NXP Semiconductors
Application Note

Document Number: AN12092
Rev. 0, February, 2018

Using the PASS module in MPC5748G to implement password-based protection for flash and debugger access

by: NXP Semiconductors

1 Introduction

Using the PASS Module in the MPC5748G or MPC5746C families to activate the password-based protection features of the device (such as debugger interface deactivation and locking flash array program-erase permissions) is a process that involves several steps. These can be divided into two categories:

1. Programming values in UTEST flash memory during the development phase. These include:
   - Setting-up Device Configuration Format (DCF) records that define the out-of-reset behavior of these protection mechanisms.
   - Setting-up a DCF record that modifies the device censorship status
   - Programming values in UTEST flash to modify the device lifecycle status
   - Programming valid passwords in UTEST flash.
2. Writing to the PASS module lock registers to temporarily unlock and reconfigure these settings during runtime, if desired.

This application note is intended to provide a summary of these steps, and function as a practical reference for developers implementing this in code. For the full-length description of all these mechanisms, the reader might want to refer to the following chapters of the MPC5748G Reference manual:

- *Chapter 11, Device Configuration Format (DCF) Records*
- *Chapter 80, Password and Device Security Module (PASS)*
- *Chapter 82, Flash Memory Programming and Configuration*

Contents

2 About passwords

The MPC5748G allows developers to define 5 passwords: one JTAG password used for enabling/disabling the JTAG interface, and four “multiple-purpose” passwords that can be used to unlock various features such as program and erase protection for flash blocks.

These five passwords are to be programmed by the user in a specific area of UTEST flash, shown below. Each password is 256 bits long. When the device's lifecycle is matured from *customer_delivery* state to a more advanced state, this password region in UTEST flash becomes read protected.

Start Address	End Address	Allocated Size [bytes]	Decription
0x004000C8	0x004000FF	56	Reserved
0x00400100	0x00400103	4	Test Mode Overide Passcode
0x00400104	0x0040011F	28	Reserved
0x00400120	0x0040013F	32	JTAG Password
0x00400140	0x0040015F	32	PASS Password Group 0
0x00400160	0x0040017F	32	PASS Password Group 1
0x00400180	0x0040019F	32	PASS Password Group 2
0x004001A0	0x004001BF	32	PASS Password Group 3
0x004001C0	0x004001DF	32	Reserved
0x004001E0	0x004001FF	32	Reserved
0x00400200	0x0040020F	16	Lifecycle Slot 0 - MCU_PROD
0x00400210	0x0040021F	16	Lifecycle Slot 1 - CUST_DEL
0x00400220	0x0040022F	16	Lifecycle Slot 2 - OEM_PROD
0x00400230	0x0040023F	16	Lifecycle Slot 3 - IN_FIELD
0x00400240	0x0040024F	16	Lifecycle Slot 4 - FA
0x00400250	0x004002FF	176	Reserved
0x00400300	0x00400307	8	DCF Start Record
0x00400308 [1, 2]	0x00400347	64	Reserved
0x00400348	0x00400FFF	3256	DCF Records
0x00401000	0x00403FFF	12288	Reserved for customer OTP data

Figure 1. UTEST memory map

These passwords allow to unlock, and therefore configure, a set of registers named PASS_LOCKx_PGn.

Precautions:

The user must be aware that the UTEST flash is a one-time-programmable (OTP) region. Once the passwords are programmed, they cannot be modified.

3 About PASS module

It is essential to be familiar with the fields of these registers to understand how the PASS module functions. A detailed view of these registers, and the mapping of flash blocks associated to each bit in the registers is provided in section *80.1.1 PASS_LOCKx_PGn register bit mapping* of the MP5748G (or MPC5746C) Reference Manual.

Most of the bits inside the PASS_LOCKx_PGn registers control the write-erase protection of the internal flash array down to a per-block granularity. There is one 1-bit field for every block of flash. If a flash block has its associated lock bit set inside a PASS_LOCKx_PGn register, it will be protected against write or erase attempts.

There are 4 lock registers, PASS_LOCK0_PGn, PASS_LOCK1_PGn, PASS_LOCK2_PGn, PASS_LOCK3_PGn. The first three registers are filled exclusively with bits that control the write-erase protection of individual flash blocks. See table below to see how bits are mapped to flash addresses.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
PASS_LOCK0_PGn	0x00400000	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	0x00620000	0x00FF0000	0x00610000	0x00FE0000	0x00FD8000	0x00FD0000	0x00FC8000	0x00FC0000	0x00404000	0x00F8C000	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	0x00FB8000	0x00FB0000	0x00FAC000	0x00FA8000	0x00FA4000	0x00FA0000	0x00F9C000	0x00F98000	0x00F94000	0x00F90000
RWW partition	0						1	1	0	0	1	1	0	0	1	0							3	2	3	3	3	3	2	2	2	2
PASS_LOCK1_PGn	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	0x00F84000	0x00F80000
RWW partition																															5	4
PASS_LOCK2_PGn	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	0x01540000	0x01500000	0x014C0000	0x01480000	0x01440000	0x01400000	0x013C0000	0x01380000	0x01340000	0x01300000	0x012C0000	0x01280000	0x01240000	0x01200000	0x011C0000	0x01180000	0x01140000	0x01100000	0x010C0000	0x01080000	0x01040000	0x01000000
RWW partition											9	9	9	8	8	8	7	7	7	7	7	7	7	7	6	6	6	6	6	6	6	6
PASS_LOCK3_PGn	PG	DBL	MO	<reserved>	MSTR				<reserved>	<reserved>	<reserved>	HSM Dflash RL	HSM Cflash RL	<reserved>	Cflash RL	UTEST RL	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>	<reserved>
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

16kB Flash block
32 kB Flash block
64kB Flash block
256 kB Flash block
Misc. Control

Figure 3. PASS_LOCKx_PGn flash block mapping

Note that these bits only control the possibility to *reprogram* specific flash blocks, but these flash blocks remain *readable*. The last register (PASS_LOCK**3**_PGn) has in addition, miscellaneous fields that control the activation of the debugger interface (DBL bit), and the “read protection” feature for different regions of flash (RLx bits).

These four PASS_LOCKx_PGn registers are then replicated 4 times (hence the suffix “n” in the name), to form a total of 16 registers. We therefore speak of 4 identical groups, called “password groups”.

The purpose of having 4 groups of registers, where each group controls identical settings, is to provide flexibility and granularity on what a password unlocks: The final state of protection will depend on the settings of its associated lock bit in *all four* of the groups: It is a logical OR function.

For example, if a flash block is write-erase protected in one password group, this protection is activated irrespective of the state of that bit in the other password groups. Various levels of protection can be implemented with this scheme: If a bit is locked in two groups, two passwords are needed to unlock it. If a bit is locked in all password groups, all four passwords are needed to unlock it.

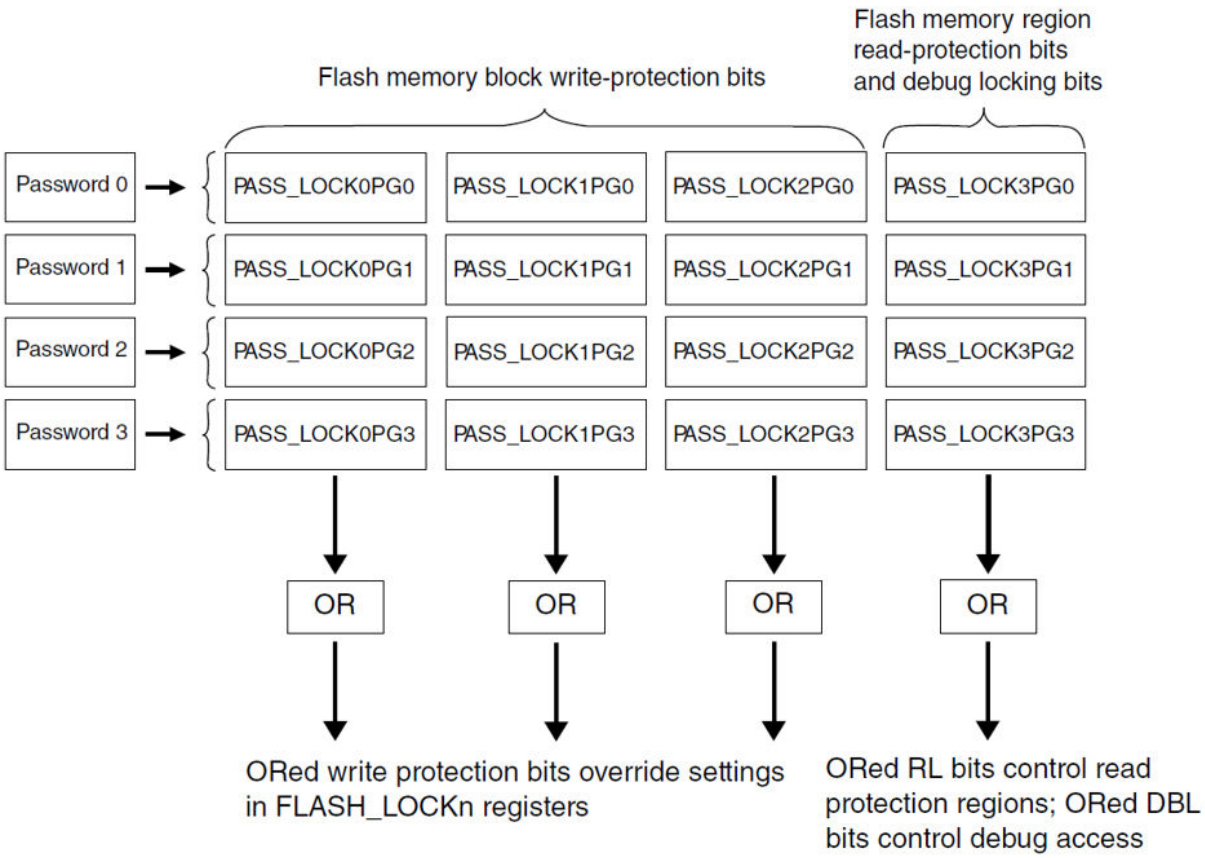


Figure 4. Secure write protection

With this four-password approach, the flash memory space can be segmented in a way that allows various programmers different flash access privileges. For example, the bootloader developer can program his application while lacking permission to erase/program the MCAL drivers and the OS; the MCAL developer can program his code but cannot erase or program the code from the bootloader and the OS vendor, and so on.

4 About unlocking PASS_LOCKx_PGn registers

Each group of PASS_LOCKx_PGn registers can be modified by providing the appropriate password to the PASS Module.

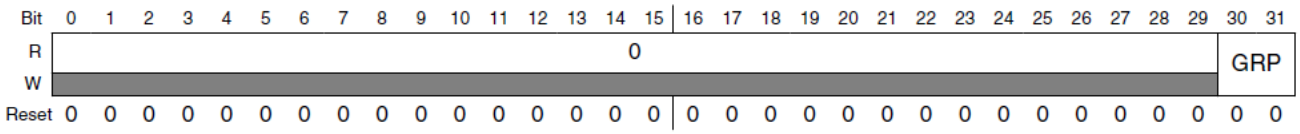
For example, PASS_LOCKx_PG**1** registers can be modified by providing the PASS module with the 256-bit password pre-programmed at address 0x00400160 in UTEST flash, which is the “PASS Password Group 1” location.

The steps needed to provide the password to the PASS module are the following:

1. Write the number of the password group (1 in this example) to the PASS_CHSEL register.
2. Immediately after, write the appropriate password to the PASS_CIN1 register.

   PASS_CINn registers are 32-bit registers. The user will need to do 8 consecutive 32-bit writes to the PASS_CIN1 register in order to enter all 256 bits of the password.

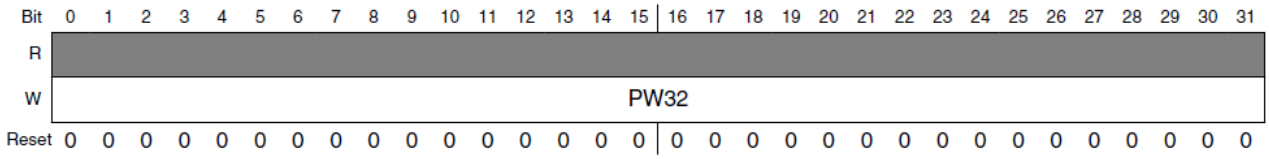
Address: FFFF_4000h base + 8h offset = FFFF_4008h



Field	Description
0–29 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30–31 GRP	Password Group to unlock 00 Password Group 0 01 Password Group 1 10 Password Group 2 11 Password Group 3

Figure 5. PASS_CHSEL register

Address: FFFF_4000h base + 20h offset + (4d × i), where i=0d to 7d



Field	Description
0–31 PW32	32 bits of the 256-bit password challenge. Only 32-bit writes may be used.

Figure 6. PASS_CINn register

5 About locking PASS_LOCKx_PGn registers

Locking or re-locking the PASS_LOCKx_PGn registers after they have been modified, can be done by setting the PGL (password group lock) bit of the group's PASS_LOCK**3**_PGn register.

Address: FFFF_4000h base + 10Ch offset + (16d × i), where i=0d to 3d

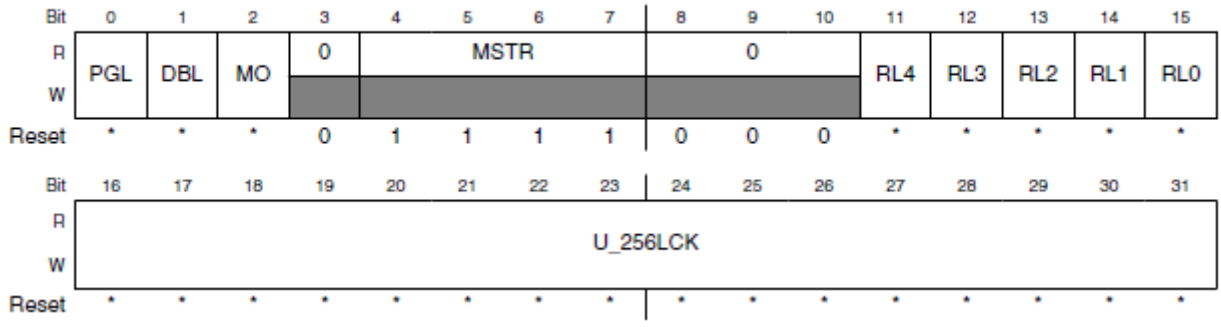


Figure 7. PASS_LOCK_PGn register

Field	Description
0 PGL	Password Group Lock. This bit may be set in software to lock the password group, when the device is older than Customer Delivery. Before that the value of the bit is always 0. This bit is only updatable from SW path. It cannot be updated from DCF load from Flash. This bit can be cleared by writing a valid password to the Password Challenge Input Registers. 0 Password group registers are unlocked. All four registers in the Password group may be read and written without restriction. 1 Password group registers are locked. Writes to the password group registers have no effect. Read accesses work normally. (If life cycle is Customer Delivery or earlier, this bit has no effect and the registers are always unlocked.)

Figure 8. PASS_LOCK_PGn field description

6 About Debugger interface lock

To block JTAG access, set the DBL bit of the PASS_LOCK3_PGn register. This should be done in a DCF record to ensure this protection is active as soon as the device comes out of reset. (see Working with DCF records on page 10 for details)

To re-activate the JTAG access two methods are possible:

- During the external debugger connection, have the debugger script provide the JTAG Password (matching the 256-bit value stored in the JTAG password location in UTEST flash). This bypasses the effect of the DBL bit.

  For example, if using the TRACE32 debugger interface, the sys.option.KEYCODE command will instruct the debugger probe to provide the 32 bytes in argument as password when attempting connection:

  ```
  sys.option.KEYCODE 0x11111111 0x22222222 0x33333333 0x44444444 0x55555555 0x66666666 0x77777777
  0x88888888
  ```

- During runtime, unlock the PASS_LOCK3_PGn registers for all password groups where the DBL but is set by providing the appropriate passwords, and clear the DBL bit.

7 About flash read protection mechanism

The PASS_LOCK3_PGn registers also contain five RLx bits that allow to activate the read protection mechanism for five flash regions.

It is important to understand that this read protection mechanism is intended to protect flash from being read **only when a debugger is connected** and activated. The flash read protection is not active when the application is running without a debugger connected.

The five flash regions that can independently be read-protected are defined below:

- RL0 : UTEST read lock
- RL1: Code Flash read lock
- RL2: Not implemented in MPC5748G
- RL3: HSM Code Flash read lock *
- RL4: HSM Data Flash read lock *

** Additionally, these regions can also be protected by means of other mechanisms during HSM configuration.*

Further details about exact addresses for these regions are provided in the table below or in *Chapter 80.1.1 : PASS_LOCKx_PGn Register bit mapping* of the MPC5748G reference manual.

8 About RESET values of lock registers

To ensure that the desired protection is enabled as soon as the device comes out of reset, the reset value for all 16 PASS_LOCKx_PGn registers should be configured via DCF records stored in UTEST flash.

When the device is initially delivered to the customer, no DCF records that initialize lock registers exist in UTEST flash. In this situation, if no DCF records to initialize PASS_LOCKx_PGn registers are found, all blocks of flash default to locked (blocked against reprogramming).

The PASS Module provides therefore several DCF clients that allow to define the reset values of the lock registers. See section Working with DCF records on page 10 for an example.

9 Further conditions for password-based protections to be active

All protection mechanisms managed by the PASS module will only go into effect when the device has been placed in a life cycle state beyond the initial “customer delivery” state that the device ships with. A device will need to be put at least in the “OEM Production” state for these protections to take effect.

10 Modifying the lifecycle state of a device

The UTEST Flash region contains predefined locations that must be programmed with a value in order to place the device in a specific life cycle. When the product is initially shipped to the customer, lifecycle slots “MCU_PROD” and “CUST_DEL” are factory programmed, meaning the product has been matured to the “customer delivery” cycle, and lifecycle slots “OEM_PROD”, “IN_FIELD” and “FA” are left in their erased states.

Start Address	End Address	Allocated Size [bytes]	Decription
0x004000C8	0x004000FF	56	Reserved
0x00400100	0x00400103	4	Test Mode Overide Passcode
0x00400104	0x0040011F	28	Reserved
0x00400120	0x0040013F	32	JTAG Password
0x00400140	0x0040015F	32	PASS Password Group 0
0x00400160	0x0040017F	32	PASS Password Group 1
0x00400180	0x0040019F	32	PASS Password Group 2
0x004001A0	0x004001BF	32	PASS Password Group 3
0x004001C0	0x004001DF	32	Reserved
0x004001E0	0x004001FF	32	Reserved
0x00400200	0x0040020F	16	Lifecycle Slot 0 - MCU_PROD
0x00400210	0x0040021F	16	Lifecycle Slot 1 - CUST_DEL
0x00400220	0x0040022F	16	Lifecycle Slot 2 - OEM_PROD
0x00400230	0x0040023F	16	Lifecycle Slot 3 - IN_FIELD
0x00400240	0x0040024F	16	Lifecycle Slot 4 - FA
0x00400250	0x004002FF	176	Reserved
0x00400300	0x00400307	8	DCF Start Record
0x00400308 [1, 2]	0x00400347	64	Reserved
0x00400348	0x00400FFF	3256	DCF Records
0x00401000	0x00403FFF	12288	Reserved for customer OTP data

Figure 9. UTEST memory map

Each lifecycle slot (16-bytes wide) is made of two fields:

- the valid field
- the invalid field

Depending on the values programmed into these fields, each lifecycle slot can have one of the 4 states shown below: erased, active, inactive, or illegal:

LC slots		LC slot value
Valid field (64 bits)	**Invalid field (64 bits)**	
Erased	Erased	Erased
Marked	Erased	Active
Marked	Marked	Inactive
Any other values		Illegal

Figure 10. LC slot status

- "Marked" refers to the value 0x55AA_50AF_55AA_50AF.
- “Erased” refers to the value 0x FFFF_FFFF_FFFF_FFFF.

The valid and invalid fields occupy the following positions inside each lifecycle slot:

Offset	LC slot word
00h	Valid[31:0]
04h	Valid[63:32]
08h	Invalid[31:0]
0Ch	Invalid[63:32]

Figure 11. LC slots in memory

The lifecycle of the device is then determined per the following table:

LC slot 0 MCU Production	LC slot 1 Customer Delivery	LC slot 2 OEM Production	LC slot 3 In Field	LC slot 4 Failure Analysis	Resulting life cycle
Active	Erased	Erased	Erased	Erased	MCU Production
Inactive	**Active**	Erased	Erased	Erased	Customer Delivery
Inactive	Inactive	**Active**	Erased	Erased	OEM Production
Inactive	Inactive	Inactive	**Active**	Erased	In Field
Inactive	Inactive	Inactive	Inactive	**Active**	Failure Analysis
Erased	Erased	Erased	Erased	Erased	System Reset
Illegal[1]	Illegal	Illegal	Illegal	Erased	In Field

1. Any value programmed other than shown in table above for Erased, Active, and Inactive would be treated as illegal.

Figure 12. LC slots

11 Censorship state of the device

Strictly speaking, moving the device to “OEM production” or “In Field” is still not enough for the Debugger interface to be blocked or the flash read protection mechanism to be active. One last switch needs to be verified: The censorship status.

Note that we are speaking here only of JTAG disabling and flash read protection. The write-erase protection of flash blocks is not affected by the censorship status.

Censorship is a DCF client that is not accessible by software, therefore the censorship state of a device can only be written via a DCF records in UTEST flash. It is a flexible parameter because other than the size of UTEST flash, there is no limitation as to how many times a DCF record can be added that modifies the value of the Censorship. The device is censored if the lower 16 bits of the Censorship DCF client are written to any value different than 0x55AA.

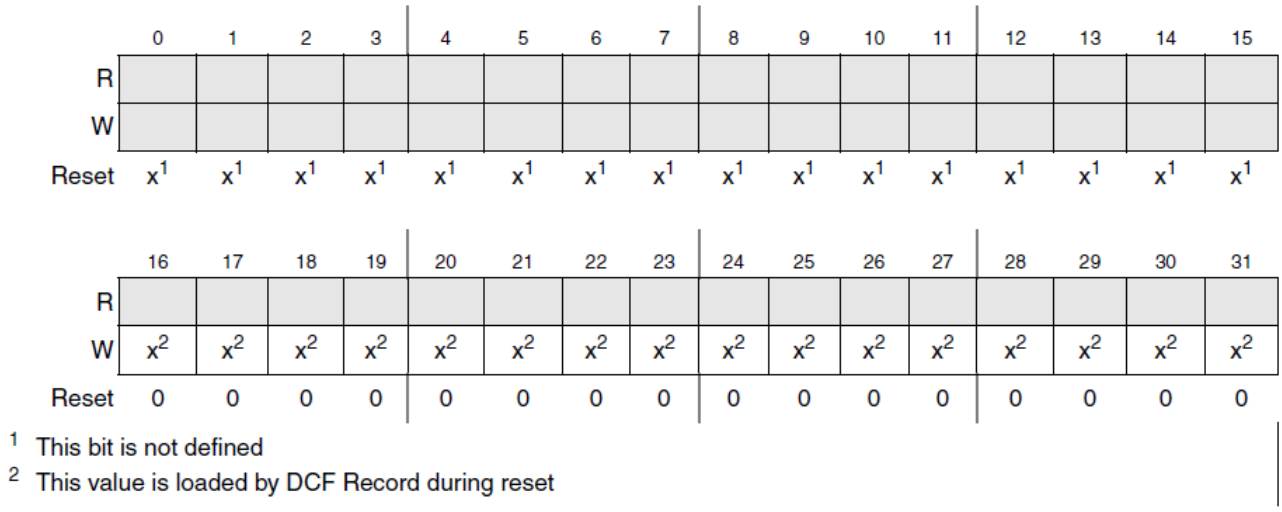


Figure 13. DCF client structure

Field	Description
16:31	Censorship If this value is 0x55AA the MCU is NOT censored If this value is anything other than 0x55AA, the MCU is censored

Figure 14. Censorship description

By default though, when no Censorship DCF record is present, the censorhip status of a device is active. Censorship is therefore active when a device is delivered from the factory. A DCF record should be written to disable censorship one first time. If this has been done, other DCFs records need to be appended to the DCF list to re-activate censorship if desired.

12 Working with DCF records

For a detailed description of how DCF records function please see *Chapter 11, Device Configuration Format (DCF) Records* of the reference manual.

Data present in the upper 32-bits of a DCF record will be written to the 32-bit DCF client register identified by the address in the lower 32-bits of the DCF record. DCF clients have different addresses than ordinary registers. A DCF client is identified with the combination of a chip select value and an address field. The lower 32-bits of a DCF record contain these chip select and address values, as well as a parity bit and a stop bit. In this case the parity bit can be left to 0 as it is not used. The stop bit should be 0 as well.

For the mechanisms discussed in this application note, the DCF clients that must be configured by inserting appropriate DCF records in UTEST flash are summarized in the table below, along with the appropriate DCF command word needed to address them.

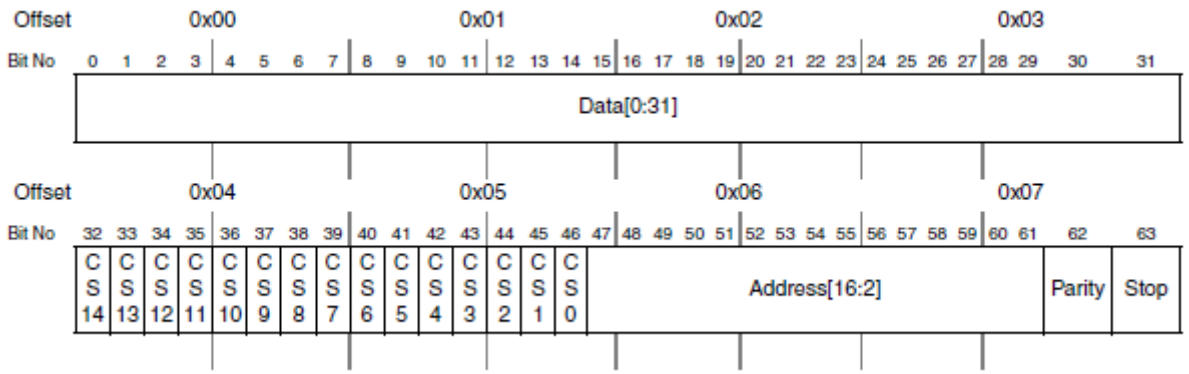


Figure 15. DCF record structure

13 Useful DCF command values

The following figure shows the useful DCF command values:

DCF Command word chip select and address values for MPC5748G (with parity bit and stop bit = 0)	DCF Client
0x0010_00B0	CENSORSHIP
0x0010_0100	LOCK0_PG0
0x0010_0104	LOCK1_PG0
0x0010_0108	LOCK2_PG0
0x0010_010C	LOCK3_PG0
0x0010_0110	LOCK0_PG1
0x0010_0114	LOCK1_PG1
0x0010_0118	LOCK2_PG1
0x0010_011C	LOCK3_PG1
0x0010_0120	LOCK0_PG2
0x0010_0124	LOCK1_PG2
0x0010_0128	LOCK2_PG2
0x0010_012C	LOCK3_PG2
0x0010_0130	LOCK0_PG3
0x0010_0134	LOCK1_PG3
0x0010_0138	LOCK2_PG3
0x0010_013C	LOCK3_PG3

Figure 16. Useful DCF command values

Example 1: Censoring the device

To censor the device, a DCF record with value 0x00000000_001000B0 should be inserted in the DCF records list in UTEST flash. The upper 4 bytes of this value (0x00000000) represent the data field, (written to a value different than 0x55AA to activate the censorship). The lower 4 bytes (0x001000B0) represent the chip select, address fields, parity bit and stop bit of the DCF record, which per the table above, correspond to the Censorhip DCF client.

To un-censor the device, a DCF record with value 0x000055AA_001000B0 should be inserted in the DCF records list in UTEST flash, immediately after the last valid record. The upper 4 bytes of this value (0x000055AA) represent the data field, written to value 0x55AA, which un-censors the device.

Example 2: Setting the reset value of register LOCK3_PG0

Similarly, programming a DCF record with value 0x40000000_0x0010010C in UTEST flash will write the upper 32-bits (0x40000000) to the LOCK3_PG0 register (identified by the address word 0x0010010C in the table above). Value 0x40000000, for the LOCK3_PG0 means the DBL bit is set and all lock bits are cleared (unlocked).

14 Combination of different protection configurations

The table below shows a summary of how the different protection mechanisms described here are combined to activate a protection or unlock it.

Input Conditions				Outcome
Lifecycle	**Censorship**	**Debug Enable (LOCK3[DBL] bit)**	**JTAG Password**	**Debug Interface Enable**
Customer Delivery	Don't Care	Don't Care	Don't Care	Enabled
OEM Production	Not censored	Don't Care	Don't Care	Enabled
	Censored	Unlocked (DBL=0)	Don't Care	Enabled
		Locked (DBL=1)	Matched	Enabled
			Not Matched	Blocked
In Field	Not censored	Don't Care	Don't Care	Enabled
	Censored	Unlocked (DBL=0)	Don't Care	Enabled
		Locked (DBL=1)	Matched	Enabled
			Not Matched	Blocked
Failure Analysis	Don't Care	Don't Care	Don't Care	Enabled

Figure 17. Debug interface enable truth table

INPUTS				OUTPUTS	
Life Cycle	**Censorship**	**Debug Interface Enable from HSM**	**Lock3[RLx]**	**Read Protected RL[0] (UTEST)**	**Read Protected RL[4:1] (Other Regions)**
Customer Delivery	Don't Care	Don't Care	Don't Care	Readable	Readable
OEM Production / IN Field	Uncensored	Don't Care	Don't Care	Readable	Readable
	Censored	Blocked	Don't Care	Readable	Readable
		Enabled	Locked (1)	Read Blocked	Read Blocked
			Unlocked (0)	Readable	Readable
Failure Analysis	Uncensored	Don't Care	Don't Care	Readable	Readable
	Censored	Don't Care	Locked (1)	Readable	Read Blocked
			Unlocked (0)	Readable	Readable

Figure 18. Flash memory read protection truth table

A - Source code example 1:

The example below will :

- Program the JTAG Password and all 4 group passwords
- Insert DCFs to determine the reset state of all 16 Lock registers. The reset state chosen is: All regions unlocked for read write or erase (lock bits all cleared), debugger blocked (DBL bit set).
- Program the censorship DCF and place the device in censored state.

- Advance life cycle from customer delivery to OEM production.

```
/************************/
/* PROGRAM PASSWORDS    */
/************************/

/* Program JTAG password */
flash_program(0x00400120, JTAG_PASSWORD_0, JTAG_PASSWORD_1);
flash_program(0x00400128, JTAG_PASSWORD_2, JTAG_PASSWORD_3);
flash_program(0x00400130, JTAG_PASSWORD_4, JTAG_PASSWORD_5);
flash_program(0x00400138, JTAG_PASSWORD_6, JTAG_PASSWORD_7);

/* Program PASS Group 0 password */
flash_program(0x00400140, PASSWORD_PG0_0, PASSWORD_PG0_1);
flash_program(0x00400148, PASSWORD_PG0_2, PASSWORD_PG0_3);
flash_program(0x00400150, PASSWORD_PG0_4, PASSWORD_PG0_5);
flash_program(0x00400158, PASSWORD_PG0_6, PASSWORD_PG0_7);

/* Program PASS Group 1 password */
flash_program(0x00400160, PASSWORD_PG1_0, PASSWORD_PG1_1);
flash_program(0x00400168, PASSWORD_PG1_2, PASSWORD_PG1_3);
flash_program(0x00400170, PASSWORD_PG1_4, PASSWORD_PG1_5);
flash_program(0x00400178, PASSWORD_PG1_6, PASSWORD_PG1_7);

/* Program PASS Group 2 password */
flash_program(0x00400180, PASSWORD_PG2_0, PASSWORD_PG2_1);
flash_program(0x00400188, PASSWORD_PG2_2, PASSWORD_PG2_3);
flash_program(0x00400190, PASSWORD_PG2_4, PASSWORD_PG2_5);
flash_program(0x00400198, PASSWORD_PG2_6, PASSWORD_PG2_7);
/* Program PASS Group 3 password */

flash_program(0x004001A0, PASSWORD_PG3_0, PASSWORD_PG3_1);
flash_program(0x004001A8, PASSWORD_PG3_2, PASSWORD_PG3_3);
flash_program(0x004001B0, PASSWORD_PG3_4, PASSWORD_PG3_5);
flash_program(0x004001B8, PASSWORD_PG3_6, PASSWORD_PG3_7);

/************************/
/* PROGRAM PASS DCFs    */
/************************/

/* Password Group 0 - program locks */
DCF_program(0x00000000, 0x00100100);   /* LOCK0_PG0 */
DCF_program(0x00000000, 0x00100104);   /* LOCK1_PG0 */
DCF_program(0x00000000, 0x00100108);   /* LOCK2_PG0 */
DCF_program(0x40000000, 0x0010010C);   /* LOCK3_PG0 [DBL=1] */

/* Password Group 1 - program locks */
DCF_program(0x00000000, 0x00100110);   /* LOCK0_PG1 */
DCF_program(0x00000000, 0x00100114);   /* LOCK1_PG1 */
DCF_program(0x00000000, 0x00100118);   /* LOCK2_PG1 */
DCF_program(0x00000000, 0x0010011C);   /* LOCK3_PG1, [DBL=0] */

/* Password Group 1 - program locks */
DCF_program(0x00000000, 0x00100120);   /* LOCK0_PG2 */
DCF_program(0x00000000, 0x00100124);   /* LOCK1_PG2 */
DCF_program(0x00000000, 0x00100128);   /* LOCK2_PG2 */
DCF_program(0x00000000, 0x0010012C);   /* LOCK3_PG2, [DBL=0] */

/* Password Group 1 - program locks */
DCF_program(0x00000000, 0x00100130);   /* LOCK0_PG3 */
DCF_program(0x00000000, 0x00100134);   /* LOCK1_PG3 */
DCF_program(0x00000000, 0x00100138);   /* LOCK2_PG3 */
DCF_program(0x00000000, 0x0010013C);   /* LOCK3_PG3, [DBL=0] */

/************************/
/* Program Censorship DCF */
/************************/
```

```
  DCF_program(0x00000000, 0x001000B0);   /* Censorship enabled for != 0x55AA */

  /*************************/
  /* PROGRAM LIFE CYCLE   */
  /*************************/

  /* Advance Life Cycle from Customer Delivery to OEM Production */

flash_program(0x00400218, 0x55AA50AF, 0x55AA50AF);/* Invalidate CustomerDel*/
flash_program(0x00400220, 0x55AA50AF, 0x55AA50AF);/* Validate OEMProduction*/
```

Auxiliary functions:

```
void flash_program(uint32_t prog_addr, uint32_t data32_0, uint32_t data32_1)
{
  /* clear lock */
  C55FMC.LOCK0.R = 0;
  C55FMC.LOCK1.R = 0;
  C55FMC.LOCK2.R = 0;
  C55FMC.LOCK3.R = 0;

  C55FMC.MCR.B.PGM = 1;

  *((uint32_t*) prog_addr)   = data32_0;
  *((uint32_t*) prog_addr+1) = data32_1;

  C55FMC.MCR.B.EHV = 1;
  while(C55FMC.MCR.B.DONE == 0);
  C55FMC.MCR.B.EHV = 0;
  C55FMC.MCR.B.PGM = 0;
  /* set lock */
  C55FMC.LOCK0.R = 0xFFFFFFFF;
  C55FMC.LOCK1.R = 0xFFFFFFFF;
  C55FMC.LOCK2.R = 0xFFFFFFFF;
  C55FMC.LOCK3.R = 0xFFFFFFFF;
}


void DCF_program(uint32_t data32_0, uint32_t data32_1)
{
  uint32_t utest_addr;
  utest_addr = find_1st_DCF_address();
  flash_program(utest_addr, data32_0, data32_1);
}


uint32_t find_1st_DCF_address(void)
{
  uint32_t *UTEST_ADDR;
  uint32_t address;
  UTEST_ADDR = 0x00400304;

  /* find first erased DCF record location after DCF stop */
  while(*UTEST_ADDR != 0xFFFFFFFF)

    UTEST_ADDR +=2; /* advance to next DCF record address */
  address = (uint32_t) (UTEST_ADDR - 1);

  return address;
}
```

Auxiliary #defines for this example:

```
#define JTAG_PASSWORD_0 0x11111111
#define JTAG_PASSWORD_1 0x22222222
#define JTAG_PASSWORD_2 0x33333333
#define JTAG_PASSWORD_3 0x44444444
#define JTAG_PASSWORD_4 0x55555555
```

```
#define JTAG_PASSWORD_5 0x66666666
#define JTAG_PASSWORD_6 0x77777777
#define JTAG_PASSWORD_7 0x88888888

#define PASSWORD_PG0_0  0x22222222
#define PASSWORD_PG0_1  0x33333333
#define PASSWORD_PG0_2  0x44444444
#define PASSWORD_PG0_3  0x55555555
#define PASSWORD_PG0_4  0x66666666
#define PASSWORD_PG0_5  0x77777777
#define PASSWORD_PG0_6  0x88888888
#define PASSWORD_PG0_7  0x99999999

#define PASSWORD_PG1_0  0x33333333
#define PASSWORD_PG1_1  0x44444444
#define PASSWORD_PG1_2  0x55555555
#define PASSWORD_PG1_3  0x66666666
#define PASSWORD_PG1_4  0x77777777
#define PASSWORD_PG1_5  0x88888888
#define PASSWORD_PG1_6  0x99999999
#define PASSWORD_PG1_7  0xAAAAAAAA

#define PASSWORD_PG2_0  0x44444444
#define PASSWORD_PG2_1  0x55555555
#define PASSWORD_PG2_2  0x66666666
#define PASSWORD_PG2_3  0x77777777
#define PASSWORD_PG2_4  0x88888888
#define PASSWORD_PG2_5  0x99999999
#define PASSWORD_PG2_6  0xAAAAAAAA
#define PASSWORD_PG2_7  0xBBBBBBBB

#define PASSWORD_PG3_0  0x55555555
#define PASSWORD_PG3_1  0x66666666
#define PASSWORD_PG3_2  0x77777777
#define PASSWORD_PG3_3  0x88888888
#define PASSWORD_PG3_4  0x99999999
#define PASSWORD_PG3_5  0xAAAAAAAA
#define PASSWORD_PG3_6  0xBBBBBBBB
#define PASSWORD_PG3_7  0xCCCCCCCC
```

B- Source code example 2:

This example demonstrates how to unlock the lock registers from within the user application by providing a password. It is assumed that only Password group 1 has been used to lock resources.

```
/* Select Password Challenge Group 1 */
  PASS.CHSEL.B.GRP = 1;

/* Enter Password - The most significant 32 bits of the password challenge must be written
to CIN0 */

  PASS.CIN[0].R = PASSWORD_PG1_7;
  PASS.CIN[1].R = PASSWORD_PG1_6;
  PASS.CIN[2].R = PASSWORD_PG1_5;
  PASS.CIN[3].R = PASSWORD_PG1_4;
  PASS.CIN[4].R = PASSWORD_PG1_3;
  PASS.CIN[5].R = PASSWORD_PG1_2;
  PASS.CIN[6].R = PASSWORD_PG1_1;
  PASS.CIN[7].R = PASSWORD_PG1_0;

  /* Clear Lock2 */
  PASS.PG[1].LOCK2.R = 0;
  if(PASS.PG[1].LOCK2.R != 0)
      while(1);  /* ERROR- bits are not cleared, PASS not unlocked */

  flash_erase_0x01080000();
```

```
if(readmem(0x01080000)!= 0xFFFFFFFF)
    while(1); /*ERROR- location was not erased, erase protection was not unlocked */
```

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Document Number: AN12092
Rev. 0, February, 2018