

Arduino Gyroscope Driver

Generated by Doxygen 1.8.9.1

Tue Aug 18 2015 22:52:17

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	1
2.1	Class List	2
3	File Index	2
3.1	File List	2
4	Class Documentation	2
4.1	ArrayByteBuffer Class Reference	2
4.1.1	Detailed Description	4
4.1.2	Constructor & Destructor Documentation	4
4.1.3	Member Function Documentation	4
4.1.4	Member Data Documentation	4
4.2	Buffer Class Reference	5
4.2.1	Detailed Description	6
4.2.2	Constructor & Destructor Documentation	6
4.2.3	Member Function Documentation	6
4.2.4	Member Data Documentation	7
4.3	ByteBuffer Class Reference	7
4.3.1	Detailed Description	9
4.3.2	Constructor & Destructor Documentation	9
4.3.3	Member Function Documentation	9
4.4	ExternalEepromByteBuffer Class Reference	10
4.4.1	Detailed Description	11
4.4.2	Constructor & Destructor Documentation	11
4.4.3	Member Function Documentation	11
4.4.4	Member Data Documentation	12
4.5	ResourceByteBuffer Class Reference	12
4.5.1	Detailed Description	13
4.5.2	Constructor & Destructor Documentation	13
4.5.3	Member Function Documentation	13
4.5.4	Member Data Documentation	14
5	File Documentation	14
5.1	ArrayByteBuffer.cpp File Reference	14
5.1.1	Macro Definition Documentation	15
5.2	ArrayByteBuffer.cpp	16
5.3	ArrayByteBuffer.h File Reference	16

5.4	ArrayByteBuffer.h	17
5.5	Buffer.cpp File Reference	18
5.5.1	Macro Definition Documentation	19
5.6	Buffer.cpp	19
5.7	Buffer.h File Reference	20
5.8	Buffer.h	20
5.9	ByteBuffer.cpp File Reference	21
5.9.1	Macro Definition Documentation	22
5.10	ByteBuffer.cpp	22
5.11	ByteBuffer.h File Reference	23
5.12	ByteBuffer.h	24
5.13	ExternalEepromByteBuffer.cpp File Reference	24
5.13.1	Macro Definition Documentation	25
5.14	ExternalEepromByteBuffer.cpp	25
5.15	ExternalEepromByteBuffer.h File Reference	26
5.16	ExternalEepromByteBuffer.h	27
5.17	main.cpp File Reference	27
5.17.1	Function Documentation	28
5.18	main.cpp	28
5.19	ResourceByteBuffer.cpp File Reference	30
5.19.1	Macro Definition Documentation	30
5.20	ResourceByteBuffer.cpp	30
5.21	ResourceByteBuffer.h File Reference	31
5.22	ResourceByteBuffer.h	32
	Index	33

1 Hierarchical Index

1.1 Class Hierarchy

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

Buffer	5
ByteBuffer	7
ArrayByteBuffer	2
ExternalEepromByteBuffer	10
ResourceByteBuffer	12

2 Class Index

2.1 Class List

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

ArrayByteBuffer	
Arduino NIO	2
Buffer	
Arduino NIO	5
ByteBuffer	
Arduino NIO	7
ExternalEepromByteBuffer	
Arduino NIO	10
ResourceByteBuffer	
Arduino NIO	12

3 File Index

3.1 File List

Here is a list of all files with brief descriptions:

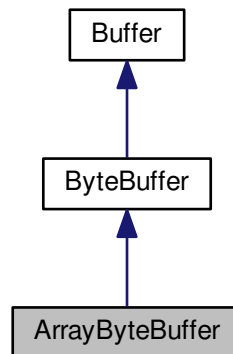
ArrayByteBuffer.cpp	14
ArrayByteBuffer.h	16
Buffer.cpp	18
Buffer.h	20
ByteBuffer.cpp	21
ByteBuffer.h	23
ExternalEepromByteBuffer.cpp	24
ExternalEepromByteBuffer.h	26
main.cpp	27
ResourceByteBuffer.cpp	30
ResourceByteBuffer.h	31

4 Class Documentation

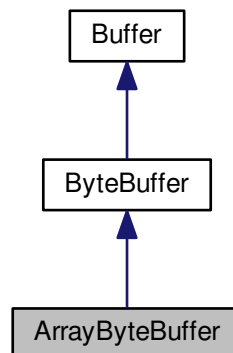
4.1 ArrayByteBuffer Class Reference

```
#include <ArrayByteBuffer.h>
```

Inheritance diagram for ArrayByteBuffer:



Collaboration diagram for ArrayByteBuffer:



Public Member Functions

- [ArrayByteBuffer](#) (unsigned char *[buf](#), unsigned int len)
- virtual unsigned char [get](#) ()
- virtual void [put](#) (unsigned char b)
- virtual unsigned char [get](#) (unsigned int index)
- virtual void [put](#) (unsigned int index, unsigned char b)
- virtual bool [isReadOnly](#) ()
- virtual bool [hasArray](#) ()
- virtual unsigned char * [getArray](#) ()

Protected Attributes

- unsigned char * [buf](#)

Additional Inherited Members

4.1.1 Detailed Description

Arduino NIO.

[ArrayByteBuffer.h](#)

Definition at line 12 of file [ArrayByteBuffer.h](#).

4.1.2 Constructor & Destructor Documentation

4.1.2.1 `ArrayByteBuffer::ArrayByteBuffer (unsigned char * buf, unsigned int len)`

Definition at line 12 of file [ArrayByteBuffer.cpp](#).

4.1.3 Member Function Documentation

4.1.3.1 `unsigned char ArrayByteBuffer::get () [virtual]`

Implements [ByteBuffer](#).

Definition at line 15 of file [ArrayByteBuffer.cpp](#).

4.1.3.2 `unsigned char ArrayByteBuffer::get (unsigned int index) [virtual]`

Implements [ByteBuffer](#).

Definition at line 28 of file [ArrayByteBuffer.cpp](#).

4.1.3.3 `unsigned char * ArrayByteBuffer::getArray () [virtual]`

Implements [Buffer](#).

Definition at line 49 of file [ArrayByteBuffer.cpp](#).

4.1.3.4 `bool ArrayByteBuffer::hasArray () [virtual]`

Implements [Buffer](#).

Definition at line 45 of file [ArrayByteBuffer.cpp](#).

4.1.3.5 `bool ArrayByteBuffer::isReadOnly () [virtual]`

Implements [Buffer](#).

Definition at line 41 of file [ArrayByteBuffer.cpp](#).

4.1.3.6 `void ArrayByteBuffer::put (unsigned char b) [virtual]`

Implements [ByteBuffer](#).

Definition at line 22 of file [ArrayByteBuffer.cpp](#).

4.1.3.7 `void ArrayByteBuffer::put (unsigned int index, unsigned char b) [virtual]`

Implements [ByteBuffer](#).

Definition at line 35 of file [ArrayByteBuffer.cpp](#).

4.1.4 Member Data Documentation

4.1.4.1 unsigned char* `ArrayByteBuffer::buf` `[protected]`

Definition at line 15 of file [ArrayByteBuffer.h](#).

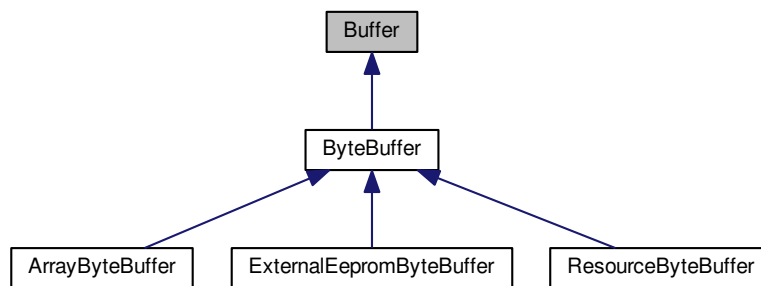
The documentation for this class was generated from the following files:

- [ArrayByteBuffer.h](#)
- [ArrayByteBuffer.cpp](#)

4.2 Buffer Class Reference

```
#include <Buffer.h>
```

Inheritance diagram for Buffer:



Public Member Functions

- unsigned int [getCapacity](#) ()
- unsigned int [getPosition](#) ()
- void [setPosition](#) (unsigned int [pos](#))
- unsigned int [getLimit](#) ()
- void [setLimit](#) (unsigned int [lim](#))
- void [mark](#) ()
- void [reset](#) ()
- void [clear](#) ()
- void [flip](#) ()
- void [rewind](#) ()
- unsigned int [getRemaining](#) ()
- bool [hasRemaining](#) ()
- virtual bool [isReadOnly](#) ()=0
- virtual bool [hasArray](#) ()=0
- virtual unsigned char * [getArray](#) ()=0

Protected Member Functions

- [Buffer](#) (unsigned int [mark](#), unsigned int [pos](#), unsigned int [lim](#), unsigned int [cap](#))

Protected Attributes

- bool [marked](#)
- unsigned int [markpos](#)
- unsigned int [pos](#)
- unsigned int [lim](#)
- unsigned int [cap](#)

4.2.1 Detailed Description

Arduino NIO.

[Buffer.h](#)

Definition at line 10 of file [Buffer.h](#).

4.2.2 Constructor & Destructor Documentation

4.2.2.1 `Buffer::Buffer (unsigned int mark, unsigned int pos, unsigned int lim, unsigned int cap)` `[protected]`

Definition at line 12 of file [Buffer.cpp](#).

4.2.3 Member Function Documentation

4.2.3.1 `void Buffer::clear ()`

Definition at line 67 of file [Buffer.cpp](#).

4.2.3.2 `void Buffer::flip ()`

Definition at line 73 of file [Buffer.cpp](#).

4.2.3.3 `virtual unsigned char* Buffer::getArray ()` `[pure virtual]`

Implemented in [ExternalEepromByteBuffer](#), [ResourceByteBuffer](#), and [ArrayByteBuffer](#).

4.2.3.4 `unsigned int Buffer::getCapacity ()`

Definition at line 24 of file [Buffer.cpp](#).

4.2.3.5 `unsigned int Buffer::getLimit ()`

Definition at line 39 of file [Buffer.cpp](#).

4.2.3.6 `unsigned int Buffer::getPosition ()`

Definition at line 28 of file [Buffer.cpp](#).

4.2.3.7 `unsigned int Buffer::getRemaining ()`

Definition at line 84 of file [Buffer.cpp](#).

4.2.3.8 `virtual bool Buffer::hasArray ()` `[pure virtual]`

Implemented in [ExternalEepromByteBuffer](#), [ResourceByteBuffer](#), and [ArrayByteBuffer](#).

4.2.3.9 `bool Buffer::hasRemaining ()`

Definition at line 88 of file [Buffer.cpp](#).

4.2.3.10 `virtual bool Buffer::isReadOnly () [pure virtual]`

Implemented in [ExternalEepromByteBuffer](#), [ResourceByteBuffer](#), and [ArrayByteBuffer](#).

4.2.3.11 `void Buffer::mark ()`

Definition at line 56 of file [Buffer.cpp](#).

4.2.3.12 `void Buffer::reset ()`

Definition at line 61 of file [Buffer.cpp](#).

4.2.3.13 `void Buffer::rewind ()`

Definition at line 79 of file [Buffer.cpp](#).

4.2.3.14 `void Buffer::setLimit (unsigned int lim)`

Definition at line 43 of file [Buffer.cpp](#).

4.2.3.15 `void Buffer::setPosition (unsigned int pos)`

Definition at line 32 of file [Buffer.cpp](#).

4.2.4 Member Data Documentation

4.2.4.1 `unsigned int Buffer::cap [protected]`

Definition at line 18 of file [Buffer.h](#).

4.2.4.2 `unsigned int Buffer::lim [protected]`

Definition at line 17 of file [Buffer.h](#).

4.2.4.3 `bool Buffer::marked [protected]`

Definition at line 14 of file [Buffer.h](#).

4.2.4.4 `unsigned int Buffer::markpos [protected]`

Definition at line 15 of file [Buffer.h](#).

4.2.4.5 `unsigned int Buffer::pos [protected]`

Definition at line 16 of file [Buffer.h](#).

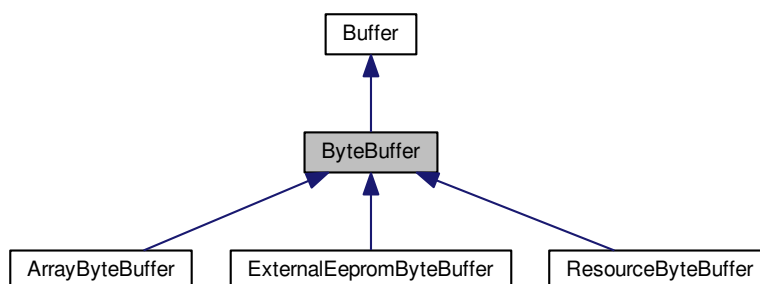
The documentation for this class was generated from the following files:

- [Buffer.h](#)
- [Buffer.cpp](#)

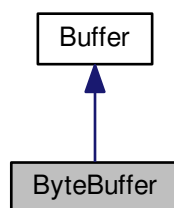
4.3 ByteBuffer Class Reference

```
#include <ByteBuffer.h>
```

Inheritance diagram for ByteBuffer:



Collaboration diagram for ByteBuffer:



Public Member Functions

- virtual unsigned char `get` ()=0
- virtual void `put` (unsigned char b)=0
- virtual unsigned char `get` (unsigned int index)=0
- virtual void `put` (unsigned int index, unsigned char b)=0
- virtual bool `get` (unsigned char *dst, int off, int len)
- bool `get` (unsigned char *dst, int len)
- virtual bool `put` (unsigned char *src, int off, int len)
- bool `put` (unsigned char *src, int len)
- virtual bool `put` (ByteBuffer *src)
- virtual bool `put` (ByteBuffer *src, int len)

Protected Member Functions

- `ByteBuffer` (unsigned int `mark`, unsigned int `pos`, unsigned int `lim`, unsigned int `cap`)

Additional Inherited Members

4.3.1 Detailed Description

Arduino NIO.

[ByteBuffer.h](#)

Definition at line 12 of file [ByteBuffer.h](#).

4.3.2 Constructor & Destructor Documentation

4.3.2.1 `ByteBuffer::ByteBuffer (unsigned int mark, unsigned int pos, unsigned int lim, unsigned int cap)` [protected]

Definition at line 12 of file [ByteBuffer.cpp](#).

4.3.3 Member Function Documentation

4.3.3.1 `virtual unsigned char ByteBuffer::get ()` [pure virtual]

Implemented in [ExternalEepromByteBuffer](#), [ResourceByteBuffer](#), and [ArrayByteBuffer](#).

4.3.3.2 `virtual unsigned char ByteBuffer::get (unsigned int index)` [pure virtual]

Implemented in [ExternalEepromByteBuffer](#), [ResourceByteBuffer](#), and [ArrayByteBuffer](#).

4.3.3.3 `bool ByteBuffer::get (unsigned char * dst, int off, int len)` [virtual]

Definition at line 15 of file [ByteBuffer.cpp](#).

4.3.3.4 `bool ByteBuffer::get (unsigned char * dst, int len)`

Definition at line 26 of file [ByteBuffer.cpp](#).

4.3.3.5 `virtual void ByteBuffer::put (unsigned char b)` [pure virtual]

Implemented in [ExternalEepromByteBuffer](#), [ResourceByteBuffer](#), and [ArrayByteBuffer](#).

4.3.3.6 `virtual void ByteBuffer::put (unsigned int index, unsigned char b)` [pure virtual]

Implemented in [ExternalEepromByteBuffer](#), [ResourceByteBuffer](#), and [ArrayByteBuffer](#).

4.3.3.7 `bool ByteBuffer::put (unsigned char * src, int off, int len)` [virtual]

Definition at line 30 of file [ByteBuffer.cpp](#).

4.3.3.8 `bool ByteBuffer::put (unsigned char * src, int len)`

Definition at line 41 of file [ByteBuffer.cpp](#).

4.3.3.9 `bool ByteBuffer::put (ByteBuffer * src)` [virtual]

Definition at line 45 of file [ByteBuffer.cpp](#).

4.3.3.10 `bool ByteBuffer::put (ByteBuffer * src, int len)` [virtual]

Definition at line 49 of file [ByteBuffer.cpp](#).

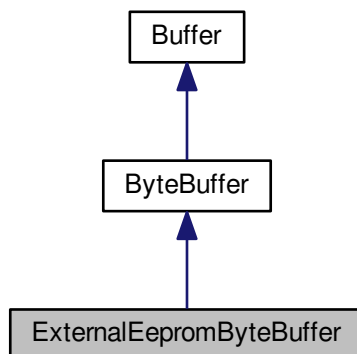
The documentation for this class was generated from the following files:

- [ByteBuffer.h](#)
- [ByteBuffer.cpp](#)

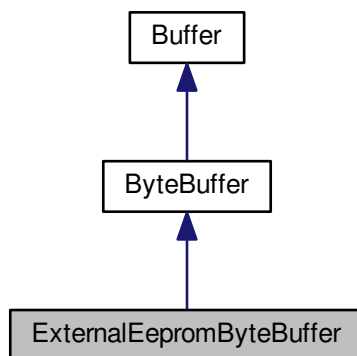
4.4 ExternalEepromByteBuffer Class Reference

```
#include <ExternalEepromByteBuffer.h>
```

Inheritance diagram for ExternalEepromByteBuffer:



Collaboration diagram for ExternalEepromByteBuffer:



Public Member Functions

- [ExternalEepromByteBuffer](#) (ExternalEeprom *externalEeprom)
- virtual unsigned char [get](#) ()
- virtual void [put](#) (unsigned char b)
- virtual unsigned char [get](#) (unsigned int index)
- virtual void [put](#) (unsigned int index, unsigned char b)
- virtual bool [isReadOnly](#) ()
- virtual bool [hasArray](#) ()
- virtual unsigned char * [getArray](#) ()

Protected Attributes

- ExternalEeprom * [externalEeprom](#)

Additional Inherited Members

4.4.1 Detailed Description

Arduino NIO.

[ExternalEepromByteBuffer.h](#)

Definition at line 13 of file [ExternalEepromByteBuffer.h](#).

4.4.2 Constructor & Destructor Documentation

4.4.2.1 ExternalEepromByteBuffer::ExternalEepromByteBuffer (ExternalEeprom * *externalEeprom*)

Definition at line 12 of file [ExternalEepromByteBuffer.cpp](#).

4.4.3 Member Function Documentation

4.4.3.1 unsigned char ExternalEepromByteBuffer::get () [virtual]

Implements [ByteBuffer](#).

Definition at line 15 of file [ExternalEepromByteBuffer.cpp](#).

4.4.3.2 unsigned char ExternalEepromByteBuffer::get (unsigned int *index*) [virtual]

Implements [ByteBuffer](#).

Definition at line 28 of file [ExternalEepromByteBuffer.cpp](#).

4.4.3.3 unsigned char * ExternalEepromByteBuffer::getArray () [virtual]

Implements [Buffer](#).

Definition at line 49 of file [ExternalEepromByteBuffer.cpp](#).

4.4.3.4 bool ExternalEepromByteBuffer::hasArray () [virtual]

Implements [Buffer](#).

Definition at line 45 of file [ExternalEepromByteBuffer.cpp](#).

4.4.3.5 bool ExternalEepromByteBuffer::isReadOnly () [virtual]

Implements [Buffer](#).

Definition at line 41 of file [ExternalEepromByteBuffer.cpp](#).

4.4.3.6 void ExternalEepromByteBuffer::put (unsigned char *b*) [virtual]

Implements [ByteBuffer](#).

Definition at line 22 of file [ExternalEepromByteBuffer.cpp](#).

4.4.3.7 void ExternalEepromByteBuffer::put (unsigned int *index*, unsigned char *b*) [virtual]

Implements [ByteBuffer](#).

Definition at line 35 of file [ExternalEepromByteBuffer.cpp](#).

4.4.4 Member Data Documentation

4.4.4.1 ExternalEeprom* ExternalEepromByteBuffer::externalEeprom [protected]

Definition at line 16 of file [ExternalEepromByteBuffer.h](#).

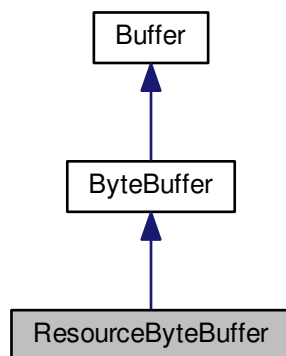
The documentation for this class was generated from the following files:

- [ExternalEepromByteBuffer.h](#)
- [ExternalEepromByteBuffer.cpp](#)

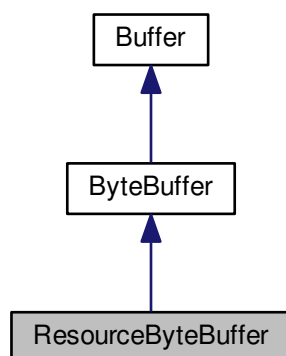
4.5 ResourceByteBuffer Class Reference

```
#include <ResourceByteBuffer.h>
```

Inheritance diagram for ResourceByteBuffer:



Collaboration diagram for ResourceByteBuffer:



Public Member Functions

- [ResourceByteBuffer](#) (Resource *[resource](#), unsigned int len)
- virtual unsigned char [get](#) ()
- virtual void [put](#) (unsigned char b)
- virtual unsigned char [get](#) (unsigned int index)
- virtual void [put](#) (unsigned int index, unsigned char b)
- virtual bool [isReadOnly](#) ()
- virtual bool [hasArray](#) ()
- virtual unsigned char * [getArray](#) ()

Protected Attributes

- Resource * [resource](#)

Additional Inherited Members

4.5.1 Detailed Description

Arduino NIO.

[ResourceByteBuffer.h](#)

Definition at line 13 of file [ResourceByteBuffer.h](#).

4.5.2 Constructor & Destructor Documentation

4.5.2.1 ResourceByteBuffer::ResourceByteBuffer (Resource * *resource*, unsigned int *len*)

Definition at line 12 of file [ResourceByteBuffer.cpp](#).

4.5.3 Member Function Documentation

4.5.3.1 unsigned char ResourceByteBuffer::get () [virtual]

Implements [ByteBuffer](#).

Definition at line 22 of file [ResourceByteBuffer.cpp](#).

4.5.3.2 unsigned char ResourceByteBuffer::get (unsigned int *index*) [virtual]

Implements [ByteBuffer](#).

Definition at line 37 of file [ResourceByteBuffer.cpp](#).

4.5.3.3 unsigned char * ResourceByteBuffer::getArray () [virtual]

Implements [Buffer](#).

Definition at line 64 of file [ResourceByteBuffer.cpp](#).

4.5.3.4 bool ResourceByteBuffer::hasArray () [virtual]

Implements [Buffer](#).

Definition at line 60 of file [ResourceByteBuffer.cpp](#).

4.5.3.5 `bool ResourceByteBuffer::isReadOnly ()` [virtual]

Implements [Buffer](#).

Definition at line 56 of file [ResourceByteBuffer.cpp](#).

4.5.3.6 `void ResourceByteBuffer::put (unsigned char b)` [virtual]

Implements [ByteBuffer](#).

Definition at line 30 of file [ResourceByteBuffer.cpp](#).

4.5.3.7 `void ResourceByteBuffer::put (unsigned int index, unsigned char b)` [virtual]

Implements [ByteBuffer](#).

Definition at line 48 of file [ResourceByteBuffer.cpp](#).

4.5.4 Member Data Documentation

4.5.4.1 `Resource* ResourceByteBuffer::resource` [protected]

Definition at line 16 of file [ResourceByteBuffer.h](#).

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

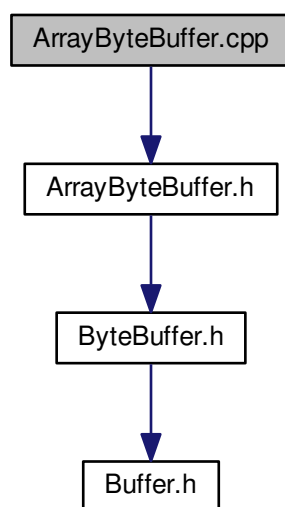
- [ResourceByteBuffer.h](#)
- [ResourceByteBuffer.cpp](#)

5 File Documentation

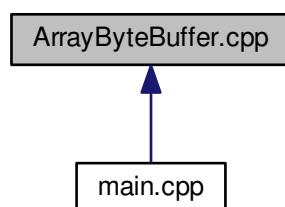
5.1 [ArrayByteBuffer.cpp](#) File Reference

```
#include "ArrayByteBuffer.h"
```


Include dependency graph for ArrayByteBuffer.cpp:



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



Macros

- `#define __ARDUINO_NIO_ARRAY_BYTE_BUFFER_CPP__ 1`

5.1.1 Macro Definition Documentation

5.1.1.1 `#define __ARDUINO_NIO_ARRAY_BYTE_BUFFER_CPP__ 1`

Arduino NIO.

[ArrayByteBuffer.cpp](#)

Definition at line 8 of file [ArrayByteBuffer.cpp](#).

5.2 ArrayByteBuffer.cpp

```

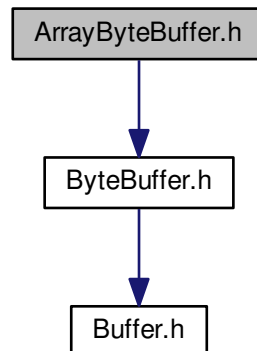
00001
00007 #ifndef __ARDUINO_NIO_ARRAY_BYTE_BUFFER_CPP__
00008 #define __ARDUINO_NIO_ARRAY_BYTE_BUFFER_CPP__ 1
00009
00010 #include "ArrayByteBuffer.h"
00011
00012 ArrayByteBuffer::ArrayByteBuffer(unsigned char* buf, unsigned int len) :
    ByteBuffer(0, 0, len, len), buf(buf) {
00013 }
00014
00015 unsigned char ArrayByteBuffer::get() {
00016     if (pos < lim) {
00017         return buf[pos++];
00018     }
00019     return 0;
00020 }
00021
00022 void ArrayByteBuffer::put(unsigned char b) {
00023     if (pos < lim) {
00024         buf[pos++] = b;
00025     }
00026 }
00027
00028 unsigned char ArrayByteBuffer::get(unsigned int index) {
00029     if (index < lim) {
00030         return buf[index];
00031     }
00032     return 0;
00033 }
00034
00035 void ArrayByteBuffer::put(unsigned int index, unsigned char b) {
00036     if (index < lim) {
00037         buf[index] = b;
00038     }
00039 }
00040
00041 bool ArrayByteBuffer::isReadOnly() {
00042     return false;
00043 }
00044
00045 bool ArrayByteBuffer::hasArray() {
00046     return true;
00047 }
00048
00049 unsigned char* ArrayByteBuffer::getArray() {
00050     return buf;
00051 }
00052
00053 #endif /* __ARDUINO_NIO_ARRAY_BYTE_BUFFER_CPP__ */

```

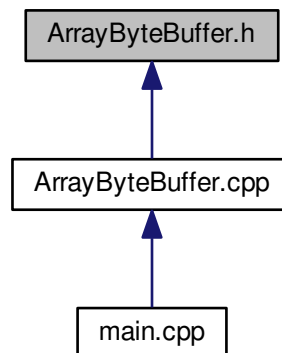
5.3 ArrayByteBuffer.h File Reference

```
#include <ByteBuffer.h>
```

Include dependency graph for ArrayByteBuffer.h:



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



Classes

- class [ArrayByteBuffer](#)

5.4 ArrayByteBuffer.h

```

00001
00007 #ifndef __ARDUINO_NIO_ARRAY_BYTE_BUFFER_H__
00008 #define __ARDUINO_NIO_ARRAY_BYTE_BUFFER_H__ 1
00009
00010 #include <ByteBuffer.h>
00011
00012 class ArrayByteBuffer : public ByteBuffer {
00013 protected:
00014
00015     unsigned char* buf;
  
```

```

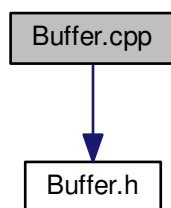
00016
00017 public:
00018
00019     ArrayByteBuffer(unsigned char* buf, unsigned int len);
00020
00021     virtual unsigned char get();
00022
00023     virtual void put(unsigned char b);
00024
00025     virtual unsigned char get(unsigned int index);
00026
00027     virtual void put(unsigned int index, unsigned char b);
00028
00029     virtual bool isReadOnly();
00030
00031     virtual bool hasArray();
00032
00033     virtual unsigned char* getArray();
00034 };
00035
00036 #endif /* __ARDUINO_NIO_ARRAY_BYTE_BUFFER_H__ */

```

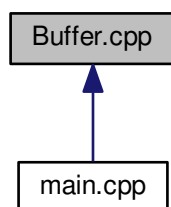
5.5 Buffer.cpp File Reference

```
#include "Buffer.h"
```

Include dependency graph for Buffer.cpp:



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



Macros

- `#define __ARDUINO_NIO_BUFFER_CPP__ 1`

5.5.1 Macro Definition Documentation

5.5.1.1 #define __ARDUINO_NIO_BUFFER_CPP__ 1

Arduino NIO.

[Buffer.cpp](#)

Definition at line 8 of file [Buffer.cpp](#).

5.6 Buffer.cpp

```

00001
00007 #ifndef __ARDUINO_NIO_BUFFER_CPP__
00008 #define __ARDUINO_NIO_BUFFER_CPP__ 1
00009
00010 #include "Buffer.h"
00011
00012 Buffer::Buffer(unsigned int mark, unsigned int pos, unsigned int lim, unsigned int cap) {
00013     this->cap = cap;
00014     this->markpos = mark;
00015     if (mark > 0) {
00016         this->marked = true;
00017     } else {
00018         this->marked = false;
00019     }
00020     setPosition(pos);
00021     setLimit(lim);
00022 }
00023
00024 unsigned int Buffer::getCapacity() {
00025     return cap;
00026 }
00027
00028 unsigned int Buffer::getPosition() {
00029     return pos;
00030 }
00031
00032 void Buffer::setPosition(unsigned int pos) {
00033     this->pos = pos;
00034     if (marked && markpos > pos) {
00035         marked = false;
00036     }
00037 }
00038
00039 unsigned int Buffer::getLimit() {
00040     return lim;
00041 }
00042
00043 void Buffer::setLimit(unsigned int lim) {
00044     if (lim > cap) {
00045         return;
00046     }
00047     this->lim = lim;
00048     if (pos > lim) {
00049         pos = lim;
00050     }
00051     if (marked && markpos > lim) {
00052         marked = false;
00053     }
00054 }
00055
00056 void Buffer::mark() {
00057     markpos = pos;
00058     marked = true;
00059 }
00060
00061 void Buffer::reset() {
00062     if (marked) {
00063         pos = markpos;
00064     }
00065 }
00066
00067 void Buffer::clear() {
00068     pos = 0;
00069     lim = cap;
00070     markpos = 0;
00071 }
00072
00073 void Buffer::flip() {
00074     lim = pos;
00075     pos = 0;

```

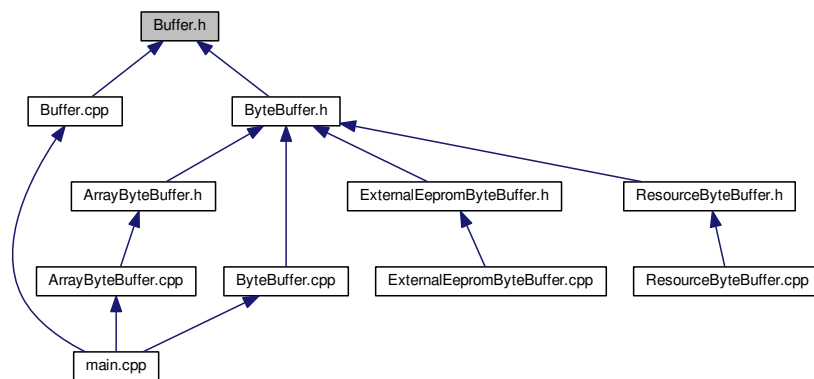
```

00076     marked = false;
00077 }
00078
00079 void Buffer::rewind() {
00080     pos = 0;
00081     marked = false;
00082 }
00083
00084 unsigned int Buffer::getRemaining() {
00085     return lim - pos;
00086 }
00087
00088 bool Buffer::hasRemaining() {
00089     return pos < lim;
00090 }
00091
00092 #endif /* __ARDUINO_NIO_BUFFER_CPP__ */

```

5.7 Buffer.h File Reference

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



Classes

- class [Buffer](#)

5.8 Buffer.h

```

00001
00007 #ifndef __ARDUINO_NIO_BUFFER_H__
00008 #define __ARDUINO_NIO_BUFFER_H__ 1
00009
00010 class Buffer {
00011
00012 protected:
00013
00014     bool marked;
00015     unsigned int markpos;
00016     unsigned int pos;
00017     unsigned int lim;
00018     unsigned int cap;
00019
00020     Buffer(unsigned int mark, unsigned int pos, unsigned int lim, unsigned int cap);
00021
00022 public:
00023
00024     unsigned int getCapacity();
00025
00026     unsigned int getPosition();
00027
00028     void setPosition(unsigned int pos);
00029

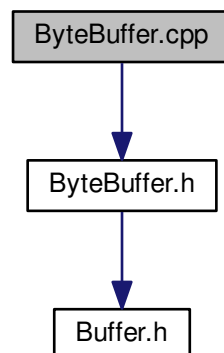
```

```
00030     unsigned int getLimit();
00031
00032     void setLimit(unsigned int lim);
00033
00034     void mark();
00035
00036     void reset();
00037
00038     void clear();
00039
00040     void flip();
00041
00042     void rewind();
00043
00044     unsigned int getRemaining();
00045
00046     bool hasRemaining();
00047
00048     virtual bool isReadOnly() = 0;
00049
00050     virtual bool hasArray() = 0;
00051
00052     virtual unsigned char* getArray() = 0;
00053 };
00054
00055 #endif /* __ARDUINO_NIO_BUFFER_H__ */
```

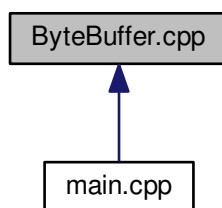
5.9 ByteBuffer.cpp File Reference

```
#include "ByteBuffer.h"
```

Include dependency graph for ByteBuffer.cpp:



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



Macros

- `#define __ARDUINO_NIO_BYTE_BUFFER_CPP__ 1`

5.9.1 Macro Definition Documentation

5.9.1.1 `#define __ARDUINO_NIO_BYTE_BUFFER_CPP__ 1`

Arduino NIO.

[ByteBuffer.cpp](#)

Definition at line 8 of file [ByteBuffer.cpp](#).

5.10 ByteBuffer.cpp

```

00001
00007 #ifndef __ARDUINO_NIO_BYTE_BUFFER_CPP__
00008 #define __ARDUINO_NIO_BYTE_BUFFER_CPP__ 1
00009
00010 #include "ByteBuffer.h"
00011
00012 ByteBuffer::ByteBuffer(unsigned int mark, unsigned int pos, unsigned int lim,
00013 unsigned int cap) : Buffer(mark, pos, lim, cap) {
00014 }
00015
00016 bool ByteBuffer::get(unsigned char* dst, int off, int len) {
00017     if (len > getRemaining()) {
00018         return false;
00019     }
00020     unsigned int end = off + len;
00021     for (int i = off; i < end; i++) {
00022         dst[i] = get();
00023     }
00024     return true;
00025 }
00026
00027 bool ByteBuffer::get(unsigned char* dst, int len) {
00028     return get(dst, 0, len);
00029 }
00030
00031 bool ByteBuffer::put(unsigned char* src, int off, int len) {
00032     if (len > getRemaining()) {
00033         return false;
00034     }
00035     unsigned int end = off + len;
00036     for (unsigned int i = off; i < end; i++) {
00037         put(src[i]);
00038     }
00039     return true;
00040 }
  
```



```

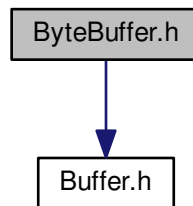
00041 bool ByteBuffer::put(unsigned char* src, int len) {
00042     return put(src, 0, len);
00043 }
00044
00045 bool ByteBuffer::put(ByteBuffer* src) {
00046     return put(src, src->getRemaining());
00047 }
00048
00049 bool ByteBuffer::put(ByteBuffer* src, int len) {
00050     if (src == this) {
00051         return false;
00052     }
00053     unsigned int n = src->getRemaining();
00054     len = (len > n) ? len : n;
00055     if (len > getRemaining()) {
00056         return false;
00057     }
00058     for (unsigned int i = 0; i < n; i++) {
00059         put(src->get());
00060     }
00061     return true;
00062 }
00063
00064 #endif /* __ARDUINO_NIO_BYTE_BUFFER_CPP__ */

```

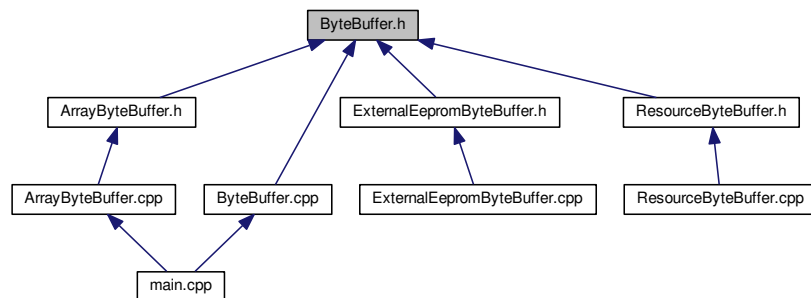
5.11 ByteBuffer.h File Reference

```
#include <Buffer.h>
```

Include dependency graph for ByteBuffer.h:



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



Classes

- class [ByteBuffer](#)

5.12 ByteBuffer.h

```

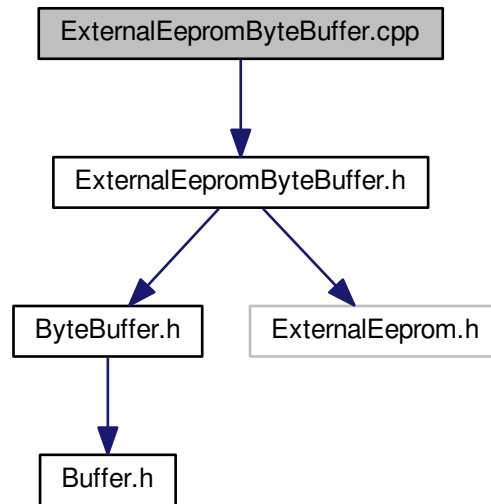
00001
00007 #ifndef __ARDUINO_NIO_BYTE_BUFFER_H__
00008 #define __ARDUINO_NIO_BYTE_BUFFER_H__ 1
00009
00010 #include <Buffer.h>
00011
00012 class ByteBuffer : public Buffer {
00013
00014 protected:
00015
00016     ByteBuffer(unsigned int mark, unsigned int pos, unsigned int
lim, unsigned int cap);
00017
00018 public:
00019
00020     virtual unsigned char get() = 0;
00021
00022     virtual void put(unsigned char b) = 0;
00023
00024     virtual unsigned char get(unsigned int index) = 0;
00025
00026     virtual void put(unsigned int index, unsigned char b) = 0;
00027
00028     virtual bool get(unsigned char* dst, int off, int len);
00029
00030     bool get(unsigned char* dst, int len);
00031
00032     virtual bool put(unsigned char* src, int off, int len);
00033
00034     bool put(unsigned char* src, int len);
00035
00036     virtual bool put(ByteBuffer* src);
00037
00038     virtual bool put(ByteBuffer* src, int len);
00039 };
00040
00041 #endif /* __ARDUINO_NIO_BYTE_BUFFER_H__ */
00042

```

5.13 ExternalEepromByteBuffer.cpp File Reference

```
#include "ExternalEepromByteBuffer.h"
```

Include dependency graph for ExternalEepromByteBuffer.cpp:



Macros

- `#define __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_CPP__ 1`

5.13.1 Macro Definition Documentation

5.13.1.1 `#define __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_CPP__ 1`

Arduino NIO.

[ExternalEepromByteBuffer.cpp](#)

Definition at line 8 of file [ExternalEepromByteBuffer.cpp](#).

5.14 ExternalEepromByteBuffer.cpp

```

00001
00007 #ifndef __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_CPP__
00008 #define __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_CPP__ 1
00009
00010 #include "ExternalEepromByteBuffer.h"
00011
00012 ExternalEepromByteBuffer::ExternalEepromByteBuffer(
00013     ExternalEeprom* externalEeprom) : ByteBuffer(0, 0, externalEeprom->getDeviceSize(), externalEeprom->
00014     getDeviceSize()), externalEeprom(internalEeprom) {
00015 }
00016
00017 unsigned char ExternalEepromByteBuffer::get() {
00018     if (pos < lim) {
00019         return externalEeprom->read(pos++);
00020     }
00021     return 0;
00022 }
00023
00024 void ExternalEepromByteBuffer::put(unsigned char b) {
00025     if (pos < lim) {
00026         externalEeprom->write(pos++, b);
00027     }
00028 }
  
```

```

00026 }
00027
00028 unsigned char ExternalEepromByteBuffer::get(unsigned int index) {
00029     if (index < lim) {
00030         return externalEeprom->read(index);
00031     }
00032     return 0;
00033 }
00034
00035 void ExternalEepromByteBuffer::put(unsigned int index, unsigned char b) {
00036     if (index < lim) {
00037         externalEeprom->write(index, b);
00038     }
00039 }
00040
00041 bool ExternalEepromByteBuffer::isReadOnly() {
00042     return false;
00043 }
00044
00045 bool ExternalEepromByteBuffer::hasArray() {
00046     return false;
00047 }
00048
00049 unsigned char* ExternalEepromByteBuffer::getArray() {
00050     return (unsigned char*) 0;
00051 }
00052
00053 #endif /* __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_CPP__ */

```

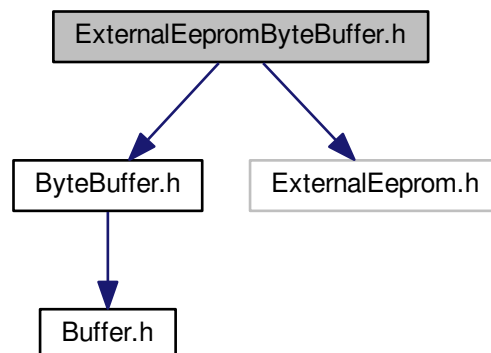
5.15 ExternalEepromByteBuffer.h File Reference

```

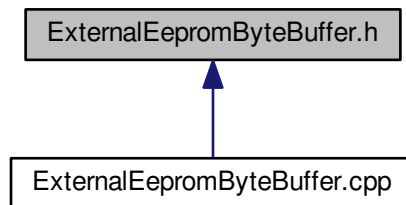
#include <ByteBuffer.h>
#include <ExternalEeprom.h>

```

Include dependency graph for ExternalEepromByteBuffer.h:



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



Classes

- class [ExternalEepromByteBuffer](#)

5.16 ExternalEepromByteBuffer.h

```

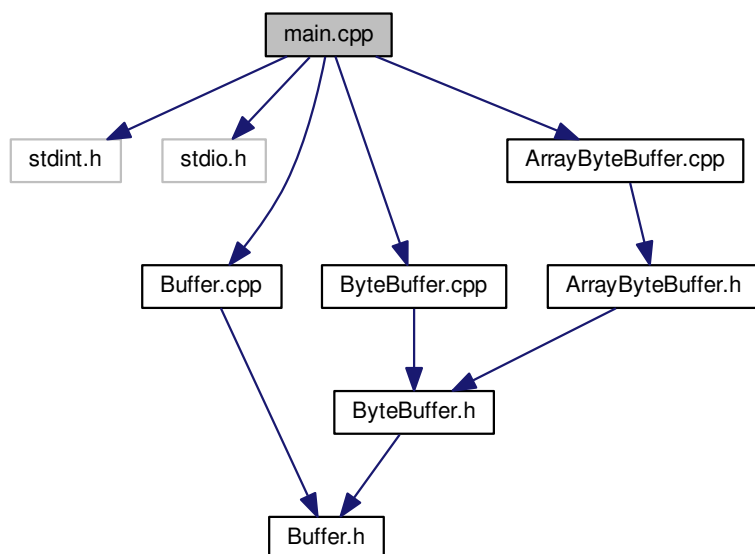
00001
00007 #ifndef __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_H__
00008 #define __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_H__ 1
00009
00010 #include <ByteBuffer.h>
00011 #include <ExternalEeprom.h>
00012
00013 class ExternalEepromByteBuffer : public ByteBuffer {
00014 protected:
00015
00016     ExternalEeprom* externalEeprom;
00017
00018 public:
00019
00020     ExternalEepromByteBuffer(ExternalEeprom* externalEeprom);
00021
00022     virtual unsigned char get();
00023
00024     virtual void put(unsigned char b);
00025
00026     virtual unsigned char get(unsigned int index);
00027
00028     virtual void put(unsigned int index, unsigned char b);
00029
00030     virtual bool isReadOnly();
00031
00032     virtual bool hasArray();
00033
00034     virtual unsigned char* getArray();
00035 };
00036
00037 #endif /* __ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_H__ */
  
```

5.17 main.cpp File Reference

```

#include <stdint.h>
#include <stdio.h>
#include <Buffer.cpp>
#include <ByteBuffer.cpp>
#include <ArrayByteBuffer.cpp>
  
```

Include dependency graph for main.cpp:



Functions

- void [testArrayByteBuffer](#) ()
- int [main](#) ()

5.17.1 Function Documentation

5.17.1.1 int main ()

Definition at line 63 of file [main.cpp](#).

5.17.1.2 void testArrayByteBuffer ()

Definition at line 8 of file [main.cpp](#).

5.18 main.cpp

```

00001 #include <stdint.h>
00002 #include <stdio.h>
00003
00004 #include <Buffer.cpp>
00005 #include <ByteBuffer.cpp>
00006 #include <ArrayByteBuffer.cpp>
00007
00008 void testArrayByteBuffer() {
00009     bool error = false;
00010     unsigned char byteArray[100];
00011     ArrayByteBuffer abb(byteArray, 100);
00012     abb.put(1);
00013     abb.put(2);
00014     abb.put(3);
00015     if (byteArray[0] != 1 || byteArray[1] != 2 || byteArray[2] != 3) {
00016         error = 1;
00017     }
00018     abb.clear();
00019     abb.put(0xaa);

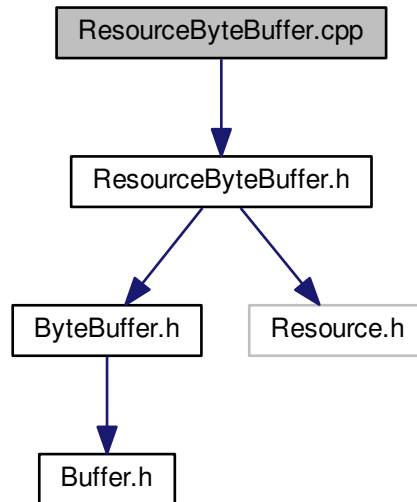
```

```
00020     abb.put(0xbb);
00021     if (byteArray[0] != 0xaa || byteArray[1] != 0xbb) {
00022         error = 1;
00023     }
00024     abb.mark();
00025     abb.put(0x00);
00026     abb.put(0x38);
00027     abb.put(0x94);
00028     abb.put(0x66);
00029     abb.reset();
00030     if (abb.get() != 0x00) {
00031         error = 1;
00032     }
00033     if (abb.get() != 0x38) {
00034         error = 1;
00035     }
00036     if (abb.get() != 0x94) {
00037         error = 1;
00038     }
00039     if (abb.get() != 0x66) {
00040         error = 1;
00041     }
00042     abb.reset();
00043     abb.put(0xf0);
00044     if (byteArray[2] != 0xf0) {
00045         error = 1;
00046     }
00047     abb.put(70, 0xfa);
00048     if (byteArray[70] != 0xfa) {
00049         error = 1;
00050     }
00051     abb.put(0x1a);
00052     if (byteArray[3] != 0x1a) {
00053         error = 1;
00054     }
00055     if (error) {
00056         printf("(F) ByteArrayOutputStream failed.\n");
00057     } else {
00058         printf("(*) ByteArrayOutputStream passed.\n");
00059     }
00060 }
00061
00062
00063 int main() {
00064     testArrayByteBuffer();
00065     return 0;
00066 }
```

5.19 ResourceByteBuffer.cpp File Reference

```
#include "ResourceByteBuffer.h"
```

Include dependency graph for ResourceByteBuffer.cpp:



Macros

- `#define __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_CPP__ 1`

5.19.1 Macro Definition Documentation

5.19.1.1 `#define __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_CPP__ 1`

Arduino NIO.

[ResourceByteBuffer.cpp](#)

Definition at line 8 of file [ResourceByteBuffer.cpp](#).

5.20 ResourceByteBuffer.cpp

```

00001
00007 #ifndef __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_CPP__
00008 #define __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_CPP__ 1
00009
00010 #include "ResourceByteBuffer.h"
00011
00012 ResourceByteBuffer::ResourceByteBuffer(Resource* resource, unsigned
int len) : ByteBuffer(0, 0, len, len), resource(resource) {
00013     if (resource->size() < len) {
00014         unsigned int needed = len - resource->size();
00015         resource->seek(Resource::SEEK_ORIGIN_BEGIN, resource->size());
00016         for (unsigned int i = 0; i < needed; i++) {
00017             resource->write(0x00);
00018         }
00019     }
00020 }
00021
00022 unsigned char ResourceByteBuffer::get() {

```



```

00023     if (pos < lim) {
00024         pos++;
00025         return resource->read();
00026     }
00027     return 0;
00028 }
00029
00030 void ResourceByteBuffer::put(unsigned char b) {
00031     if (pos < lim) {
00032         pos++;
00033         resource->write(b);
00034     }
00035 }
00036
00037 unsigned char ResourceByteBuffer::get(unsigned int index) {
00038     if (index < lim) {
00039         unsigned char b = 0;
00040         resource->seek(Resource::SEEK_ORIGIN_BEGIN, index);
00041         b = resource->read();
00042         resource->seek(Resource::SEEK_ORIGIN_BEGIN, pos);
00043         return b;
00044     }
00045     return 0;
00046 }
00047
00048 void ResourceByteBuffer::put(unsigned int index, unsigned char b) {
00049     if (index < lim) {
00050         resource->seek(Resource::SEEK_ORIGIN_BEGIN, index);
00051         resource->write(b);
00052         resource->seek(Resource::SEEK_ORIGIN_BEGIN, pos);
00053     }
00054 }
00055
00056 bool ResourceByteBuffer::isReadOnly() {
00057     return resource->isReadOnly();
00058 }
00059
00060 bool ResourceByteBuffer::hasArray() {
00061     return false;
00062 }
00063
00064 unsigned char* ResourceByteBuffer::getArray() {
00065     return (unsigned char *) 0;
00066 }
00067
00068 #endif /* __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_CPP__ */
00069

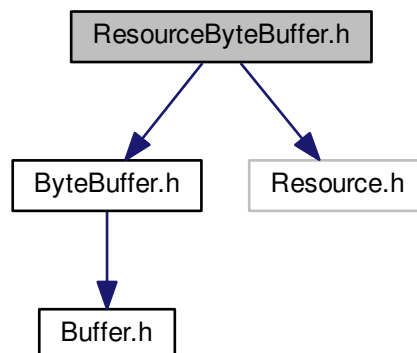
```

5.21 ResourceByteBuffer.h File Reference

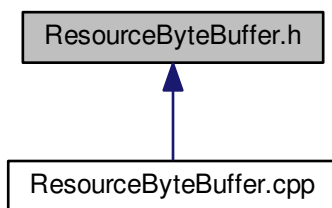
```
#include <ByteBuffer.h>
```

```
#include <Resource.h>
```

Include dependency graph for ResourceByteBuffer.h:



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



Classes

- class [ResourceByteBuffer](#)

5.22 ResourceByteBuffer.h

```

00001
00007 #ifndef __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_H__
00008 #define __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_H__ 1
00009
00010 #include <ByteBuffer.h>
00011 #include <Resource.h>
00012
00013 class ResourceByteBuffer : public ByteBuffer {
00014 protected:
00015
00016     Resource* resource;
00017
00018 public:
00019
00020     ResourceByteBuffer(Resource* resource, unsigned int len);
00021
00022     virtual unsigned char get();
00023
00024     virtual void put(unsigned char b);
00025
00026     virtual unsigned char get(unsigned int index);
00027
00028     virtual void put(unsigned int index, unsigned char b);
00029
00030     virtual bool isReadOnly();
00031
00032     virtual bool hasArray();
00033
00034     virtual unsigned char* getArray();
00035 };
00036
00037 #endif /* __ARDUINO_NIO_RESOURCE_BYTE_BUFFER_H__ */
  
```

Index

- `__ARDUINO_NIO_ARRAY_BYTE_BUFFER_CPP__`
 - ArrayByteBuffer.cpp, 15
 - `__ARDUINO_NIO_BUFFER_CPP__`
 - Buffer.cpp, 19
 - `__ARDUINO_NIO_BYTE_BUFFER_CPP__`
 - ByteBuffer.cpp, 22
 - `__ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_CPP__`
 - ExternalEepromByteBuffer.cpp, 25
 - `__ARDUINO_NIO_RESOURCE_BYTE_BUFFER_CPP__`
 - ResourceByteBuffer.cpp, 30
- ArrayByteBuffer, 2
 - ArrayByteBuffer, 4
 - buf, 4
 - get, 4
 - getArray, 4
 - hasArray, 4
 - isReadOnly, 4
 - put, 4
- ArrayByteBuffer.cpp, 14, 16
 - `__ARDUINO_NIO_ARRAY_BYTE_BUFFER_CPP__`, 15
- ArrayByteBuffer.h, 16, 17
- buf
 - ArrayByteBuffer, 4
- Buffer, 5
 - Buffer, 6
 - cap, 7
 - clear, 6
 - flip, 6
 - getArray, 6
 - getCapacity, 6
 - getLimit, 6
 - getPosition, 6
 - getRemaining, 6
 - hasArray, 6
 - hasRemaining, 6
 - isReadOnly, 6
 - lim, 7
 - mark, 7
 - marked, 7
 - markpos, 7
 - pos, 7
 - reset, 7
 - rewind, 7
 - setLimit, 7
 - setPosition, 7
- Buffer.cpp, 18, 19
 - `__ARDUINO_NIO_BUFFER_CPP__`, 19
- Buffer.h, 20
- ByteBuffer, 7
 - ByteBuffer, 9
 - get, 9
 - put, 9
- ByteBuffer.cpp, 21, 22
 - `__ARDUINO_NIO_BYTE_BUFFER_CPP__`, 22
- ByteBuffer.h, 23, 24
- cap
 - Buffer, 7
- clear
 - Buffer, 6
- externalEeprom
 - ExternalEepromByteBuffer, 12
- ExternalEepromByteBuffer, 10
 - externalEeprom, 12
 - ExternalEepromByteBuffer, 11
 - get, 11
 - getArray, 11
 - hasArray, 11
 - isReadOnly, 11
 - put, 11
- ExternalEepromByteBuffer.cpp, 24, 25
 - `__ARDUINO_NIO_EXTERNAL_EEPROM_BYTE_BUFFER_CPP__`, 25
- ExternalEepromByteBuffer.h, 26, 27
- flip
 - Buffer, 6
- get
 - ArrayByteBuffer, 4
 - ByteBuffer, 9
 - ExternalEepromByteBuffer, 11
 - ResourceByteBuffer, 13
- getArray
 - ArrayByteBuffer, 4
 - Buffer, 6
 - ExternalEepromByteBuffer, 11
 - ResourceByteBuffer, 13
- getCapacity
 - Buffer, 6
- getLimit
 - Buffer, 6
- getPosition
 - Buffer, 6
- getRemaining
 - Buffer, 6
- hasArray
 - ArrayByteBuffer, 4
 - Buffer, 6
 - ExternalEepromByteBuffer, 11
 - ResourceByteBuffer, 13
- hasRemaining
 - Buffer, 6
- isReadOnly
 - ArrayByteBuffer, 4

- Buffer, [6](#)
- ExternalEepromByteBuffer, [11](#)
- ResourceByteBuffer, [13](#)

lim

- Buffer, [7](#)

main

- main.cpp, [28](#)

main.cpp, [27](#), [28](#)

- main, [28](#)
- testArrayByteBuffer, [28](#)

mark

- Buffer, [7](#)

marked

- Buffer, [7](#)

markpos

- Buffer, [7](#)

pos

- Buffer, [7](#)

put

- ArrayByteBuffer, [4](#)
- ByteBuffer, [9](#)
- ExternalEepromByteBuffer, [11](#)
- ResourceByteBuffer, [14](#)

reset

- Buffer, [7](#)

resource

- ResourceByteBuffer, [14](#)

ResourceByteBuffer, [12](#)

- get, [13](#)
- getArray, [13](#)
- hasArray, [13](#)
- isReadOnly, [13](#)
- put, [14](#)
- resource, [14](#)
- ResourceByteBuffer, [13](#)

ResourceByteBuffer.cpp, [30](#)

- __ARDUINO_NIO_RESOURCE_BYTE_BUFFER__↵
R_CPP__, [30](#)

ResourceByteBuffer.h, [31](#), [32](#)

rewind

- Buffer, [7](#)

setLimit

- Buffer, [7](#)

setPosition

- Buffer, [7](#)

testArrayByteBuffer

- main.cpp, [28](#)