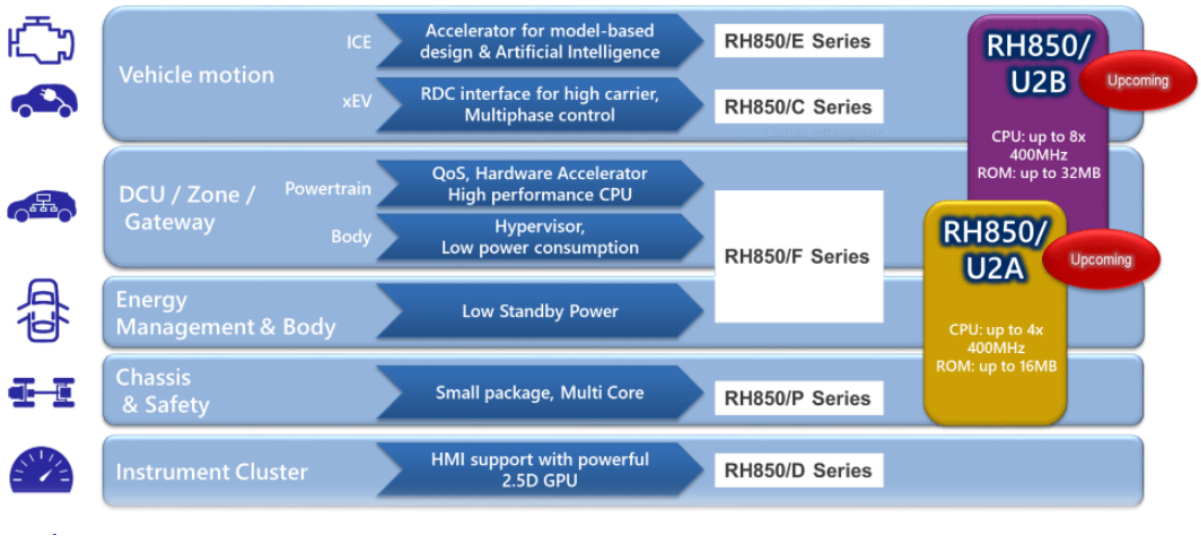
DC11593
Driver 3.40
January 2024
F. Devescovi

Interfacing FlashRunner with RH850 family MCUs (F1x, F1xx, D1x, P1x, P1xx and U2Ax series)

1. Introduction

The **`RH850 Family`** of *32-bit automotive microcontrollers* (MCUs) offers high performance balanced with very low power consumption over a wide and scalable range of products. This family offers rich functional safety and embedded security features needed for new and advanced automotive applications.



You can download the latest version of this document from this static link: [Interfacing FlashRunner with RH850 family MCUs].

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

2. Contents

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com

info@smh-tech.com

3. Tips and tricks to flash RH850 MCUs

In this section, we want to explain which operations are performed in the target devices and how these operations work. This knowledge may help you to optimize even more your application.

FlashRunner uses **CSI** (**C**locked **S**erial **I**nterface, with **H**and**S**hake) to access to the memories of these devices.

CSI/HS is a *3-wired serial synchronous interface* that makes use of the following signals:

1. **FPCK** (Serial clock)
2. **FPDR** (TxD, programmer side)
3. **FPDT** (RxD, programmer side)

In addition to these protocol lines, FlashRunner also need to control the **FLMD0** and **FLMD1** lines for activating the factory shipped bootloader upon **RST** signal release. Reset line is very important and very critical as well. It is crucial to have no interferences on the reset line because it could cause strange behaviours of the device and the operations could likely fail.

The following option must be also properly set according to board's configuration.

```
#TCSETPAR FOSC <xtal_freq>
```

Programming algorithm will automatically choose and handshake the maximum available CPU frequency so to optimize performances.
Caution: A wrong setting of oscillator frequency parameter could affect communication stability or prevent communication to happen at all.

Once the bootloader has been activated, communication between FlashRunner and the MCU takes place by means of Request Frames (issued by the programmer) and Response Frames (issued by the target device).

The above mentioned serial approach is the same of **Renesas Flash Programmer** and/or **FP6 Terminal**.

So said, the variety of supported commands depends on the factory shipped bootloader, that is, the ones implemented by Renesas.

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale
Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255
D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021
→smh-tech.com
info@smh-tech.com

Let us now analyse the standard commands:

- **MASSERASE**

  This command erases all the contents of selected memory.

  Supported memories for this command are:

-	Code Flash	[**F**]	Both Flash Technologies
-	Data Flash	[**E**]	Both Flash Technologies
-	User Boot Area	[**U**]	Both Flash Technologies
-	Extended Data Area	[**D**]	RV28F Flash Technology
-	Chip	[**C**]	Both Flash Technologies

  Optionally, it is also possible to send additional parameters to this command to select only some blocks of memory to be erased. This can be useful to reduce the erase time and it could be needed if the user wants to preserve some data into the memory and partially reprogram it. See chapter **Block Erase** for more information.
  Masserase duration depends only on the target device characteristics and there is not much that we can do to improve this. Typically, it is extremely long, even longer than all the other operations combined together.
  The suggestion is to skip this operation in case the device is virgin (i.e. factory shipped). This can be done using a conditional script based on the result of the blankcheck operation:

```
#IFERR TPCMD BLANKCHECK F
#THEN TPCMD MASSERASE F
#THEN TPCMD BLANKCHECK F
```

  Care must be taken on target devices with **ICU-S** (Intelligent Cryptographic Unit).
  On those MCUs the last **1KB** or **2KB** of Data Flash are reserved for the **cryptographic keys** of **ICU-S**.
  Once the ICU-S has been activated, **#TPCMD MASSERASE E** command will skip that reserved area, since it is not meant to be an user available area.
  In case ICU-S area can be erased (and that depends on target device), it is still possible to erase it with the following command:

```
#TPCMD MASSERASE C
```

  This command will fully erase **Code Flash**, **Data Flash** and **User Boot Area**, so once issued, it could then be followed by a **#TPCMD BLANKCHECK** command for all the target memories and no further masserases are necessary.
  To know whether **ICU-S** is available on target device, activated and erasable the following option can be used:

```
#TCSETPAR PRINT_INFO_LOG <YES/NO>
```


HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

When set to **YES** following information will be added to the log when connecting:

```
01|3|231120-18:00:36.743|Code Flash granularity is 256 bytes
01|3|231120-18:00:36.743|Data Flash granularity is 4 bytes
01|3|231120-18:00:36.743|ICU-S Region Area is last 2KB of Data Flash
01|3|231120-18:00:36.744|ICU-S Region Area Erasable
01|3|231120-18:00:36.744|ICU-S is OFF
```

- **BLANKCHECK**

  This command checks that all the bits of the selected memory are set to 1 (erased state).

  Supported memories for this command are:

  - Code Flash [**F**] Both Flash Technologies
  - Data Flash [**E**] Both Flash Technologies
  - User Boot Area [**U**] Both Flash Technologies
  - Extended Data Area [**D**] RV28F Flash Technology

  This operation is executed internally by the device and it is typically extremely fast.
  Optionally, it is also possible to send two additional parameters to this command: the address from where to start checking and the number of bytes to check.
  These parameters will be previuosly checked by the algorithm to prevent alignment issues (granularity, which can also be checked when).
  The duration of this operation depends on the size of the memory to check and communication speed doesn't speed up the process.

  Care must be taken for devices having **ICU-S** (Intelligent Cryptographic Unit).
  When a blankcheck of Data Flash on these devices is performed, if no additional parameters are provided, ICU-S reserved area blankcheck is skipped by the algorithm.

- **PROGRAM**

  This command takes the Customer's data from the FRB file and programs them into selected memory.

  Supported memories for this command are:

  - Code Flash [**F**] Both Flash Technologies
  - Data Flash [**E**] Both Flash Technologies
  - User Boot Area [**U**] Both Flash Technologies
  - Option Bytes [**O**] RV40F Flash Technology
  - Configuration Setting Area [**A**] RV28F Flash Technology
  - Security Setting Area [**S**] RV28F Flash Technology
  - Extended Data Area [**D**] RV28F Flash Technology

  Optionally, it is also possible to send two additional parameters to this command: the address from where to start programming and the number of bytes to be programmed (see example).

```
#TPCMD PROGRAM F <start_address> <size>
```

As above said, these parameters will be previuosly checked by the algorithm to prevent alignment issues (granularity mismatch).

The duration of this command depends on many factors which are all important:

- The target device characteristics, in other words, how much time is needed to write data into the memory.
- The target device frequency because it determines how fast the device is able to process incoming data.
- Protocol frequency because it sets the bitrate of the communication between FlashRunner and the target device.

Care must be taken when programming Code Flash, User Boot Area and Data Flash because of the following two options:

```
#TCSETPAR FILL_CODE_FLASH <YES/NO>
#TCSETPAR FILL_DATA_FLASH <YES/NO>
```

They default to **NO** and determine whether the holes on the source file should be also programmed with blank value (0xFF) or corresponding memory locations have to be left in erased state. It makes difference because of the ECC calculation.
User should be aware of that.

- **VERIFY READOUT**
  This command checks that data contained in the memory of the device corresponds to FRB data.

  Supported memories for this command are:

-	Code Flash	[**F**]	Both Flash Technologies
-	Data Flash	[**E**]	Both Flash Technologies
-	User Boot Area	[**U**]	Both Flash Technologies
-	Option Bytes	[**O**]	RV40F Flash Technology
-	Configuration Setting Area	[**A**]	RV28F Flash Technology
-	Security Setting Area	[**S**]	RV28F Flash Technology
-	Extended Data Area	[**D**]	RV28F Flash Technology

  Optionally, it is also possible to send two additional parameters to this command: the address from where to start checking and the number of bytes to check (see example).

```
#TPCMD VERIFY F R <start_address> <size>
```

As above said, these parameters will be previuosly checked by the algorithm to prevent alignment issues (granularity mismatch).

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

This command works exactly like the program command with the only exception that the device reads (instead of writing) data from the memory and compares that with the data received from FlashRunner.
Same as for programming, also when verify Code Flash, User Boot Area and Data Flash, the following parameters will be taken into account.

```
#TCSETPAR FILL_CODE_FLASH <YES/NO>
#TCSETPAR FILL_DATA_FLASH <YES/NO>
```

- **VERIFY CHECKSUM**

This command asks the target device to calculate the **32-bits CRC (Cyclic Redundancy Check)** of selected memory region, meanwhile, the FlashRunner calculates the expected 32-bits CRC according to FRB data and then the two values are compared.

Supported memories for this command are:

-	Code Flash	[**F**]	Both Flash Technologies
-	Data Flash	[**E**]	Both Flash Technologies
-	User boot Area	[**U**]	Both Flash Technologies
-	Configuration Setting Area	[**A**]	RV28F Flash Technology
-	Security Setting Area	[**S**]	RV28F Flash Technology
-	Extended Data Area	[**D**]	RV28F Flash Technology

On **RH850/F1x**, **RH850/F1xx**, **RH850/D1x**, **RH850/P1x** and **RH850/P1xx** series, verify checksum of Data Flash [**E**] is available only if:

```
#TCSETPAR FILL_DATA_FLASH YES
```

Verify checksum operation is executed internally by the device and it is typically faster than the readout method. An example of this command is provided below:

```
#TPCMD VERIFY F S
```

Optionally, it is also possible to send up to three additional parameters to this command: the address from where to start the checksum calculation, the number of bytes to consider in the calculation, and the expected checksum value. This will result faster because the FlashRunner does not need to spend time for calculating the checksum of the FRB file.

```
#TPCMD VERIFY F S <start_address> <size> <cks>
```

To get the value to use as the expected checksum parameter, you can use the following command

```
#TPCMD CALC_FRB_CHKSUM
```


HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

which can be executed by FlashRunner without being connected to the target device because it is just an internal calculation and it will return the commands to use in the **real-time log**.

We also have an additional feature about checksum: some customers requested to return the checksum of the data when performing the verify checksum command and we gave them the possibility to enable this feature by manually setting the following parameter.

```
#TCSETPAR PRINT_CHKSUM_ON_VERIFY YES
```

This parameter does not give any benefits in terms of performance, in fact, this feature aims to give additional feedback to the customer's application which can verify that FlashRunner is working with the expected data. For example, running a project which has the checksum print enabled, the FlashRunner will return the following string on **Terminal** when executing the verify checksum:

```
#1*RUN TEST_R7F701760.prj
01|Start = 0x00000000, Size = 0x00200000, CheckSum = 0xDCF338E4
01|#1*RUN TEST_R7F701760.prj
01|>
```

Note: the algorithm used to calculate the **32-bit CRC** is the one defined by Renesas.

After this explanation should be clear what can be improved by the user, what can be improved by SMH, and what cannot be improved because it depends only on the characteristics of the target device.

Warning: performing the program and verify commands using an FRB which does not contain any data for the selected memory region will return pass. The driver has been designed in this way to be more flexible so, basically, if a customer gives no data to program and verify, then the driver does not perform any operation and it just returns pass after completing the research for the data.

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com

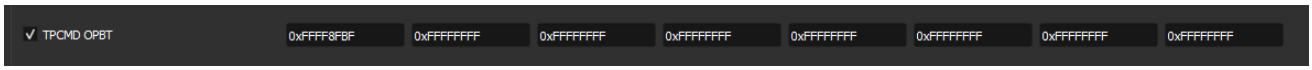
info@smh-tech.com

4. Option Bytes

All **RH850** devices (both **RV40F** and **RV28F** Flash Technologies) have a set of non-volatile registers where boot relevant data are taken from and loaded into corresponding volatile registers once the reset is released. They are known as **Option Bytes** (OPBT).

RV40F flash technology

There are a couple of ways to program Option Bytes.
The easiest one is to choose from Project Wizard the following additional command:



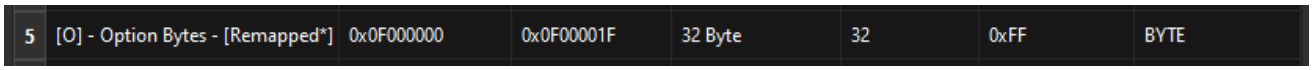
Once selected the following command will be added to the project:

```
#TPCMD OPBT 0xFFFF8FBF 0xFFFFFFFF 0xFFFFFFFF 0xFFFFFFFF 0xFFFFFFFF 0xFFFFFFFF 0xFFFFFFFF 0xFFFFFFFF
```

The above command will program and verify **OPBT0-OPBT7** interval of any of **RV40F Flash Technology** devices.

NOTE: On devices with **ICU-M**, if it is enabled, verify will be skipped by the algorithm and verify should be done separately being aware that some of them will be read as **0s** due brecause of the **ICU-M** itself.

In case Option Bytes should be dynamically programmed (or traditionally programmed) we have reserved a region on device's memory map whose size is **32 bytes** (that is, 8 x 4B, given 4B as OPBT size in bytes).



Note that Start Address of such memory region doesn't correspond to actual address of device's Option Bytes so we can state that it is "**virtual**".
Once data being placed in that region they will be programmed by using:

```
#TPCMD PROGRAM O or #TPCMD PROGRAM O <start_address> <size>
```

Be careful of data endianness, data of each OPBT must be input in little endian order (i.e to program 0xFFFF8FBF the stream should be 0xBF 0x8F 0xFF 0xFF)

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

RV28F flash technology

On **RH850/U2A** series devices Option Bytes must be programmed normally using:

```
#TPCMD PROGRAM A  or #TPCMD PROGRAM A <start_address> <size>
```

since they effectively belong to Configuration Setting Area [**A**].

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com

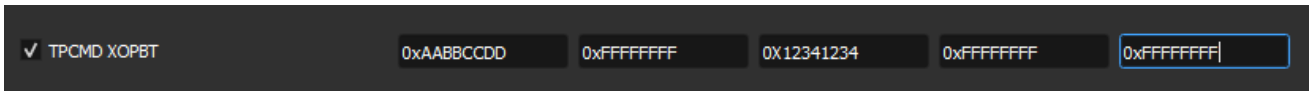
info@smh-tech.com

5. Extended Option Bytes

Some **RH850** devices series with **RV40F Flash Technology** also have an additional set of Option Bytes which can be programmed (and in some cases, also verified).
They are known as **Extended Option Bytes** (XOPBT) and there are a couple of ways to program them.

Caution: Since the Extended Option Bytes contain important settings for the MCU, take care with the data to be programmed if perform this operation. Don't use this option unless you have a particular reason to do otherwise.

The easiest way is to choose from Project Wizard the following additional command:



According to the **RH850** series the above command will program:

1) [**OPBT8**] **Standard RH850** with **ICU-S** (Intelligent Cryptographic Unit)
2) [**OPBT8-OPBT12**] **Standard RH850** with **ICU-M** (Intelligent Cryptographic Unit)
3) [**OPBT8-OPBT15**] **Custom RH850**, such as **P1xC** series devices

Once selected the following command will be added to the project (in following example we will consider OPBT8):

```
#TPCMD XOPBT 0xAABBCCDD
```

With the above approach, verify of Extended Option Bytes is, at this moment, not performed within this additional command.

In case Extended Option Bytes should be dynamically programmed (or traditionally programmed) we have reserved a region on device's memory map whose size is:

1) **4 bytes** (that is, 1 x 4B, given 4B as an OPBT size) for devices with **ICU-S**.

6	[O] - Option Bytes - [Remapped*]	0x0F000020	0x0F000023	4 Byte	4	0xFF	BYTE

2) **20 bytes** (that is, 5 x 4B, given 4B as an OPBT size) for devices with **ICU-M**.

6	[O] - Option Bytes - [Remapped*]	0x0F000020	0x0F000033	20 Byte	20	0xFF	BYTE

3) **32 bytes** (that is, 8 x 4B, given 4B as an OPBT size) for **P1xC** series devices.

5	[O] - Option Bytes - [Remapped*]	0x0F000020	0x0F00003F	32 Byte	32	0xFF	BYTE

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

Note that Start Address of such memory region doesn't correspond to actual address of device's Extended Option Bytes so we can state that it is "**`virtual`**".

Be careful of data endianness, data of each OPBT must be input in little endian order (i.e to program `0xAABBCCDD` the stream should be `0xDD 0xCC 0xBB 0xAA`)

Once data being placed in that region they will be programmed using the following command:

```
#TPCMD PROGRAM O or #TPCMD PROGRAM O <start_address> <size>
```

Where allowed it is also possible to verify the Extended Option Bytes using the following command:

```
#TPCMD VERIFY O R or #TPCMD VERIFY O R <start_addr> <size>
```


HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com

info@smh-tech.com

6. Security Management

RV40F flash technology

As above mentioned, according to RH850 series name, there are two different kinds of devices:

- Standard devices (F1x, F1xx, P1x, P1xx, R1x)
- Custom devices (P1xC)

In **Standard RH850** devices Security Management involves a couple of 16 bytes keys [**IDCODE**].
In **Custom RH850** devices Security Management involves a couple of three 32 bytes each keys [**IDCODE**, **CFPEID**, **DFPEID**].

The above approach allows dynamic management of passwords (i.e. CODES) by using DYNAMIC MEMORY. In fact similarly to Option Bytes Management also for **Security Management** there is a "**virtual**" memory area named **P** that is suitable for entering **Authentication Codes** (if required) and setting **Security Codes** (if needed).

Dedicated memory area address ranges differ between **Standard** and **Custom** devices.

For **Standard RH850** devices, 2 regions are available:

1) **16 bytes** for **Authentication Code** when serial programming is protected:

3	[P] - ID Code - [Remapped*]	0x0E000000	0x0E00000F	16 Byte	16	0xFF	BYTE

2) **16 bytes** for storing **Security Code** to be used whenever the following command has to be performed.

```
#TPCMD SET_SECURITY_CODE <A|C>
```

A = Authentication protection mode
C = Command protection mode

4	[P] - ID Code - [Remapped*]	0x0E000010	0x0E00001F	16 Byte	16	0xFF	BYTE

It is also possible to prevent **Erase**, **Write** and **Read** commands to be executed once protection being enabled through the following command:

```
#TPCMD PROTECT R W E
```

R = Read protection
W = Write protection
E = Erase protection

UNIVERSAL PRODUCTION IN-SYSTEM PROGRAMMING

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

Caution: Once any of the functions for disabling commands has been set for an MCU, some of the security settings cannot be reversed.
To reverse the security settings for the MCU, select **#TPCMD UNPROTECT** to clear the flash options.

For **Custom RH850** devices, 2 regions are also available:

1) **96 bytes** for **Authentication Codes** when Serial Programming is protected:

2	[P] - Authentication Area - [Remapped*]	0x0E000000	0x0E00005F	96 Byte	32	0xFF	BYTE

This region is further composed into the following fields:

IDCODE (32 bytes)	CFPEID (32 bytes)	DFPEID (32 bytes)
[0xE000000 : 0xE00001F]	*[0xE000020 : 0xE00003F]*	*[0xE000040 : 0xE00005F]*

2) **96 bytes** for storing **Security Codes** whenever the following command has to be performed.

```
#TPCMD SET_SECURITY_CODE <A|C>
```

3	[P] - Authentication Area - [Remapped*]	0x0E000060	0x0E0000BF	96 Byte	32	0xFF	BYTE

This region is further composed into the following fields:

IDCODE (32 bytes)	CFPEID (32 bytes)	DFPEID (32 bytes)
[0xE000060 : 0xE00007F]	*[0xE000080 : 0xE00009F]*	*[0xE0000A0 : 0xE0000BF]*

If Authentication is required on "**programming entry**" and no Authentication Codes have been given an all **0xFFs** code will be used by default.

If Authentication Codes and Security Codes are not defined inside a source file setup a variable data area on related regions by using **DYNMEMSET** command.

RV28F flash technology

Differently from **RV40F flash technology** devices those group of devices have several **IDCODE** values that are used to protect different storage areas from **Erase** and **Write** operations. Programming desired **IDCODE** is very simple because you just have to program security related data on **Security Settings Area** through

```
#TPCMD PROGRAM S
```


UNIVERSAL PRODUCTION IN-SYSTEM PROGRAMMING

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

Similarly to RV40F Flash Technology group of devices there is a "**virtual**" memory region named **P** that is suitable for entering **Authentication Codes** (if required by programming flow). One **256 bytes** region is available (the one for unlocking locked features) and it is mapped as follows:

7	[P] - ID Code - [Remapped*]	0x0E000000	0x0E00001F	32 Byte	32	0xFF	BYTE
8	[P] - ID Code - [Remapped*]	0x0E000020	0x0E00003F	32 Byte	32	0xFF	BYTE
9	[P] - ID Code - [Remapped*]	0x0E000040	0x0E00005F	32 Byte	32	0xFF	BYTE
10	[P] - ID Code - [Remapped*]	0x0E000060	0x0E00007F	32 Byte	32	0xFF	BYTE
11	[P] - ID Code - [Remapped*]	0x0E000080	0x0E00009F	32 Byte	32	0xFF	BYTE
12	[P] - ID Code - [Remapped*]	0x0E0000A0	0x0E0000BF	32 Byte	32	0xFF	BYTE
13	[P] - ID Code - [Remapped*]	0x0E0000C0	0x0E0000DF	32 Byte	32	0xFF	BYTE
14	[P] - ID Code - [Remapped*]	0x0E0000E0	0x0E0000FF	32 Byte	32	0xFF	BYTE

The following table links each memory entry to its corresponding ID CODE.

7	[0xE000000 – 0xE00001F]	ID Code
8	[0xE000020 – 0xE00003F]	Data Flash ID
9	[0xE000040 – 0xE00005F]	Customer ID A
10	[0xE000060 – 0xE00007F]	Customer ID B
11	[0xE000080 – 0xE00009F]	Customer ID C
12	[0xE0000A0 – 0xE0000BF]	OCD ID
13	[0xE0000C0 – 0xE0000DF]	RHSFIF ID
14	[0xE0000E0 – 0xE0000FF]	C-TEST ID

If no **ID code** has been provided for authentication an all **0xFFs** code will be used by default. When other authentication codes are not set, related authentication command will be skipped by the algorithm.

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

7. ICU-S Management

The **ICU_S** is a region of Data Flash [**E**] available on some RH850 devices with RV40F Flash Technology.

This **ICU_S** region size is device dependant:

- Size 0KB -> *Not present*
- Size 1KB -> *ICU Region area is last 1KB of Data Flash*
- Size 2KB -> *ICU Region area is last 2KB of Data Flash*

The **ICU_S** region can be in *Enabled* or *Disabled* state, use

```
#TPCMD CHECK_ICU_S
```

command for retrieving status information.

The **ICU_S** region can be **Activated** by performing

```
#TPCMD VALIDATE_ICU_S
```

Power on Reset (PoR) is necessary in order activation to be effective.

Furthermore **ICU-S** region of Data Flash must be programmed with **all 0xFFs** (or random data) before this command to be executed (i.e. it shouldn't be in erased state).

The **ICU_S** region can be **Deactivated** performing

```
#TPCMD UNPROTECT
```

NOTE: Only for **RH850/F1K** series, once **ICU_S** has been enabled, it's no longer possible to use **UNPROTECT** command. This is because **ICU_S Erase** internal command is not supported anymore by *Renesas*.

If **ICU_S** region is enabled, it cannot be programmed anymore, unless erased and disabled.

In those cases **#TPCMD UNPROTECT** before **#TPCMD PROGRAM E** is mandatory. (Except RH850/F1K Family)

UNIVERSAL PRODUCTION IN-SYSTEM PROGRAMMING

HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com
info@smh-tech.com

8. Block Erase

It is also possible to partially erase the target device's memory by using the following command:

```
#TPCMD BLOCK_ERASE <mem_type> <start_address> <size>
```

`start_address` and **`size`** parameters should be given so that they are block aligned.

Next example hopefully will clarify that.

Region	Start	End	Size	Lockbit	OTP
Block 4	0x00008000	0x00009FFF	8 KB	☐	☐
Block 5	0x0000A000	0x0000BFFF	8 KB	☐	☐
Block 6	0x0000C000	0x0000DFFF	8 KB	☐	☐
Block 7	0x0000E000	0x0000FFFF	8 KB	☐	☐
Block 8	0x00010000	0x00017FFF	32 KB	☐	☐
Block 9	0x00018000	0x0001FFFF	32 KB	☐	☐
Block 10	0x00020000	0x00027FFF	32 KB	☐	☐
Block 11	0x00028000	0x0002FFFF	32 KB	☐	☐
Block 12	0x00030000	0x00037FFF	32 KB	☐	☐
Block 13	0x00038000	0x0003FFFF	32 KB	☐	☐
Block 14	0x00040000	0x00047FFF	32 KB	☐	☐
Block 15	0x00048000	0x0004FFFF	32 KB	☐	☐

We want to erase only the **Code Flash** blocks **5** to **12**.

According to what we have above stated the corresponding FlashRunning command would then be:

```
#TPCMD BLOCK_ERASE F 0xA000 0x2E000
```


HQ and Registered Office
Via Giovanni Agnelli 1
33083 Villotta di Chions (PN) Italy
Società Unipersonale

Capitale sociale €102.040
P.I. 01697470936
C.F. 01697470936
REA PN-97255

D-U-N-S® 51-724-9350
T + 39 0434 421 111
F + 39 0434 639 021

→smh-tech.com

info@smh-tech.com