Arduino Resource Based File System Library

Generated by Doxygen 1.8.9.1

Wed Aug 19 2015 01:07:23

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

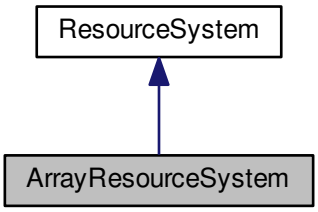
Here is a list of all files with brief descriptions:

4 Class Documentation

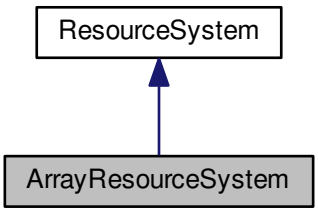
4.1 ArrayResourceSystem Class Reference

```
#include <ArrayResourceSystem.h>
```

Inheritance diagram for ArrayResourceSystem:



Collaboration diagram for ArrayResourceSystem:



Public Member Functions

- ArrayResourceSystem (unsigned char *array, unsigned int size, rbfs_t *rbfs)

Protected Member Functions

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

Private Attributes

- unsigned char * array
- unsigned int size

Additional Inherited Members

4.1.1 Detailed Description

Arduino - A simple resource implementation.

SimpleArrayResourceIO.h

This is the Resource IO representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 17 of file ArrayResourceSystem.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 ArrayResourceSystem::ArrayResourceSystem (unsigned char * *array*, unsigned int *size*, rbfs_t * *rbfs*)

Definition at line 16 of file ArrayResourceSystem.cpp.

4.1.3 Member Function Documentation

4.1.3.1 int ArrayResourceSystem::readBytes (unsigned int *address*, unsigned char * *buf*, unsigned int *len*) [protected], [virtual]

Definition at line 19 of file ArrayResourceSystem.cpp.

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

Definition at line 31 of file ArrayResourceSystem.cpp.

4.1.4 Member Data Documentation

4.1.4.1 unsigned char* ArrayResourceSystem::array [private]

Definition at line 20 of file ArrayResourceSystem.h.

4.1.4.2 unsigned int ArrayResourceSystem::size [private]

Definition at line 21 of file ArrayResourceSystem.h.

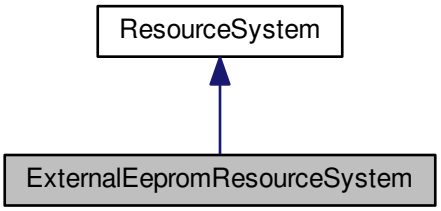
The documentation for this class was generated from the following files:

- ArrayResourceSystem.h
- ArrayResourceSystem.cpp

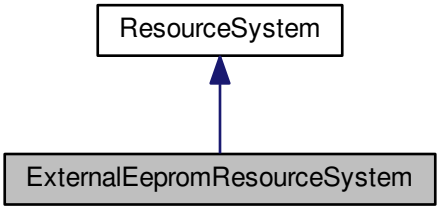
4.2 ExternalEepromResourceSystem Class Reference

```
#include <ExternalEepromResourceSystem.h>
```

Inheritance diagram for ExternalEepromResourceSystem:



Collaboration diagram for ExternalEepromResourceSystem:



Public Member Functions

- ExternalEepromResourceSystem (ExternalEeprom ∗eeprom, rbfs_t ∗rbfs)

Protected Member Functions

- virtual int readBytes (unsigned int address, unsigned char ∗buf, int len)
- virtual void writeBytes (unsigned int address, unsigned char ∗buf, int len)

Private Attributes

- ExternalEeprom ∗ eeprom

Additional Inherited Members

4.2.1 Detailed Description

Arduino - A simple resource implementation.

SimpleExternlaEepromResourceIO.h

This is the Resource IO representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 17 of file ExternalEepromResourceSystem.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 ExternalEepromResourceSystem::ExternalEepromResourceSystem (ExternalEeprom * *eeprom,* rbfs_t * *rbfs*)

Definition at line 17 of file ExternalEepromResourceSystem.cpp.

4.2.3 Member Function Documentation

4.2.3.1 int ExternalEepromResourceSystem::readBytes (unsigned int *address,* unsigned char * *buf,* int *len*) [protected], [virtual]

Implements ResourceSystem.

Definition at line 20 of file ExternalEepromResourceSystem.cpp.

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

Implements ResourceSystem.

Definition at line 24 of file ExternalEepromResourceSystem.cpp.

4.2.4 Member Data Documentation

4.2.4.1 ExternalEeprom* ExternalEepromResourceSystem::eeprom [private]

Definition at line 20 of file ExternalEepromResourceSystem.h.

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

- ExternalEepromResourceSystem.h
- ExternalEepromResourceSystem.cpp

4.3 Resource Class Reference

`#include <Resource.h>`

Public Types

- enum ResourceOperationResult {
  OPERATION_SUCCESS = 0, OPERATION_ERROR_RESOURCE_OPENED = 1, OPERATION_ERROR↩
  _RESOURCE_CLOSED = 2, OPERATION_ERROR_RESOURCE_READ_ONLY = 3,
  OPERATION_ERROR_NO_SPACE_AVAILABLE = 4, OPERATION_ERROR_DRIVER_BUSY = 5, OPE↩
  RATION_ERROR_SEEK_OUT_OF_BOUND = 6, OPERATION_ERROR_RESOURCE_DOES_NOT_ALL↩
  OCATED = 7,
  OPERATION_ERROR_DRIVER_NOT_MOUNTED = 8, OPERATION_ERROR_IO_ERROR = 9 }
- enum OpenOptions { OPEN_READ_WRITE = 0, OPEN_READ_ONLY = 1 }
- enum ResourceSeekOrigin { SEEK_ORIGIN_BEGIN = 0, SEEK_ORIGIN_CURRENT = 1 }

Public Member Functions

- Resource (rbfs_resource_code_t code, rbfs_t *rbfs)
- ResourceOperationResult getLastOperationResult ()
- void setCode (int code)
- int getCode ()
- void setRbfs (rbfs_t *rbfs)
- rbfs_t * getRbfs ()
- bool open (OpenOptions options)
- bool close ()
- void write (unsigned char b)
- void writeBytes (unsigned char *buf, int len)
- int read ()
- int readBytes (unsigned char *buf, int len)
- bool seek (ResourceSeekOrigin origin, unsigned int offset)
- bool truncate ()
- void sync ()
- bool rewind ()
- void release ()
- unsigned int size ()
- unsigned int tell ()
- bool eor ()
- bool error ()
- bool isReadOnly ()

Private Attributes

- rbfs_resource_code_t code
- rbfs_resource_t resource
- rbfs_t * rbfs
- ResourceOperationResult lastOperationResult

4.3.1 Detailed Description

Arduino - A simple resource implementation.

SimpleResource.h

This is the Resource representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 16 of file Resource.h.

4.3.2 Member Enumeration Documentation

4.3.2.1 enum Resource::OpenOptions

Enumerator

OPEN_READ_WRITE

OPEN_READ_ONLY

Definition at line 33 of file Resource.h.

4.3.2.2 enum Resource::ResourceOperationResult

Enumerator

OPERATION_SUCCESS

OPERATION_ERROR_RESOURCE_OPENED

OPERATION_ERROR_RESOURCE_CLOSED

OPERATION_ERROR_RESOURCE_READ_ONLY

OPERATION_ERROR_NO_SPACE_AVAILABLE

OPERATION_ERROR_DRIVER_BUSY

OPERATION_ERROR_SEEK_OUT_OF_BOUND

OPERATION_ERROR_RESOURCE_DOES_NOT_ALLOCATED

OPERATION_ERROR_DRIVER_NOT_MOUNTED

OPERATION_ERROR_IO_ERROR

Definition at line 20 of file Resource.h.

4.3.2.3 enum Resource::ResourceSeekOrigin

Enumerator

SEEK_ORIGIN_BEGIN

SEEK_ORIGIN_CURRENT

Definition at line 38 of file Resource.h.

4.3.3 Constructor & Destructor Documentation

4.3.3.1 Resource::Resource (rbfs_resource_code_t *code*, rbfs_t * *rbfs*)

Definition at line 16 of file Resource.cpp.

4.3.4 Member Function Documentation

4.3.4.1 bool Resource::close ( )

Definition at line 25 of file Resource.cpp.

4.3.4.2 bool Resource::eor ( )

Definition at line 102 of file Resource.cpp.

4.3.4.3 bool Resource::error ( )

Definition at line 106 of file Resource.cpp.

4.3.4.4 int Resource::getCode ( ) [inline]

Definition at line 62 of file Resource.h.

4.3.4.5 ResourceOperationResult Resource::getLastOperationResult ( ) [inline]

Definition at line 54 of file Resource.h.

4.3.4.6 rbfs_t* Resource::getRbfs ( ) [inline]

Definition at line 70 of file Resource.h.

4.3.4.7 bool Resource::isReadOnly ( )

Definition at line 110 of file Resource.cpp.

4.3.4.8 bool Resource::open (OpenOptions *options*)

Definition at line 20 of file Resource.cpp.

4.3.4.9 int Resource::read ( )

Definition at line 42 of file Resource.cpp.

4.3.4.10 int Resource::readBytes (unsigned char * *buf*, int *len*)

Definition at line 49 of file Resource.cpp.

4.3.4.11 void Resource::release ( )

Definition at line 89 of file Resource.cpp.

4.3.4.12 bool Resource::rewind ( )

Definition at line 84 of file Resource.cpp.

4.3.4.13 bool Resource::seek (ResourceSeekOrigin *origin*, unsigned int *offset*)

Definition at line 69 of file Resource.cpp.

4.3.4.14 void Resource::setCode (int *code*) [inline]

Definition at line 58 of file Resource.h.

4.3.4.15 void Resource::setRbfs (rbfs_t * *rbfs*) [inline]

Definition at line 66 of file Resource.h.

4.3.4.16 unsigned int Resource::size ( )

Definition at line 94 of file Resource.cpp.

4.3.4.17 void Resource::sync ( )

Definition at line 79 of file Resource.cpp.

4.3.4.18 unsigned int Resource::tell ( )

Definition at line 98 of file Resource.cpp.

4.3.4.19 bool Resource::truncate ( )

Definition at line 74 of file Resource.cpp.

4.3.4.20 void Resource::write (unsigned char *b*)

Definition at line 31 of file Resource.cpp.

4.3.4.21 void Resource::writeBytes (unsigned char * *buf*, int *len*)

Definition at line 35 of file Resource.cpp.

4.3.5 Member Data Documentation

4.3.5.1 rbfs_resource_code_t Resource::code [private]

Definition at line 45 of file Resource.h.

4.3.5.2 ResourceOperationResult Resource::lastOperationResult [private]

Definition at line 48 of file Resource.h.

4.3.5.3 rbfs_t* Resource::rbfs [private]

Definition at line 47 of file Resource.h.

4.3.5.4 rbfs_resource_t Resource::resource [private]

Definition at line 46 of file Resource.h.

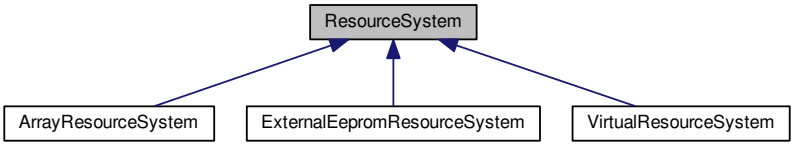
The documentation for this class was generated from the following files:

- Resource.h
- Resource.cpp

4.4 ResourceSystem Class Reference

`#include <ResourceSystem.h>`

Inheritance diagram for ResourceSystem:



Public Types

- enum MountOptions { MOUNT_READ_WRITE = 0, MOUNT_READ_ONLY = 1 }

Public Member Functions

- ResourceSystem (rbfs_t *rbfs)
- Resource::ResourceOperationResult getLastOperationResult ()
- virtual bool format ()
- virtual bool mount (MountOptions options)
- virtual bool umount ()
- virtual Resource allocResource ()
- virtual Resource loadResource (int code)
- virtual unsigned int totalSpace ()
- virtual unsigned int availableSpace ()
- virtual void flush ()

Protected Member Functions

- virtual int readBytes (unsigned int address, unsigned char *buf, int len)=0
- virtual void writeBytes (unsigned int address, unsigned char *buf, int len)=0

Private Member Functions

- unsigned char read (unsigned char driver, unsigned int address)
- void write (unsigned char driver, unsigned int address, unsigned char b)
- void checkCache (uint16_t address)

Private Attributes

- rbfs_t * rbfs
- Resource::ResourceOperationResult lastOperationResult
- bool wasCacheChanged
- bool wasCacheInitialized
- unsigned int cacheMemoryAddress
- unsigned char cache [RESOURCE_SYSTEM_CACHE_SIZE]
- unsigned int cacheMiss
- unsigned int cacheHit
- unsigned int validCacheSize

4.4.1 Detailed Description

Definition at line 20 of file ResourceSystem.h.

4.4.2 Member Enumeration Documentation

4.4.2.1 enum ResourceSystem::MountOptions

Enumerator

MOUNT_READ_WRITE

MOUNT_READ_ONLY

Definition at line 33 of file ResourceSystem.h.

4.4.3 Constructor & Destructor Documentation

4.4.3.1 ResourceSystem::ResourceSystem (rbfs_t * *rbfs*)

Definition at line 16 of file ResourceSystem.cpp.

4.4.4 Member Function Documentation

4.4.4.1 Resource ResourceSystem::allocResource ( ) [virtual]

Definition at line 43 of file ResourceSystem.cpp.

4.4.4.2 unsigned int ResourceSystem::availableSpace ( ) [virtual]

Definition at line 66 of file ResourceSystem.cpp.

4.4.4.3 void ResourceSystem::checkCache (uint16_t *address*) [private]

Definition at line 86 of file ResourceSystem.cpp.

4.4.4.4 void ResourceSystem::flush ( ) [virtual]

Reimplemented in VirtualResourceSystem.

Definition at line 31 of file ResourceSystem.cpp.

4.4.4.5 bool ResourceSystem::format ( ) [virtual]

Definition at line 21 of file ResourceSystem.cpp.

4.4.4.6 Resource::ResourceOperationResult ResourceSystem::getLastOperationResult ( ) [inline]

Definition at line 40 of file ResourceSystem.h.

4.4.4.7 Resource ResourceSystem::loadResource (int *code*) [virtual]

Definition at line 54 of file ResourceSystem.cpp.

4.4.4.8 bool ResourceSystem::mount (MountOptions *options*) [virtual]

Definition at line 26 of file ResourceSystem.cpp.

4.4.4.9 unsigned char ResourceSystem::read (unsigned char *driver,* unsigned int *address*) [private]

Definition at line 71 of file ResourceSystem.cpp.

4.4.4.10 virtual int ResourceSystem::readBytes (unsigned int *address,* unsigned char * *buf,* int *len*) [protected], [pure virtual]

Implemented in VirtualResourceSystem, and ExternalEepromResourceSystem.

4.4.4.11 unsigned int ResourceSystem::totalSpace ( ) [virtual]

Definition at line 62 of file ResourceSystem.cpp.

4.4.4.12 bool ResourceSystem::umount ( ) [virtual]

Definition at line 37 of file ResourceSystem.cpp.

4.4.4.13 void ResourceSystem::write (unsigned char *driver,* unsigned int *address,* unsigned char *b*) [private]

Definition at line 80 of file ResourceSystem.cpp.

4.4.4.14 virtual void ResourceSystem::writeBytes (unsigned int *address,* unsigned char * *buf,* int *len*) [protected], [pure virtual]

Implemented in VirtualResourceSystem, and ExternalEepromResourceSystem.

4.4.5 Member Data Documentation

4.4.5.1 unsigned char ResourceSystem::cache[RESOURCE_SYSTEM_CACHE_SIZE] [private]

Definition at line 27 of file ResourceSystem.h.

4.4.5.2 unsigned int ResourceSystem::cacheHit [private]

Definition at line 28 of file ResourceSystem.h.

4.4.5.3 unsigned int ResourceSystem::cacheMemoryAddress [private]

Definition at line 26 of file ResourceSystem.h.

4.4.5.4 unsigned int ResourceSystem::cacheMiss [private]

Definition at line 28 of file ResourceSystem.h.

4.4.5.5 Resource::ResourceOperationResult ResourceSystem::lastOperationResult [private]

Definition at line 23 of file ResourceSystem.h.

4.4.5.6 rbfs_t* ResourceSystem::rbfs [private]

Definition at line 22 of file ResourceSystem.h.

4.4.5.7 unsigned int ResourceSystem::validCacheSize [private]

Definition at line 29 of file ResourceSystem.h.

4.4.5.8 bool ResourceSystem::wasCacheChanged [private]

Definition at line 25 of file ResourceSystem.h.

4.4.5.9 bool ResourceSystem::wasCacheInitialized [private]

Definition at line 25 of file ResourceSystem.h.

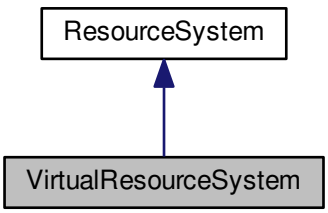
The documentation for this class was generated from the following files:

- ResourceSystem.h
- ResourceSystem.cpp

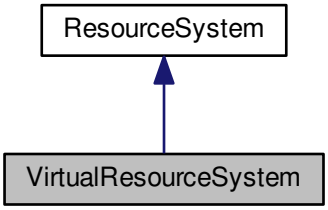
4.5 VirtualResourceSystem Class Reference

`#include <VirtualResourceSystem.h>`

Inheritance diagram for VirtualResourceSystem:



Collaboration diagram for VirtualResourceSystem:



Public Member Functions

- VirtualResourceSystem (rbfs_t *rbfs, char *fileName)
- virtual bool open ()
- virtual void flush ()
- virtual void close ()

Protected Member Functions

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

Private Attributes

- char * fileName
- FILE * fp

Additional Inherited Members

4.5.1 Detailed Description

Arduino - A simple resource implementation.

SimpleVirtualResourceIO.h

This is the Resource IO representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 18 of file VirtualResourceSystem.h.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 VirtualResourceSystem::VirtualResourceSystem (rbfs_t * *rbfs*, char * *fileName*)

Definition at line 19 of file VirtualResourceSystem.cpp.

4.5.3 Member Function Documentation

4.5.3.1 void VirtualResourceSystem::close ( ) [virtual]

Definition at line 37 of file VirtualResourceSystem.cpp.

4.5.3.2 void VirtualResourceSystem::flush ( ) [virtual]

Reimplemented from ResourceSystem.

Definition at line 32 of file VirtualResourceSystem.cpp.

4.5.3.3 bool VirtualResourceSystem::open ( ) [virtual]

Definition at line 23 of file VirtualResourceSystem.cpp.

4.5.3.4 int VirtualResourceSystem::readBytes (unsigned int *address*, unsigned char * *buf*, int *len*) [protected], [virtual]

Implements ResourceSystem.

Definition at line 42 of file VirtualResourceSystem.cpp.

4.5.3.5 void VirtualResourceSystem::writeBytes (unsigned int *address*, unsigned char * *buf*, int *len*) [protected], [virtual]

Implements ResourceSystem.

Definition at line 47 of file VirtualResourceSystem.cpp.

4.5.4 Member Data Documentation

4.5.4.1 char* VirtualResourceSystem::fileName [private]

Definition at line 20 of file VirtualResourceSystem.h.

4.5.4.2 FILE* VirtualResourceSystem::fp [private]

Definition at line 21 of file VirtualResourceSystem.h.

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

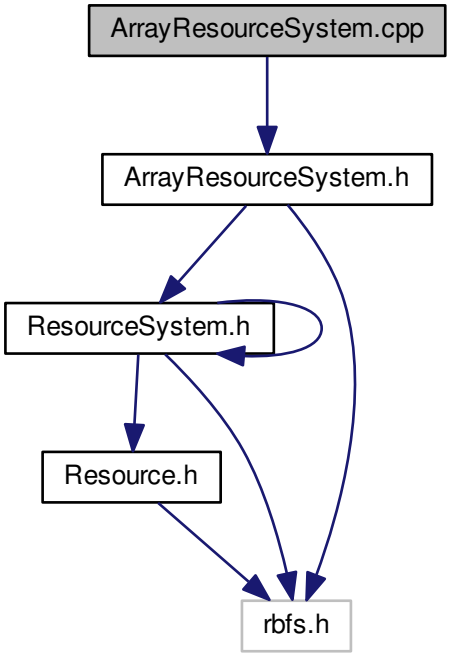
- VirtualResourceSystem.h
- VirtualResourceSystem.cpp

5 File Documentation

5.1 ArrayResourceSystem.cpp File Reference

```
#include "ArrayResourceSystem.h"
```

Include dependency graph for ArrayResourceSystem.cpp:



Macros

- #define __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_CPP__ 1

5.1.1 Macro Definition Documentation

5.1.1.1 #define __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_CPP__ 1

Arduino - A simple resource implementation.

SimpleArrayResourceIO.cpp

This is the Resource IO representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file ArrayResourceSystem.cpp.

5.2 ArrayResourceSystem.cpp

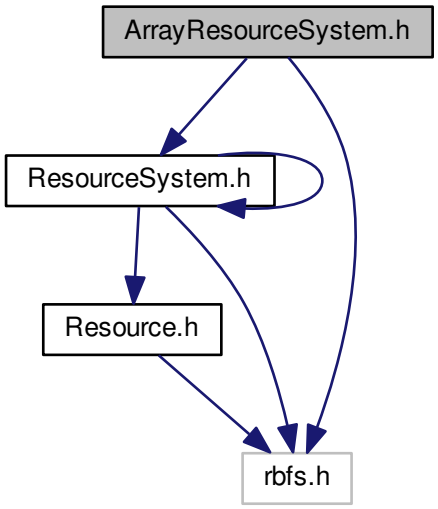
```
00001
00011 #ifndef __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_CPP__
00012 #define __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_CPP__ 1
00013
00014 #include "ArrayResourceSystem.h"
00015
00016 ArrayResourceSystem::ArrayResourceSystem(unsigned char* array,
```

```
      unsigned int size, rbfs_t *rbfs) : ResourceSystem(rbfs), array(array), size(size) {
00017 }
00018
00019 int ArrayResourceSystem::readBytes(unsigned int address, unsigned char* buf,
      unsigned int len) {
00020     unsigned int available = (size - address);
00021     if (available < 1) {
00022         return -1;
00023     }
00024     len = (len > available) ? available : len;
00025     for (unsigned int i = 0; i < len; i++) {
00026         buf[i] = array[address + i];
00027     }
00028     return len;
00029 }
00030
00031 void ArrayResourceSystem::writeBytes(unsigned int address, unsigned char*
      buf, unsigned int len) {
00032     for (unsigned int i = 0; i < len && (address + i) < size; i++) {
00033         array[address + i] = buf[i];
00034     }
00035 }
00036
00037 #endif  /* __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_CPP__ */
00038
```

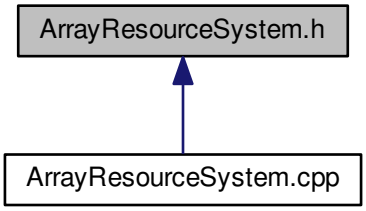
5.3 ArrayResourceSystem.h File Reference

```
#include <ResourceSystem.h>
#include <rbfs.h>
```

Include dependency graph for ArrayResourceSystem.h:



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



Classes

- class ArrayResourceSystem

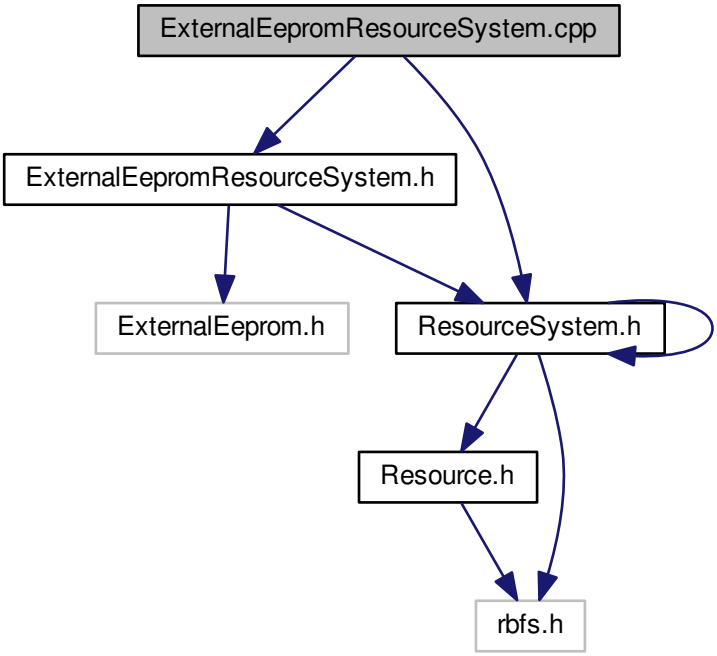
5.4 ArrayResourceSystem.h

```
00001
00011 #ifndef __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_H__
00012 #define __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_H__ 1
00013
00014 #include <ResourceSystem.h>
00015 #include <rbfs.h>
00016
00017 class ArrayResourceSystem : public ResourceSystem {
00018
00019 private:
00020     unsigned char* array;
00021     unsigned int size;
00022
00023 public:
00024
00025     ArrayResourceSystem(unsigned char* array, unsigned int size, rbfs_t *
      rbfs);
00026
00027 protected:
00028
00029     virtual int readBytes(unsigned int address, unsigned char* buf, unsigned int len);
00030
00031     virtual void writeBytes(unsigned int address, unsigned char* buf, unsigned int len);
00032 };
00033
00034 #endif  /* __ARDUINO_SIMPLE_ARRAY_RESOURCE_IO_H__ */
00035
```

5.5 ExternalEepromResourceSystem.cpp File Reference

```
#include "ExternalEepromResourceSystem.h"
#include <ResourceSystem.h>
```

Include dependency graph for ExternalEepromResourceSystem.cpp:



Macros

- #define __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_CPP__ 1

5.5.1 Macro Definition Documentation

5.5.1.1 #define __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_CPP__ 1

Arduino - A simple resource implementation.

SimpleExternalEepromResourceIO.cpp

This is the Resource IO representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file ExternalEepromResourceSystem.cpp.

5.6 ExternalEepromResourceSystem.cpp

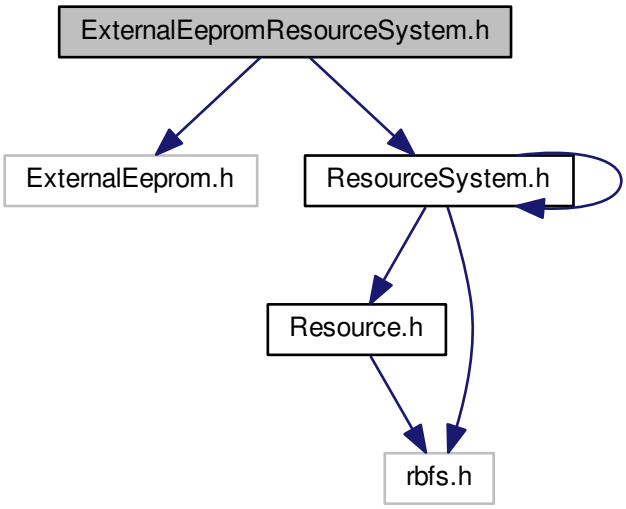
```
00001
00011 #ifndef __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_CPP__
00012 #define __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_CPP__ 1
00013
00014 #include "ExternalEepromResourceSystem.h"
00015 #include <ResourceSystem.h>
00016
```

```
00017 ExternalEepromResourceSystem::ExternalEepromResourceSystem
      (ExternalEeprom* eeprom, rbfs_t *rbfs) : ResourceSystem(rbfs), eeprom(eeprom) {
00018 }
00019
00020 int ExternalEepromResourceSystem::readBytes(unsigned int address,
      unsigned char* buf, int len) {
00021     return eeprom->readBytes(address, buf, len);
00022 }
00023
00024 void ExternalEepromResourceSystem::writeBytes(unsigned int address,
       unsigned char* buf, int len) {
00025     eeprom->writeBytes(address, buf, len);
00026 }
00027
00028 #endif  /* __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_CPP__ */
00029
```

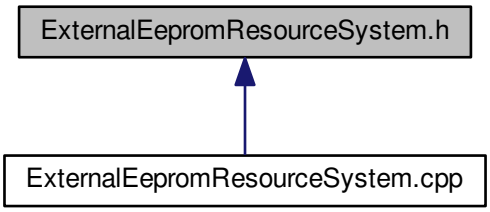
5.7 ExternalEepromResourceSystem.h File Reference

```
#include <ExternalEeprom.h>
#include <ResourceSystem.h>
```

Include dependency graph for ExternalEepromResourceSystem.h:



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



Classes

- class ExternalEepromResourceSystem

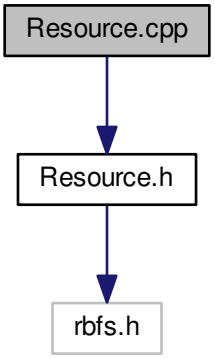
5.8 ExternalEepromResourceSystem.h

```
00001
00011 #ifndef __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_H__
00012 #define __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_H__ 1
00013
00014 #include <ExternalEeprom.h>
00015 #include <ResourceSystem.h>
00016
00017 class ExternalEepromResourceSystem: public
      ResourceSystem {
00018
00019 private:
00020     ExternalEeprom* eeprom;
00021
00022 public:
00023
00024     ExternalEepromResourceSystem(ExternalEeprom* eeprom, rbfs_t *
      rbfs);
00025
00026 protected:
00027
00028     virtual int readBytes(unsigned int address, unsigned char* buf, int len);
00029
00030     virtual void writeBytes(unsigned int address, unsigned char* buf, int len);
00031 };
00032
00033 #endif  /* __ARDUINO_SIMPLE_EXTERNAL_EEPROM_RESOURCE_IO_H__ */
00034
```

5.9 Resource.cpp File Reference

```
#include "Resource.h"
```

Include dependency graph for Resource.cpp:



Macros

- #define __ARDUINO_SIMPLE_RESOURCE_CPP__ 1

5.9.1 Macro Definition Documentation

5.9.1.1 #define __ARDUINO_SIMPLE_RESOURCE_CPP__ 1

Arduino - A simple resource implementation.

Resource.cpp

This is the Resource representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file Resource.cpp.

5.10 Resource.cpp

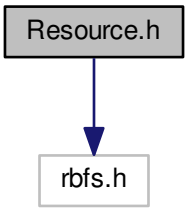
```
00001
00011 #ifndef __ARDUINO_SIMPLE_RESOURCE_CPP__
00012 #define __ARDUINO_SIMPLE_RESOURCE_CPP__ 1
00013
00014 #include "Resource.h"
00015
00016 Resource::Resource(rbfs_resource_code_t code, rbfs_t* rbfs) : code(code), rbfs(rbfs) {
00017     lastOperationResult = OPERATION_SUCCESS;
00018 }
00019
00020 bool Resource::open(OpenOptions options) {
00021     lastOperationResult = (ResourceOperationResult) rbfs_open(
      rbfs, code, &resource, (rbfs_open_resource_options_t) options);
00022     return (lastOperationResult == OPERATION_SUCCESS);
00023 }
00024
00025 bool Resource::close() {
00026     sync();
00027     lastOperationResult = (ResourceOperationResult) rbfs_close(
      rbfs, &resource);
00028     return (lastOperationResult == OPERATION_SUCCESS);
```

```
00029 }
00030
00031 void Resource::write(unsigned char b) {
00032     lastOperationResult = (ResourceOperationResult) rbfs_write(
      rbfs, &resource, b);
00033 }
00034
00035 void Resource::writeBytes(unsigned char* buf, int count) {
00036     lastOperationResult = OPERATION_SUCCESS;
00037     for (int i = 0; i < count && lastOperationResult ==
      OPERATION_SUCCESS; i++) {
00038         write(buf[i]);
00039     }
00040 }
00041
00042 int Resource::read() {
00043     if (eor()) {
00044         return -1;
00045     }
00046     return rbfs_read(rbfs, &resource);
00047 }
00048
00049 int Resource::readBytes(unsigned char* buf, int count) {
00050     int i, c;
00051     if (buf == (unsigned char*) 0) {
00052         return 0;
00053     }
00054     c = read();
00055     if (c == -1) {
00056         return -1;
00057     }
00058     buf[0] = c;
00059     for (i = 1; i < count; i++) {
00060         c = read();
00061         if (c == -1) {
00062             break;
00063         }
00064         buf[i] = c;
00065     }
00066     return i;
00067 }
00068
00069 bool Resource::seek(ResourceSeekOrigin origin, unsigned int offset) {
00070     lastOperationResult = (ResourceOperationResult) rbfs_seek(
      rbfs, &resource, (rbfs_seek_origin_t) origin, (rbfs_seek_int_t) offset);
00071     return (lastOperationResult == OPERATION_SUCCESS);
00072 }
00073
00074 bool Resource::truncate() {
00075     lastOperationResult = (ResourceOperationResult) rbfs_truncate
      (rbfs, &resource);
00076     return (lastOperationResult == OPERATION_SUCCESS);
00077 }
00078
00079 void Resource::sync() {
00080     rbfs_sync(rbfs, &resource);
00081     // TODO: flush();
00082 }
00083
00084 bool Resource::rewind() {
00085     lastOperationResult = (ResourceOperationResult) rbfs_rewind(
      rbfs, &resource);
00086     return (lastOperationResult == OPERATION_SUCCESS);
00087 }
00088
00089 void Resource::release() {
00090     sync();
00091     rbfs_release(rbfs, &resource);
00092 }
00093
00094 unsigned int Resource::size() {
00095     return (unsigned int) rbfs_size(&resource);
00096 }
00097
00098 unsigned int Resource::tell() {
00099     return (unsigned int) rbfs_tell(&resource);
00100 }
00101
00102 bool Resource::eor() {
00103     return (rbfs_eor(&resource) != 0);
00104 }
00105
00106 bool Resource::error() {
00107     return (rbfs_error(&resource) != 0);
00108 }
00109
00110 bool Resource::isReadOnly() {
```

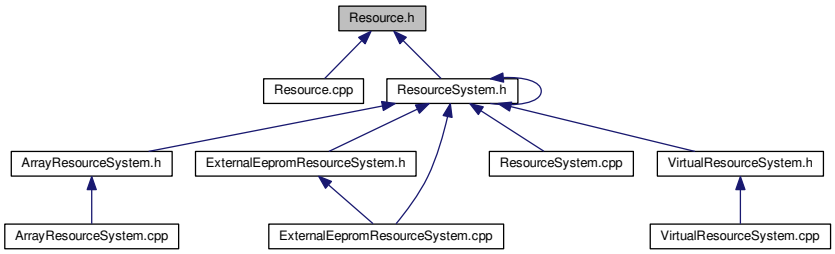
```
00111     return (rbfs->flags & RBFS_RESOURCE_FLAG_BIT_READ_ONLY) != 0;
00112 }
00113
00114 #endif /* __ARDUINO_SIMPLE_RESOURCE_CPP__ */
```

5.11 Resource.h File Reference

`#include <rbfs.h>`
Include dependency graph for Resource.h:



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



Classes

- class Resource

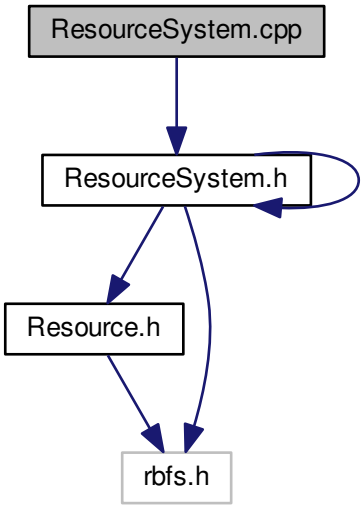
5.12 Resource.h

```
00001
00011 #ifndef __ARDUINO_SIMPLE_RESOURCE_H__
00012 #define __ARDUINO_SIMPLE_RESOURCE_H__ 1
00013
00014 #include <rbfs.h>
00015
00016 class Resource {
00017
00018 public:
00019
00020     enum ResourceOperationResult {
00021         OPERATION_SUCCESS = 0,
00022         OPERATION_ERROR_RESOURCE_OPENED = 1,
00023         OPERATION_ERROR_RESOURCE_CLOSED = 2,
00024         OPERATION_ERROR_RESOURCE_READ_ONLY = 3,
00025         OPERATION_ERROR_NO_SPACE_AVAILABLE = 4,
```

```
00026         OPERATION_ERROR_DRIVER_BUSY = 5,
00027         OPERATION_ERROR_SEEK_OUT_OF_BOUND = 6,
00028         OPERATION_ERROR_RESOURCE_DOES_NOT_ALLOCATED = 7,
00029         OPERATION_ERROR_DRIVER_NOT_MOUNTED = 8,
00030         OPERATION_ERROR_IO_ERROR = 9
00031     };
00032
00033     enum OpenOptions {
00034         OPEN_READ_WRITE = 0,
00035         OPEN_READ_ONLY = 1
00036     };
00037
00038     enum ResourceSeekOrigin {
00039         SEEK_ORIGIN_BEGIN = 0,
00040         SEEK_ORIGIN_CURRENT = 1
00041     };
00042
00043 private:
00044
00045     rbfs_resource_code_t code;
00046     rbfs_resource_t resource;
00047     rbfs_t* rbfs;
00048     ResourceOperationResult lastOperationResult;
00049
00050 public:
00051
00052     Resource(rbfs_resource_code_t code, rbfs_t* rbfs);
00053
00054     ResourceOperationResult getLastOperationResult() {
00055         return lastOperationResult;
00056     }
00057
00058     void setCode(int code) {
00059         this->code = (rbfs_resource_code_t) code;
00060     }
00061
00062     int getCode() {
00063         return (int) this->code;
00064     }
00065
00066     void setRbfs(rbfs_t* rbfs) {
00067         this->rbfs = rbfs;
00068     }
00069
00070     rbfs_t* getRbfs() {
00071         return this->rbfs;
00072     }
00073
00074     bool open(OpenOptions options);
00075
00076     bool close();
00077
00078     void write(unsigned char b);
00079
00080     void writeBytes(unsigned char* buf, int len);
00081
00082     int read();
00083
00084     int readBytes(unsigned char* buf, int len);
00085
00086     bool seek(ResourceSeekOrigin origin, unsigned int offset);
00087
00088     bool truncate();
00089
00090     void sync();
00091
00092     bool rewind();
00093
00094     void release();
00095
00096     unsigned int size();
00097
00098     unsigned int tell();
00099
00100     bool eor();
00101
00102     bool error();
00103
00104     bool isReadOnly();
00105 };
00106
00107 #endif // __ARDUINO_SIMPLE_RESOURCE_H__
```

5.13 ResourceSystem.cpp File Reference

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



Macros

- #define __ARDUINO_SIMPLE_RESOURCE_SYSTEM_CPP__ 1

5.13.1 Macro Definition Documentation

5.13.1.1 #define __ARDUINO_SIMPLE_RESOURCE_SYSTEM_CPP__ 1

Arduino - A simple resource implementation.

ResourceSystem.cpp

This is the Resource system itself.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file ResourceSystem.cpp.

5.14 ResourceSystem.cpp

```
00001
00011 #ifndef __ARDUINO_SIMPLE_RESOURCE_SYSTEM_CPP__
00012 #define __ARDUINO_SIMPLE_RESOURCE_SYSTEM_CPP__ 1
00013
00014 #include "ResourceSystem.h"
00015
00016 ResourceSystem::ResourceSystem(rbfs_t *rbfs)
00017         : rbfs(rbfs) {
00018     lastOperationResult = Resource::OPERATION_SUCCESS;
```

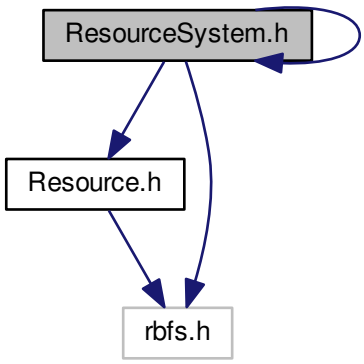
```
00019 }
00020
00021 bool ResourceSystem::format() {
00022     Resource::ResourceOperationResult o = (
      Resource::ResourceOperationResult) rbfs_format(
      rbfs);
00023     return (o == Resource::OPERATION_SUCCESS);
00024 }
00025
00026 bool ResourceSystem::mount(MountOptions options) {
00027     lastOperationResult = (Resource::ResourceOperationResult
      ) rbfs_mount(rbfs->driver, rbfs, (rbfs_mount_options_t) options);
00028     return (lastOperationResult ==
      Resource::OPERATION_SUCCESS);
00029 }
00030
00031 void ResourceSystem::flush() {
00032     if (wasCacheChanged) {
00033         writeBytes(cacheMemoryAddress, cache,
      validCacheSize);
00034     }
00035 }
00036
00037 bool ResourceSystem::umount() {
00038     flush();
00039     lastOperationResult = (Resource::ResourceOperationResult
      ) rbfs_umount(rbfs);
00040     return (lastOperationResult ==
      Resource::OPERATION_SUCCESS);
00041 }
00042
00043 Resource ResourceSystem::allocResource() {
00044     flush();
00045     Resource resource(RBFS_NULL_RESOURCE_CODE, rbfs);
00046     rbfs_resource_code_t code;
00047     code = rbfs_alloc(rbfs);
00048     if (code != RBFS_NULL_RESOURCE_CODE) {
00049         resource.setCode(code);
00050     }
00051     return resource;
00052 }
00053
00054 Resource ResourceSystem::loadResource(int code) {
00055     flush();
00056     Resource resource((rbfs_resource_code_t) code, rbfs);
00057     resource.setRbfs(rbfs);
00058     resource.setCode(code);
00059     return resource;
00060 }
00061
00062 unsigned int ResourceSystem::totalSpace() {
00063     return (unsigned int) rbfs_total_space(rbfs);
00064 }
00065
00066 unsigned int ResourceSystem::availableSpace() {
00067     return (unsigned int) rbfs_available_space(rbfs);
00068 }
00069
00070
00071 unsigned char ResourceSystem::read(unsigned char driver, unsigned int address) {
00072     checkCache(address);
00073     if (validCacheSize < 1) {
00074         lastOperationResult =
      Resource::OPERATION_ERROR_IO_ERROR;
00075         return 0;
00076     }
00077     return (uint8_t) cache[address - cacheMemoryAddress];
00078 }
00079
00080 void ResourceSystem::write(unsigned char driver, unsigned int address, unsigned char b
      ) {
00081     checkCache(address);
00082     cache[address - cacheMemoryAddress] = b;
00083     wasCacheChanged = true;
00084 }
00085
00086 void ResourceSystem::checkCache(uint16_t address) {
00087     if (!wasCacheInitialized || (address < cacheMemoryAddress || address >= (
      cacheMemoryAddress + validCacheSize))) {
00088         flush();
00089         validCacheSize = readBytes(address, cache,
      RESOURCE_SYSTEM_CACHE_SIZE);
00090         cacheMemoryAddress = address;
00091         wasCacheChanged = false;
00092         wasCacheInitialized = true;
00093         cacheMiss++;
00094     } else {
```

```
00095         cacheHit++;
00096     }
00097 }
00098
00099 #endif /* __ARDUINO_SIMPLE_RESOURCE_SYSTEM_CPP__ */
```

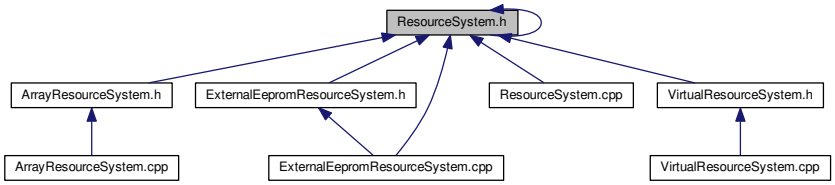
5.15 ResourceSystem.h File Reference

```
#include "ResourceSystem.h"
#include <Resource.h>
#include <rbfs.h>
```

Include dependency graph for ResourceSystem.h:



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



Classes

- class ResourceSystem

Macros

- #define RESOURCE_SYSTEM_CACHE_SIZE 32

5.15.1 Macro Definition Documentation

5.15.1.1 #define RESOURCE_SYSTEM_CACHE_SIZE 32

Arduino - A simple resource implementation.

ResourceSystem.h

This is the Resource system itself.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 18 of file ResourceSystem.h.

5.16 ResourceSystem.h

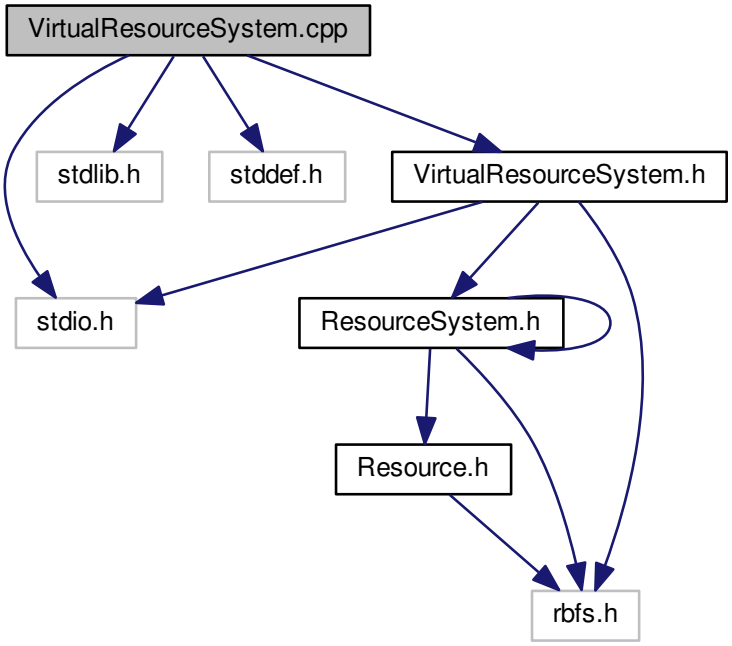
```
00001
00011 #ifndef __ARDUINO_SIMPLE_RESOURCE_SYSTEM_H__
00012 #define __ARDUINO_SIMPLE_RESOURCE_SYSTEM_H__ 1
00013
00014 #include "ResourceSystem.h"
00015 #include <Resource.h>
00016 #include <rbfs.h>
00017
00018 #define RESOURCE_SYSTEM_CACHE_SIZE 32
00019
00020 class ResourceSystem {
00021
00022     rbfs_t *rbfs;
00023     Resource::ResourceOperationResult
      lastOperationResult;
00024
00025     bool wasCacheChanged, wasCacheInitialized;
00026     unsigned int cacheMemoryAddress;
00027     unsigned char cache[RESOURCE_SYSTEM_CACHE_SIZE];
00028     unsigned int cacheMiss, cacheHit;
00029     unsigned int validCacheSize;
00030
00031 public:
00032
00033     enum MountOptions {
00034         MOUNT_READ_WRITE = 0,
00035         MOUNT_READ_ONLY = 1
00036     };
00037
00038     ResourceSystem(rbfs_t *rbfs);
00039
00040     Resource::ResourceOperationResult
      getLastOperationResult() {
00041         return lastOperationResult;
00042     }
00043
00044     virtual bool format();
00045
00046     virtual bool mount(MountOptions options);
00047
00048     virtual bool umount();
00049
00050     virtual Resource allocResource();
00051
00052     virtual Resource loadResource(int code);
00053
00054     virtual unsigned int totalSpace();
00055
00056     virtual unsigned int availableSpace();
00057
00058     virtual void flush();
00059
00060 protected:
00061
00062     virtual int readBytes(unsigned int address, unsigned char* buf, int len) = 0;
00063
00064     virtual void writeBytes(unsigned int address, unsigned char* buf, int len) = 0;
00065
00066 private:
00067
00068     unsigned char read(unsigned char driver, unsigned int address);
00069
00070     void write(unsigned char driver, unsigned int address, unsigned char b);
00071
00072     void checkCache(uint16_t address);
```

```
00073 };
00074
00075 #endif // __ARDUINO_SIMPLE_RESOURCE_SYSTEM_H__
```

5.17 VirtualResourceSystem.cpp File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include "VirtualResourceSystem.h"
```

Include dependency graph for VirtualResourceSystem.cpp:



Macros

- #define __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_CPP__ 1

5.17.1 Macro Definition Documentation

5.17.1.1 #define __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_CPP__ 1

Arduino - A simple resource implementation.

VirtualResourceSystem.cpp

This is the Resource IO representation.

Author

Dalmir da Silva dalmirdasilva@gmail.com

Definition at line 12 of file VirtualResourceSystem.cpp.

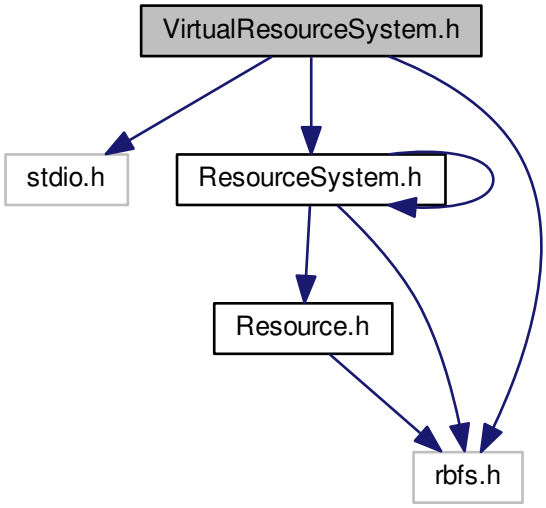
5.18 VirtualResourceSystem.cpp

```
00001
00011 #ifndef __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_CPP__
00012 #define __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_CPP__ 1
00013
00014 #include <stdio.h>
00015 #include <stdlib.h>
00016 #include <stddef.h>
00017 #include "VirtualResourceSystem.h"
00018
00019 VirtualResourceSystem::VirtualResourceSystem(srbfs_t *srbfs,
      char *fileName) : ResourceSystem(srbfs), fileName(fileName) {
00020     open();
00021 }
00022
00023 bool VirtualResourceSystem::open() {
00024     fp = fopen(fileName, "rb+");
00025     if (fp == NULL) {
00026         printf("Error when opening file: %s.\n", fileName);
00027         exit(1);
00028     }
00029     return true;
00030 }
00031
00032 void VirtualResourceSystem::flush() {
00033     SimpleResourceIO::flush();
00034     fflush(fp);
00035 }
00036
00037 void VirtualResourceSystem::close() {
00038     SimpleResourceIO::close();
00039     fclose(fp);
00040 }
00041
00042 int VirtualResourceSystem::readBytes(unsigned int address, unsigned char*
      buf, int len) {
00043     fseek(fp, address, 0);
00044     return (int) fread(buf, sizeof(unsigned char), len, fp);
00045 }
00046
00047 void VirtualResourceSystem::writeBytes(unsigned int address, unsigned char
      * buf, int len) {
00048     fseek(fp, address, 0);
00049     fwrite(buf, sizeof(unsigned char), len, fp);
00050 }
00051
00052 #endif  /* __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_CPP__ */
00053
```

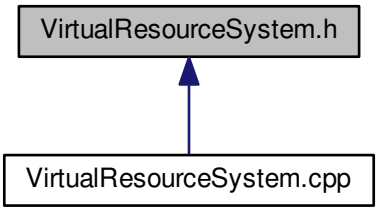
5.19 VirtualResourceSystem.h File Reference

```
#include <stdio.h>
#include <ResourceSystem.h>
#include <rbfs.h>
```

Include dependency graph for VirtualResourceSystem.h:



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



Classes

- class VirtualResourceSystem

5.20 VirtualResourceSystem.h

```
00001
00011 #ifndef __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_H__
00012 #define __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_H__ 1
00013
00014 #include <stdio.h>
00015 #include <ResourceSystem.h>
00016 #include <rbfs.h>
00017
00018 class VirtualResourceSystem : public ResourceSystem {
00019 private:
```

```
00020     char *fileName;
00021     FILE *fp;
00022 public:
00023
00024     VirtualResourceSystem(rbfs_t* rbfs, char *fileName);
00025
00026     virtual bool open();
00027
00028     virtual void flush();
00029
00030     virtual void close();
00031
00032 protected:
00033
00034     virtual int readBytes(unsigned int address, unsigned char* buf, int len);
00035
00036     virtual void writeBytes(unsigned int address, unsigned char* buf, int len);
00037 };
00038
00039 #endif  /* __ARDUINO_SIMPLE_VIRTUAL_RESOURCE_IO_H__ */
00040
```

Index