Arduino Driver - Memory

Generated by Doxygen 1.8.9.1

Sat Aug 15 2015 15:34:14

Contents

1 Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

WiredDevice

- **ExternalEeprom** **9**
  - **External24cl256Eeprom** **2**
  - **External24x16Eeprom** **4**
  - **ExternalByteArrayEeprom** **6**
  - **ExternalMappedEeprom** **13**

2 Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

3 File Index

3.1 File List

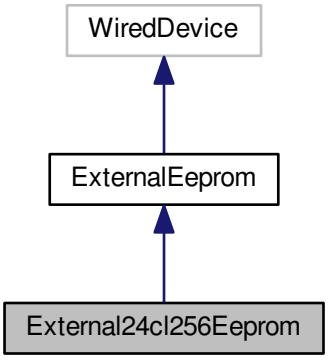
Here is a list of all files with brief descriptions:

4 Class Documentation

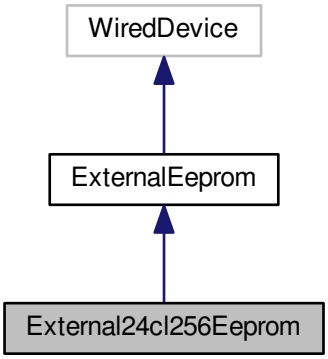
4.1 External24cl256Eeprom Class Reference

```
#include <External24cl256Eeprom.h>
```

Inheritance diagram for External24cl256Eeprom:



Collaboration diagram for External24cl256Eeprom:



Public Member Functions

- External24cl256Eeprom (unsigned char deviceAddress)

Protected Member Functions

- virtual void writeBlock (unsigned int address, unsigned char ∗buf, int len)
- virtual void readBlock (unsigned int address, unsigned char ∗buf, int len)

4.1.1 Detailed Description

Arduino - External 24cl256 eeprom.

External24cl256Eeprom.h

This an implementation of 24cl256 eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 16 of file External24cl256Eeprom.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 External24cl256Eeprom::External24cl256Eeprom (unsigned char *deviceAddress*)

Public constructor.

Parameters

deviceAddress	The i2c addredd of the device.

Definition at line 18 of file External24cl256Eeprom.cpp.

4.1.3 Member Function Documentation

4.1.3.1 void External24cl256Eeprom::readBlock (unsigned int *address*, unsigned char * *buf*, int *len*) [protected], [virtual]

Reads a block of bytes from the device.

Parameters

address	
buf	
len	

Implements ExternalEeprom.

Definition at line 35 of file External24cl256Eeprom.cpp.

4.1.3.2 void External24cl256Eeprom::writeBlock (unsigned int *address*, unsigned char * *buf*, int *len*) [protected], [virtual]

Writes a block of bytes separately by pages to the device.

All bytes during a page write operation must reside on the same page.

Parameters

address	
buf	
len	

Implements ExternalEeprom.

Definition at line 23 of file External24cl256Eeprom.cpp.

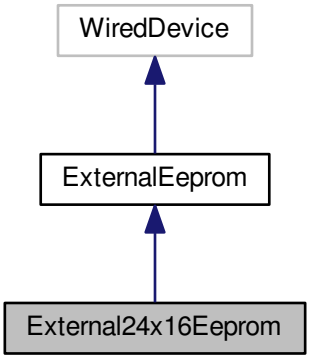
The documentation for this class was generated from the following files:

- External24cl256Eeprom.h
- External24cl256Eeprom.cpp

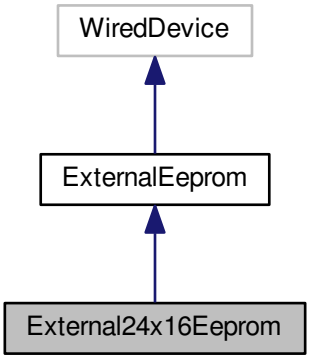
4.2 External24x16Eeprom Class Reference

`#include <External24x16Eeprom.h>`

Inheritance diagram for External24x16Eeprom:



Collaboration diagram for External24x16Eeprom:



Public Member Functions

- External24x16Eeprom (unsigned char deviceAddress)

Protected Member Functions

- virtual void writeBlock (unsigned int address, unsigned char ∗buf, int len)
- virtual void readBlock (unsigned int address, unsigned char ∗buf, int len)

4.2.1 Detailed Description

Arduino - External 24x16 eeprom.

External24x16Eeprom.h

This an implementation of 24X16 eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 16 of file External24x16Eeprom.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 External24x16Eeprom::External24x16Eeprom (unsigned char *deviceAddress*)

Public constructor.

Parameters

device	The i2c address of the device.

Definition at line 18 of file External24x16Eeprom.cpp.

4.2.3 Member Function Documentation

4.2.3.1 void External24x16Eeprom::readBlock (unsigned int *address*, unsigned char * *buf*, int *len*) [protected], [virtual]

Reads a block of bytes from the device.

Parameters

address	
buf	
len	

Implements ExternalEeprom.

Definition at line 36 of file External24x16Eeprom.cpp.

4.2.3.2 void External24x16Eeprom::writeBlock (unsigned int *address*, unsigned char * *buf*, int *len*) [protected], [virtual]

Writes a block of bytes separately by pages to the device.

All bytes during a page write operation must reside on the same page.

Parameters

address	
buf	
len	

Implements ExternalEeprom.

Definition at line 23 of file External24x16Eeprom.cpp.

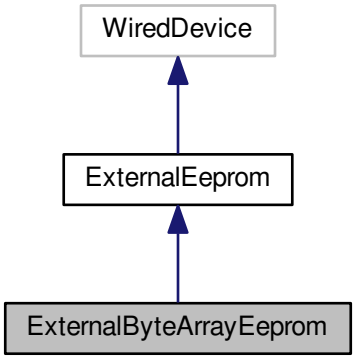
The documentation for this class was generated from the following files:

- External24x16Eeprom.h
- External24x16Eeprom.cpp

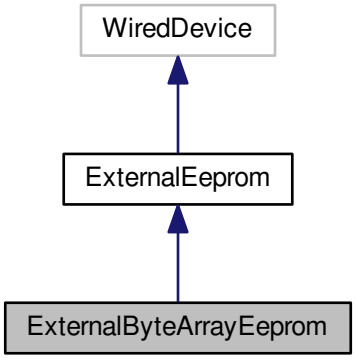
4.3 ExternalByteArrayEeprom Class Reference

```
#include <ExternalByteArrayEeprom.h>
```

Inheritance diagram for ExternalByteArrayEeprom:



Collaboration diagram for ExternalByteArrayEeprom:



Public Member Functions

- ExternalByteArrayEeprom (unsigned char *byteArray, unsigned int deviceSize)

Protected Member Functions

- virtual void writeBlock (unsigned int address, unsigned char *buf, int len)
- virtual void readBlock (unsigned int address, unsigned char *buf, int len)

Private Attributes

- unsigned char * byteArray

4.3.1 Detailed Description

Arduino - External Virtual eeprom.

ExternalByteArrayEeprom.h

This an implementation of VIRTUAL eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 16 of file ExternalByteArrayEeprom.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 ExternalByteArrayEeprom::ExternalByteArrayEeprom (unsigned char * *byteArray,* unsigned int *deviceSize*)

Public constructor.

Parameters

device	

Definition at line 16 of file ExternalByteArrayEeprom.cpp.

4.3.3 Member Function Documentation

4.3.3.1 void ExternalByteArrayEeprom::readBlock (unsigned int *address,* unsigned char * *buf,* int *len*) [protected], [virtual]

Reads a block of bytes from the device.

Parameters

address	
buf	
len	

Implements ExternalEeprom.

Definition at line 26 of file ExternalByteArrayEeprom.cpp.

4.3.3.2 void ExternalByteArrayEeprom::writeBlock (unsigned int *address,* unsigned char * *buf,* int *len*) [protected], [virtual]

Writes a block of bytes separately by pages to the device.

All bytes during a page write operation must reside on the same page.

Parameters

address	
buffer	
len	

Implements ExternalEeprom.

Definition at line 20 of file ExternalByteArrayEeprom.cpp.

4.3.4 Member Data Documentation

4.3.4.1 unsigned char* ExternalByteArrayEeprom::byteArray [private]

The used buffer.

Definition at line 21 of file ExternalByteArrayEeprom.h.

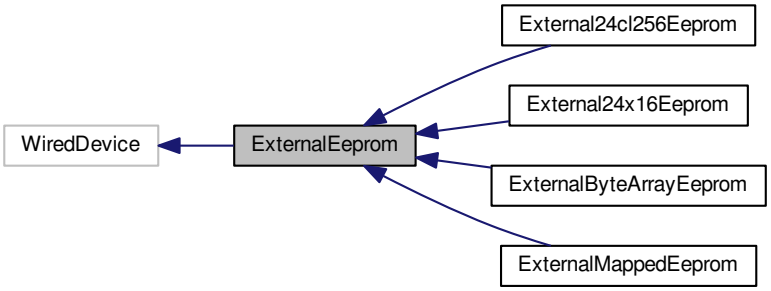
The documentation for this class was generated from the following files:

- ExternalByteArrayEeprom.h
- ExternalByteArrayEeprom.cpp

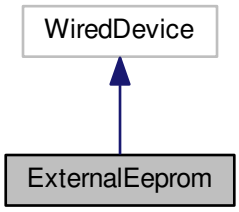
4.4 ExternalEeprom Class Reference

`#include <ExternalEeprom.h>`

Inheritance diagram for ExternalEeprom:



Collaboration diagram for ExternalEeprom:



Public Member Functions

- virtual void write (unsigned int address, unsigned char b)
- virtual void writeBytes (unsigned int address, unsigned char *buf, int len)
- virtual int read (unsigned int address)
- virtual int readBytes (unsigned int address, unsigned char *buf, int len)
- virtual int setBytes (unsigned int address, unsigned char b, int len)

- int getPageSize ()
- unsigned int getDeviceSize ()
- virtual void writeBlock (unsigned int address, unsigned char *buf, int len)=0
- virtual void readBlock (unsigned int address, unsigned char *buf, int len)=0

Protected Member Functions

- ExternalEeprom (unsigned char deviceAddress, int pageSize, unsigned int deviceSize)
- unsigned int endOfPage (unsigned int address)

Private Attributes

- unsigned int deviceSize
- int pageSize

4.4.1 Detailed Description

Arduino - External eeprom.

ExternalEeprom.h

This is an abstract class of external eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 16 of file ExternalEeprom.h.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 ExternalEeprom::ExternalEeprom (unsigned char *deviceAddress*, int *pageSize*, unsigned int *deviceSize*) [protected]

Protected constructor.

Parameters

device	

Definition at line 18 of file ExternalEeprom.cpp.

4.4.3 Member Function Documentation

4.4.3.1 unsigned int ExternalEeprom::endOfPage (unsigned int *address*) [protected]

Determines the length until first multiple of 'pageSize' of an address so writing always occurs up to 'pageSize' unsigned char boundaries.

Parameters

address	

Returns

Definition at line 110 of file ExternalEeprom.cpp.

4.4.3.2 unsigned int ExternalEeprom::getDeviceSize ( ) [inline]

Gets the total size of the device.

Returns

Definition at line 86 of file ExternalEeprom.h.

4.4.3.3 int ExternalEeprom::getPageSize ( ) [inline]

Gets the page size of the device.

Returns

Definition at line 77 of file ExternalEeprom.h.

4.4.3.4 int ExternalEeprom::read (unsigned int *address*) [virtual]

Reads a unsigned char from the device.

Parameters

address	

Returns

Definition at line 51 of file ExternalEeprom.cpp.

4.4.3.5 virtual void ExternalEeprom::readBlock (unsigned int *address*, unsigned char * *buf*, int *len*) [pure virtual]

Reads a block of bytes from the device.

Parameters

address	
buffer	
len	

Implemented in ExternalMappedEeprom, ExternalByteArrayEeprom, External24cl256Eeprom, and External24x16Eeprom.

4.4.3.6 int ExternalEeprom::readBytes (unsigned int *address*, unsigned char * *buf*, int *len*) [virtual]

Reads a buffer with len bytes from the device.

Parameters

address	
buf	
len	

Definition at line 59 of file ExternalEeprom.cpp.

4.4.3.7 int ExternalEeprom::setBytes (unsigned int *address*, unsigned char *b*, int *len*) [virtual]

Writes len bytes at the address with data.

Parameters

address	
data	
len	

Definition at line 80 of file ExternalEeprom.cpp.

4.4.3.8 void ExternalEeprom::write (unsigned int *address*, unsigned char *b*) [virtual]

Writes a unsigned char at the address into the device.

Parameters

address	The address where the data will be written.
b	The data to be written.

Definition at line 22 of file ExternalEeprom.cpp.

4.4.3.9 virtual void ExternalEeprom::writeBlock (unsigned int *address*, unsigned char * *buf*, int *len*) [pure virtual]

Writes a block of bytes separately by pages to the device.

All bytes during a page write operation must reside on the same page.

Parameters

address	
buf	
len	

Implemented in ExternalMappedEeprom, ExternalByteArrayEeprom, External24cl256Eeprom, and External24x16Eeprom.

4.4.3.10 void ExternalEeprom::writeBytes (unsigned int *address*, unsigned char * *buf*, int *len*) [virtual]

Writes a buffer of bytes at the address into the device.

Parameters

address	
buffer	
len	

Definition at line 26 of file ExternalEeprom.cpp.

4.4.4 Member Data Documentation

4.4.4.1 unsigned int ExternalEeprom::deviceSize [private]

The size of the device.

Definition at line 21 of file ExternalEeprom.h.

4.4.4.2 int ExternalEeprom::pageSize [private]

The size of the device page.

Definition at line 26 of file ExternalEeprom.h.

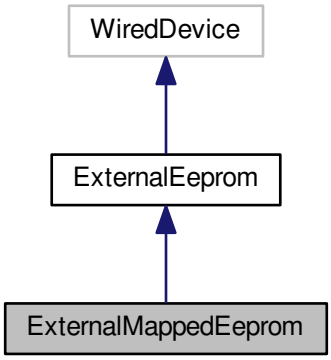
The documentation for this class was generated from the following files:

- ExternalEeprom.h
- ExternalEeprom.cpp

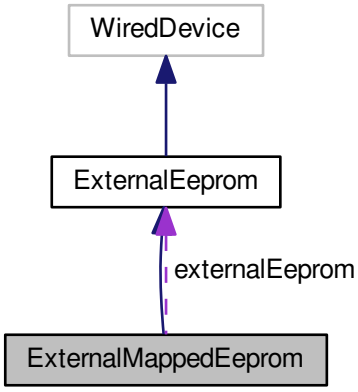
4.5 ExternalMappedEeprom Class Reference

`#include <ExternalMappedEeprom.h>`

Inheritance diagram for ExternalMappedEeprom:



Collaboration diagram for ExternalMappedEeprom:



Public Member Functions

- ExternalMappedEeprom (ExternalEeprom *externalEeprom, unsigned int startAddress, unsigned int endAddress)
- unsigned int getDeviceSize ()

Protected Member Functions

- virtual void writeBlock (unsigned int address, unsigned char *buf, int len)

- virtual void readBlock (unsigned int address, unsigned char *buf, int len)

Private Attributes

- ExternalEeprom * externalEeprom
- unsigned int startAddress
- unsigned int endAddress

4.5.1 Detailed Description

Arduino - External eeprom.

ExternalMappedEeprom.h

This is an abstract class of external eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 16 of file ExternalMappedEeprom.h.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 ExternalMappedEeprom::ExternalMappedEeprom (ExternalEeprom * *externalEeprom,* unsigned int *startAddress,* unsigned int *endAddress*)

Public constructor.

Parameters

externalEeprom	
startAddress	
endAddress	

Definition at line 16 of file ExternalMappedEeprom.cpp.

4.5.3 Member Function Documentation

4.5.3.1 unsigned int ExternalMappedEeprom::getDeviceSize ( ) [inline]

Device size.

Returns

Definition at line 48 of file ExternalMappedEeprom.h.

4.5.3.2 void ExternalMappedEeprom::readBlock (unsigned int *address,* unsigned char * *buf,* int *len*) [protected], [virtual]

Reads a block of bytes from the device.

Parameters

address	
buffer	
len	

Implements ExternalEeprom.

Definition at line 31 of file ExternalMappedEeprom.cpp.

4.5.3.3 void ExternalMappedEeprom::writeBlock (unsigned int *address*, unsigned char * *buf*, int *len*) [protected], [virtual]

Writes a block of bytes separately by pages to the device.

All bytes during a page write operation must reside on the same page.

Parameters

address	
buf	
len	

Implements ExternalEeprom.

Definition at line 22 of file ExternalMappedEeprom.cpp.

4.5.4 Member Data Documentation

4.5.4.1 unsigned int ExternalMappedEeprom::endAddress [private]

The end address of the map.

Definition at line 31 of file ExternalMappedEeprom.h.

4.5.4.2 ExternalEeprom* ExternalMappedEeprom::externalEeprom [private]

The underlying external eeprom.

Definition at line 21 of file ExternalMappedEeprom.h.

4.5.4.3 unsigned int ExternalMappedEeprom::startAddress [private]

The start address of the map.

Definition at line 26 of file ExternalMappedEeprom.h.

The documentation for this class was generated from the following files:

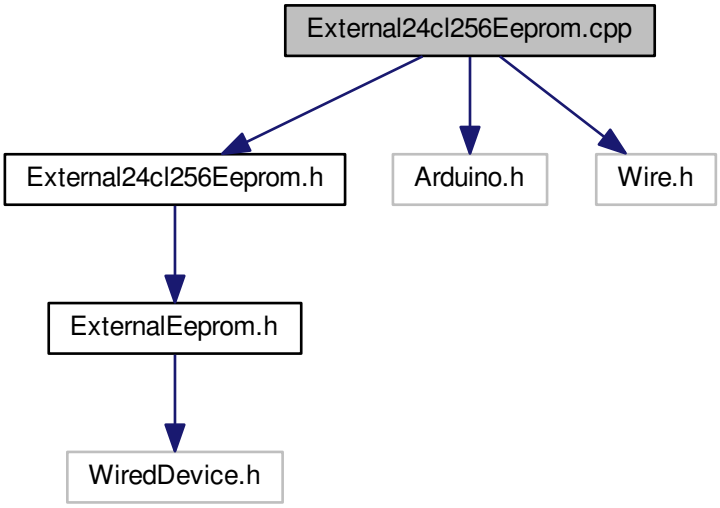
- ExternalMappedEeprom.h
- ExternalMappedEeprom.cpp

5 File Documentation

5.1 External24cl256Eeprom.cpp File Reference

```
#include "External24cl256Eeprom.h"
#include <Arduino.h>
#include <Wire.h>
```

Include dependency graph for External24cl256Eeprom.cpp:



Macros

- #define __ARDUINO_EXTERNAL_24CL256_EEPROM_CPP__ 1

5.1.1 Macro Definition Documentation

5.1.1.1 #define __ARDUINO_EXTERNAL_24CL256_EEPROM_CPP__ 1

Arduino - External 24cl256 eeprom.

External24cl256Eeprom.cpp

This an implementation of 24cl256 eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file External24cl256Eeprom.cpp.

5.2 External24cl256Eeprom.cpp

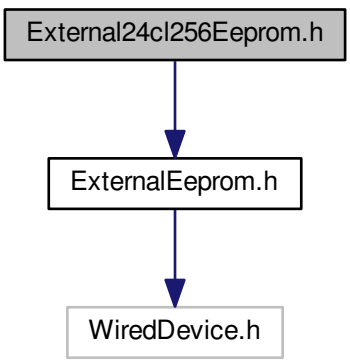
```
00001
00011 #ifndef __ARDUINO_EXTERNAL_24CL256_EEPROM_CPP__
00012 #define __ARDUINO_EXTERNAL_24CL256_EEPROM_CPP__ 1
00013
00014 #include "External24cl256Eeprom.h"
00015 #include <Arduino.h>
00016 #include <Wire.h>
00017
00018 External24cl256Eeprom::External24cl256Eeprom(unsigned char
      deviceAddress) :
00019         ExternalEeprom(deviceAddress, 0x40, 0x7fff) {
00020     Wire.begin();
00021 }
00022
```

```
00023 void External24cl256Eeprom::writeBlock(unsigned int address, unsigned char
      * buf,
00024         int len) {
00025     Wire.beginTransmission(getDeviceAddress());
00026     Wire.write((unsigned char) (address >> 8) & 0xff);
00027     Wire.write((unsigned char) (address & 0xff));
00028     for (int i = 0; i < len; i++) {
00029         Wire.write(buf[i]);
00030     }
00031     Wire.endTransmission();
00032     delay(5);
00033 }
00034
00035 void External24cl256Eeprom::readBlock(unsigned int address, unsigned char*
      buf,
00036         int len) {
00037     Wire.beginTransmission(getDeviceAddress());
00038     Wire.write((unsigned char) (address >> 8) & 0xff);
00039     Wire.write((unsigned char) (address & 0xff));
00040     Wire.endTransmission();
00041     Wire.requestFrom((int) getDeviceAddress(), len);
00042     for (int i = 0; i < len; i++) {
00043         while (!Wire.available())
00044             ;
00045         buf[i] = Wire.read();
00046     }
00047 }
00048
00049 #endif /* __ARDUINO_EXTERNAL_24CL256_EEPROM_CPP__ */
```

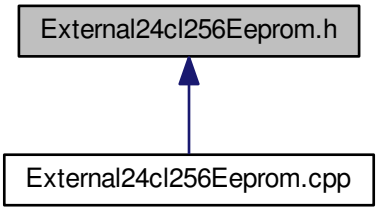
5.3 External24cl256Eeprom.h File Reference

`#include <ExternalEeprom.h>`

Include dependency graph for External24cl256Eeprom.h:



This graph shows which files directly or indirectly include this file:



Classes

- class External24cl256Eeprom

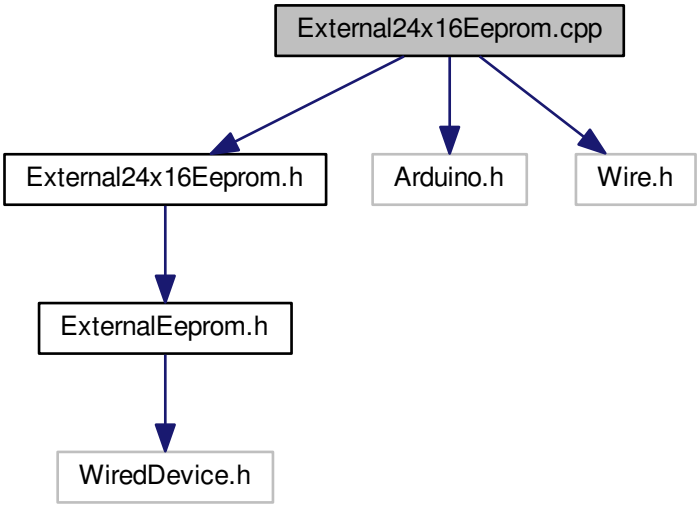
5.4 External24cl256Eeprom.h

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_24CL256_EEPROM_H__
00012 #define __ARDUINO_EXTERNAL_24CL256_EEPROM_H__ 1
00013
00014 #include <ExternalEeprom.h>
00015
00016 class External24cl256Eeprom : public ExternalEeprom {
00017 public:
00018
00024     External24cl256Eeprom(unsigned char deviceAddress);
00025
00026 protected:
00027
00036     virtual void writeBlock(unsigned int address, unsigned char* buf, int len);
00037
00045     virtual void readBlock(unsigned int address, unsigned char* buf, int len);
00046 };
00047
00048 #endif /* __ARDUINO_EXTERNAL_24CL256_EEPROM_H__ */
```

5.5 External24x16Eeprom.cpp File Reference

```
#include "External24x16Eeprom.h"
#include <Arduino.h>
#include <Wire.h>
```

Include dependency graph for External24x16Eeprom.cpp:



Macros

- #define __ARDUINO_EXTERNAL_24X16_EEPROM_CPP__ 1

5.5.1 Macro Definition Documentation

5.5.1.1 #define __ARDUINO_EXTERNAL_24X16_EEPROM_CPP__ 1

Arduino - External 24x16 eeprom.

External24x16Eeprom.cpp

This an implementation of 24x16 eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file External24x16Eeprom.cpp.

5.6 External24x16Eeprom.cpp

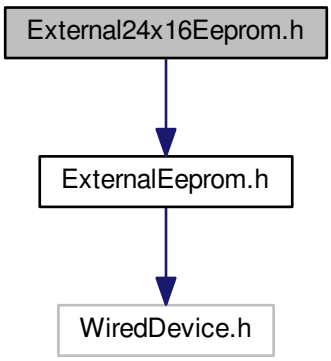
```
00001
00011 #ifndef __ARDUINO_EXTERNAL_24X16_EEPROM_CPP__
00012 #define __ARDUINO_EXTERNAL_24X16_EEPROM_CPP__ 1
00013
00014 #include "External24x16Eeprom.h"
00015 #include <Arduino.h>
00016 #include <Wire.h>
00017
00018 External24x16Eeprom::External24x16Eeprom(unsigned char
      deviceAddress) :
00019         ExternalEeprom(0x20, 0x7ff, deviceAddress) {
00020     Wire.begin();
00021 }
00022
```

```
00023 void External24x16Eeprom::writeBlock(unsigned int address, unsigned char*
      buf,
00024         int len) {
00025     unsigned char block;
00026     block = (unsigned char) ((address >> 8) & 0x07);
00027     Wire.beginTransmission(getDeviceAddress() | block);
00028     Wire.write((unsigned char) (address & 0xff));
00029     for (int i = 0; i < len; i++) {
00030         Wire.write(buf[i]);
00031     }
00032     Wire.endTransmission();
00033     delay(5);
00034 }
00035
00036 void External24x16Eeprom::readBlock(unsigned int address, unsigned char* buf,
00037         int len) {
00038     unsigned char block, blockAddress;
00039     block = (unsigned char) ((address >> 8) & 0x07);
00040     blockAddress = (getDeviceAddress() | block);
00041     Wire.beginTransmission(blockAddress);
00042     Wire.write((unsigned char) (address & 0xff));
00043     Wire.endTransmission();
00044     Wire.requestFrom((int) blockAddress, len);
00045     for (int i = 0; i < len; i++) {
00046         while (!Wire.available())
00047             ;
00048         buf[i] = Wire.read();
00049     }
00050 }
00051
00052 #endif /* __ARDUINO_EXTERNAL_24X16_EEPROM_CPP__ */
```

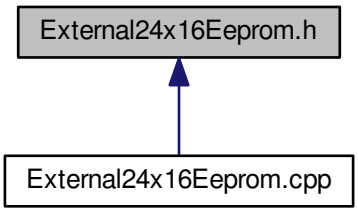
5.7 External24x16Eeprom.h File Reference

`#include <ExternalEeprom.h>`

Include dependency graph for External24x16Eeprom.h:



This graph shows which files directly or indirectly include this file:



Classes

- class External24x16Eeprom

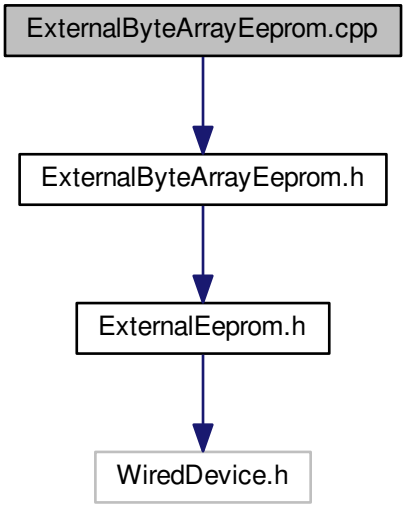
5.8 External24x16Eeprom.h

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_24X16_EEPROM_H__
00012 #define __ARDUINO_EXTERNAL_24X16_EEPROM_H__ 1
00013
00014 #include <ExternalEeprom.h>
00015
00016 class External24x16Eeprom : public ExternalEeprom {
00017 public:
00018
00024     External24x16Eeprom(unsigned char deviceAddress);
00025
00026 protected:
00027
00036     virtual void writeBlock(unsigned int address, unsigned char* buf, int len);
00037
00045     virtual void readBlock(unsigned int address, unsigned char* buf, int len);
00046 };
00047
00048 #endif /* __ARDUINO_EXTERNAL_24X16_EEPROM_H__ */
```

5.9 ExternalByteArrayEeprom.cpp File Reference

```
#include "ExternalByteArrayEeprom.h"
```

Include dependency graph for ExternalByteArrayEeprom.cpp:



Macros

- #define __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_CPP__ 1

5.9.1 Macro Definition Documentation

5.9.1.1 #define __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_CPP__ 1

Arduino - External Virtual eeprom.

ExternalByteArrayEeprom.cpp

This an implementation of Virtual eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file ExternalByteArrayEeprom.cpp.

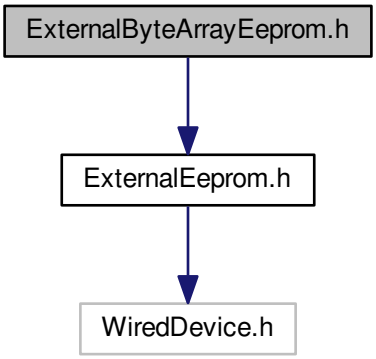
5.10 ExternalByteArrayEeprom.cpp

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_CPP__
00012 #define __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_CPP__ 1
00013
00014 #include "ExternalByteArrayEeprom.h"
00015
00016 ExternalByteArrayEeprom::ExternalByteArrayEeprom(unsigned
      char* byteArray, unsigned int deviceSize)
00017         : ExternalEeprom(0, 16, deviceSize), byteArray(byteArray) {
00018 }
00019
00020 void ExternalByteArrayEeprom::writeBlock(unsigned int address, unsigned
      char* buf, int len) {
00021     for (int i = 0; i < len; i++) {
```

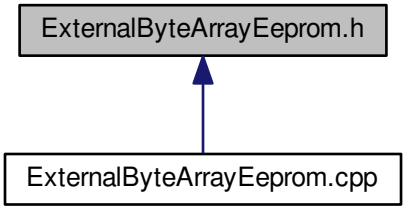
```
00022         byteArray[address + i] = buf[i];
00023     }
00024 }
00025
00026 void ExternalByteArrayEeprom::readBlock(unsigned int address, unsigned
      char* buf, int len) {
00027     for (int i = 0; i < len; i++) {
00028         buf[i] = byteArray[address + i];
00029     }
00030 }
00031
00032 #endif /* __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_CPP__ */
```

5.11 ExternalByteArrayEeprom.h File Reference

`#include <ExternalEeprom.h>`
Include dependency graph for ExternalByteArrayEeprom.h:



This graph shows which files directly or indirectly include this file:



Classes

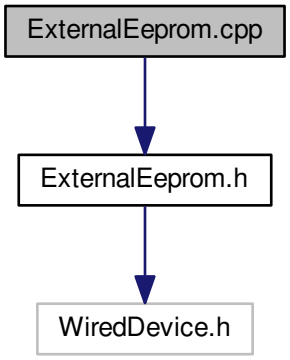
- class ExternalByteArrayEeprom

5.12 ExternalByteArrayEeprom.h

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_H__
00012 #define __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_H__ 1
00013
00014 #include <ExternalEeprom.h>
00015
00016 class ExternalByteArrayEeprom : public ExternalEeprom {
00017
00021     unsigned char* byteArray;
00022
00023 public:
00024
00030     ExternalByteArrayEeprom(unsigned char* byteArray, unsigned int
      deviceSize);
00031
00032 protected:
00033
00042     virtual void writeBlock(unsigned int address, unsigned char* buf, int len);
00043
00051     virtual void readBlock(unsigned int address, unsigned char* buf, int len);
00052 };
00053
00054 #endif /* __ARDUINO_EXTERNAL_BYTE_ARRAY_EEPROM_H__ */
```

5.13 ExternalEeprom.cpp File Reference

`#include "ExternalEeprom.h"`
Include dependency graph for ExternalEeprom.cpp:



Macros

- #define __ARDUINO_EXTERNAL_EEPROM_CPP__ 1
- #define min(a, b) ((a > b) ? b : a)

5.13.1 Macro Definition Documentation

5.13.1.1 #define __ARDUINO_EXTERNAL_EEPROM_CPP__ 1

Arduino - External eeprom.

ExternalEeprom.cpp

This is an abstract class of external eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file ExternalEeprom.cpp.

5.13.1.2 #define min(*a*, *b*) ((a > b) ? b : a)

Definition at line 14 of file ExternalEeprom.cpp.

5.14 ExternalEeprom.cpp

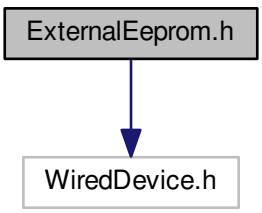
```
00001
00011 #ifndef __ARDUINO_EXTERNAL_EEPROM_CPP__
00012 #define __ARDUINO_EXTERNAL_EEPROM_CPP__ 1
00013
00014 #define min(a, b) ((a > b) ? b : a)
00015
00016 #include "ExternalEeprom.h"
00017
00018 ExternalEeprom::ExternalEeprom(unsigned char deviceAddress, int pageSize,
      unsigned int deviceSize)
00019         : WiredDevice(0x50 | (deviceAddress & 0x07)), pageSize(pageSize), deviceSize(deviceSize) {
00020 }
00021
00022 void ExternalEeprom::write(unsigned int address, unsigned char b) {
00023     writeBlock(address, &b, 1);
00024 }
00025
00026 void ExternalEeprom::writeBytes(unsigned int address, unsigned char* buf, int len
      ) {
00027     unsigned int eop, room;
00028     int chunkSize;
00029     room = (deviceSize - address);
00030     if (room == 0) {
00031         return;
00032     }
00033     len = (room < (unsigned char) len) ? room : len;
00034     eop = endOfPage(address);
00035     chunkSize = min(eop, (unsigned char) len);
00036     if (chunkSize > 0) {
00037         writeBlock(address, buf, chunkSize);
00038         address += chunkSize;
00039         buf += chunkSize;
00040         len -= chunkSize;
00041     }
00042     while (len > 0) {
00043         chunkSize = min(len, pageSize);
00044         writeBlock(address, buf, chunkSize);
00045         address += chunkSize;
00046         buf += chunkSize;
00047         len -= chunkSize;
00048     }
00049 }
00050
00051 int ExternalEeprom::read(unsigned int address) {
00052     unsigned char b;
00053     if (readBytes(address, &b, 1) == -1) {
00054         return -1;
00055     }
00056     return (int) b;
00057 }
00058
00059 int ExternalEeprom::readBytes(unsigned int address, unsigned char* buf, int len) {
00060     int cnt, chunkSize = pageSize;
00061     unsigned int available;
00062     if (address >= deviceSize) {
00063         return -1;
00064     }
00065     available = (deviceSize - address);
00066     if (available < (unsigned char) len) {
00067         len = (int) available;
00068     }
00069     cnt = len;
00070     while (len > 0) {
00071         chunkSize = min(len, pageSize);
00072         readBlock(address, buf, chunkSize);
00073         address += chunkSize;
00074         buf += chunkSize;
00075         len -= chunkSize;
00076     }
00077     return cnt;
```

```
00078 }
00079
00080 int ExternalEeprom::setBytes(unsigned int address, unsigned char b, int len) {
00081     unsigned char buf[pageSize];
00082     int eop, chunkSize;
00083     unsigned int room;
00084     if (address >= deviceSize) {
00085         return -1;
00086     }
00087     room = (deviceSize - address);
00088     if (room < (unsigned char) len) {
00089         len = (int) room;
00090     }
00091     for (int i = 0; i < pageSize; i++) {
00092         buf[i] = b;
00093     }
00094     eop = endOfPage(address);
00095     if (eop > 0) {
00096         chunkSize = min(eop, len);
00097         writeBlock(address, buf, chunkSize);
00098         address += chunkSize;
00099         len -= chunkSize;
00100     }
00101     while (len > 0) {
00102         chunkSize = min(len, pageSize);
00103         writeBlock(address, buf, chunkSize);
00104         address += chunkSize;
00105         len -= chunkSize;
00106     }
00107     return len;
00108 }
00109
00110 unsigned int ExternalEeprom::endOfPage(unsigned int address) {
00111     // Why / and then * by the same number?
00112     unsigned int eopAddr = ((address + pageSize - 1) / pageSize) *
      pageSize;
00113     return (eopAddr - address);
00114 }
00115
00116 #endif /* __ARDUINO_EXTERNAL_EEPROM_CPP__ */
```

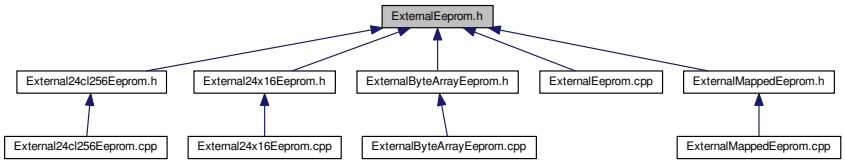
5.15 ExternalEeprom.h File Reference

`#include <WiredDevice.h>`

Include dependency graph for ExternalEeprom.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ExternalEeprom

5.16 ExternalEeprom.h

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_EEPROM_H__
00012 #define __ARDUINO_EXTERNAL_EEPROM_H__ 1
00013
00014 #include <WiredDevice.h>
00015
00016 class ExternalEeprom : public WiredDevice {
00017
00021     unsigned int deviceSize;
00022
00026     int pageSize;
00027 public:
00028
00035     virtual void write(unsigned int address, unsigned char b);
00036
00044     virtual void writeBytes(unsigned int address, unsigned char* buf, int len);
00045
00052     virtual int read(unsigned int address);
00053
00061     virtual int readBytes(unsigned int address, unsigned char* buf, int len);
00062
00070     virtual int setBytes(unsigned int address, unsigned char b, int len);
00071
00077     int getPageSize() {
00078         return pageSize;
00079     }
00080
00086     unsigned int getDeviceSize() {
00087         return deviceSize;
00088     }
00089
00098     virtual void writeBlock(unsigned int address, unsigned char* buf, int len) = 0;
00099
00107     virtual void readBlock(unsigned int address, unsigned char* buf, int len) = 0;
00108
00109 protected:
00110
00116     ExternalEeprom(unsigned char deviceAddress, int pageSize, unsigned int deviceSize);
00117
00125     unsigned int endOfPage(unsigned int address);
00126 };
00127
00128 #endif /* __ARDUINO_EXTERNAL_EEPROM_H__ */
```

5.17 ExternalFileEeprom.cpp File Reference

Macros

- #define __ARDUINO_EXTERNAL_FILE_EEPROM_CPP__ 1

5.17.1 Macro Definition Documentation

5.17.1.1 #define __ARDUINO_EXTERNAL_FILE_EEPROM_CPP__ 1

Arduino - External Virtual eeprom.

ExternalFileEeprom.cpp

This an implementation of Virtual eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file ExternalFileEeprom.cpp.

5.18 ExternalFileEeprom.cpp

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_FILE_EEPROM_CPP__
00012 #define __ARDUINO_EXTERNAL_FILE_EEPROM_CPP__ 1
00013
00014 #if USE_FILE_LIBRARIES
00015
00016 #include "ExternalFileEeprom.h"
00017 #include <stddef.h>
00018 #include <stdlib.h>
00019 #include <stddef.h>
00020
00021 ExternalFileEeprom::ExternalFileEeprom(char *fileName, unsigned int deviceSize) :
      ExternalEeprom(0, 16, deviceSize), fileName(fileName) {
00022     fp = fopen(fileName, "rb+");
00023     if (fp == NULL) {
00024         printf("Error when opening file: %s.\n", fileName);
00025         exit(1);
00026     }
00027 }
00028
00029 void ExternalFileEeprom::writeBlock(unsigned int address, unsigned char* buffer, int len) {
00030     fseek(fp, address, 0);
00031     fwrite(buffer, sizeof(unsigned char), len, fp);
00032 }
00033
00034 void ExternalFileEeprom::readBlock(unsigned int address, unsigned char* buffer, int len) {
00035     fseek(fp, address, 0);
00036     fread(buffer, sizeof(unsigned char), len, fp);
00037 }
00038
00039 #endif /* USE_FILE_LIBRARIES */
00040
00041 #endif /* __ARDUINO_EXTERNAL_FILE_EEPROM_CPP__ */
```

5.19 ExternalFileEeprom.h File Reference

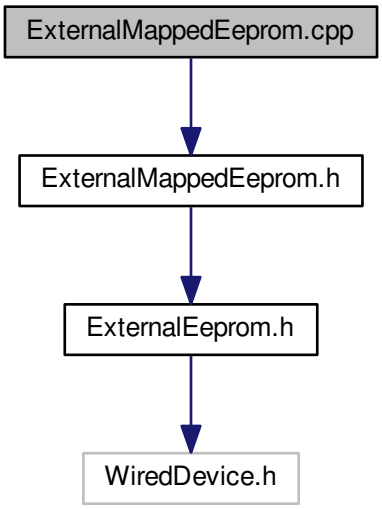
5.20 ExternalFileEeprom.h

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_FILE_EEPROM_H__
00012 #define __ARDUINO_EXTERNAL_FILE_EEPROM_H__ 1
00013
00014 #if USE_FILE_LIBRARIES
00015
00016 #include <ExternalEeprom.h>
00017
00018 class ExternalFileEeprom : public ExternalEeprom {
00019 private:
00020     char *fileName;
00021     FILE *fp;
00022 public:
00023
00029     ExternalFileEeprom(char *fileName, unsigned int size);
00030
00031 protected:
00032
00041     virtual void writeBlock(unsigned int address, unsigned char* buf, int len);
00042
00050     virtual void readBlock(unsigned int address, unsigned char* buf, int len);
00051 };
```

```
00052
00053 #endif /* USE_FILE_LIBRARIES */
00054
00055 #endif /* __ARDUINO_EXTERNAL_FILE_EEPROM_H__ */
```

5.21 ExternalMappedEeprom.cpp File Reference

`#include "ExternalMappedEeprom.h"`
Include dependency graph for ExternalMappedEeprom.cpp:



Macros

- #define __ARDUINO_EXTERNAL_MAPPED_EEPROM_CPP__ 1

5.21.1 Macro Definition Documentation

5.21.1.1 #define __ARDUINO_EXTERNAL_MAPPED_EEPROM_CPP__ 1

Arduino - External eeprom.

ExternalMappedEeprom.cpp

This is an abstract class of external eeprom.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file ExternalMappedEeprom.cpp.

5.22 ExternalMappedEeprom.cpp

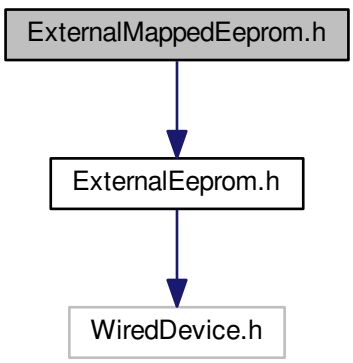
```
00001
00011 #ifndef __ARDUINO_EXTERNAL_MAPPED_EEPROM_CPP__
```

```
00012 #define __ARDUINO_EXTERNAL_MAPPED_EEPROM_CPP__ 1
00013
00014 #include "ExternalMappedEeprom.h"
00015
00016 ExternalMappedEeprom::ExternalMappedEeprom(
      ExternalEeprom* externalEeprom, unsigned int startAddress, unsigned int endAddress)
00017         : ExternalEeprom(externalEeprom->getDeviceAddress(), externalEeprom->getPageSize(),
      externalEeprom->getDeviceSize()), externalEeprom(externalEeprom) {
00018     this->startAddress = startAddress;
00019     this->endAddress = endAddress;
00020 }
00021
00022 void ExternalMappedEeprom::writeBlock(unsigned int address, unsigned char*
      buf, int len) {
00023     unsigned int mappedAddress = (address + startAddress);
00024     if (mappedAddress < endAddress) {
00025         unsigned int available = (endAddress - mappedAddress);
00026         len = (len > available) ? available : len;
00027         externalEeprom->writeBlock(mappedAddress, buf, len);
00028     }
00029 }
00030
00031 void ExternalMappedEeprom::readBlock(unsigned int address, unsigned char*
      buf, int len) {
00032     unsigned int mappedAddress = (address + startAddress);
00033     if (mappedAddress < endAddress) {
00034         unsigned int available = (endAddress - mappedAddress);
00035         len = (len > available) ? available : len;
00036         externalEeprom->readBlock(mappedAddress, buf, len);
00037     }
00038 }
00039
00040 #endif /* __ARDUINO_EXTERNAL_MAPPED_EEPROM_CPP__ */
```

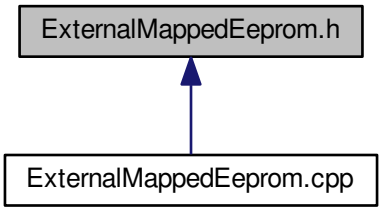
5.23 ExternalMappedEeprom.h File Reference

`#include <ExternalEeprom.h>`

Include dependency graph for ExternalMappedEeprom.h:



This graph shows which files directly or indirectly include this file:



Classes

- class ExternalMappedEeprom

5.24 ExternalMappedEeprom.h

```
00001
00011 #ifndef __ARDUINO_EXTERNAL_MAPPED_EEPROM_H__
00012 #define __ARDUINO_EXTERNAL_MAPPED_EEPROM_H__ 1
00013
00014 #include <ExternalEeprom.h>
00015
00016 class ExternalMappedEeprom: public ExternalEeprom {
00017
00021     ExternalEeprom* externalEeprom;
00022
00026     unsigned int startAddress;
00027
00031     unsigned int endAddress;
00032 public:
00033
00041     ExternalMappedEeprom(ExternalEeprom* externalEeprom, unsigned int
      startAddress, unsigned int endAddress);
00042
00048     unsigned int getDeviceSize() {
00049         return (endAddress - startAddress);
00050     }
00051
00052 protected:
00053
00062     virtual void writeBlock(unsigned int address, unsigned char* buf, int len);
00063
00071     virtual void readBlock(unsigned int address, unsigned char* buf, int len);
00072 };
00073
00074 #endif /* __ARDUINO_EXTERNAL_MAPPED_EEPROM_H__ */
```

Index