DIGITAL MULTIPHASE

GEN 2 PROGRAMMING GUIDE

JUNE 2020

BIG IDEAS
FOR EVERY SPACE

OVERVIEW

- **This guide specifies the algorithm for programming Renesas generation 2 digital multiphase products via PMBus communication.**

- **Unless noted otherwise, timing and voltage requirements are outlined in the PMBus specification version 1.3.**

- **The following sections are shown in this guide:**
  - Section 1: Programming devices
  - Section 2: HEX file CRC information

Note: This guide is valid only for devices using firmware versions:

2.0.x.y where x > 1

or

2.0.0.x where x > 7

Prerequisites and Conventions

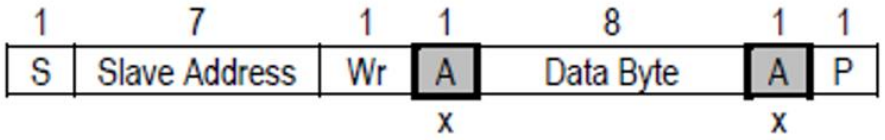
- HEX configuration files must be generated using PowerNavigator version 5.4.198 or earlier.
- In this guide, address 0x60 (7-bit format) is used in all examples.
- Data on the bus may be reversed. Follow examples for correct byte order.

Minimum Pin and Component Requirements

The following pins must be connected when programming a device:

- PMSCL and PMSDA (I^2C clock and data pins). These are open drain and must be pulled to 3.3V via a resistor (1kΩ maximum).
- VCC, provided with an external 3.3V supply. A 1uF decoupling capacitor from this pin to ground is also needed.
- VCCS must be decoupled with 4.7µF or greater MLCC (X5R or better).
- Ground pin must be connected to ground.
- ADDRESS pin must have an address set resistor to ground. Connect directly to ground for address 0x60.
- All EN pins held low.
- Other pins may be floated.

PMBus Communication Key



S	Start Condition
Sr	Repeated Start Condition
Rd	Read (bit value of 1)
Wr	Write (bit value of 0)
x	Shown under a field indicates that that field is required to have the value of 'x'
A	Acknowledge (this bit position may be '0' for an ACK or '1' for a NACK)
P	Stop Condition
PEC	Packet Error Code
	Master-to-Slave
	Slave-to-Master
...	Continuation of protocol

Note: See PMBus/SMBus spec for additional details and timing requirements.

Direct Memory Access (DMA) Command Codes

Actual device programming is completed through 3 command codes:

- **DMA Address (Command Code 0xC7):** Used to set the register address to use with other DMA commands.
- **DMA Data (Command Code 0xC5):** Used to read from or write to the register selected by the DMA Address command.
- **DMA Sequential (Command Code 0xC6):** Used to read from or write to the register selected by the DMA Address command, then automatically increment the register address by 1.

Important Notes

- The command that causes the OTP to burn is the last line of the HEX file. Device programming can be aborted at any point before this, and no contents will be burned to OTP.

- The last byte of each line in the HEX file is a CRC8 check for that line in the file only. It does not relate to any hardware value.

SECTION 1: PROGRAMMING DEVICES

PROGRAMMING ALGORITHM OVERVIEW

1. **Determine number of NVM slots available. (Optional)**

2. **Verify device and file versions.**
   a. Read and parse header data from HEX file. Go to the Step 2a section for more information.
   b. Read IC_DEVICE_ID from device. Verify the value matches the Device Table.
   c. Read IC_DEVICE_REV from device. Verify the value matches the HEX file.

3. **Read and parse one line from HEX file. Write to device.**
   - This step must be repeated for all configuration lines in the HEX file.

4. **Verify programming success.**
   a. Poll PROGRAMMER_STATUS register until programming is complete.
   b. Read the BANK_STATUS registers to confirm all configurations were programmed successfully.

PROGRAMMING ALGORITHM OVERVIEW

5. **Reload firmware**

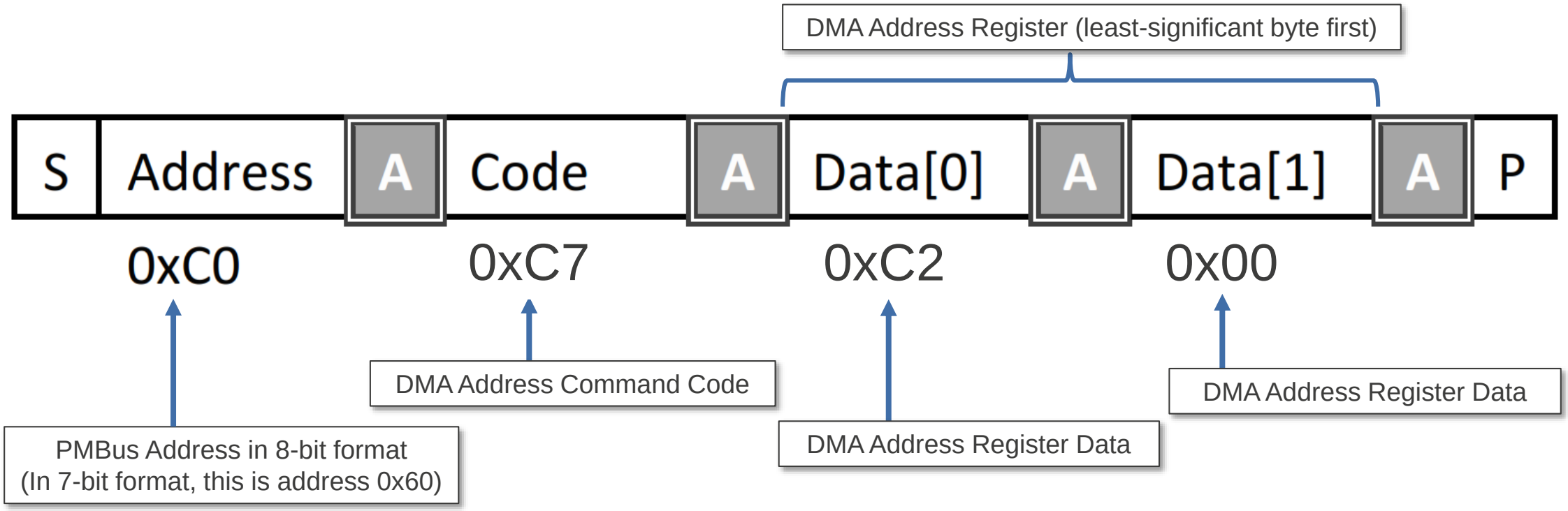
6. **Recompute OTP bounds**
   - Without POR, this routine must be started manually

7. **Start device firmware**

8. **Verify CRCs**

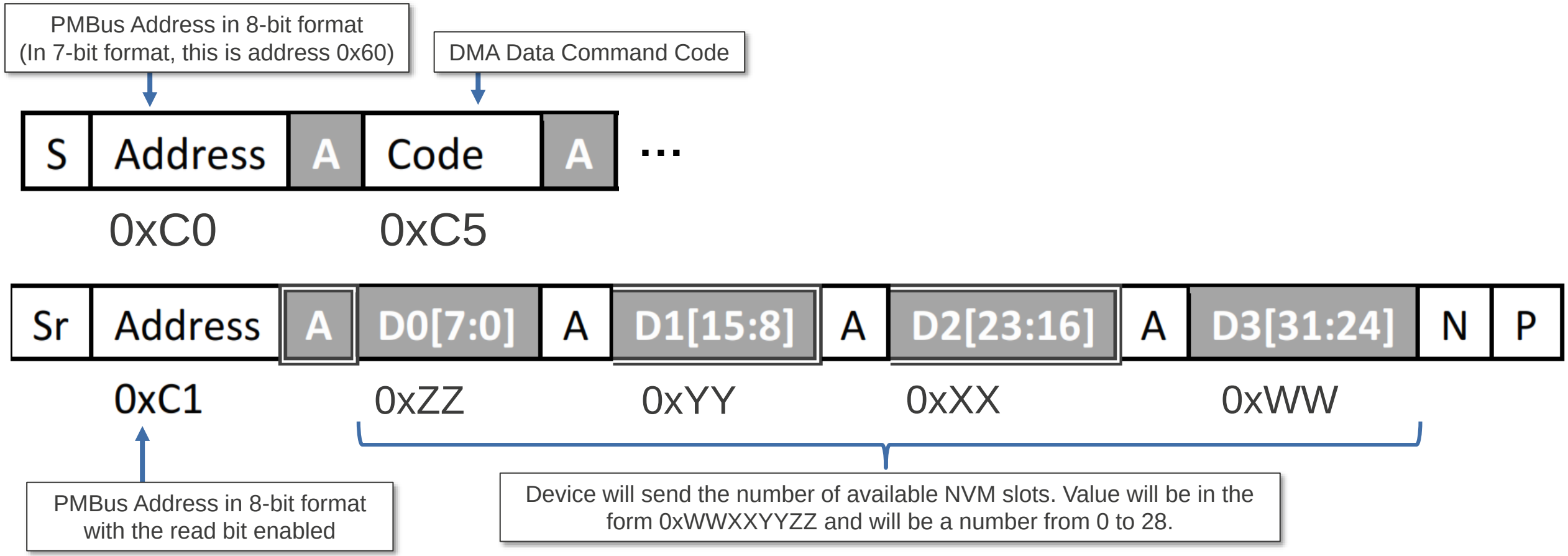
Step 1a – Write to DMA Address Register

To read the number of available NVM slots, first set the DMA address, as shown below.

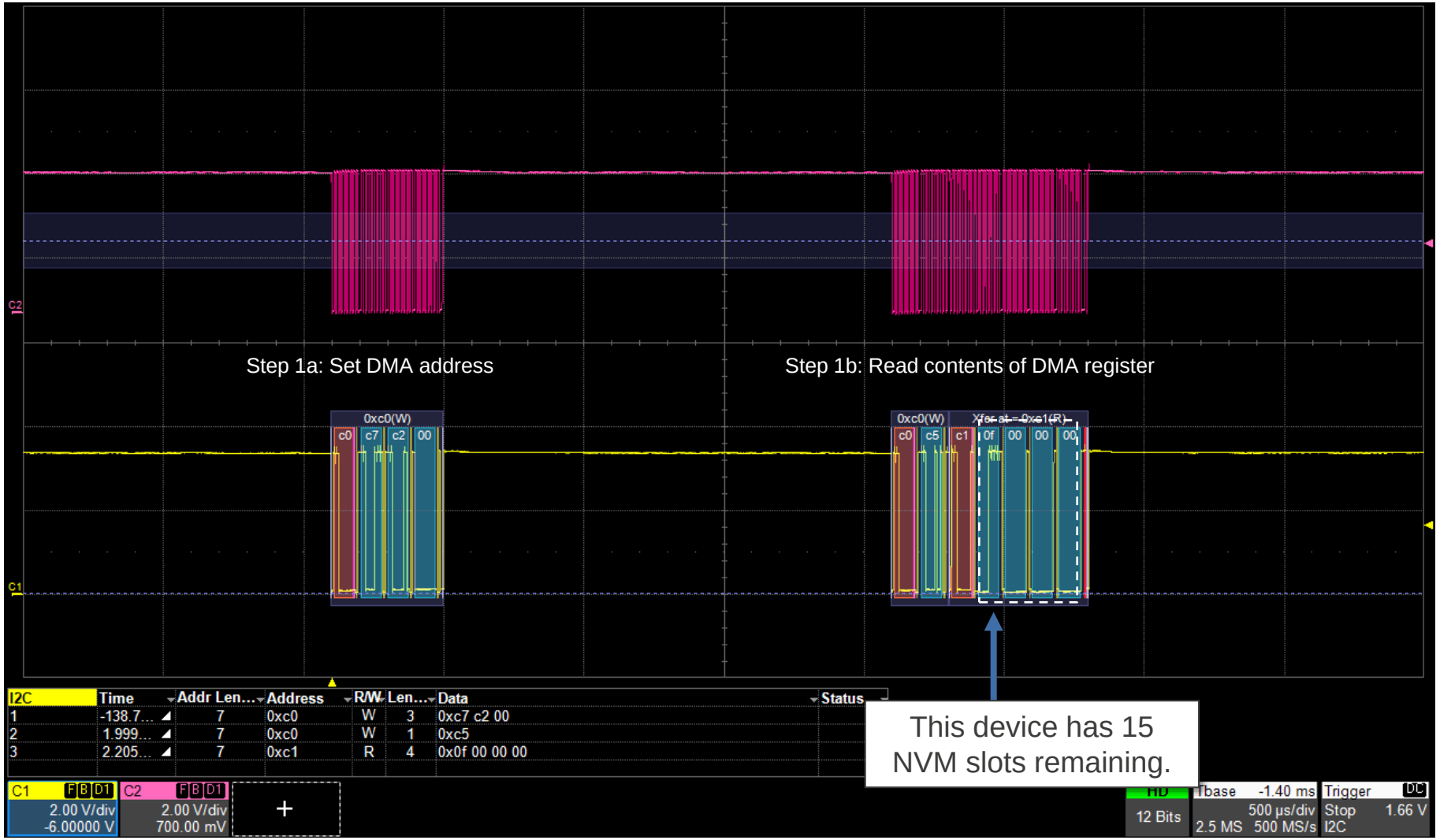


Step 1b – Read DMA Data Register

Next, Read the content of the register pointed to in step 1.



Step 1 – Example Waveforms



Step 2a – Parse Header in HEX File

- The first 5 lines in the HEX file (starting with 0x49) are part of the HEX file header and should not be written to the device.

- The HEX header contains IC_DEVICE_ID, IC_DEVICE_REV and HEX_VERSION information.
  - IC_DEVICE_ID in HEX file must match IC_DEVICE_ID read back from device (see step 2b).

Step 2a – Example HEX File

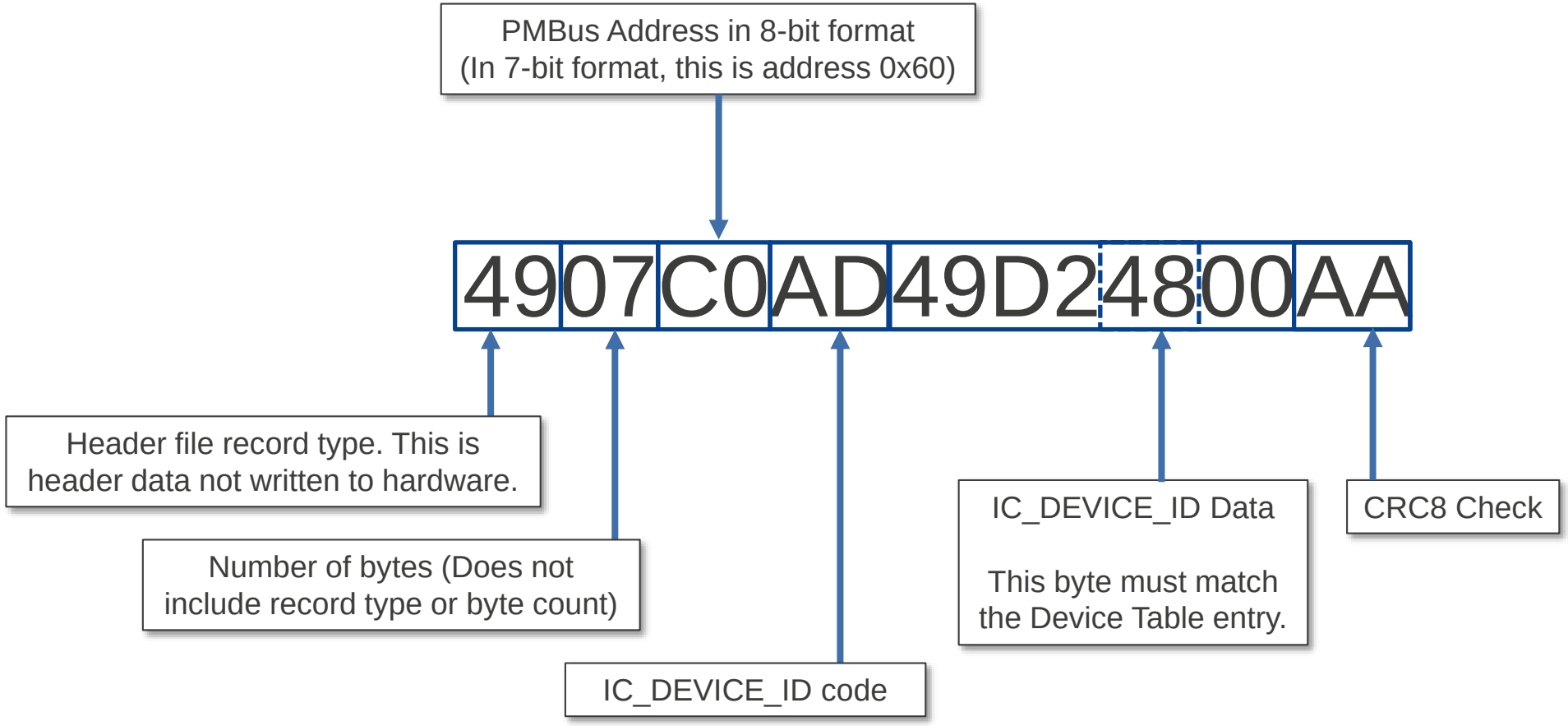
```
1   4907C0AD49D24800AA
2   4907C0AE0200000017
3   4907C00000000200B0
4   490AC001352E342E31333573
5   490BC00200000168C930F47B6F
6   0005C0E6020033
7   0005C0C7000724
8   0007C0C6E1000000FC
9   0007C0C6001C0000D6
10  0007C0C6840400004C

                ...

640   0005C0C70C07D8
641   0007C0C60100000098
642   0005C0C721018D
643   0007C0C6AD1D0000BC
644   0005C0C7DB001C
645   0007C0C6000000008E
646   0005C0C7DD0062
647   0007C0C6000000008E
648   0005C0E6060067
649
```

Parse first two lines and verify the HEX file was generated for the device being programmed (see steps 2b and 2c).

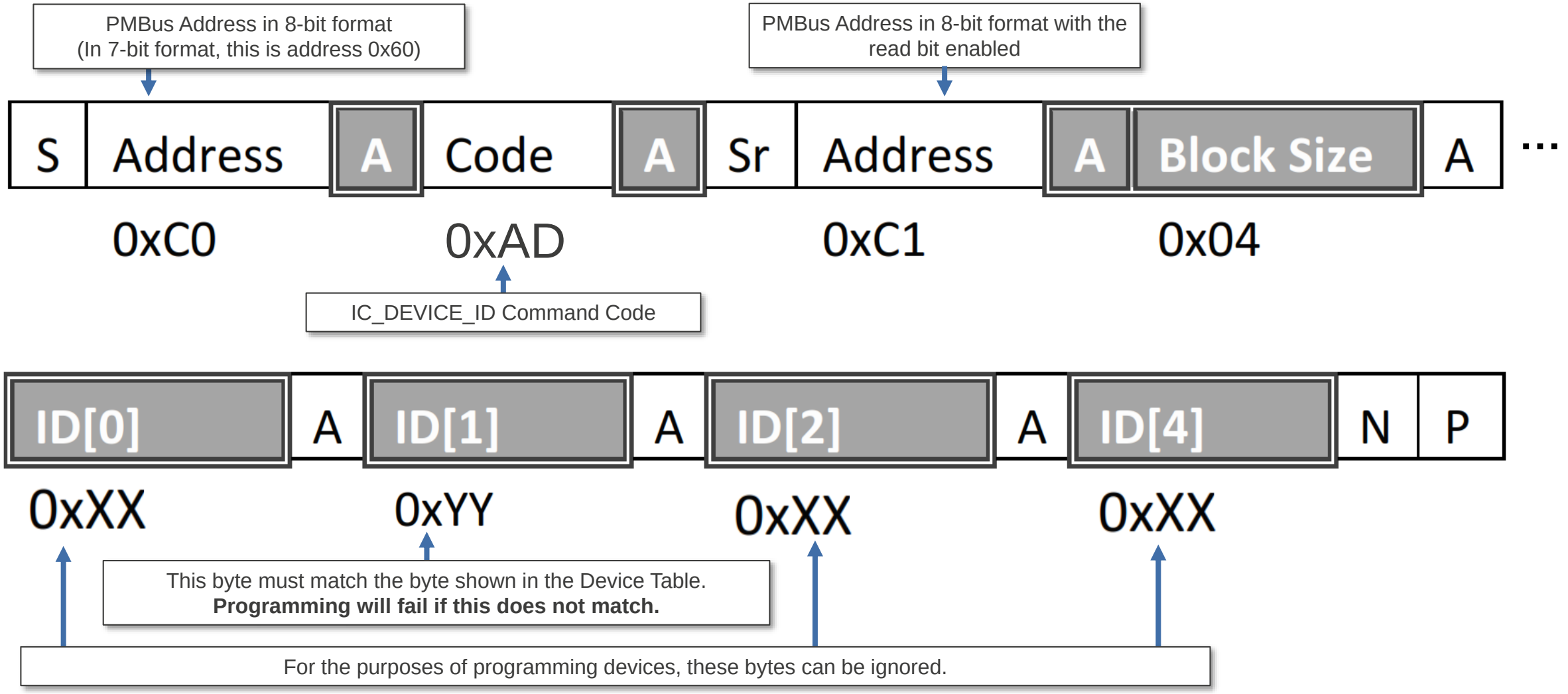
- Line 1 = IC_DEVICE_ID
- Line 2 = IC_DEVICE_REV
- Line 3 = HEX_VERSION

Value contained in file	Record Type	Command Code
IC_DEVICE_ID	0x49	0xAD
IC_DEVICE_REV	0x49	0xAE
HEX_VERSION	0x49	0x00

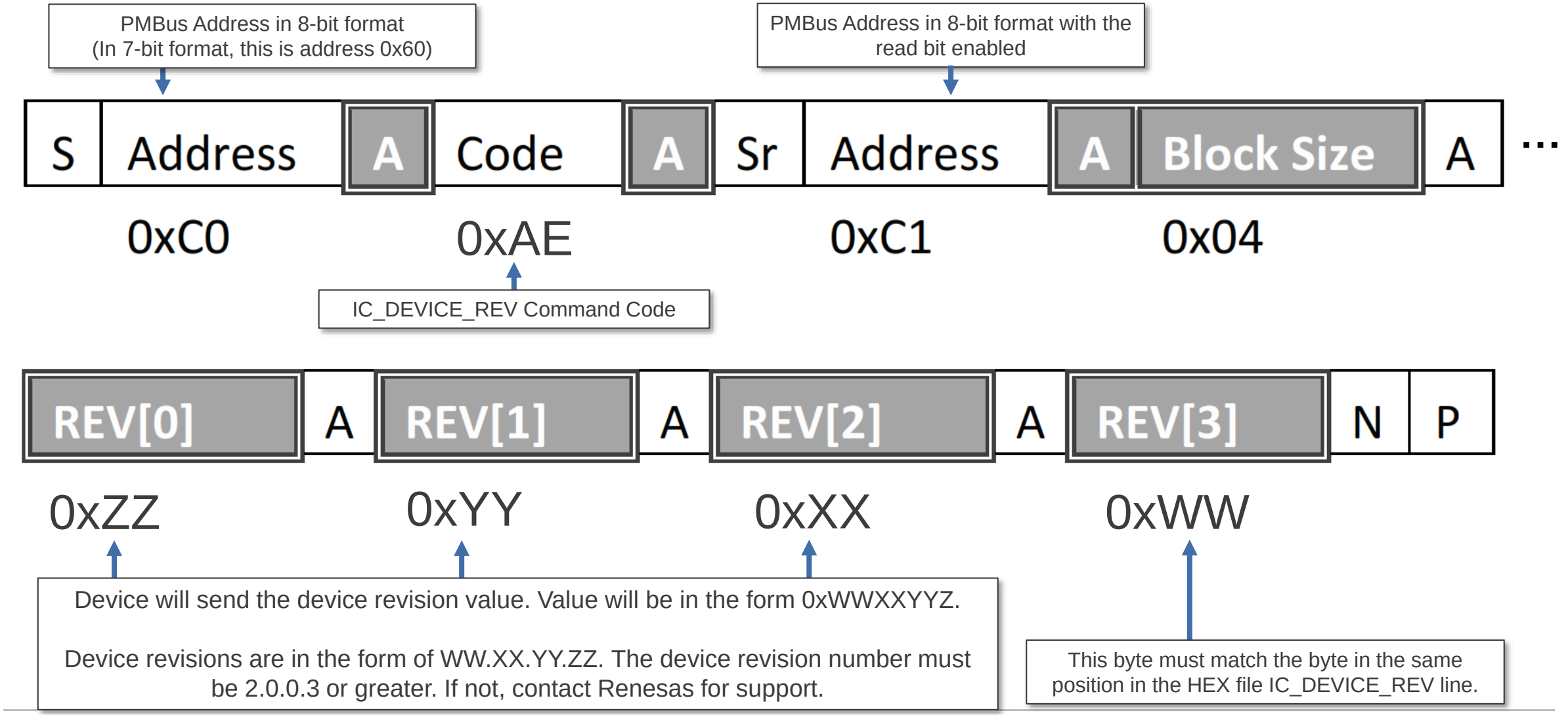
Step 2a – Example HEX File Header Decode



Step 2b – Read IC_DEVICE_ID from Device



Step 2c – Read IC_DEVICE_REV from Device



Step 2d – Compare data to HEX file

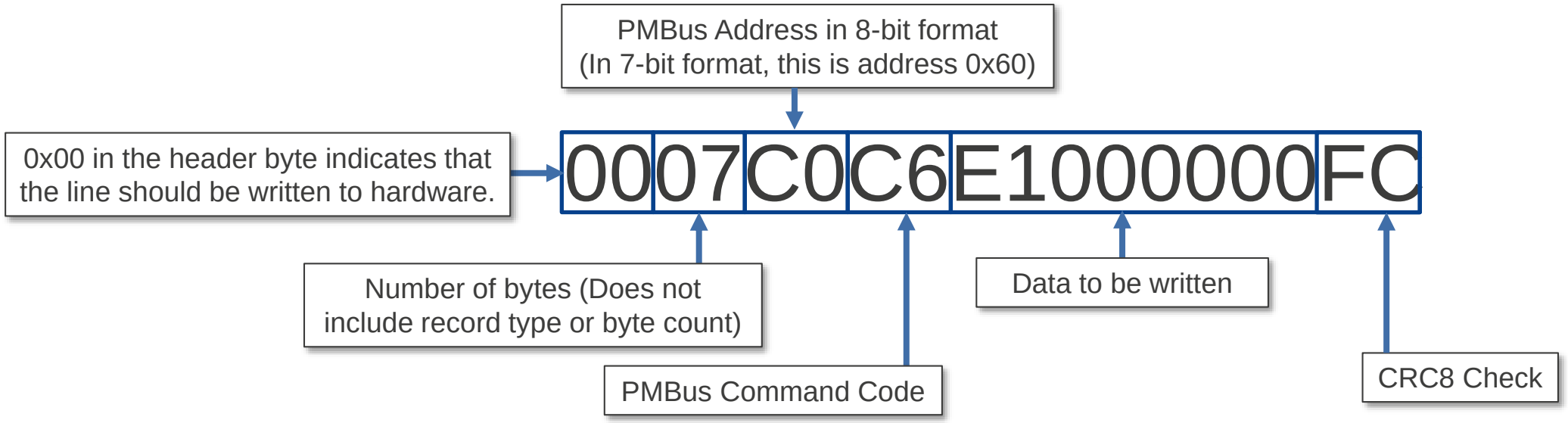
- If the IC_DEVICE_ID and IC_DEVICE_REV bytes in the HEX file match the bytes read back from the device, proceed to step 3.

- If it does not match, the HEX file was generated for a different device and device programming should be halted.

Step 3 – Parse HEX File and Write to Hardware

Parse remaining lines from HEX file (all lines not starting with 0x49), and write to device.

Example HEX File Line:



Step 3 – Example HEX File

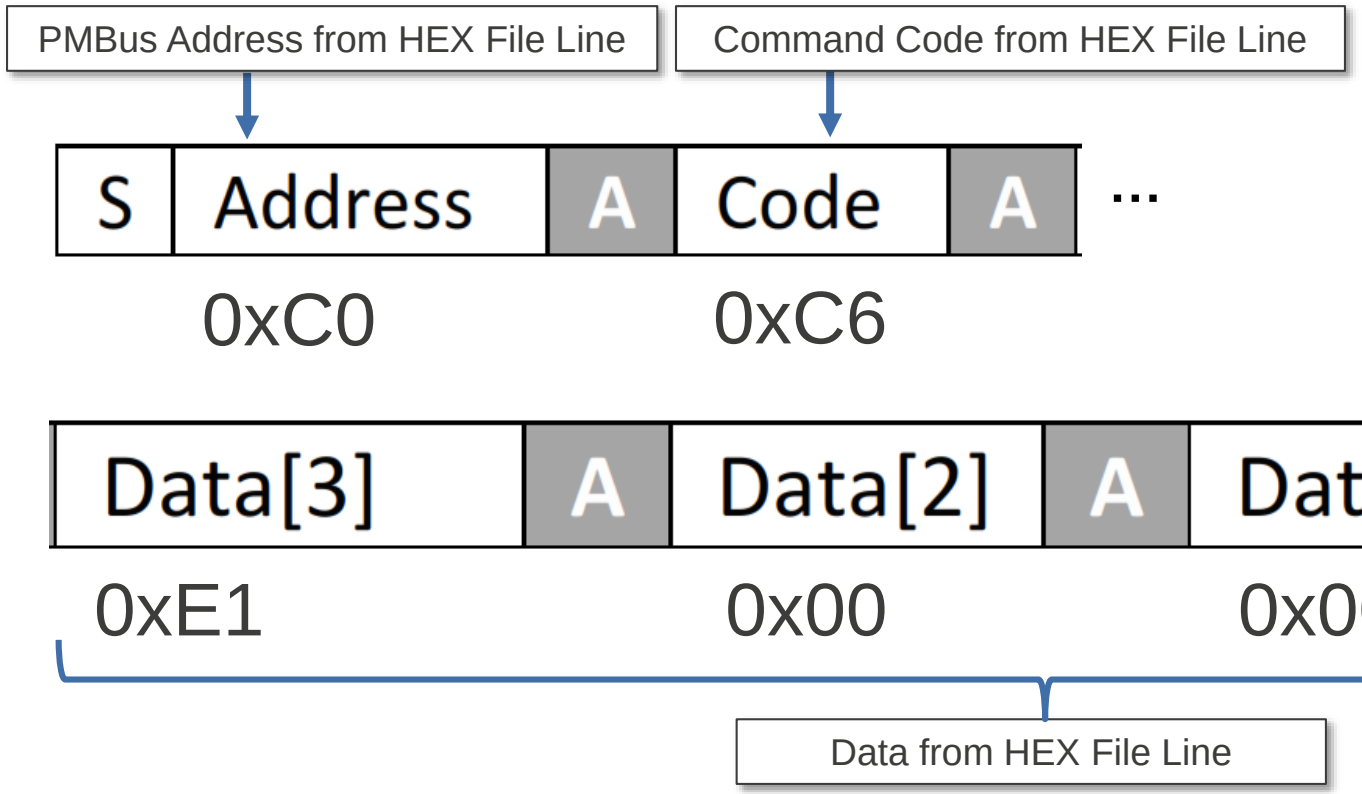
```
 1  4907C0AD49D24800AA
 2  4907C0AE0200000017
 3  4907C00000000200B0
 4  490AC001352E342E31333573
 5  490BC00200000168C930F47B6F
 6  0005C0E6020033
 7  0005C0C7000724
 8  0007C0C6E1000000FC
 9  0007C0C6001C0000D6
10  0007C0C6840400004C

          ...

640  0005C0C70C07D8
641  0007C0C60100000098
642  0005C0C721018D
643  0007C0C6AD1D0000BC
644  0005C0C7DB001C
645  0007C0C6000000008E
646  0005C0C7DD0062
647  0007C0C6000000008E
648  0005C0E6060067
649
```

Step 3: Parse and write to device all lines in HEX file that do not start with 0x49

When last line has been written to device, Step 3 is complete.

Step 3 – Example Write of HEX File Data Line

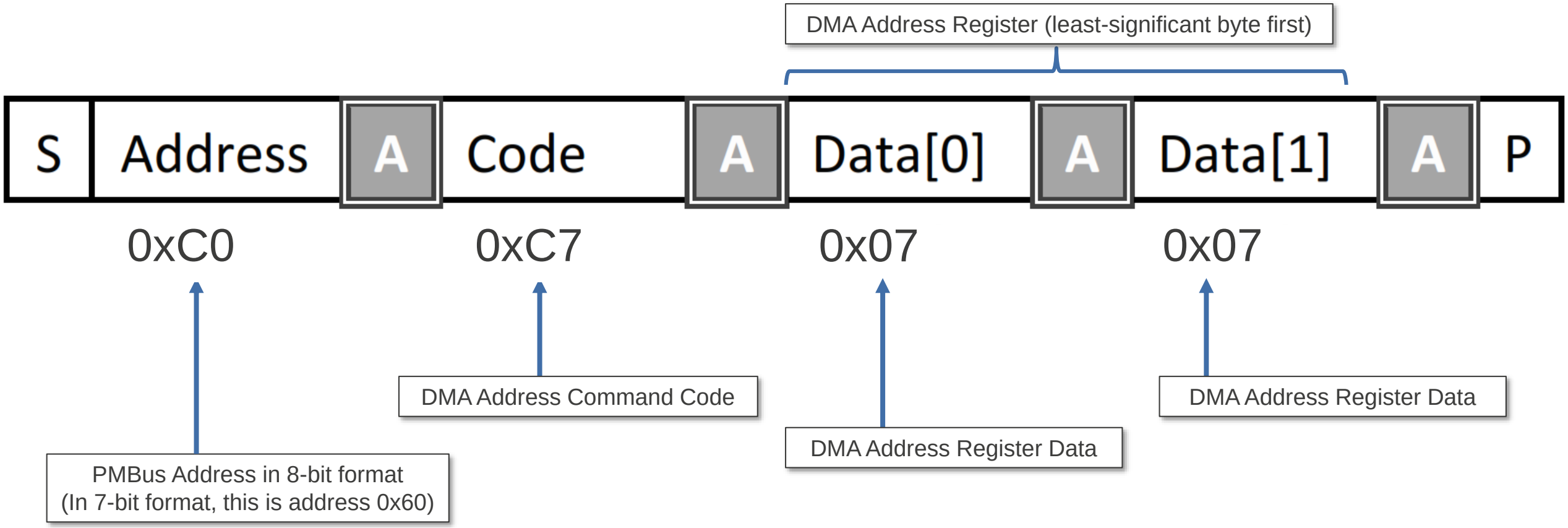
Example PMBus Command:



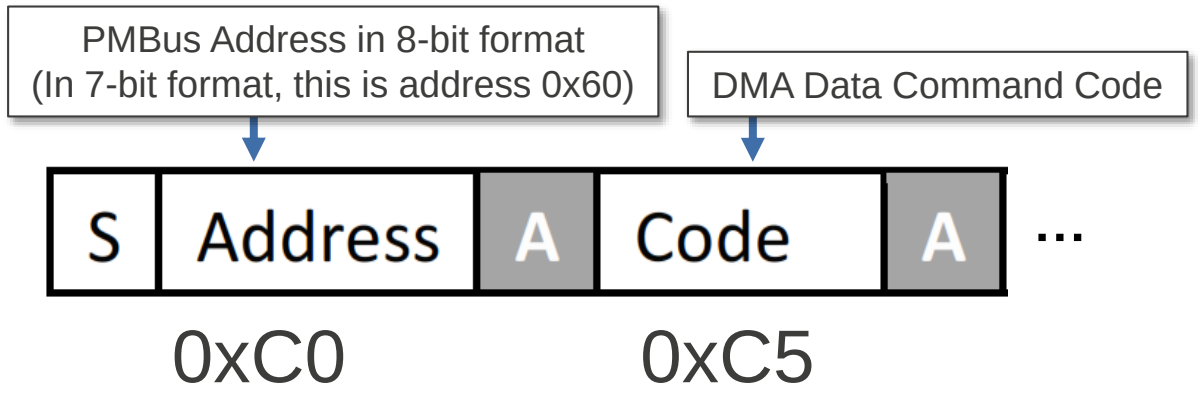
Hex File Line Data: 0007C0C6**E1000000**FC

Step 4a – Poll PROGRAMMER_STATUS Register

To check for the completion of device programming, first write to the DMA address register as shown below then read the DMA data register until bit 0 is set to 1.

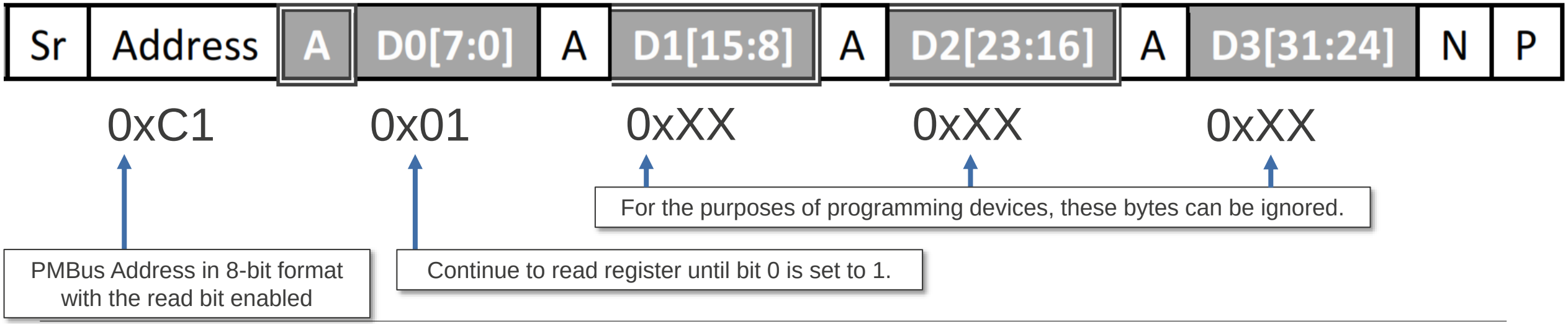


Step 4b – Poll PROGRAMMER_STATUS Register



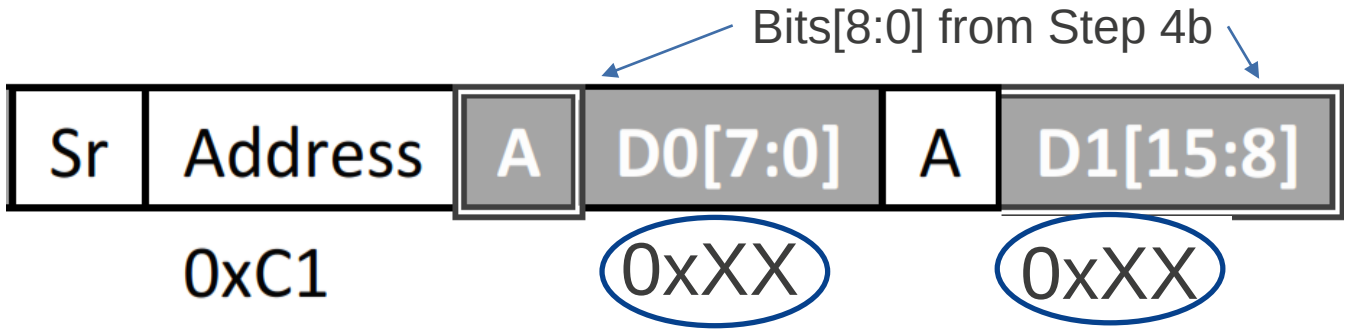
After completing step 4a, use the DMA read command to poll register until bit 0 is set to 1.

If after 2s timeout, bit 0 has not been set to 1, the part has failed programming. See step 4c for more details.



Step 4c – Programming Failure

To determine programming failure, take bits[8:0] from Step 4b and decode using the following…



b0X0X0XXX0

If bit 8 is 1, the HEX file contains more configurations than are available. Programming fails before OTP banks are consumed.

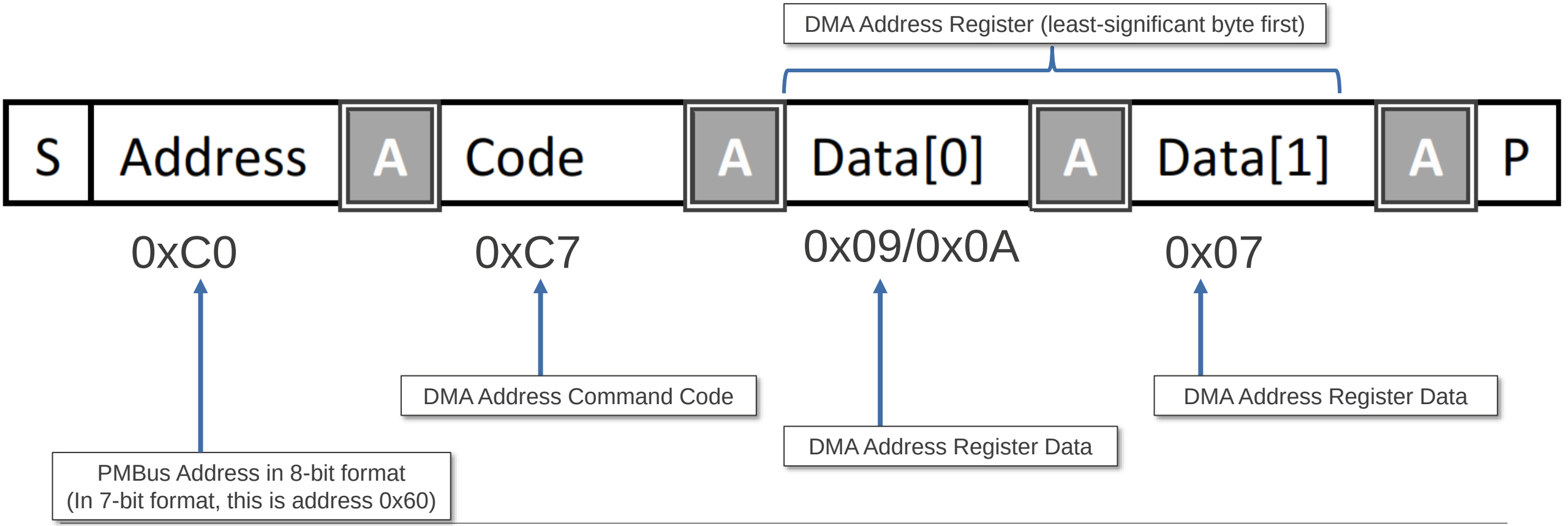
If bit 6 is 1, the CRC check fails on the OTP memory. Programming fails **after** OTP banks are consumed.

If bit 4 is 1, a CRC mismatch exists within the configuration data. Programming fails before OTP banks are consumed.

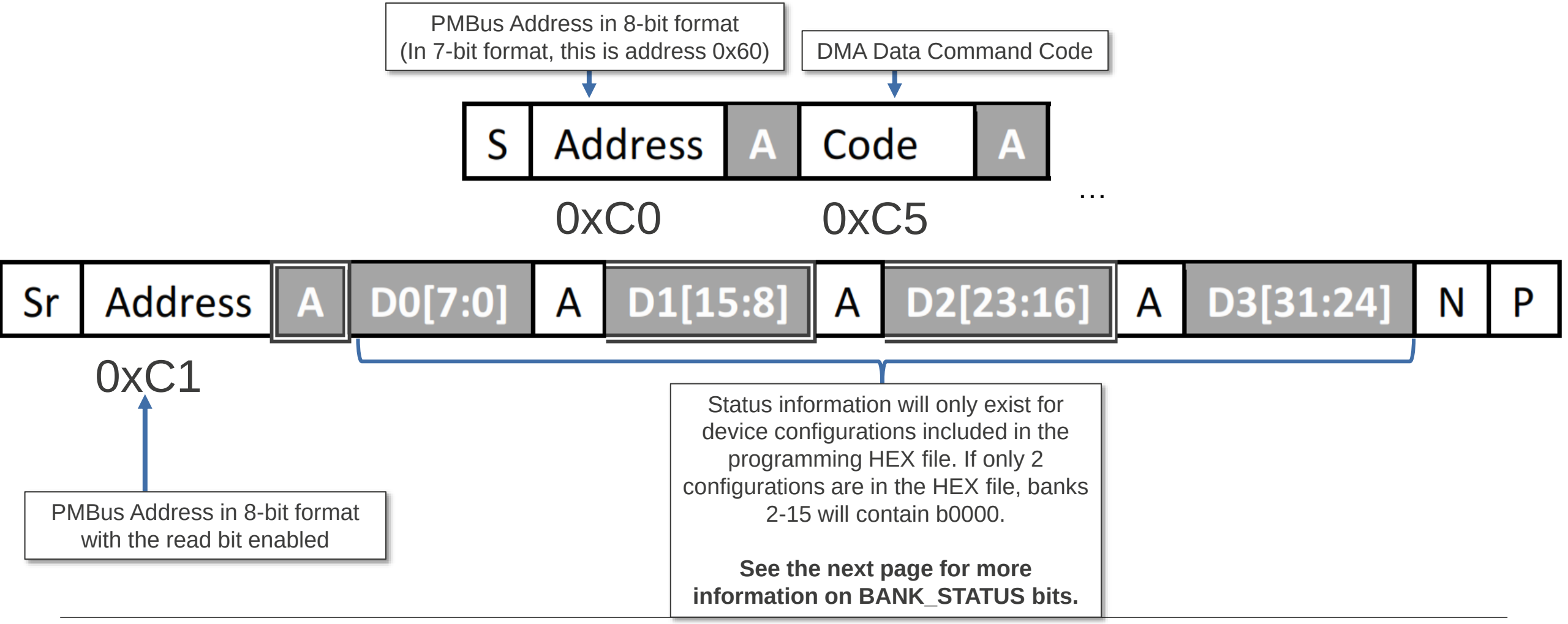
If this bit is 0, programming has failed.

Step 4d – Read BANK_STATUS Registers

To read the BANK_STATUS registers, first write to the DMA address register as shown below then read the DMA data register to retrieve BANK_STATUS data.



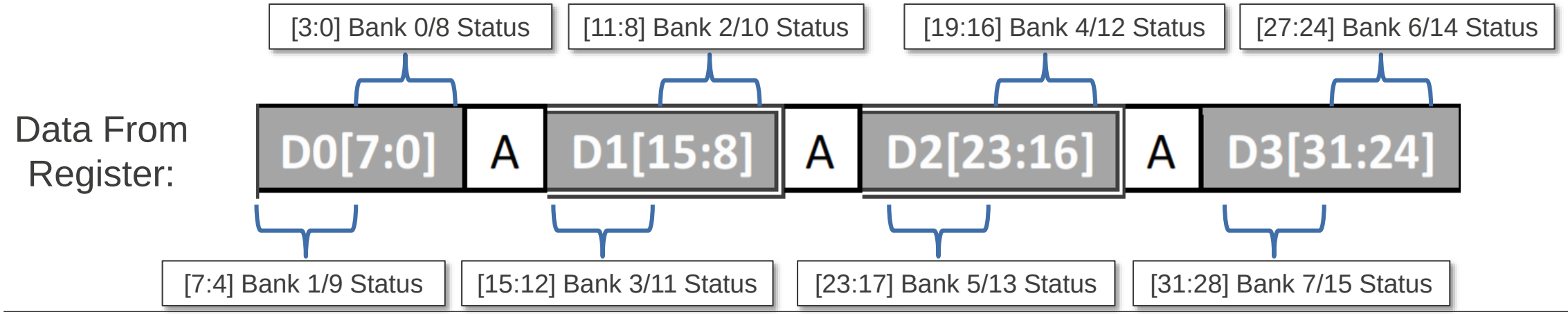
Step 4d – Read BANK_STATUS Registers



Step 4d – Read BANK_STATUS Registers

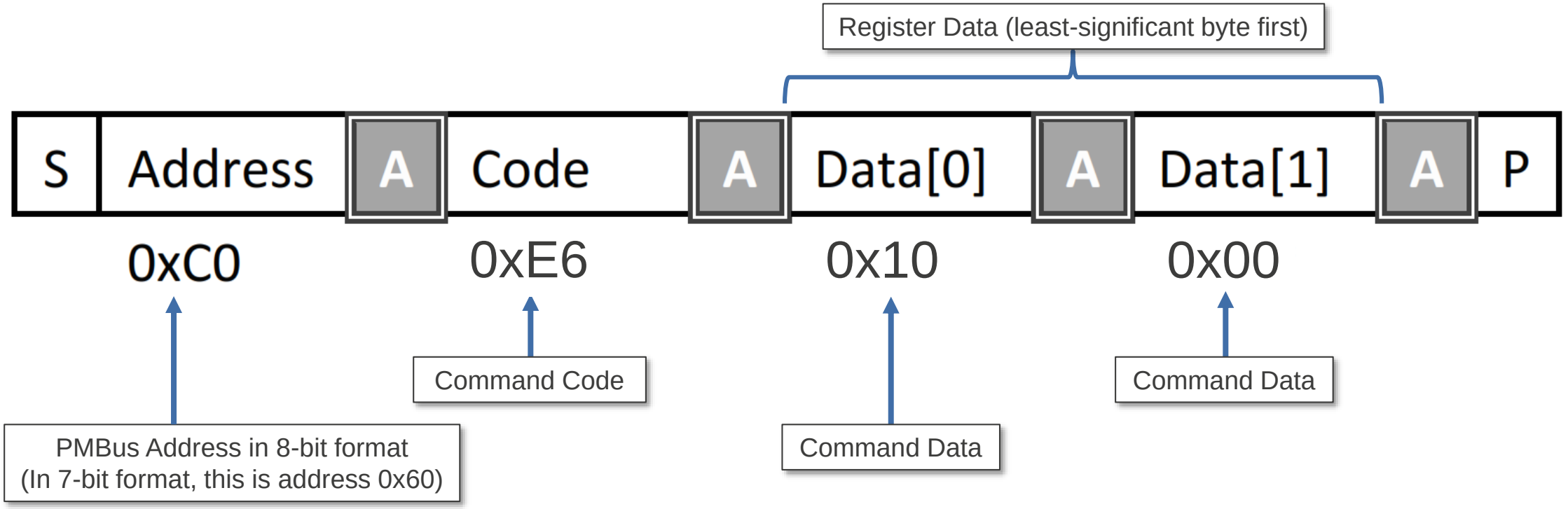
Every set of bank status bits will correspond to the table below:

Bank Status Bits	Description
b1000	Fail: CRC mismatch OTP
b0100	Fail: CRC mismatch RAM
b0010	Reserved
b0001	Bank Written (No Failures)
b0000	Bank Unaffected



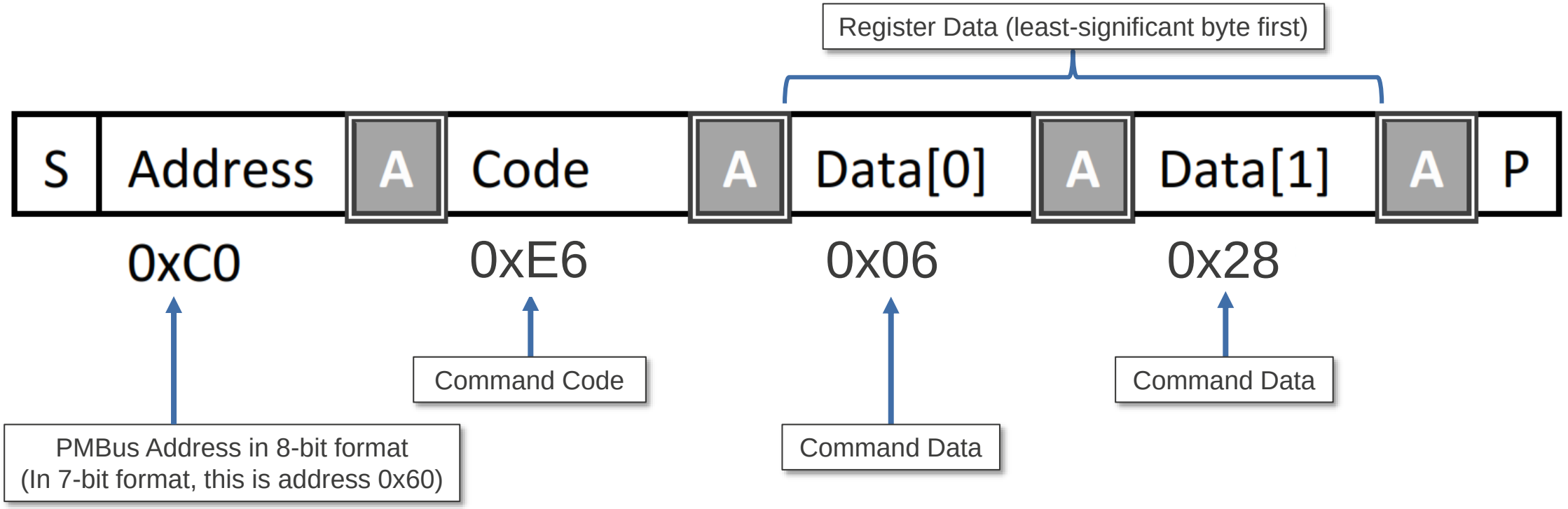
Step 5 – Load Firmware to RAM

Next, send 0x0010 using command code 0xE6. **Programmer must wait at least 1ms after completing this step.**



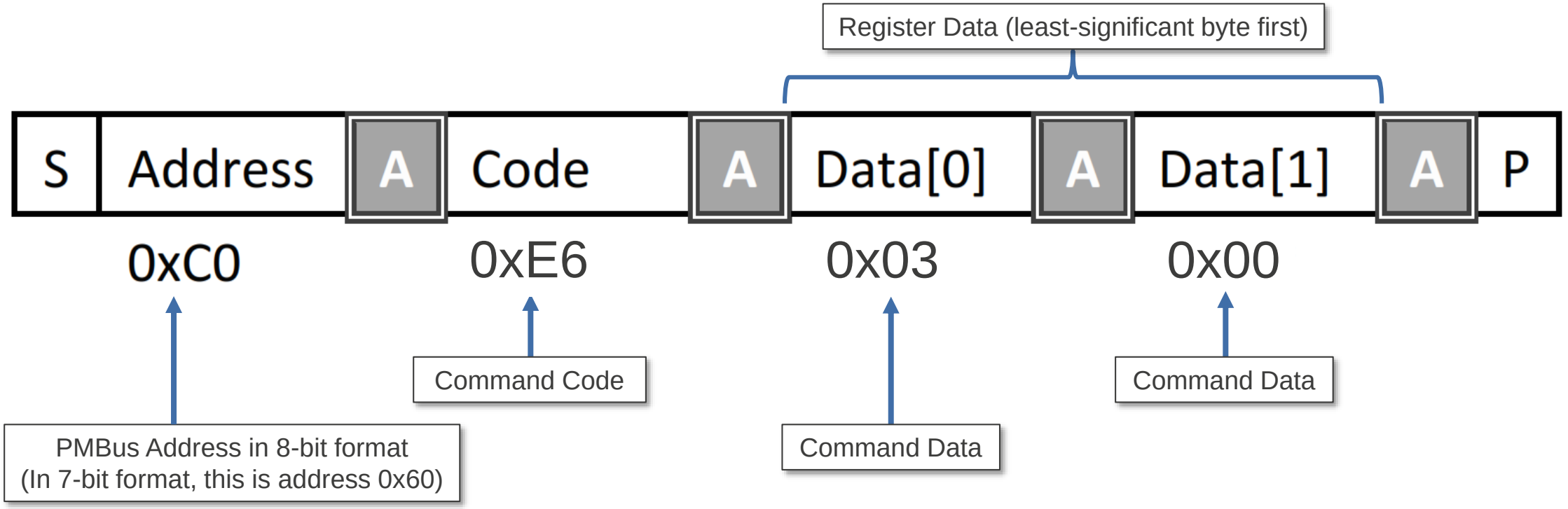
Step 6 – Recompute OTP Bounds

Next, send 0x2806 using command code 0xE6. **Programmer must wait at least 1ms after completing this step.**



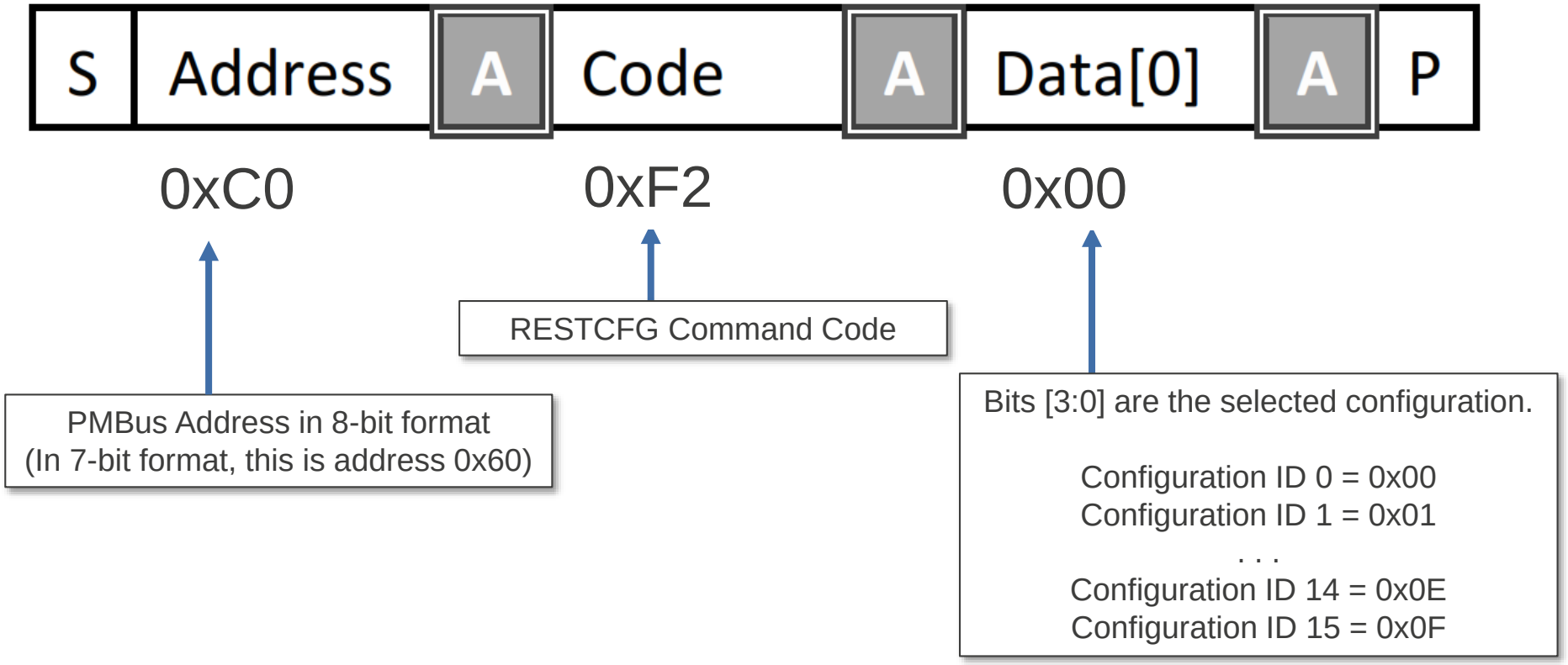
Step 7 – Wake Firmware

Next, send 0x0003 using command code 0xE6. **Programmer must wait at least 1ms after completing this step.**



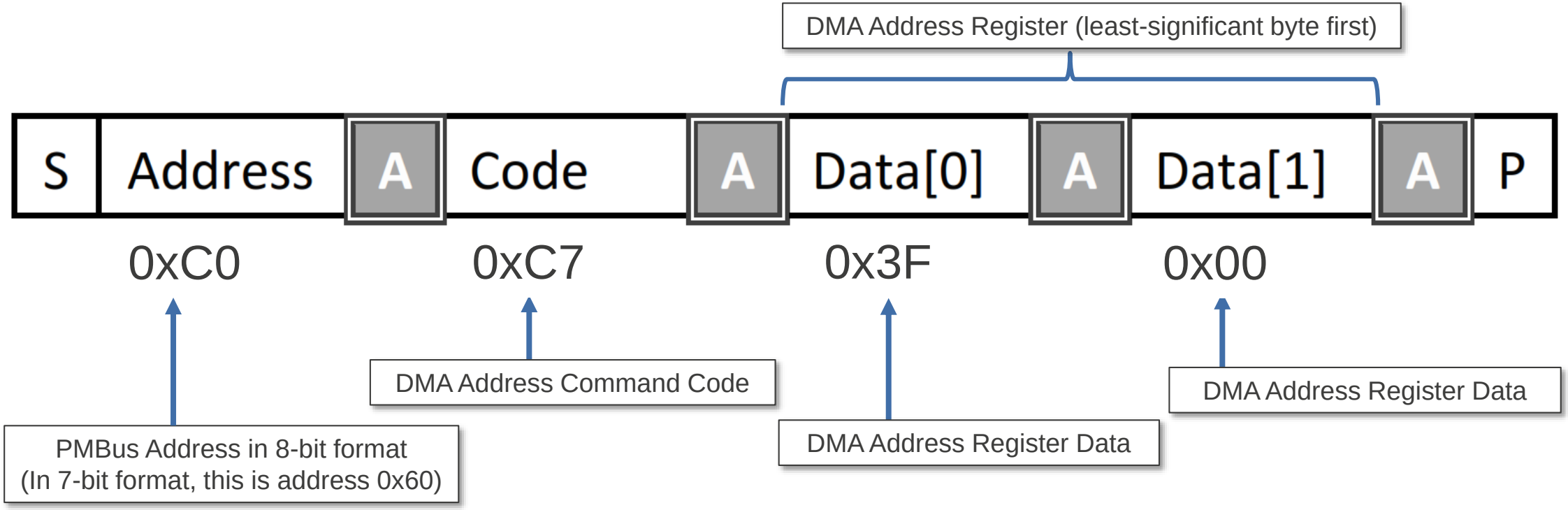
Step 8a – Use RESTCFG Command

Use the RESTCFG command code in the format shown below to load a config from OTP into RAM. Do **not** use this command if the device is regulating.



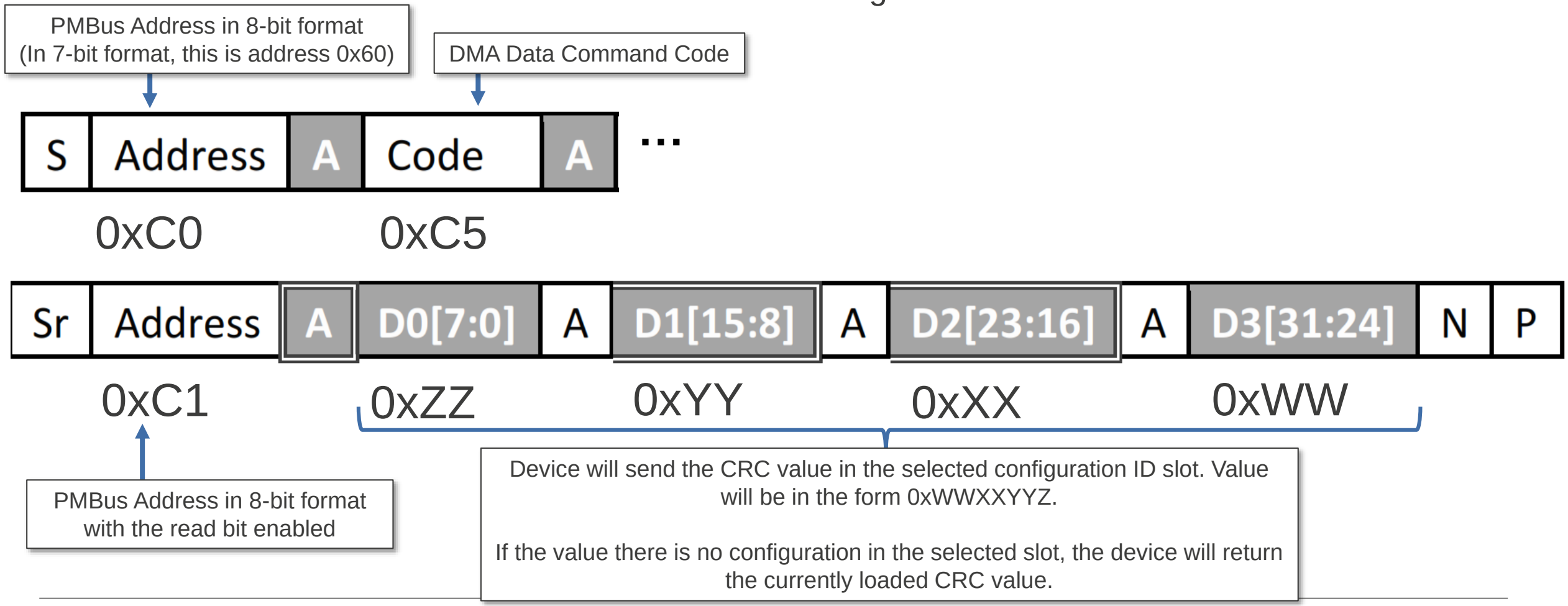
Step 8b – Write to DMA Address Register

To read the CRC value from the Configuration ID selected in Step 8a, first set the DMA address as shown below.



Step 8c – Read DMA Data Register

Next, Read the content of the register pointed to in Step 8b. This is the CRC value in the selected Configuration ID slot.



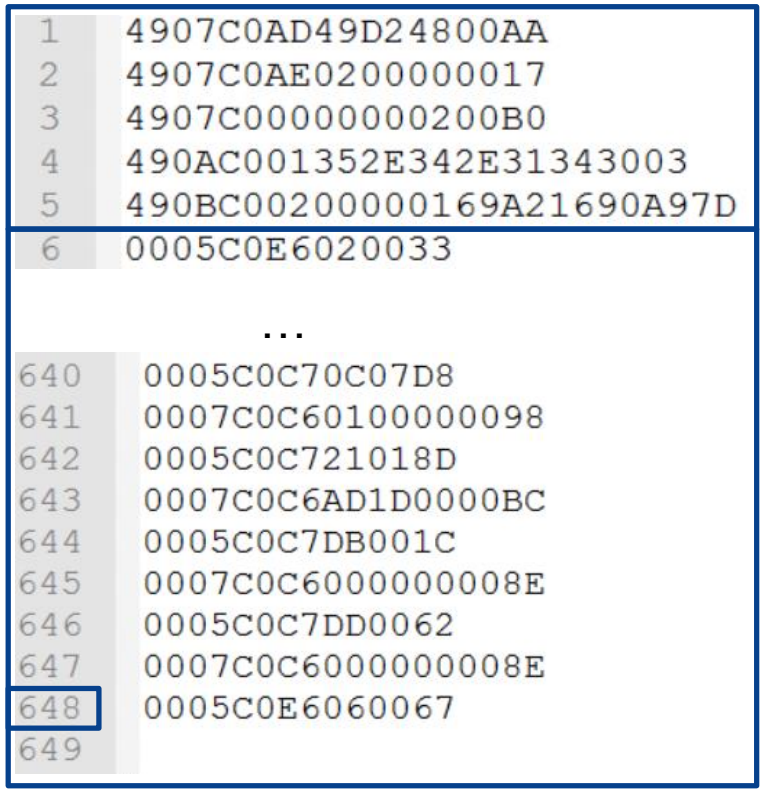
Algorithm Completion

- After successfully completing programming, outputs may be re-enabled.

SECTION 2: HEX FILE CRC INFORMATION

HEX Files – Total Number of Configurations

$$\text{Total number of configurations} = \frac{\text{\# of lines in HEX file} - 290}{358}$$



In the example HEX file, there are 648 lines.

$$\text{Total number of configurations} = \frac{\text{\# of lines in HEX file} - 290}{358}$$

$$\text{Total number of configurations} = \frac{648 - 290}{358}$$

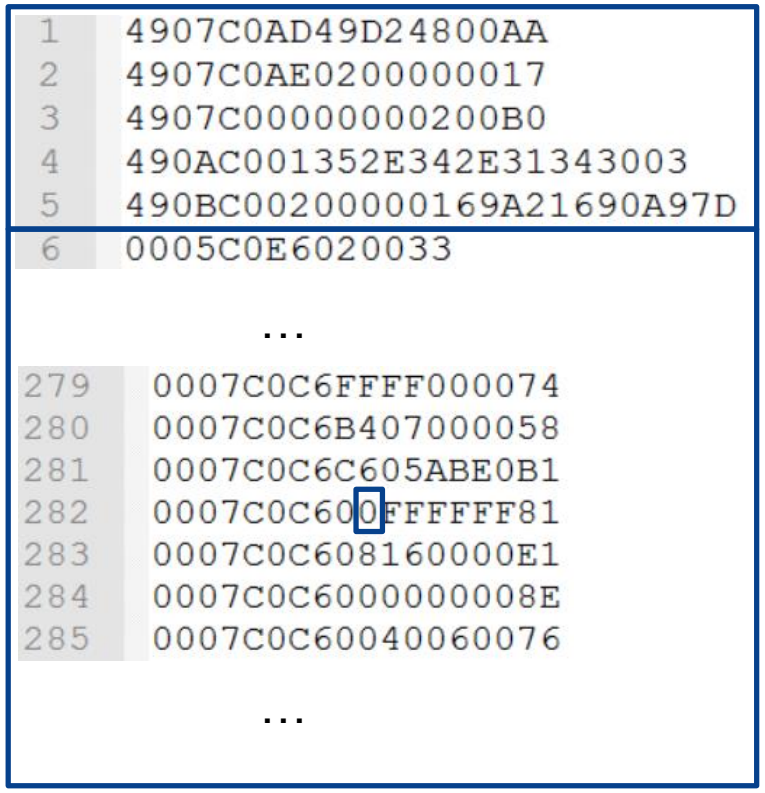
$$\text{Total number of configurations} = \frac{358}{358}$$

Total number of configurations = 1

The example HEX file contains **1 configuration**.

HEX Files – Configuration Slot IDs

Slot ID line number = (N ∗ 358) + 282, N = 0, 1, . . ., # of configurations in file – 1



In the example HEX file, there are 648 lines and 1 configuration.

N = 0, 1, . . ., # of configurations in file – 1

Slot ID line number = (N ∗ 358) + 282

For the 1st configuration in the file, N = 0.

Slot ID line number = (0 ∗ 358) + 282

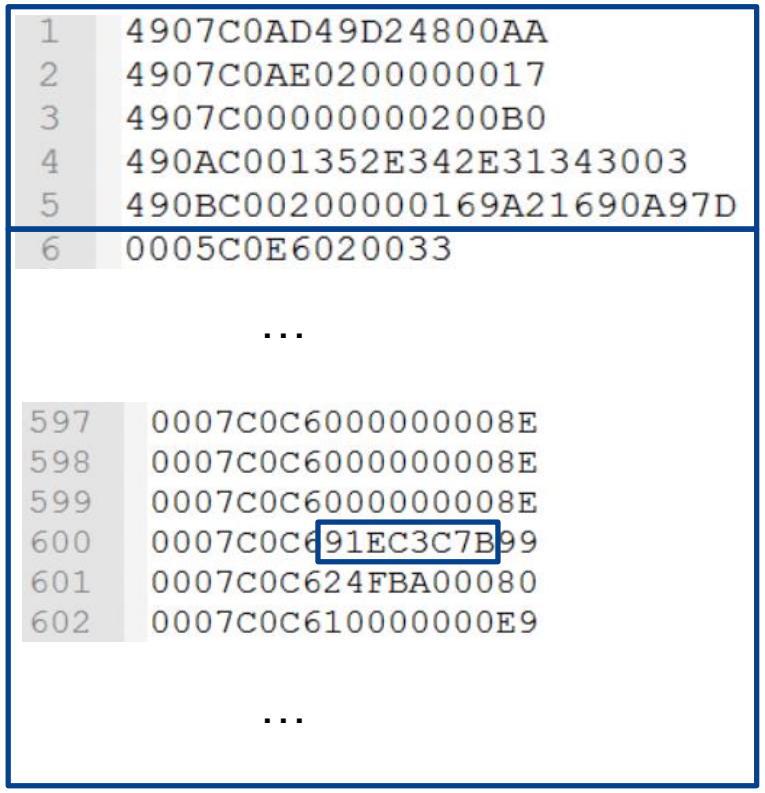
Slot ID line number = 282

The 10th character in the line is the Slot ID.

The example HEX file contains a configuration in **Slot ID 0**.

HEX Files – Configuration CRCs

CRC line number = (N ∗ 358) + 600, N = 0, 1, . . ., # of configurations in file – 1



In the example HEX file, there are 648 lines and 1 configuration.

N = 0, 1, . . ., # of configurations in file – 1

CRC line number = (N ∗ 358) + 600

For the 1st configuration in the file, N = 0.

CRC line number = (0 ∗ 358) + 600

CRC line number = 600

The 1st configuration CRC is **0x7B3CEC91**.

BIG IDEAS FOR EVERY SPACE

Renesas.com