

PSOC™ Control C3 (PSC3P2xx, PSC3M3xx, PSC3P5xx, PSC3M5xx)- Emulated EEPROM memory (Em_EEPROM)

Customer training workshop (CTW)

February 2026

public

Table of contents

1 Target products 3

2 EEPROM memory 5

3 Operating modes 8

4 Em_EEPROM storage variable location and size 10

Copyright © Infineon Technologies AG 2026. All rights reserved.

Table of contents

1 **Target products** **3**

2 EEPROM memory 5

3 Operating modes 8

4 Em_EEPROM storage variable location and size 10

Copyright © Infineon Technologies AG 2026. All rights reserved.

Target products

– Target product list for this training material

Family category	Series	Code flash memory size
PSOC™ control C3 entry line	PSC3P2xD	Up to 256 KB
PSOC™ control C3 entry line	PSC3M3xD	Up to 256 KB
PSOC™ control C3 main line	PSC3P5xD	Up to 256 KB
PSOC™ control C3 main line	PSC3M5xD	Up to 256 KB

Copyright © Infineon Technologies AG 2026. All rights reserved.

Table of contents

1 Target products 3

2 **EEPROM memory** **5**

3 Operating modes 8

4 Em_EEPROM storage variable location and size 10

Copyright © Infineon Technologies AG 2026. All rights reserved.

EEPROM memory

- The Emulated EEPROM middleware emulates an EEPROM device in the device's non-volatile memory
- The EEPROM middleware operates on top of the block storage library that abstracts memories by specifying how to perform memory operations such as read, write, erase, and provides auxiliary functions to retrieve erase and program size and checks whether an address is within the memory range
- Use the Emulated EEPROM to store nonvolatile data on a target device, with wear levelling and restoring corrupted data from a redundant copy
- **Features**
  - Nonvolatile storage similar to EEPROM
  - Easy-to-use read and write API
  - Optional wear levelling
  - Optional redundant data storage
  - Supports any storage which can be modelled as a block device, including internal NVM or external flash (for example, via QSPI)
  - Designed to be resilient to power failures
  - Designed to promote even wear of the storage (when wear levelling is enabled)

Copyright © Infineon Technologies AG 2026. All rights reserved.

EEPROM memory (contd.)

– Function

A type of non-volatile memory used to store small amounts of data that must be saved when power is removed.

Note: To ensure data integrity and prevent potential hard faults, do not attempt to read a sector that is actively being erased or written. To mitigate this, wrap EEPROM write/erase operations in a critical section by disabling interrupts or Place high-priority ISRs in SRAM so they can run while flash is busy.

– Common use

- Configuration settings: Storing user preferences and system configuration
- Calibration data: Keeping calibration constant for sensor and analog Components
- Firmware storage: Holding bootloaders or small firmware updates
- Data logging: Recording events, usage data, or logs that need to persist between power cycles

Hint bar

See the architecture reference manual, Flash geometry (section 5.2.2), for additional details.

Copyright © Infineon Technologies AG 2026. All rights reserved.

Table of contents

1 Target products 3

2 EEPROM memory 5

3 **Operating modes** **8**

4 Em_EEPROM storage variable location and size 10

Copyright © Infineon Technologies AG 2026. All rights reserved.

Operating modes

- **Simple mode**
  - Simple mode, when enabled (1 - enabled, 0 - disabled), means no additional service information is stored by the Em_EEPROM middleware, like checksums, headers, several writes, and so on
  - Data is stored directly by the specified address. The size of Em_EEPROM storage is equal to the number of bytes specified in the EEPROM Size parameter, rounded up to a full row size CY_EM_EEPROM_FLASH_SIZEOF_ROW
  - The wear levelling and redundant copy features are disabled, i.e., wearLevelingFactor and redundantCopy parameters are ignored
- **Redundant copy**
  - Depending on how critical it is for you to recover information, configure the redundant copy feature
  - If enabled (1 - enabled, 0 - disabled), a checksum (stored in a row) is calculated on each row of data, while a redundant copy of Em_EEPROM is stored in another location
  - When data is read, first the checksum is checked. If that checksum is bad, and the redundant copy's checksum is good, the copy is restored
- **Wear levelling**
  - Depending on whether you want to increase the flash memory endurance or not, enable or disable the wear levelling. The higher the value is, the more flash is used, but a higher number of erase/write cycles can be done on Em_EEPROM.
  - Multiply this number by the datasheet write endurance spec to determine the maximum number of write cycles
  - The amount of wear levelling from 1 to 10
    - 1 means no wear levelling is used

Copyright © Infineon Technologies AG 2026. All rights reserved.

Table of contents

1 Target products 3

2 EEPROM memory 5

3 Operating modes 8

4 **Em_EEPROM storage variable location and size** **10**

Copyright © Infineon Technologies AG 2026. All rights reserved.

Em_EEPROM storage variable location and size

- The user is responsible for allocating space in flash for Em_EEPROM (further the Em_EEPROM storage)
- For PSOC™ Control the Em_EEPROM storage can be placed
  - In the application flash
  - The storage can be placed at a fixed address in the application flash

- **Em_EEPROM location in the application flash**
  - The example below shows placing the Em_EEPROM storage in the application flash for the GCC compiler:

```
CY_ALIGN(CY_EM_EEPROM_FLASH_SIZEOF_ROW)
const uint8_t emEepromStorage[STORAGE_SIZE] = {0u};
```

where STORAGE_SIZE is the size of the Em_EEPROM storage. For convenience, use the CY_EM_EEPROM_GET_PHYSICAL_SIZE macro to get the needed Em_EEPROM storage size depending on the configuration.

Copyright © Infineon Technologies AG 2026. All rights reserved.

Em_EEPROM storage variable location and size (contd.)

- **Em_EEPROM Location in the application flash at a fixed address**
  - To allocate the Em_EEPROM storage at a fixed address in the application flash, modify the linker script
  - This approach demonstrates adding the storage reservation in the application flash after the application. The user calculates the application end address and selects the address of the Em_EEPROM storage so that the memory spaces of the storage and the application do not overlap
  - The user also adds some offset between the application end address and the Em_EEPROM storage start address to ensure there is extra space in case the project code grows

```
EM_EEPROM_START_ADDRESS = <EEPROM Storage Address>;
.my_emulated_eeprom EM_EEPROM_START_ADDRESS :
{
KEEP(*(.my_emulated_eeprom))
} > flash
```

where,

- EEPROM Storage Address is an absolute address in flash where the Em_EEPROM operates. Ensure the address is aligned to the size of the device's flash row
- my_emulated_eeprom is the name of the section where the Em_EEPROM storage will be placed

Copyright © Infineon Technologies AG 2026. All rights reserved.

Table of contents

1 Target products 3

2 EEPROM memory 5

3 Operating modes 8

4 Em_EEPROM storage variable location and size 10

Copyright © Infineon Technologies AG 2026. All rights reserved.

Revision history

Document revision	Date	Description of change
**	2026-02-13	Initial release

Copyright © Infineon Technologies AG 2026. All rights reserved.


Infineon