

## EEPROM LIBRARY

Generated by Doxygen 1.9.6



<b>1 EEPROM Library Integration Guide</b>	<b>1</b>
1.1 Step 1: Download the library files	1
1.2 Step 2: Add the library files to your project	1
1.2.1 STM32Cube IDE	1
1.2.2 Other IDEs	1
1.3 Step 3: Include the library header in main.c	2
1.4 Step 4: Usage Examples	2
1.4.1 Write data to EEPROM	2
1.4.2 Read data from EEPROM	2
1.4.3 Erase a page in EEPROM	2
1.4.4 Comprehensive example on how to use it in main.c	2
<b>2 File Index</b>	<b>5</b>
2.1 File List	5
<b>3 File Documentation</b>	<b>7</b>
3.1 EEPROM.c File Reference	7
3.1.1 Detailed Description	8
3.1.2 Function Documentation	8
3.1.2.1 EEPROM_PageErase()	8
3.1.2.2 EEPROM_Read()	8
3.1.2.3 EEPROM_Read_NUM()	9
3.1.2.4 EEPROM_Write()	9
3.1.2.5 EEPROM_Write_NUM()	10
3.2 EEPROM.h File Reference	10
3.2.1 Detailed Description	11
3.2.2 Function Documentation	11
3.2.2.1 EEPROM_PageErase()	11
3.2.2.2 EEPROM_Read()	12
3.2.2.3 EEPROM_Read_NUM()	13
3.2.2.4 EEPROM_Write()	13
3.2.2.5 EEPROM_Write_NUM()	14
3.3 EEPROM.h	14
<b>Index</b>	<b>17</b>



# Chapter 1

## EEPROM Library Integration Guide

This guide provides instructions on how to integrate the EEPROM library into your STM32 project using STM32Cube IDE or any other compatible IDE.

### 1.1 Step 1: Download the library files

Download the following EEPROM library files:

1. [EEPROM.h](#) - The header file containing the function prototypes and necessary definitions.
2. [EEPROM.c](#) - The source file containing the function implementations.

### 1.2 Step 2: Add the library files to your project

Follow these steps to add the EEPROM library files to your project:

#### 1.2.1 STM32Cube IDE

1. In STM32Cube IDE, open your STM32 project.
2. Navigate to the project tree in the "Project Explorer" tab.
3. Place the [EEPROM.h](#) file into the "Inc" folder (or the folder where header files are stored in your project).
4. Place the [EEPROM.c](#) file into the "Src" folder (or the folder where source files are stored in your project).

#### 1.2.2 Other IDEs

1. Open your STM32 project in the IDE you are using.
2. Place the [EEPROM.h](#) file in the folder where header files are stored in your project (usually an "include" or "inc" folder).
3. Place the [EEPROM.c](#) file in the folder where source files are stored in your project (usually a "source" or "src" folder).

## 1.3 Step 3: Include the library header in main.c

In the `main.c` file of your project, add the following include statement at the beginning of the file, along with other include statements:

```
#include "EEPROM.h"
```

## 1.4 Step 4: Usage Examples

Here are some examples of how to use the EEPROM library in your STM32 project:

### 1.4.1 Write data to EEPROM

```
uint8_t data_to_write[] = "Hello, EEPROM!";  
EEPROM_Write(0, 0, data_to_write, sizeof(data_to_write));
```

### 1.4.2 Read data from EEPROM

```
uint8_t read_buffer[16];  
EEPROM_Read(0, 0, read_buffer, sizeof(read_buffer));
```

### 1.4.3 Erase a page in EEPROM

```
EEPROM_PageErase(2);
```

### 1.4.4 Comprehensive example on how to use it in main.c

```
#include "main.h"  
#include "EEPROM.h"  
  
I2C_HandleTypeDef hi2c1;  
  
void SystemClock_Config(void);  
static void MX_GPIO_Init(void);  
static void MX_I2C1_Init(void);  
  
int main(void)  
{  
    HAL_Init();  
    SystemClock_Config();  
  
    MX_GPIO_Init();  
    MX_I2C1_Init();  
  
    // Write data to EEPROM  
    uint8_t data_to_write[] = "Hello, EEPROM!";  
    EEPROM_Write(0, 0, data_to_write, sizeof(data_to_write));  
  
    // Read data from EEPROM  
    uint8_t read_buffer[16];  
    EEPROM_Read(0, 0, read_buffer, sizeof(read_buffer));  
  
    // Write a float number to EEPROM  
    float number_to_write = 3.14159265;  
    EEPROM_Write_NUM(1, 0, number_to_write);  
  
    // Read a float number from EEPROM  
    float read_number;  
    read_number = EEPROM_Read_NUM(1, 0);  
  
    // Erase a page in EEPROM  
    EEPROM_PageErase(2);  
  
    while (1)  
    {  
        // Main loop  
    }  
}
```

```
}

void SystemClock_Config(void)
{
    // System clock configuration code...
}

static void MX_GPIO_Init(void)
{
    // GPIO initialization code...
}

static void MX_I2C1_Init(void)
{
    // I2C1 initialization code...
}
```





## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">EEPROM.c</a>	
Using the HAL I2C Functions . . . . .	<a href="#">7</a>
<a href="#">EEPROM.h</a>	
Using the HAL I2C Functions . . . . .	<a href="#">10</a>



## Chapter 3

# File Documentation

### 3.1 EEPROM.c File Reference

Using the HAL I2C Functions.

```
#include "EEPROM.h"
#include "math.h"
#include "string.h"
```

#### Macros

- `#define EEPROM_I2C` &hi2c1
- `#define EEPROM_ADDR` 0xA0
- `#define PAGE_SIZE` 64
- `#define PAGE_NUM` 512

#### Functions

- void [EEPROM\\_Write](#) (uint16\_t page, uint16\_t offset, uint8\_t \*data, uint16\_t size)  
*Write data to the EEPROM.*
- void [EEPROM\\_Write\\_NUM](#) (uint16\_t page, uint16\_t offset, float data)  
*Write a float/integer value to the EEPROM.*
- float [EEPROM\\_Read\\_NUM](#) (uint16\_t page, uint16\_t offset)  
*Read a single float/integer value from the EEPROM.*
- void [EEPROM\\_Read](#) (uint16\_t page, uint16\_t offset, uint8\_t \*data, uint16\_t size)  
*Read data from the EEPROM.*
- void [EEPROM\\_PageErase](#) (uint16\_t page)  
*Erase a page in the EEPROM Memory.*

#### Variables

- I2C\_HandleTypeDef **hi2c1**

### 3.1.1 Detailed Description

Using the HAL I2C Functions.

#### Author

ControllersTech

#### Date

Feb 16, 2021

#### Attention

Copyright (C) 2017 ControllersTech.com

This is a free software under the GNU license, you can redistribute it and/or modify it under the terms of the GNU General Public License version 3 as published by the Free Software Foundation. This software library is shared with public for educational purposes, without WARRANTY and Author is not liable for any damages caused directly or indirectly by this software, read more about this on the GNU General Public License.

### 3.1.2 Function Documentation

#### 3.1.2.1 EEPROM\_PageErase()

```
void EEPROM_PageErase (
    uint16_t page )
```

Erase a page in the EEPROM Memory.

#### Parameters

<i>page</i>	Number of page to erase In order to erase multiple pages, just use this function in the for loop
-------------	--

#### Return values

<i>None</i>	
-------------	--

#### 3.1.2.2 EEPROM\_Read()

```
void EEPROM_Read (
    uint16_t page,
```

```
uint16_t offset,  
uint8_t * data,  
uint16_t size )
```

Read data from the EEPROM.

#### Parameters

<i>page</i>	Number of the start page. Range from 0 to PAGE_NUM-1.
<i>offset</i>	Start byte offset in the page. Range from 0 to PAGE_SIZE-1.
<i>data</i>	Pointer to the data to write in bytes.
<i>size</i>	Size of the data.

#### Return values

<i>None.</i>	
--------------	--

### 3.1.2.3 EEPROM\_Read\_NUM()

```
float EEPROM_Read_NUM (  
    uint16_t page,  
    uint16_t offset )
```

Read a single float/integer value from the EEPROM.

Read a float or integer value from the EEPROM.

#### Parameters

<i>page</i>	Number of the start page. Range from 0 to PAGE_NUM-1.
<i>offset</i>	Start byte offset in the page. Range from 0 to PAGE_SIZE-1.

#### Return values

<i>Float/integer</i>	value.
----------------------	--------

### 3.1.2.4 EEPROM\_Write()

```
void EEPROM_Write (  
    uint16_t page,  
    uint16_t offset,  
    uint8_t * data,  
    uint16_t size )
```

Write data to the EEPROM.

**Parameters**

<i>page</i>	Start page number (0 to PAGE_NUM-1).
<i>offset</i>	Start byte offset in the page (0 to PAGE_SIZE-1).
<i>data</i>	Pointer to the data to write in bytes.
<i>size</i>	Size of the data.

**Return values**

<i>None</i>	
-------------	--

**3.1.2.5 EEPROM\_Write\_NUM()**

```
void EEPROM_Write_NUM (
    uint16_t page,
    uint16_t offset,
    float data )
```

Write a float/integer value to the EEPROM.

Write a float or integer value to the EEPROM.

**Parameters**

<i>page</i>	Number of the start page. Range from 0 to PAGE_NUM-1.
<i>offset</i>	Start byte offset in the page. Range from 0 to PAGE_SIZE-1.
<i>data</i>	Float/integer value that you want to write.

**Return values**

<i>None.</i>	
--------------	--

**3.2 EEPROM.h File Reference**

Using the HAL I2C Functions.

```
#include "stdint.h"
#include "stm32f4xx_hal.h"
```

**Functions**

- void [EEPROM\\_Write](#) (uint16\_t page, uint16\_t offset, uint8\_t \*data, uint16\_t size)  
*Write data to the EEPROM.*

- void [EEPROM\\_Read](#) (uint16\_t page, uint16\_t offset, uint8\_t \*data, uint16\_t size)  
*Read data from the EEPROM.*
- void [EEPROM\\_PageErase](#) (uint16\_t page)  
*Erase a page in the EEPROM Memory.*
- void [EEPROM\\_Write\\_NUM](#) (uint16\_t page, uint16\_t offset, float fdata)  
*Write a float or integer value to the EEPROM.*
- float [EEPROM\\_Read\\_NUM](#) (uint16\_t page, uint16\_t offset)  
*Read a float or integer value from the EEPROM.*

### 3.2.1 Detailed Description

Using the HAL I2C Functions.

#### Author

ControllersTech

#### Date

Feb 16, 2021

#### Attention

Copyright (C) 2017 ControllersTech.com

This is a free software under the GNU license, you can redistribute it and/or modify it under the terms of the GNU General Public License version 3 as published by the Free Software Foundation. This software library is shared with public for educational purposes, without WARRANTY and Author is not liable for any damages caused directly or indirectly by this software, read more about this on the GNU General Public License.

### 3.2.2 Function Documentation

#### 3.2.2.1 EEPROM\_PageErase()

```
void EEPROM_PageErase (  
    uint16_t page )
```

Erase a page in the EEPROM Memory.

#### Parameters

<i>page</i>	Page number to erase.
-------------	-----------------------

## Return values

<i>None</i>	
-------------	--

## Parameters

<i>page</i>	Number of page to erase In order to erase multiple pages, just use this function in the for loop
-------------	--

## Return values

<i>None</i>	
-------------	--

### 3.2.2.2 EEPROM\_Read()

```
void EEPROM_Read (
    uint16_t page,
    uint16_t offset,
    uint8_t * data,
    uint16_t size )
```

Read data from the EEPROM.

## Parameters

<i>page</i>	Start page number (0 to PAGE_NUM-1).
<i>offset</i>	Start byte offset in the page (0 to PAGE_SIZE-1).
<i>data</i>	Pointer to the data to read in bytes.
<i>size</i>	Size of the data.

## Return values

<i>None</i>	
-------------	--

## Parameters

<i>page</i>	Number of the start page. Range from 0 to PAGE_NUM-1.
<i>offset</i>	Start byte offset in the page. Range from 0 to PAGE_SIZE-1.
<i>data</i>	Pointer to the data to write in bytes.
<i>size</i>	Size of the data.

## Return values

<i>None.</i>	
--------------	--



### 3.2.2.3 EEPROM\_Read\_NUM()

```
float EEPROM_Read_NUM (
    uint16_t page,
    uint16_t offset )
```

Read a float or integer value from the EEPROM.

#### Parameters

<i>page</i>	Start page number (0 to PAGE_NUM-1).
<i>offset</i>	Start byte offset in the page (0 to PAGE_SIZE-1).

#### Return values

<i>Float</i>	or integer value read from the EEPROM.
--------------	--

Read a float or integer value from the EEPROM.

#### Parameters

<i>page</i>	Number of the start page. Range from 0 to PAGE_NUM-1.
<i>offset</i>	Start byte offset in the page. Range from 0 to PAGE_SIZE-1.

#### Return values

<i>Float/integer</i>	value.
----------------------	--------

### 3.2.2.4 EEPROM\_Write()

```
void EEPROM_Write (
    uint16_t page,
    uint16_t offset,
    uint8_t * data,
    uint16_t size )
```

Write data to the EEPROM.

#### Parameters

<i>page</i>	Start page number (0 to PAGE_NUM-1).
<i>offset</i>	Start byte offset in the page (0 to PAGE_SIZE-1).
<i>data</i>	Pointer to the data to write in bytes.
<i>size</i>	Size of the data.

## Return values

<i>None</i>	
-------------	--

### 3.2.2.5 EEPROM\_Write\_NUM()

```
void EEPROM_Write_NUM (
    uint16_t page,
    uint16_t offset,
    float data )
```

Write a float or integer value to the EEPROM.

## Parameters

<i>page</i>	Start page number (0 to PAGE_NUM-1).
<i>offset</i>	Start byte offset in the page (0 to PAGE_SIZE-1).
<i>fdata</i>	Float or integer value to write.

## Return values

<i>None</i>	
-------------	--

Write a float or integer value to the EEPROM.

## Parameters

<i>page</i>	Number of the start page. Range from 0 to PAGE_NUM-1.
<i>offset</i>	Start byte offset in the page. Range from 0 to PAGE_SIZE-1.
<i>data</i>	Float/integer value that you want to write.

## Return values

<i>None.</i>	
--------------	--

## 3.3 EEPROM.h

[Go to the documentation of this file.](#)

```
00001
00019 #ifndef INC_EEPROM_H_
00020 #define INC_EEPROM_H_
00021
00022 #include "stdint.h"
00023 #include "stm32f4xx_hal.h"
00024
00033 void EEPROM_Write(uint16_t page, uint16_t offset, uint8_t *data, uint16_t size);
00034
00043 void EEPROM_Read(uint16_t page, uint16_t offset, uint8_t *data, uint16_t size);
```

```
00044
00050 void EEPROM_PageErase(uint16_t page);
00051
00059 void EEPROM_Write_NUM(uint16_t page, uint16_t offset, float fdata);
00060
00067 float EEPROM_Read_NUM(uint16_t page, uint16_t offset);
00068
00069 #endif /* INC_EEPROM_H_ */
```



# Index

- EEPROM.c, [7](#)
  - EEPROM\_PageErase, [8](#)
  - EEPROM\_Read, [8](#)
  - EEPROM\_Read\_NUM, [9](#)
  - EEPROM\_Write, [9](#)
  - EEPROM\_Write\_NUM, [10](#)
- EEPROM.h, [10](#)
  - EEPROM\_PageErase, [11](#)
  - EEPROM\_Read, [12](#)
  - EEPROM\_Read\_NUM, [12](#)
  - EEPROM\_Write, [13](#)
  - EEPROM\_Write\_NUM, [14](#)
- EEPROM\_PageErase
  - EEPROM.c, [8](#)
  - EEPROM.h, [11](#)
- EEPROM\_Read
  - EEPROM.c, [8](#)
  - EEPROM.h, [12](#)
- EEPROM\_Read\_NUM
  - EEPROM.c, [9](#)
  - EEPROM.h, [12](#)
- EEPROM\_Write
  - EEPROM.c, [9](#)
  - EEPROM.h, [13](#)
- EEPROM\_Write\_NUM
  - EEPROM.c, [10](#)
  - EEPROM.h, [14](#)