em microelectronic

A COMPANY OF THE SWATCH GROUP

EM9305 Security lock bits

Release 1.1

EMM

Apr 29, 2024

CONTENTS

CHAPTER

ONE

LOCK BITS

1.1 Glossary

AES:	Advanced Encryption Standard
CRC:	Cyclic Redundancy Check
ID:	IDentifier
JTAG:	Joint Test Action Group
NVM:	Non Volatile Memory
PML:	Power Management Logic
POR:	Power On Reset
RFU:	Reserved for Further Usage
SDK:	Software Development Kit

1.2 Introduction

EM9305 implements various lock bits. For instance, there are lock bits to prevent the use of JTAG, to prohibit the erasing of flash pages, or to enable features.

This documentation is built as follows:

1. The notion of PML registers is explained.
2. All the lock bits are described.
3. The methods to set and reset the lock bits are explained. This section explains the role of InfoPage2 and InfoPage3 flash pages.
4. The typical configuration of lock bits at shipment time is given.

1.3 PML Registers

Before describing the lock bits by themselves, a reminder on PML registers is necessary.

The lock bits are implemented in PML registers. PML stands for **Power Management Logic**.

The PML lock registers have the following characteristics:

- The values of the registers are **retained** when the chip goes to sleep or deep sleep mode.
- The registers are **one-way** only:
  - writing 1 over a 0 is possible.
  - writing 0 over a 1 is **not** possible. More precisely writing 0 over a 1 has no effect.
- The registers can only be reset with a POR.

Next figure illustrates how the lock bits work:

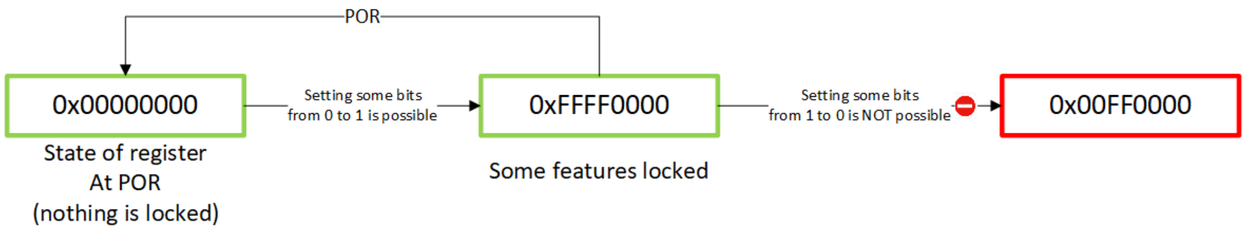
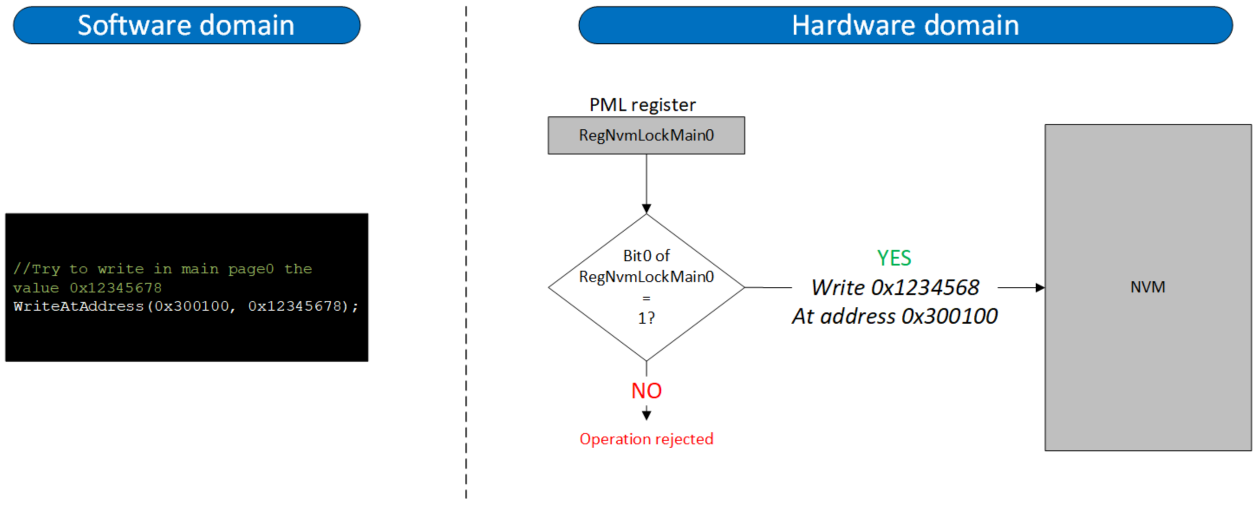


Fig. 1: PML register principle

The PML registers are managed by the hardware. The access to the feature, to the page or to the key is controlled by the hardware assessment of the PML register. If the corresponding lock bit is set to "1", the feature is locked or the access is forbidden.

Next figure illustrates the principle. In this example we suppose that the software wants to write a data in the page0 of the main area. In that case, the NVM controller assesses the bit 0 of PML register *RegNvmLockMain0*. If the bit 0 is set to 0, the write operation is allowed. The data is written at the specified address. If the bit is set to 1, it means that the page 0 is locked. The write operation is rejected by the hardware.



```
//Try to write in main page0 the
value 0x12345678
WriteAtAddress(0x300100, 0x12345678);
```

Fig. 2: PML hardware verification principle

Note: When **accessing** the feature, only the PML register is checked. The content of the NVM InfoPage2 or InfoPage3 is not read. The copy of the InfoPages into the PML registers is performed only at boot time. For further information see *Lock bits management* section.

1.4 Lock bits description

In this chapter, each lock bit is described.

Fundamentally, EM9305 implements 3 categories of lock bits:

1. **System lock bits**: These lock bits enable or disable features.
2. **NVM Page operations lock bits**: These lock bits prevent erase or write operations on NVM pages.
3. **AES crypto keys lock bits**: These lock bits prevent operations to some crypto containers.

Note:

- Bits are numbered from 31 to 0.
- When not described, the bits are **not implemented (RFU)**. The RFU bits are not used. They can be set by the user to "0" or "1" value. Nonetheless, for consistency reasons, one will preferably set them to the reset value, that is to say to the value "0".
- Unless mentioned differently, the lock bits are one-way.

1.4.1 System lock bits

One PML register *RegPmlLockBits* is dedicated to the system lock bits. It configures the JTAG, the test mode and the USB interface.

Table 1: The register RegPmlLockBits: the system lock bits

PML register name	PML address	Bit name	Bits	Description
RegPmlLockBits	0x00F00420	*PmlJtag2wEn*	bit 25	JTAG 2-wire enable
RegPmlLockBits	0x00F00420	*PmlJtag4wEn*	bit 24	JTAG 4-wire enable
RegPmlLockBits	0x00F00420	*PmlTxPaPwrLock*	bits 21 to 16	Max value of authorized output power
RegPmlLockBits	0x00F00420	*PmlLdoAntTrimLock*	bits 11 to 8	Max value of authorized gain
RegPmlLockBits	0x00F00420	*PmlTmLock*	bit 2	Test mode lock
RegPmlLockBits	0x00F00420	*PmlUsbLock*	bit 1	USB lock
RegPmlLockBits	0x00F00420	*PmlJtagLock*	bit 0	JTAG lock

In next subsections the bits of RegPmlLockBits are further described.

1.4.1.1 JTAG management: PmlJtag2wEn, PmlJtag4wEn and PmlJtagLock

Three lock bits of the *RegPmlLockBits* permit to manage the JTAG. Namely:

- PmlJtag2wEn
- PmlJtag4wEn
- PmlJtagLock

Those three bits permit to lock the JTAG and select the protocol.

Essentially:

- **At boot time, JTAG is always disabled**. In other words, *PmlJtag4wEn=0* and *PmlJtag2wEn=0*.
- *PmlJtagLock* controls *PmlJtag2wEn* and *PmlJtag4wEn* , preventing those 2 registers to be set to 1.
- Setting *PmlJtag2wEn* or *PmlJtag4wEn* enables JTAG when authorized.
- Enabling JTAG requires an intentional activation, setting *PmlJtag2wEn* or *PmlJtag4wEn*.

Next table details how to configure the 3 JTAG bits.

Table 2: JTAG configuration

PmlJtagLock	*PmlJtag2wEn*	*PmlJtag4wEn*	Status	Can be enabled?
0	0	0	Disabled	YES
0	0	1	JTAG 4-wire Enabled	YES
0	1	0	JTAG 2-wire Enabled	YES
0	1	1	JTAG 2-wire and 4-wire Enabled	YES
1	0	0	Disabled	NO
1	0	1	Disabled	NO
1	1	0	Disabled	NO
1	1	1	Disabled	NO

Fundamentally:

- JTAG is always disabled when *PmlJtagLock* is locked.
- Can be enabled or disabled if *PmlJtagLock* is not locked. The two registers *PmlJtag2wEn* and *PmlJtag4wEn* are two ways, so that the application may decide to disable JTAG during the application setting PmlJtag4wEn\PmlJtag2wEn back to 0.

Note:

- EM typically ships the parts with JTAG lock not set (PmlJtagLock='0') and mlJtag4wEn\PmlJtag2wEn not set. This configuration allows the debugging with JTAG.
- It is highly recommended to disable JTAG when the parts are in the field. *PmlJtagLock* shall be set to 1 to avoid any manual enabling setting PmlJtag4wEn\PmlJtag2wEn registers.
- If JTAG is locked in InfoPage2, one can always unlock it after reconfiguring the InfoPage2 following a USER authentication.

1.4.1.2 PmlTxPaPwrLock and PmlLdoAntTrimLock

These two registers allow to limit the gain and the output power of the parts.

If *PmlTxPaPwrLock* value is greater than TX_PA_PWR RF register, then TX_PA_PWR value is applied. Otherwise the output power is limited by the value given by *PmlTxPaPwrLock*.

Similarly, if *PmlLdoAntTrimLock* value is greater than LDO_ANT_TRIM, the value of LDO_ANT_TRIM applies. Otherwise the gain is capped by the value set in *PmlLdoAntTrimLock*.

Warning: If *RegPmlLockBits* register is modified by InfoPage2 (user), it is mandatory to program the same values as the ones pre-defined by EM in InfoPage3.

1.4.1.3 PmlTmLock

This lock prevents the access to test circuitry and registers.

Note: EM always delivers the parts with test mode locked.

1.4.1.4 PmlUsbLock

This lock prevents the utilization of the USB interface.

If *PmlUsbLock='1'*, the USB block is not clocked.

1.4.2 NVM pages operations

The second type of lock bits are related to the flash pages management.

Four 32-bit registers manage the NVM operations:

- *RegNvmLockMain0*
- *RegNvmLockMain1*
- *RegNvmLockInfo*
- *RegNvmLockMaster*

Table 3: Flash pages management lock bits

PML register name	PML address	Lock bit name	Bits	Description
RegNvmLockMain0	0x00F00490	Lock main pages	0..31	Lock erase and write operations of the main pages 0..31
RegNvmLockMain1	0x00F00494	Lock main pages	0..31	Lock erase and write operations of the main pages 32..64
RegNvmLockInfo	0x00F00498	*NvmEraseLockInfoPage0*	bit 17	Lock the InfoPage 0 **erase** operation
RegNvmLockInfo	0x00F00498	*NvmWriteLockInfoPage0*	bit 16	Lock the InfoPage 0 **write** operation
RegNvmLockInfo	0x00F00498	*NvmLockInfoPage*	bits 3 to 0	Lock the InfoPages 0..3
RegNvmLockMaster	0x00F0049C	*NvmLockMaster*	bit 16	Lock the other lock bits
RegNvmLockMaster	0x00F0049C	*NvmLockRedund*	bit 8	Lock bit for redundancy pages mapping
RegNvmLockMaster	0x00F0049C	*NvmLockEraseFull*	bit 1	Lock bit for **full mass erase** operation
RegNvmLockMaster	0x00F0049C	*NvmLockEraseMain*	bit 0	Lock bit for **main mass erase** operation

1.4.2.1 RegNvmLockMain0 and RegNvmLockMain1

These two registers aim at locking the main NVM pages. Each of the 64 pages can be locked individually. When locked the page cannot be erased and cannot be written.

Next table specifies the bits of RegNvmLockMain0:

Table 4: Main pages (0 to 31) lock bits

bit 31	bit 30	…	bit 1	bit 0
Page31	Page30	…	Page1	Page0

Next table specifies the bits of RegNvmLockMain1:

Table 5: Main pages (32 to 63) lock bits

bit 31	bit 30	…	bit 1	bit 0
Page63	Page62	…	Page33	Page32

As an example, if *RegNvmLockMain0=0x00FF0000* and *RegNvmLockMain1=0x000000FF*, the pages 16 to 23 and the pages 32 to 39 are locked. The other ones are unlocked and can be written or erased.

1.4.2.2 RegNvmLockInfo

This register aims at managing the 4 InfoPages. 6 bits configure the pages.

Next table specifies the bits of RegNvmLockInfo:

Table 6: Info Pages lock bits

bit 31	…	bit 17	bit 16	…	bit 3	bit 2	bit 1	bit 0
RFU	…	NvmErase-LockIn-foPage0	NvmWrite-LockIn-foPage0	…	In-foPage3	In-foPage2	In-foPage1	In-foPage0

Each InfoPage can be individually locked against write and erase. The bit 0,1,2,3 respectively lock the InfoPage0, InfoPage1, InfoPage2 and InfoPage3.

As an example, if *RegNvmLockInfo=0x00000009*, InfoPage3 and InfoPage0 are locked. InfoPage1 and InfoPage2 are not locked.

1.4.2.2.1 The special case of InfoPage0

In addition, the InfoPage0 has two extra lock bits:

- **NvmWriteLockInfoPage0**: This lock prevents **write operations on** the InfoPage0. Erasing is still possible.
- **NvmEraseLockInfoPage0**: This lock prevents **erase operations on** the InfoPage0. Writing the page is still possible.

InfoPage0 is the unique page that benefits such a granularity with the locks. As a reminder, InfoPage0 is dedicated to the AES key containers.

Thanks to these two extra lock bits, the user can lock the page against the erase operation and therefore protect keys already present in the InfoPage0 and keep the ability to program new keys (in other key containers).

To illustrate the functioning of those lock bits, two use cases are described:

1- All the keys in the key containers are set at the same time, for instance at production time. InfoPage0 can be locked with the bit 0 of *NvmLockInfoPage*. Alternatively, *NvmWriteLockInfoPage0* and *NvmEraseLockInfoPage0* can be set. All the keys are protected against corruption or erasing.

2- Keys are written in several steps. The first set of keys should be protected against erasing. Therefore the page should be locked with *NvmEraseLockInfoPage0* lock bit. Nonetheless, since other keys have to be written later on, the page should not be protected against write. Therefore *NvmLockInfoPage* lock bit and *NvmWriteLockInfoPage0* should stay unlocked. They both can be locked at the second stage, when all the keys have been provisioned.

Note:

- If no AES key is provisioned by EM before shipment, InfoPage0 is delivered fully unlocked. (*NvmWriteLockInfoPage0='0'*, *NvmEraseLockInfoPage0='0'*, *NvmLockInfoPage-bit0='0'*).
- If the parts are delivered with AES keys, InfoPage0 is locked against erase only. (*NvmWriteLockInfoPage0='0'*, *NvmEraseLockInfoPage0='1'*, *NvmLockInfoPage-bit0='0'*).

In addition to the management at the page level, the 8 first key containers can be individually protected against write (see section *AES crypto keys lock bits*).

1.4.2.3 NvmLockMaster

NvmLockMaster prevents to reconfigure the memory management lock bits.

More precisely, *NvmLockMaster* protects:

- all the bits of *RegNvmLockMain0*
- all the bits of *RegNvmLockMain1*
- all the bits of *RegNvmLockInfo*
- **all the bits of *RegNvmLockMaster*:**
    - *NvnLockMaster* by itself
    - *NvmLockRedund*
    - *NvmLockEraseFull*
    - *NvmLockEraseMain*

If NvmLockMaster is set:

- The covered lock bits cannot be set to 1.
- In addition, since the PML registers are one-way, they cannot be set back to 0.

Therefore, setting NvmLockMaster freezes the configuration of all memory management lock bits.

Warning: The locks are read sequentially from NVM (see section *Lock bits management*).

If NvmLockMaster is the first record in the locks list, the next lock bits will be ignored.

Therefore, it is recommended that RegNvmLockMaster is the last register in the NVM list. Alternatively, RegNvmLockMaster can be set in two records. Other bits but the *NvmLockMaster* can be set in any record and *NvmLockMaster* individually set in the very last record.

Note: *NvmLockMaster* does not affect the locks of the crypto containers. *RegNvmKcLockKey* register can be written independently of *NvmLockMaster*.

1.4.2.4 NvmLockRedund

EM9305 NVM IP block has two extra pages for redundancy. Up to two failing pages can be remapped to the redundancy pages. *NvmLockRedund* lock prevents to modify the redundancy configuration.

Note:

- Only the main pages can be remapped. InfoPages cannot be remapped.
- The use of redundancy pages is transparent for the user.
- The configuration of the redundancy pages is pre-defined by EM before shippment.
- *NvmLockRedund* is always locked by EM before shippment.

1.4.2.5 NvmLockEraseFull and NvmLockEraseMain

Two "mass erase" operations are supported by EM9305:

- **Full mass erase:** all the pages of the main area are erased. The four info pages are also **erased**. NvmLockEraseFull lock prevents this operation.
- **Main mass erase:** all the pages of the main area are erased. The four info pages are **not erased**. NvmLockEraseMain lock prevents this operation.

Note: Since InfoPage3 is dedicated to EM, the parts are usually shipped with InfoPage3 locked. The full mass erase is therefore prohibited and *NvmLockEraseFull* is locked. On the other hand, *NvmLockEraseMain* is normally not locked. If one needs to erase all the memory but InfoPage3, he should use the main mass erase, together with individual InfoPage0, InfoPage1 and InfoPage2 page erase.

Warning: The two mass erase locks have priority on the individual page locks.

It means:

- If *NvmLockEraseFull* is not locked, all the pages including the InfoPages can be erased independently of the individual lock configuration of the pages.
- If *NvmLockEraseMain* is not locked, all the main pages can be erased independently of the individual lock configuration of the main pages.
- If *NvmLockEraseFull* and *NvmLockEraseMain* are locked, the individual locks of the pages determine if the page can be erased or not.

It is recommended to lock *NvmLockEraseMain* as soon as one main page is locked.

1.4.3 AES crypto keys lock bits

The third type of lock bits is related to the AES keys stored in the crypto containers.

RegNvmKcLockKey register allows to lock individually AES keys located in the key containers (InfoPage0) against write.

- Only the **eight** first keys, with ID=0..7, can be individually locked.
- Keys with ID greater or equal to 8 cannot be individually locked against write operation. Protection of those keys is accomplished locking bit 0,16 or 17 of *RegNvmLockInfo*.

PML register name	PML address	Lock bit name	Bits	Description
RegNvmKcLockKey	0x00F004A0	*LockKey0*	bit 0	Lock for the key container 0
RegNvmKcLockKey	0x00F004A0	*LockKey1*	bit 1	Lock for the key container 1
RegNvmKcLockKey	0x00F004A0	*LockKey2*	bit 2	Lock for the key container 2
RegNvmKcLockKey	0x00F004A0	*LockKey3*	bit 3	Lock for the key container 3
RegNvmKcLockKey	0x00F004A0	*LockKey4*	bit 4	Lock for the key container 4
RegNvmKcLockKey	0x00F004A0	*LockKey5*	bit 5	Lock for the key container 5
RegNvmKcLockKey	0x00F004A0	*LockKey6*	bit 6	Lock for the key container 6
RegNvmKcLockKey	0x00F004A0	*LockKey7*	bit 7	Lock for the key container 7

As an example, if *RegNvmKcLockKey=0x00000032*, the keys 1,4,5 are locked against write operation.

Detailed information on the key containers can be found in the crypto lib documentation.

1.5 Lock bits management

1.5.1 Setting locks

There are two ways to set the PML lock bit registers:

- Setting directly the registers addressing the registers by software.
- Setting the lock bits in InfoPage2 and\or InfoPage3.

1.5.1.1 Direct addressing by SW

The user can directly set the PML registers in the application software.

For instance the instruction:

PML->RegNvmLockInfo.r32 = NVM_WRITE_LOCK_INFO_PAGE0(1);

would lock the InfoPage0 against write.

In the case registers are set in the application software, the effect is immediate. The feature is immediately locked by the hardware.

Warning: The locks directly set with this method only apply until **next POR or any system reset**. Indeed at next POR, all registers are reset to the null value. See *How to reset the locks?* section for more details. For permanent locking, the configuration should be defined in InfoPage2\InfoPage3.

1.5.1.2 Automatic loading from NVM

At boot time, the lock bits assigned in InfoPage3 and InfoPage2 are **automatically** copied into the PML registers.

Practically, InfoPage3 and InfoPage2 store a list of records.

Each record is composed of two fields:

- the address of a lock bit register.
- the value of the register.

At boot time, the list of records is scanned. Each record is read sequentially and the value of the register is copied into the corresponding PML lock bit register.

Practically:

- If the boot is performed in **application mode**, InfoPage3 is read, followed by the InfoPage2. In other words both locks of InfoPage2 and InfoPage3 apply.
- If the boot is performed in **USER configuration mode**, only InfoPage3 locks are scanned.
- If the boot is performed in **EM configuration mode**, the locks do not apply.

Next sections show the mapping of InfoPage3 and InfoPage2. It also illustrates how the records are copied into the PML registers.

1.5.1.2.1 NVM InfoPage3 lock bits container

Next figure illustrates how the automatic copy from InfoPage3 to PML registers works.

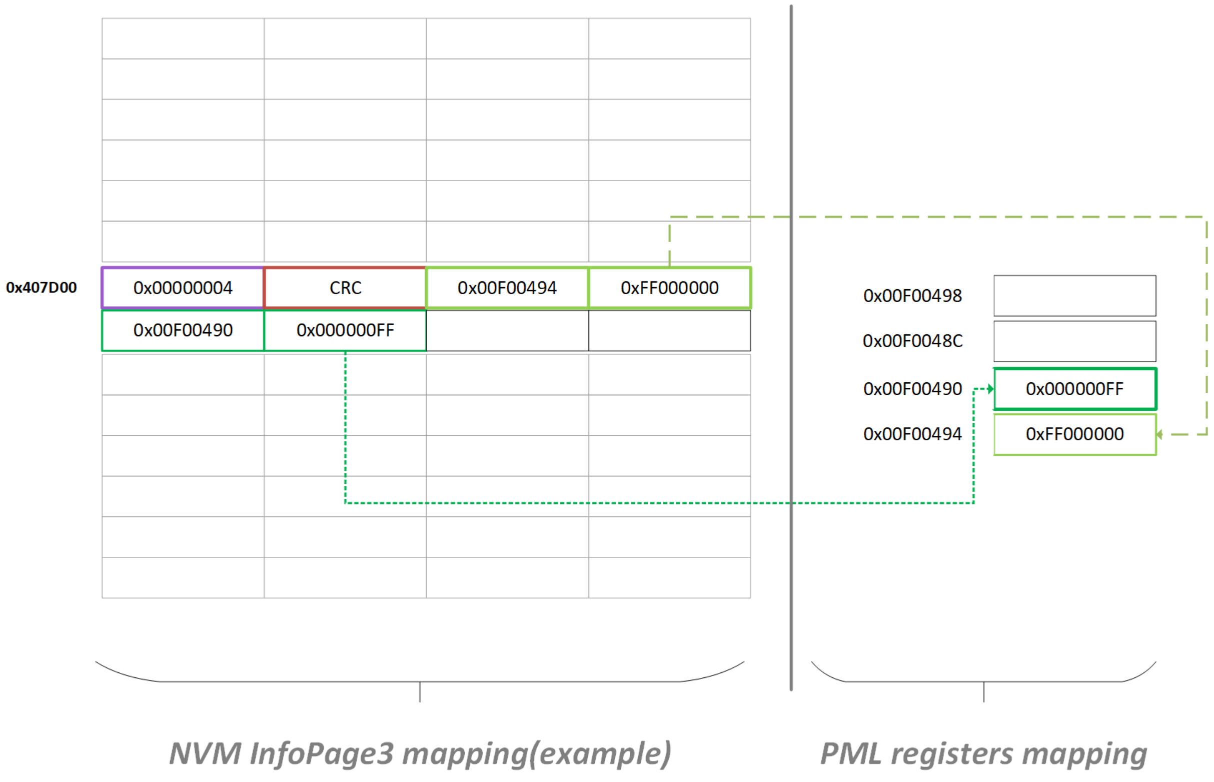


Fig. 3: Example of InfoPage3 lock bits container

The locks of InfoPage3 are stored from the address **0x407D00**. It is composed of a list of lock bits registers.

More precisely:

- The first 32-bit word (in purple) indicates how many 32-bit words are set after the CRC-32. It corresponds to the number of lock bits registers that are specified multiplied by 2.
- The second 32-bit word (in brown) is a CRC-32 of the complete record.
- Then (in green), there is a list of lock bits composed each of two 32-bit words:
  - First word is the address of the PML register corresponding to the lock bit.
  - Second word is the value of the PML register.

In our example, we have 2 records:

- First record, in light green, aims at setting the value 0xFF000000 to the PML register 0x00F00494. Since 0x00F00494 is the lock register to lock the main pages 32..63, this setting (0xFF000000) will lock the main pages 56..63 and keep unlocked the pages 32..57.
- Second record, in dark green, aims at setting the value 0x000000FF to the PML register 0x00F00490. Since 0x00F00490 is the lock register to lock the main pages 0..31, this setting (0x000000FF) will lock the main pages 0..7 and keep unlocked the pages 8..31.

This example shows:

- One can set as many lock bits he wants, as long as the two first words are properly filled.
- The lock bits can be set in any order. In our example, we first set the locks of the pages 32..63 before setting the locks of pages 0..31.

Except for *NvmLockMaster*, the order of appearance of the locks in NVM is not relevant.

At boot time, in application or in USER configuration mode, the locks located in InfoPage3 are copied to the PML registers. Once this copy is performed, the locks are effective.

Note: Because the list of records is protected by the CRC and the length field, it is recommended to set all the lock bits at the same time. The modification of the lock bits requires to erase the complete container, therefore the full InfoPage3.

1.5.1.2.2 NVM InfoPage2 lock bits container

The InfoPage2 lock bits container structure is very similar to the InfoPage3 one. The only difference is that the list of lock bits is located at the address **0x405D00**. The overall size of the lock bits container is 128 bytes. Since 8 bytes are reserved for the CRC and the number of records that are set, up to 15 lock bits registers can be set.

Next figure shows how 2 lock registers are configured in InfoPage2.

We see:

- First record, in light green, aims at setting the value 0x0000004 to the PML register 0x00F00498. Since 0x00F00498 is the lock register for the information pafes *RegNVMLockInfo*, this setting (0x0000004) will lock the InfoPage2 (bit 2). This example illustrates how InfoPage2 can self-lock InfoPage2.
- Second record, in dark green, aims at setting the value 0xF0000000 to the PML register 0x00F00490. Since 0x00F00490 is the lock register to lock the main pages 0..31, this setting (0xF0000000) will lock the main pages 28..31 and keep unlocked the pages 0..27.

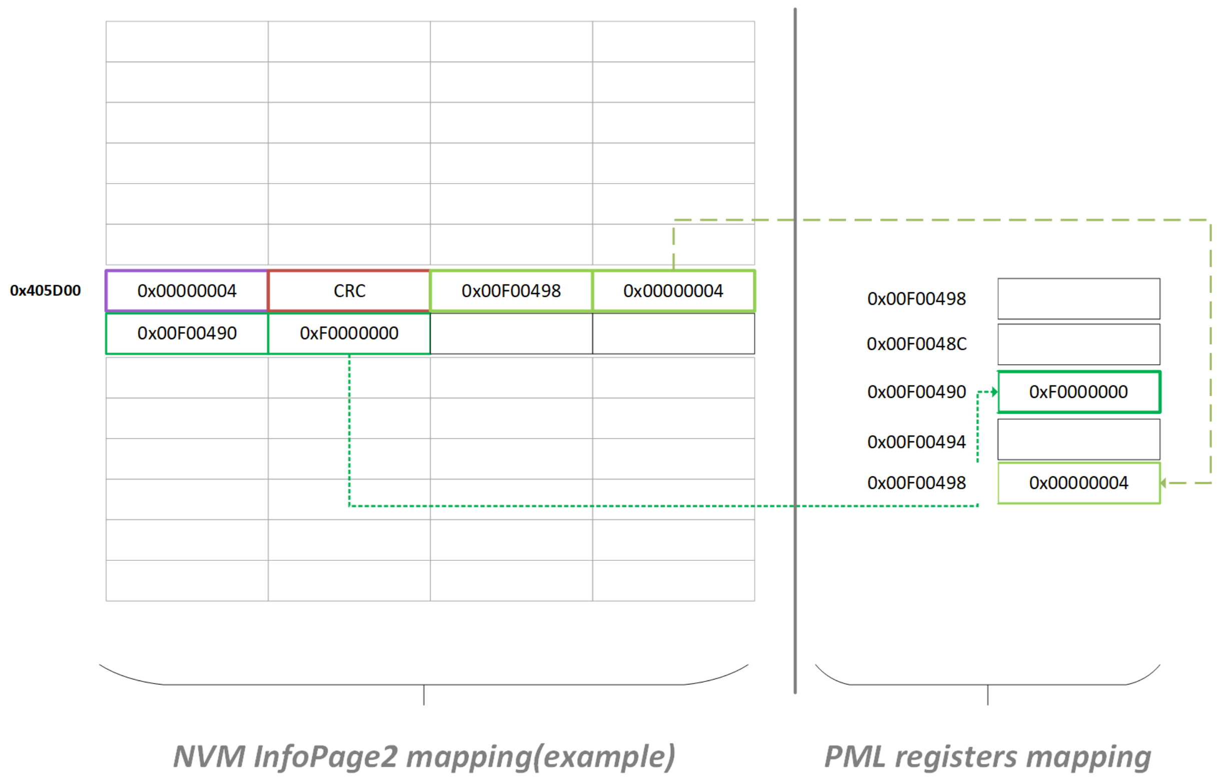


Fig. 4: Example of InfoPage2 lock bits container

Note: Similarly to InfoPage3, because the list of records is protected by the CRC and the length field, it is recommended to set the all the lock bits at the same time. The modification of the lock bits requires to erase the complete container, therefore the full InfoPage2.

Note: The primary goal of the InfoPage2 lock bits container is for user to set the lock bits registers. Nonetheless, when the copy from InfoPage2 to the PML is performed, there is no check that the destination PML register is a lock bit register. Therefore, this mechanism can potentially set any PML register. For instance, user might set trimming registers or any other register. Nevertheless, EM does not recommend it as not initially designed for this purpose. In any case, user will be extremely careful with the order of operations and the cases when those settings apply or not. For instance, if trimming data are set in the lock bit container, one should understand that trimming setting would not apply if USER authentication is not required.

1.5.1.2.3 The combination of InfoPage2 and InfoPage3

It might happen that InfoPage2 and InfoPage3 set the same PML registers. In that case, compared to the bits locked by InfoPage3, InfoPage2 can **only** lock additional bits.

Since the PML registers are one-way only, the loading of the InfoPage3, then the loading of InfoPage2 implies that the **OR** of the InfoPage3 and InfoPage2 configuration is loaded in the registers. For the two directional bits (for instance the JTAG bits), InfoPage2 configuration is loaded in application.

As an example, if we look at the two previous figures:

InfoPage3:

- Set 0x00F00494 to 0xFF000000 (lock of main pages 56..63)
- *Set 0x00F00490 to 0x000000FF (lock of main pages 0..7)*

InfoPage2:

- Set 0x00F00498 to 0x00000004 (lock of InfoPage2)
- *Set 0x00F00490 to 0xF0000000 (lock of main pages 28..31)*

When booting in application mode, the loading from NVM InfoPage3 and InfoPage2 will lead to:

- Set 0x00F00494 to 0xFF000000 (lock of main pages 56..63)
- set 0x00F00498 to 0x00000004 (lock of InfoPage2)
- *Set 0x00F00490 to (0x000000FF OR 0xF0000000), i.e. 0xF00000FF.* That is to say the main pages 28..31 **AND** 0..7 are locked.

1.5.2 How to reset the locks?

Fundamentally, **the locks are designed to be permanent**.

Nonetheless, there are 2 options to reset the locks:

1-Apply a POR or any system reset

When there is a POR, all the PML registers are reset to the null value. It results that no lock is active.

Nonetheless, if the chip is booted in application mode, the locks located in NVM InfoPage2 and InfoPage3 are copied into the PML registers. So one will ensure that the locks are also reset in the NVM pages.

2-Reconfigure InfoPage2 and\or InfoPage3

When entering in **EM configuration mode**, no lock applies. InfoPage3 content is not copied into the PML registers. It results that InfoPage3 can be erased and fully reconfigured.

When entering in **User configuration mode**, InfoPage2 content is not copied into the PML registers. Therefore InfoPage2 can be erased and reconfigured entering **User configuration mode**. This is possible unless InfoPage3 locked InfoPage2 (which is normally not the case).

1.5.3 When do the locks apply?

The locks are active **as soon as the PML registers are set**.

When the locks are set by the software application code, the locks apply immediately.

For the locks set in the NVM InfoPage3 and InfoPage2, the locks apply or not depending on the device mode.

- If the device is in **EM configuration mode**, no lock applies. EM may erase the InfoPage2 and the InfoPage3 and completely reconfigure the part.
- If the device is in **USER configuration mode**, the locks of InfoPage3 still apply. On the other hand, the locks defined in InfoPage2 do not apply. It allows the user to reset most of the configuration and use JTAG.
- If the device is in **application mode**, all the locks apply before jumping into the flash code.

For more information on the configuration modes and how to enter in these modes, the reader is referred to the document "Configuration modes".

1.6 Configuration of InfoPage3 at delivery

The typical configuration of InfoPage3 is as follows:

Table 7: Typical InfoPage3 configuration at shippment time

Bit name	Status
PmlJtag2wEn	0
PmlJtag4wEn	0
PmlTxPaPwrLock	Depends on the product configuration
PmlLdoAntTrimLock	Depends on the product configuration
PmlTmLock	**Locked**
PmlUsbLock	Depends on the product configuration
PmlJtagLock	Not locked
RegNvmLockMain0	Not locked
RegNvmLockMain1	Not locked
NvmEraseLockInfoPage0	Not locked unless a key was provisionned in InfoPage0
NvmWriteLockInfoPage0	Not locked
NvmLockInfoPage	InfoPage3 **locked**. Others unlocked
NvmLockRedund	**Locked**
NvmLockEraseFull	**Locked**
NvmLockEraseMain	Not locked
NvmKcLockKey	None locked unless one key is provisionned in InfoPage0

1.7 Configuration of InfoPage2

InfoPage2 is typically delivered fully erased.

InfoPage2 can be built manually or using a script.

A python script "**lifecycle_ctrl.py**" together with a json file is provided to ease the generation of the InfoPage2. "**lifecycle_ctrl.py**" is included in the SDK.

1.8 Document version

Table 8: Document versions

Version	Date	Description
1.0	1st February 2023	Initial Version
1.1	29th April 2024	Minor fixes and clarifications