EEPROM Library

Documentation for usage of the EEPROM library. EEPROM is a memory whose values are kept when the board is powered off.

Author • Arduino

Last revision • 02/02/2024

memory whose values are kept when the board is turned off (like a tiny hard drive). This library enables you to read and write those bytes.

The supported micro-controllers on the various Arduino and Genuino boards have different amounts of EEPROM: 1024 bytes on the ATmega328P, 512 bytes on the ATmega168 and ATmega8, 4 KB (4096 bytes) on the ATmega1280 and ATmega2560. The Arduino and Genuino 101 boards have an emulated EEPROM space of 1024 bytes.

To use this library

COPY

```
#include <EEPROM.h>
```

Examples

To see a list of examples for the EEPROM library, click the link below:

Functions

read()

Description

Reads a byte from the EEPROM. Locations that have never been written to have the value of 255.

Syntax

COPY

```
EEPROM.read(address)
```

Parameters

address: the location to read from, starting from 0 (int)

the value stored in that location (byte)

Example

```
#include <EEPROM.h>

int a = 0;
int value;

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  value = EEPROM.read(a);

  Serial.print(a);
  Serial.print("\t");
  Serial.print(value);
  Serial.println();

  a = a + 1;

  if (a == 512)
    a = 0;

  delay(500);
}
```

write()

Write a byte to the EEPROM.

Syntax

COPY

```
EEPROM.write(address, value)
```

Parameters

address: the location to write to, starting from 0 (int)

value: the value to write, from 0 to 255 (byte)

Returns

none

Note: An EEPROM write takes 3.3 ms to complete. The EEPROM memory has a specified life of 100,000 write/erase cycles, so you may

Example

COPY

```
#include <EEPROM.h>

void setup()
{
  for (int i = 0; i < 255; i++)
    EEPROM.write(i, i);
}

void loop()
{
}
```

update()

Description

Write a byte to the EEPROM. The value is written only if differs from the one already saved at the same address.

COPY

```
EEPROM.update(address, value)
```

Parameters

address: the location to write to, starting from 0 (int)

value: the value to write, from 0 to 255 (byte)

Returns

none

> *Note: An EEPROM write takes 3.3 ms to complete. The EEPROM memory has a specified life of 100,000 write/erase cycles, so using this function instead of write() can save cycles if the written data does not change often*

Example

```
#include <EEPROM.h>

void setup()
{
  for (int i = 0; i < 255; i++) {
    // this performs as EEPROM.wr
    EEPROM.update(i, i);
  }
  for (int i = 0; i < 255; i++) {
    // write value "12" to cell 3
    // will not write the cell th
    EEPROM.update(3, 12);
  }
}

void loop()
{
}
```

get()

Description

Read any data type or object from the EEPROM.

```
EEPROM.get(address, data)
```

Parameters

address: the location to read from, starting from 0 (int)

data: the data to read, can be a primitive type (eg. float) or a custom struct

Returns

A reference to the data passed in

Example

```
#include <EEPROM.h>

struct MyObject{
  float field1;
  byte field2;
  char name[10];
};

void setup(){

  float f = 0.00f;   //Variable t
  int eeAddress = 0; //EEPROM add

  Serial.begin( 9600 );
  while (!Serial) {
    ; // wait for serial port to
  }
  Serial.print( "Read float from

  //Get the float data from the E
  EEPROM.get( eeAddress, f );
  Serial.println( f, 3 );  //This

  // get() can be used with custo
  eeAddress = sizeof(float); //Mo
  MyObject customVar; //Variable
  EEPROM.get( eeAddress, customVa

  Serial.println( "Read custom ob
```

Description

Write any data type or object to the EEPROM.

Syntax

COPY

```
EEPROM.put(address, data)
```

Parameters

address: the location to write to, starting from 0 (int)

data: the data to write, can be a primitive type (eg. float) or a custom struct

Returns

A reference to the data passed in

> *value if it didn't change.*

Example

```
#include <EEPROM.h>

struct MyObject {
  float field1;
  byte field2;
  char name[10];
};

void setup() {

  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to
  }

  float f = 123.456f;  //Variable
  int eeAddress = 0;   //Location


  //One simple call, with the add
  EEPROM.put(eeAddress, f);

  Serial.println("Written float d

  /** Put is designed for use wit

  //Data to store.
  MyObject customVar = {
    3.14f,
```

Description

This operator allows using the identifier `EEPROM` like an array. EEPROM cells can be read and written directly using this method.

Syntax

COPY

```
EEPROM[address]
```

Parameters

address: the location to read/write from, starting from 0 (int)

Returns

A reference to the EEPROM cell

```
#include <EEPROM.h>

void setup(){

  unsigned char val;

  //Read first EEPROM cell.
  val = EEPROM[ 0 ];

  //Write first EEPROM cell.
  EEPROM[ 0 ] = val;

  //Compare contents
  if( val == EEPROM[ 0 ] ){
    //Do something...
  }
}

void loop(){ /* Empty loop */ }
```

length()

This function returns an unsigned int containing the number of cells in the EEPROM.

This function returns an `unsigned int` containing the number of cells in the EEPROM.

Syntax

COPY

```
EEPROM.length()
```

Returns

Number of cells in the EEPROM as an `unsigned int` .
