Arduino Device Driver

Generated by Doxygen 1.8.9.1

Tue Aug 18 2015 22:51:25

Contents

1 Hierarchical Index

1.1 Class Hierarchy

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

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 Device Class Reference

```
#include <Device.h>
```

Public Member Functions

- Device ()

4.1.1 Detailed Description

Arduino - Device.

Device.cpp

Author

Dalmir da Silva `dalmirdasilva@gmail.com`

Definition at line 12 of file Device.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Device::Device ( )

Public constructor.

Parameters

address	The wire address.

Definition at line 3 of file Device.cpp.

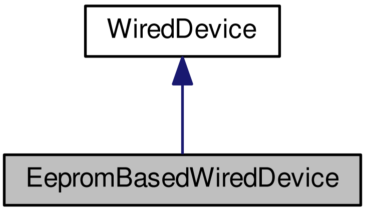
The documentation for this class was generated from the following files:

- Device.h
- Device.cpp

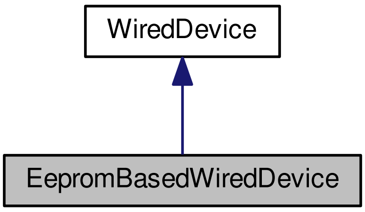
4.2 EepromBasedWiredDevice Class Reference

```
#include <EepromBasedWiredDevice.h>
```

Inheritance diagram for EepromBasedWiredDevice:



Collaboration diagram for EepromBasedWiredDevice:



Public Types

- enum { LITTLE_ENDIAN = 0x00, BIG_ENDIAN = 0x01 }

Public Member Functions

- EepromBasedWiredDevice (unsigned char deviceAddress, char addressSize, unsigned char endianness)
- EepromBasedWiredDevice (unsigned char deviceAddress, char addressSize)
- EepromBasedWiredDevice (unsigned char deviceAddress)
- void writeBlock (unsigned int address, unsigned char ∗buf, int len)
- void readBlock (unsigned int address, unsigned char ∗buf, int len)
- void setAddressSize (char addressSize)
- char getAddressSize ()

Public Attributes

- enum EepromBasedWiredDevice:: { ... } Endianness

Private Attributes

- char addressSize
- unsigned char endianness

Static Private Attributes

- static const char MAX_RETRIES_ON_READING = 0x7f

4.2.1 Detailed Description

Definition at line 16 of file EepromBasedWiredDevice.h.

4.2.2 Member Enumeration Documentation

4.2.2.1 anonymous enum

Enumerator

LITTLE_ENDIAN

BIG_ENDIAN

Definition at line 25 of file EepromBasedWiredDevice.h.

4.2.3 Constructor & Destructor Documentation

4.2.3.1 EepromBasedWiredDevice::EepromBasedWiredDevice (unsigned char *deviceAddress*, char *addressSize*, unsigned char *endianness*)

Public constructor.

Parameters

deviceAddress	The wire address.
addressSize	How long is the internal device addresses.
endianness	The endianness.

Definition at line 5 of file EepromBasedWiredDevice.cpp.

4.2.3.2 EepromBasedWiredDevice::EepromBasedWiredDevice (unsigned char *deviceAddress*, char *addressSize*)

Public constructor.

Definition at line 9 of file EepromBasedWiredDevice.cpp.

4.2.3.3 EepromBasedWiredDevice::EepromBasedWiredDevice (unsigned char *deviceAddress*)

Public constructor.

Definition at line 13 of file EepromBasedWiredDevice.cpp.

4.2.4 Member Function Documentation

4.2.4.1 char EepromBasedWiredDevice::getAddressSize ( )

Gets the address size.

addressSize

Definition at line 55 of file EepromBasedWiredDevice.cpp.

4.2.4.2 void EepromBasedWiredDevice::readBlock (unsigned int *address*, unsigned char ∗ *buf*, int *len*)

Reads a block of bytes from the device.

Sends the address MSB fist.

Parameters

address	
buf	
len	

Definition at line 29 of file EepromBasedWiredDevice.cpp.

4.2.4.3 void EepromBasedWiredDevice::setAddressSize (char *addressSize*)

Sets the address size.

Parameters

addressSize	

Definition at line 51 of file EepromBasedWiredDevice.cpp.

4.2.4.4 void EepromBasedWiredDevice::writeBlock (unsigned int *address,* unsigned char ∗ *buf,* int *len*)

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

Parameters

address	
buf	
len	

Definition at line 17 of file EepromBasedWiredDevice.cpp.

4.2.5 Member Data Documentation

4.2.5.1 char EepromBasedWiredDevice::addressSize `[private]`

Definition at line 20 of file EepromBasedWiredDevice.h.

4.2.5.2 unsigned char EepromBasedWiredDevice::endianness `[private]`

Definition at line 21 of file EepromBasedWiredDevice.h.

4.2.5.3 enum { ... } EepromBasedWiredDevice::Endianness

4.2.5.4 const char EepromBasedWiredDevice::MAX_RETRIES_ON_READING = 0x7f `[static]`, `[private]`

Definition at line 18 of file EepromBasedWiredDevice.h.

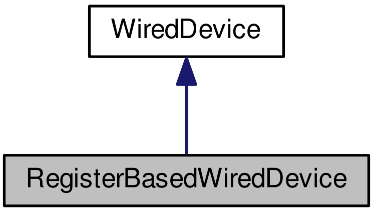
The documentation for this class was generated from the following files:

- EepromBasedWiredDevice.h
- EepromBasedWiredDevice.cpp

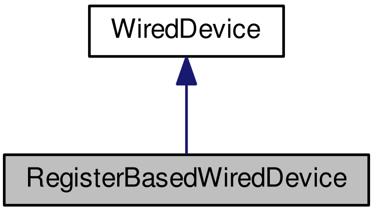
4.3 RegisterBasedWiredDevice Class Reference

```
#include <RegisterBasedWiredDevice.h>
```

Inheritance diagram for RegisterBasedWiredDevice:



Collaboration diagram for RegisterBasedWiredDevice:



Public Member Functions

- RegisterBasedWiredDevice (unsigned char address)
- void configureRegisterBits (unsigned char reg, unsigned char mask, unsigned char d)
- unsigned char writeRegister (unsigned char reg, unsigned char d)
- int readRegister (unsigned char reg)

Static Private Attributes

- static const unsigned char MAX_RETRIES_ON_READING = 10

4.3.1 Detailed Description

Arduino - Register Based Wire Device.

RegisterBasedWireDevice.cpp

Author

Dalmir da Silva `dalmirdasilva@gmail.com`

Definition at line 14 of file RegisterBasedWiredDevice.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 RegisterBasedWiredDevice::RegisterBasedWiredDevice (unsigned char *address*)

Public constructor.

Parameters

address	The wire address.

Definition at line 5 of file RegisterBasedWiredDevice.cpp.

4.3.3 Member Function Documentation

4.3.3.1 void RegisterBasedWiredDevice::configureRegisterBits (unsigned char *reg,* unsigned char *mask,* unsigned char *d*)

Configures a register.

Parameters

reg	The register number.
mask	The mask to be used.
d	The value to be used.

Definition at line 10 of file RegisterBasedWiredDevice.cpp.

4.3.3.2 int RegisterBasedWiredDevice::readRegister (unsigned char *reg*)

Reades a value from a register.

Parameters

reg	The register number.

Returns

The register value.

Definition at line 27 of file RegisterBasedWiredDevice.cpp.

4.3.3.3 unsigned char RegisterBasedWiredDevice::writeRegister (unsigned char *reg,* unsigned char *d*)

Writes a value to a register.

Parameters

reg	The register number.
d	The value to be used.

Definition at line 19 of file RegisterBasedWiredDevice.cpp.

4.3.4 Member Data Documentation

4.3.4.1 const unsigned char RegisterBasedWiredDevice::MAX_RETRIES_ON_READING = 10 `[static]`, `[private]`

Definition at line 16 of file RegisterBasedWiredDevice.h.

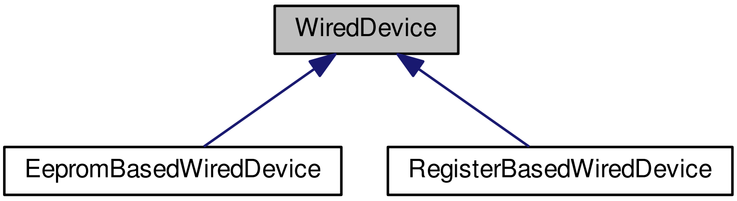
The documentation for this class was generated from the following files:

- RegisterBasedWiredDevice.h
- RegisterBasedWiredDevice.cpp

4.4 WiredDevice Class Reference

```
#include <WiredDevice.h>
```

Inheritance diagram for WiredDevice:



Public Member Functions

- WiredDevice (unsigned char deviceAddress)
- unsigned char getDeviceAddress ()
- void setDeviceAddress (unsigned char deviceAddress)

Private Attributes

- unsigned char deviceAddress

4.4.1 Detailed Description

Arduino - Wired Device.

WiredDevice.cpp

Author

Dalmir da Silva `dalmirdasilva@gmail.com`

Definition at line 12 of file WiredDevice.h.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 WiredDevice::WiredDevice (unsigned char *deviceAddress*)

Public constructor.

Parameters

address	The wire address.

Definition at line 4 of file WiredDevice.cpp.

4.4.3 Member Function Documentation

4.4.3.1 unsigned char WiredDevice::getDeviceAddress ( )

Gets the device address.

Returns

address

Definition at line 9 of file WiredDevice.cpp.

4.4.3.2 void WiredDevice::setDeviceAddress (unsigned char *deviceAddress*)

Sets the device address.

Parameters

address	The device address.

Definition at line 13 of file WiredDevice.cpp.

4.4.4 Member Data Documentation

4.4.4.1 unsigned char WiredDevice::deviceAddress [private]

Definition at line 14 of file WiredDevice.h.

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

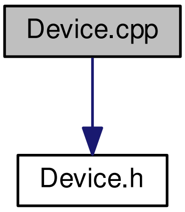
- WiredDevice.h
- WiredDevice.cpp

5 File Documentation

5.1 Device.cpp File Reference

```
#include "Device.h"
```

Include dependency graph for Device.cpp:



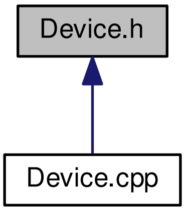
5.2 Device.cpp

```
00001 #include "Device.h"
```

```

Device::Device() {
}
```

5.3 Device.h File Reference

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



Classes

- class Device

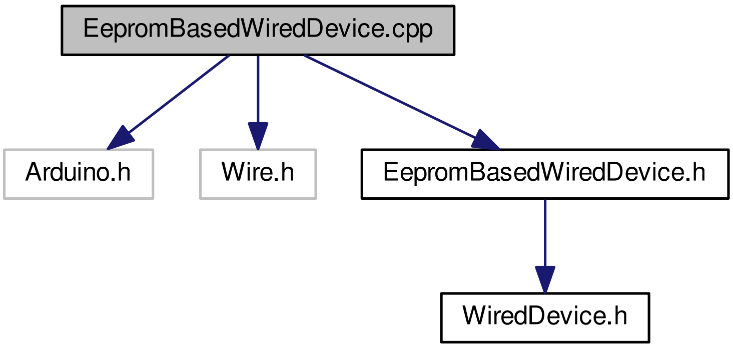
5.4 Device.h

```

#ifndef __ARDUINO_DRIVER_DEVICE_H__
#define __ARDUINO_DRIVER_DEVICE_H__ 1

class Device {

public:

    Device();
};

#endif /* __ARDUINO_DRIVER_DEVICE_H__ */
```

5.5 EepromBasedWiredDevice.cpp File Reference

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

Include dependency graph for EepromBasedWiredDevice.cpp:



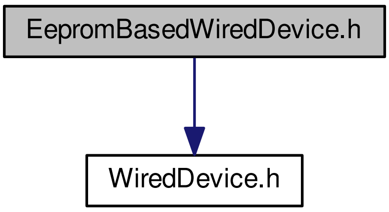
5.6 EepromBasedWiredDevice.cpp

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

EepromBasedWiredDevice::EepromBasedWiredDevice(unsigned char deviceAddress, char addressSize, unsigned char endianness)
        : WiredDevice(deviceAddress), addressSize(addressSize), endianness(endianness) {
}

EepromBasedWiredDevice::EepromBasedWiredDevice(unsigned char deviceAddress, char addressSize)
        : EepromBasedWiredDevice(deviceAddress, addressSize, LITTLE_ENDIAN) {
}

EepromBasedWiredDevice::EepromBasedWiredDevice(unsigned char deviceAddress)
        : EepromBasedWiredDevice(deviceAddress, 0x02) {
}

void EepromBasedWiredDevice::writeBlock(unsigned int address, unsigned char* buf, int len) {
    Wire.beginTransmission(getDeviceAddress());
    for (char i = addressSize; i > 0; i--) {
        Wire.write((unsigned char) (address >> ((i - 1) * 8)) & 0xff);
    }
    for (int i = 0; i < len; i++) {
        Wire.write(buf[i]);
    }
    Wire.endTransmission();
    delay(EEPROM_BASED_WIRED_DEVICE_AFTER_WRITE_DELAY);
}

void EepromBasedWiredDevice::readBlock(unsigned int address, unsigned char * buf, int len) {
    char tries;
    unsigned char last = len - 1;
    Wire.beginTransmission(getDeviceAddress());
    for (char i = addressSize; i > 0; i--) {
        Wire.write((unsigned char) (address >> ((i - 1) * 8)) & 0xff);
    }
    Wire.endTransmission();
    delay(EEPROM_BASED_WIRED_DEVICE_AFTER_WRITE_DELAY);
    Wire.requestFrom((int) getDeviceAddress(), len);
    for (int i = 0; i < len; i++) {
        tries = MAX_RETRIES_ON_READING;
        while (!Wire.available() && --tries > 0) {
            delayMicroseconds(1);
        }
        if (tries == 0) {
            return;
        }
        buf[(endianness == BIG_ENDIAN) ? last - i : i] = Wire.read();
    }
}

void EepromBasedWiredDevice::setAddressSize(char addressSize) {
    this->addressSize = addressSize;
}

char EepromBasedWiredDevice::getAddressSize() {
    return addressSize;
}
```

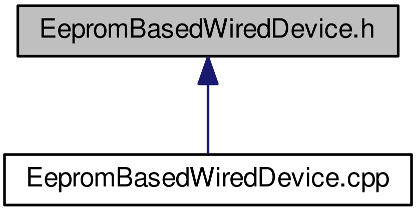
5.7 EepromBasedWiredDevice.h File Reference

`#include <WiredDevice.h>`

Include dependency graph for EepromBasedWiredDevice.h:



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



Classes

- class EepromBasedWiredDevice

Macros

- #define EEPROM_BASED_WIRED_DEVICE_AFTER_WRITE_DELAY 5

5.7.1 Macro Definition Documentation

5.7.1.1 #define EEPROM_BASED_WIRED_DEVICE_AFTER_WRITE_DELAY 5

Arduino - Register Based Wire Device.

RegisterBasedWireDevice.cpp

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file EepromBasedWiredDevice.h.

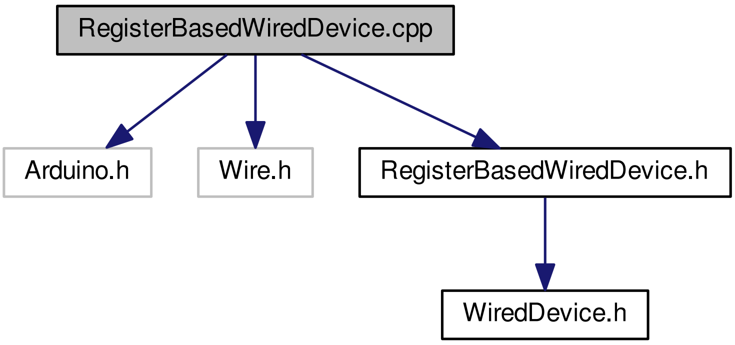
5.8 EepromBasedWiredDevice.h

```
00001
00009 #ifndef __ARDUINO_DRIVER_EEPROM_BASED_WIRED_DEVICE_H__
00010 #define __ARDUINO_DRIVER_EEPROM_BASED_WIRED_DEVICE_H__ 1
00011
00012 #define EEPROM_BASED_WIRED_DEVICE_AFTER_WRITE_DELAY     5
00013
00014 #include <WiredDevice.h>
00015
00016 class EepromBasedWiredDevice: public WiredDevice {
00017
00018     const static char MAX_RETRIES_ON_READING = 0x7f;
00019
00020     char addressSize;
00021     unsigned char endianness;
00022
00023 public:
00024
00025     enum {
00026         LITTLE_ENDIAN = 0x00,
00027         BIG_ENDIAN = 0x01
00028     } Endianness;
00029
00037     EepromBasedWiredDevice(unsigned char deviceAddress, char addressSize
      , unsigned char endianness);
00038
00042     EepromBasedWiredDevice(unsigned char deviceAddress, char addressSize
      );
00043
00047     EepromBasedWiredDevice(unsigned char deviceAddress);
00048
00056     void writeBlock(unsigned int address, unsigned char* buf, int len);
00057
00067     void readBlock(unsigned int address, unsigned char* buf, int len);
00068
00074     void setAddressSize(char addressSize);
00075
00081     char getAddressSize();
00082 };
00083
00084 #endif /* __ARDUINO_DRIVER_EEPROM_BASED_WIRED_DEVICE_H__ */
```

5.9 RegisterBasedWiredDevice.cpp File Reference

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

Include dependency graph for RegisterBasedWiredDevice.cpp:



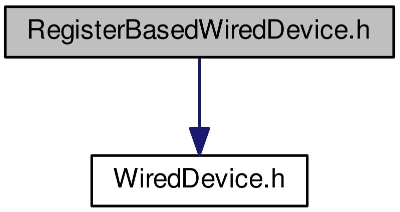
5.10 RegisterBasedWiredDevice.cpp

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

RegisterBasedWiredDevice::RegisterBasedWiredDevice(
        unsigned char address) :
        WiredDevice(address) {
}

void RegisterBasedWiredDevice::configureRegisterBits(
unsigned char reg,
        unsigned char mask, unsigned char d) {
    unsigned char n;
    n = readRegister(reg);
    n &= ~(mask);
    n |= d & mask;
    writeRegister(reg, n);
}

unsigned char RegisterBasedWiredDevice::writeRegister(unsigned char
reg,
        unsigned char d) {
    Wire.beginTransmission(getDeviceAddress());
    Wire.write(reg);
    Wire.write(d);
    return Wire.endTransmission();
}

int RegisterBasedWiredDevice::readRegister(unsigned char reg) {
    char tries = MAX_RETRIES_ON_READING;
    Wire.beginTransmission(getDeviceAddress());
    Wire.write(reg);
    char status = Wire.endTransmission(false);
    if (status != 0) {
        return -(status);
    }
    Wire.requestFrom(getDeviceAddress(), (unsigned char) 1);
    while (!Wire.available() && --tries > 0) {
        delayMicroseconds(1);
    }
    if (tries == 0) {
        return -5;
    }
    return Wire.read();
}
```

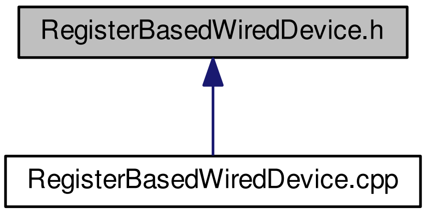
5.11 RegisterBasedWiredDevice.h File Reference

```
#include <WiredDevice.h>
```

Include dependency graph for RegisterBasedWiredDevice.h:



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



Classes

- class RegisterBasedWiredDevice

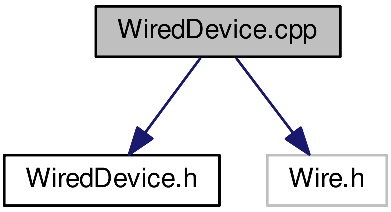
5.12 RegisterBasedWiredDevice.h

```

#ifndef __ARDUINO_DRIVER_REGISTER_BASED_WIRED_DEVICE_H__
#define __ARDUINO_DRIVER_REGISTER_BASED_WIRED_DEVICE_H__ 1

#include <WiredDevice.h>

class RegisterBasedWiredDevice: public WiredDevice {

    const static unsigned char MAX_RETRIES_ON_READING = 10;

public:

    RegisterBasedWiredDevice(unsigned char address);

    void configureRegisterBits(unsigned char reg, unsigned char mask,
            unsigned char d);

    unsigned char writeRegister(unsigned char reg, unsigned char d);

    int readRegister(unsigned char reg);
};

#endif /* __ARDUINO_DRIVER_REGISTER_BASED_WIRED_DEVICE_H__ */
```

5.13 WiredDevice.cpp File Reference

```
#include "WiredDevice.h"
#include <Wire.h>
```

Include dependency graph for WiredDevice.cpp:

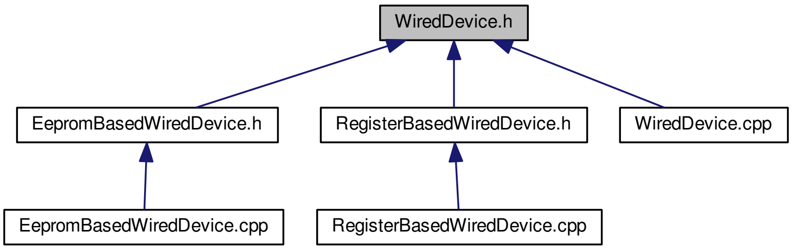


5.14 WiredDevice.cpp

```
#include "WiredDevice.h"
#include <Wire.h>

WiredDevice::WiredDevice(unsigned char deviceAddress)
        : deviceAddress(deviceAddress) {
    Wire.begin();
}

unsigned char WiredDevice::getDeviceAddress() {
    return deviceAddress;
}

void WiredDevice::setDeviceAddress(unsigned char deviceAddress) {
    this->deviceAddress = deviceAddress;
}
```

5.15 WiredDevice.h File Reference

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


WiredDevice.h
EepromBasedWiredDevice.h
RegisterBasedWiredDevice.h
WiredDevice.cpp
EepromBasedWiredDevice.cpp
RegisterBasedWiredDevice.cpp

Classes

- class WiredDevice

5.16 WiredDevice.h

```
#ifndef __ARDUINO_DRIVER_WIRED_DEVICE_H__
#define __ARDUINO_DRIVER_WIRED_DEVICE_H__ 1

class WiredDevice {

    unsigned char deviceAddress;

public:

     WiredDevice(unsigned char deviceAddress);

     unsigned char getDeviceAddress();

     void setDeviceAddress(unsigned char deviceAddress);
};

#endif /* __ARDUINO_DRIVER_WIRED_DEVICE_H__ */
```

Index