj
55|2|211006-08:25:15.656|>|
```

If the following flashing operations are not accessing the EEPROM backup reserved area, they will pass



```
04|2|211006-08:26:52.251|---#TPCMD CONNECT
04|2|211006-08:26:52.354|Protocol clock = 25.00 MHz
04|2|211006-08:26:52.415|Good samples: 5 [Range 3-7].
04|2|211006-08:26:52.415|IDCODE: 0x2BA01477.
04|1|211006-08:26:52.415|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211006-08:26:52.415|Device not locked.
04|2|211006-08:26:52.436|>|
04|2|211006-08:26:52.436|---#TPCMD UNSECURE
04|1|211006-08:26:52.436|Unlocking and erasing...
04|1|211006-08:26:52.670|Device not locked.
04|1|211006-08:26:52.691|Time for Unsecure: 255 ms
04|2|211006-08:26:52.691|>|
04|2|211006-08:26:52.692|---#TPCMD PARTITION 0 0 0 0x2 0x3
04|1|211006-08:26:52.714|Resetting the device...
04|1|211006-08:26:52.774|Device not locked.
04|1|211006-08:26:52.795|Time for PARTITON: 104 ms
04|2|211006-08:26:52.795|>|
04|2|211006-08:26:52.795|---#TPCMD SECTOR_ERASE D 0x10000000 0x2000
04|1|211006-08:26:52.812|Time for sector erase D: 16 ms
04|2|211006-08:26:52.812|>|
04|2|211006-08:26:52.812|---#TPCMD DISCONNECT
04|2|211006-08:26:52.914|>|
04|2|211006-08:26:52.915|---#TPEND
04|2|211006-08:26:53.017|>|
04|2|211006-08:26:53.018|>|#4*RUN S32K142_AllMem.prj
```

Executing the **PARTITION** command after programming the **FlexNVM** memory is not allowed since it will delete the firmware just stored inside



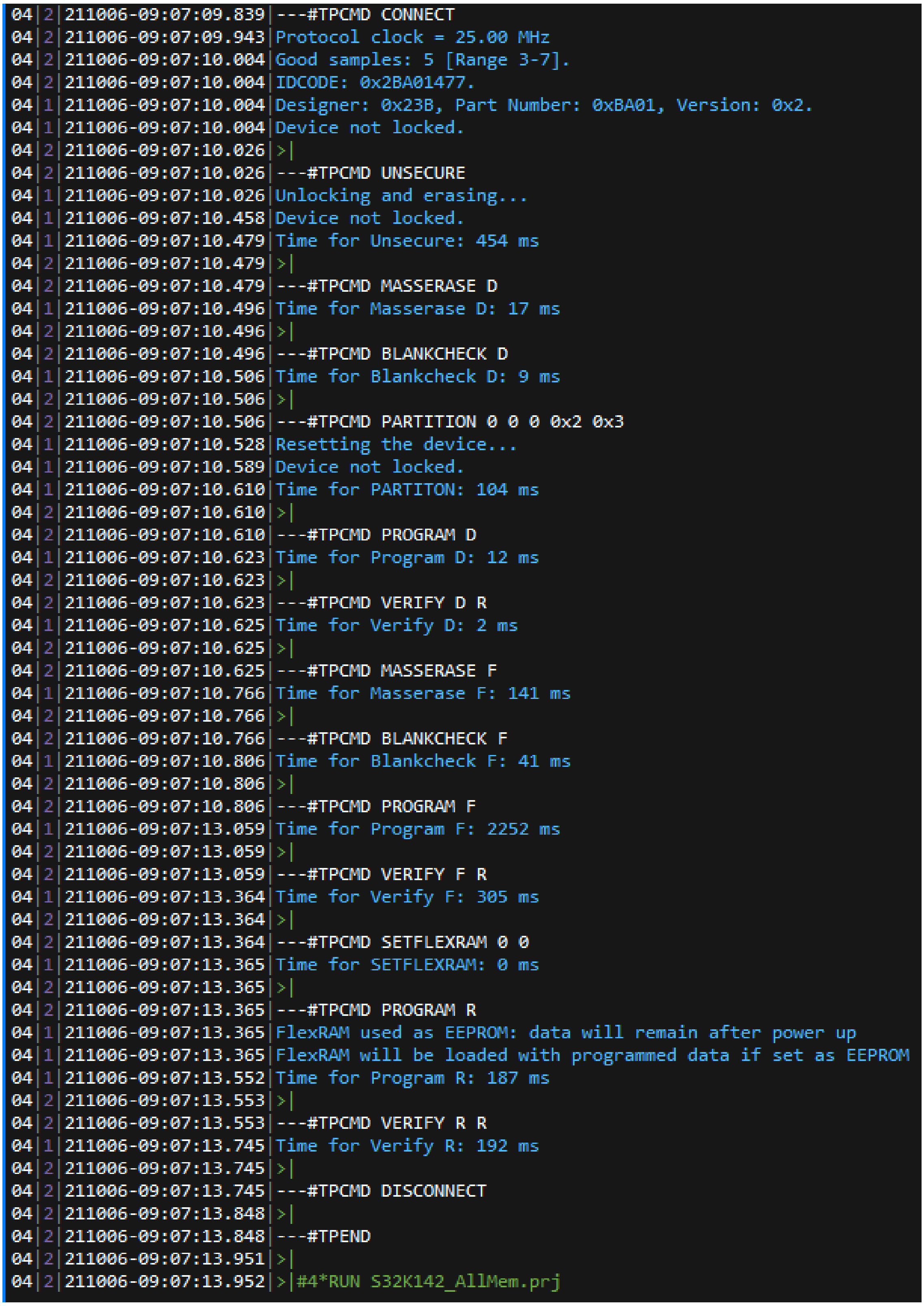
```
04|2|211006-08:46:52.820|---#TPCMD CONNECT
04|2|211006-08:46:52.924|Protocol clock = 25.00 MHz
04|2|211006-08:46:52.984|Good samples: 5 [Range 3-7].
04|2|211006-08:46:52.984|IDCODE: 0x2BA01477.
04|1|211006-08:46:52.984|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211006-08:46:52.984|Device not locked.
04|2|211006-08:46:53.006|>|
04|2|211006-08:46:53.006|---#TPCMD UNSECURE
04|1|211006-08:46:53.006|Unlocking and erasing...
04|1|211006-08:46:53.239|Device not locked.
04|1|211006-08:46:53.261|Time for Unsecure: 255 ms
04|2|211006-08:46:53.261|>|
04|2|211006-08:46:53.261|---#TPCMD MASSERASE D
04|1|211006-08:46:53.278|Time for Masserase D: 16 ms
04|2|211006-08:46:53.278|>|
04|2|211006-08:46:53.278|---#TPCMD BLANKCHECK D
04|1|211006-08:46:53.288|Time for Blankcheck D: 9 ms
04|2|211006-08:46:53.288|>|
04|2|211006-08:46:53.288|---#TPCMD PROGRAM D
04|1|211006-08:46:53.300|Time for Program D: 13 ms
04|2|211006-08:46:53.300|>|
04|2|211006-08:46:53.300|---#TPCMD VERIFY D R
04|1|211006-08:46:53.302|Time for Verify D: 1 ms
04|2|211006-08:46:53.302|>|
04|2|211006-08:46:53.302|---#TPCMD PARTITION 0 0 0 0x2 0x3
04|2|211006-08:46:53.302|05002C1D!|
04|4|211006-08:46:53.302|04|ERR--05002C1D|PARTITION command after PROGRAM D not allowed: it will delete the FlexNVM content.
04|4|211006-08:46:53.303|04|ERR--05002C1D|(null)|[file ../Src/pi-algo.c, line 550, funct cmd_TPCMD()]
04|2|211006-08:46:53.406|05002C1D!|#4*RUN S32K142_AllMem.prj
```

6) The hardest programming process to handle is when the **FLASH** firmware locks the access to the device, the **FlexNVM** needs to be partitioned for both **DFLASH** data and **EEPROM** backup and some data needs to be stored inside the **FlexRAM** memory used as **EEPROM**.

First of all let us consider the secure state of the FLASH firmware: there must be no reset after storing this firmware inside the **FLASH.** The **PARTITION** command performs a reset so it must be done before flashing (user case 3).
The **FlexNVM** is heavily influenced by the **PARTITION** (user case 5): both **MASSERASE D** and **BLANKCHECK D** must be done before if needed. After the **PARTITION** command, the only available memory of the FlexNVM will be the one reserved to the **DFLASH data.** Every attempts to access the area reserved to the EEPROM backup will fail: be sure that the flashing operations done after the PARTITION command only deals with the DFLASH part.

To program the FlexRAM, "**SET_FLEXRAM**" command needs to be executed, but this cannot be done before flashing the **FLASH** or the **FlexNVM** memories (user case 4): this is the reason why they must be done at the end of the project.

So first comes **MASSERASE D** and **BLANKCHECK D,** then **PARTITION.**
After this, flashing operations on the **FlexNVM** and on the **FLASH** memories can be performed, but we need to be sure that **EEPROM** backup reserved area in the **FlexNVM** after **PARTITION** is not accessed by any of them.
At the end FlexRAM flashing can be executed.



```
04|2|211006-09:07:09.839|---#TPCMD CONNECT
04|2|211006-09:07:09.943|Protocol clock = 25.00 MHz
04|2|211006-09:07:10.004|Good samples: 5 [Range 3-7].
04|2|211006-09:07:10.004|IDCODE: 0x2BA01477.
04|1|211006-09:07:10.004|Designer: 0x23B, Part Number: 0xBA01, Version: 0x2.
04|1|211006-09:07:10.004|Device not locked.
04|2|211006-09:07:10.026|>|
04|2|211006-09:07:10.026|---#TPCMD UNSECURE
04|1|211006-09:07:10.026|Unlocking and erasing...
04|1|211006-09:07:10.458|Device not locked.
04|1|211006-09:07:10.479|Time for Unsecure: 454 ms
04|2|211006-09:07:10.479|>|
04|2|211006-09:07:10.479|---#TPCMD MASSERASE D
04|1|211006-09:07:10.496|Time for Masserase D: 17 ms
04|2|211006-09:07:10.496|>|
04|2|211006-09:07:10.496|---#TPCMD BLANKCHECK D
04|1|211006-09:07:10.506|Time for Blankcheck D: 9 ms
04|2|211006-09:07:10.506|>|
04|2|211006-09:07:10.506|---#TPCMD PARTITION 0 0 0 0x2 0x3
04|1|211006-09:07:10.528|Resetting the device...
04|1|211006-09:07:10.589|Device not locked.
04|1|211006-09:07:10.610|Time for PARTITON: 104 ms
04|2|211006-09:07:10.610|>|
04|2|211006-09:07:10.610|---#TPCMD PROGRAM D
04|1|211006-09:07:10.623|Time for Program D: 12 ms
04|2|211006-09:07:10.623|>|
04|2|211006-09:07:10.623|---#TPCMD VERIFY D R
04|1|211006-09:07:10.625|Time for Verify D: 2 ms
04|2|211006-09:07:10.625|>|
04|2|211006-09:07:10.625|---#TPCMD MASSERASE F
04|1|211006-09:07:10.766|Time for Masserase F: 141 ms
04|2|211006-09:07:10.766|>|
04|2|211006-09:07:10.766|---#TPCMD BLANKCHECK F
04|1|211006-09:07:10.806|Time for Blankcheck F: 41 ms
04|2|211006-09:07:10.806|>|
04|2|211006-09:07:10.806|---#TPCMD PROGRAM F
04|1|211006-09:07:13.059|Time for Program F: 2252 ms
04|2|211006-09:07:13.059|>|
04|2|211006-09:07:13.059|---#TPCMD VERIFY F R
04|1|211006-09:07:13.364|Time for Verify F: 305 ms
04|2|211006-09:07:13.364|>|
04|2|211006-09:07:13.364|---#TPCMD SETFLEXRAM 0 0
04|1|211006-09:07:13.365|Time for SETFLEXRAM: 0 ms
04|2|211006-09:07:13.365|>|
04|2|211006-09:07:13.365|---#TPCMD PROGRAM R
04|1|211006-09:07:13.365|FlexRAM used as EEPROM: data will remain after power up
04|1|211006-09:07:13.365|FlexRAM will be loaded with programmed data if set as EEPROM
04|1|211006-09:07:13.552|Time for Program R: 187 ms
04|2|211006-09:07:13.553|>|
04|2|211006-09:07:13.553|---#TPCMD VERIFY R R
04|1|211006-09:07:13.745|Time for Verify R: 192 ms
04|2|211006-09:07:13.745|>|
04|2|211006-09:07:13.745|---#TPCMD DISCONNECT
04|2|211006-09:07:13.848|>|
04|2|211006-09:07:13.848|---#TPEND
04|2|211006-09:07:13.951|>|
04|2|211006-09:07:13.952|>|#4*RUN S32K142_AllMem.prj
```

S32K3xx HSE

S32K3xx HSE is a peripheral that can be configured to run its own firmware. From a programmer point of view, there are two ways to install it:

1) By downloading the HSE FW inside the FLASH and executing it
2) Hybrid approach: writing a value different from 0xFFFFFFFFFFFFFFFF at UTEST addresses [0x1B000000 – 0x1B000007], downloading HSE FW inside the FLASH and execute it

Two commands can be used to execute it:

1) RUN <time [ms]>
2) FUNCTIONAL_RESET <time [ms]>.

The choice of what methods should be used to program the HSE is up to the customer and how the FW has been designed: same applies to what command needs to be performed to start the installation process and the time needed for it.

The installation process just copies the HSE FW downloaded in FLASH to its secure memory. After that, the FLASH memory can be erased to store final application.

After having successfully installed the HSE, this peripheral will take some part of CodeFlash/DataFlash; the reserved parts and their size may vary depending on the configuration chosen and the HSE FW used and cannot be accessed by any programmer/debugger.

The CONNECT command is able to understand if the device is configured to work on FULL_MEM mode or AB_SWAP mode and the related reserved size in CodeFlsah/DataFlash that the HSE requires to work properly.

The MASSERASE F|D and BLANKCHECK F|D skips the HSE reserved sectors if the HSE peripheral has been enabled.

The usage of the HSE is not mandatory: it depends on customers requirements and purposes. These devices can be also used without it.

S32K3xx UNLOCK/LOCK

To lock an S32K3xx MCU, two steps are needed:

1) A "password" that can be flashed by the programmer for static mode
2) Lifecycle advance that requires a FW execution

If the customer requires to lock the device in static mode, it is possible to choose between an hybrid approach where the programmer flashes the 16 bytes password at address [0x1B000080 – 0x1B00008F] and its firmware advances the lifecycle or just executing a FW that installs the password and advance the lifecycle.

For the dynamic mode everything is handled by the HSE and it requires a FW execution.

FlashRunner CONNECT command is able to recognize if a device is locked and unlocks it only if the static mode lock has been chosen. Dynamic mode unlock would require an implementation activity and collaboration with the FW design team.

#TCSETPAR PWD_ADDR **<address>** is used to indicate the FRB or dynamic memory address of the password that needs to be used to unsecure the device during the CONNECT command.

S32K3xx DCFs

DCF configures certain registers of this chip during system boot while the reset signal asserts. An individual DCF record points to an internal register in the chip and the data to be written to that register.

The DCF records that customers supply must be added in a contiguous manner immediately following the factory-written DCF records. This area must never have an unprogrammed record in the series of DCF records because that is interpreted as a stop record. This is the reason why this area is not treated as a normal non-volatile memory when executing yhe related PROGRAM command. When flashing this area, the algorithm will look for the first blank space in the area, ignoring the addresses in the dynamic memory/FRB file and place the DCF records one after another as soon as it finds the first empty location. In this way, it is guranateed that the functionality of this area is never compromised.

If the customer requires it to be treaten as a normal non-volatile storage, it can execute WRITE_UTEST U <startAddress> <size> to specify the address and the size of data to be programmed. By giving as source dynamic memory or FRB, the customer can program this area from the specified address as a normal non-volatile memory and not for the function it has been designed for. The related READ_UTEST U <startAddress> <size> command allows the customer to verify these data.