

## ZSamba Library

Generated by Doxygen 1.8.14

## Contents

<b>1</b>	<b>SAMBA</b>	<b>1</b>
<b>2</b>	<b>Data Structure Index</b>	<b>2</b>
2.1	Data Structures . . . . .	2
<b>3</b>	<b>File Index</b>	<b>2</b>
3.1	File List . . . . .	2
<b>4</b>	<b>Data Structure Documentation</b>	<b>3</b>
4.1	t_monitor_if Struct Reference . . . . .	3
4.1.1	Detailed Description . . . . .	3
4.1.2	Field Documentation . . . . .	3
<b>5</b>	<b>File Documentation</b>	<b>5</b>
5.1	driver_usb.cpp File Reference . . . . .	5
5.1.1	Macro Definition Documentation . . . . .	6
5.1.2	Function Documentation . . . . .	7
5.2	driver_usb.h File Reference . . . . .	11
5.2.1	Function Documentation . . . . .	12
5.2.2	Variable Documentation . . . . .	15
5.3	README.md File Reference . . . . .	15
5.4	sam_ba_cdc.cpp File Reference . . . . .	15
5.5	sam_ba_cdc.h File Reference . . . . .	16
5.6	sam_ba_monitor.cpp File Reference . . . . .	16
5.6.1	Macro Definition Documentation . . . . .	18
5.6.2	Function Documentation . . . . .	18
5.6.3	Variable Documentation . . . . .	20
5.7	sam_ba_monitor.h File Reference . . . . .	22
5.7.1	Macro Definition Documentation . . . . .	24
5.7.2	Function Documentation . . . . .	25
5.7.3	Variable Documentation . . . . .	27

5.8	<a href="#">sam_ba_uart.cpp File Reference</a>	27
5.8.1	<a href="#">Function Documentation</a>	28
5.8.2	<a href="#">Variable Documentation</a>	36
5.9	<a href="#">sam_ba_uart.h File Reference</a>	39
5.9.1	<a href="#">Macro Definition Documentation</a>	40
5.9.2	<a href="#">Function Documentation</a>	42
5.10	<a href="#">sam_ba_usb.cpp File Reference</a>	50
5.11	<a href="#">sam_ba_usb.h File Reference</a>	51
5.12	<a href="#">sam_ba_wire.cpp File Reference</a>	52
5.12.1	<a href="#">Function Documentation</a>	53
5.12.2	<a href="#">Variable Documentation</a>	61
5.13	<a href="#">sam_ba_wire.h File Reference</a>	63
5.13.1	<a href="#">Macro Definition Documentation</a>	65
5.13.2	<a href="#">Function Documentation</a>	66
5.14	<a href="#">util.cpp File Reference</a>	74
5.14.1	<a href="#">Macro Definition Documentation</a>	74
5.14.2	<a href="#">Function Documentation</a>	75
5.14.3	<a href="#">Variable Documentation</a>	77
5.15	<a href="#">util.h File Reference</a>	77
5.15.1	<a href="#">Macro Definition Documentation</a>	79
5.15.2	<a href="#">Function Documentation</a>	82
5.15.3	<a href="#">Variable Documentation</a>	85
	<a href="#">Index</a>	87

## 1 SAMBA

Arduino Library for samba boot loader compatible with ATSAMDx/ATSAMCx/ATSAMLx product family and bossac.

Based on an original work of zoubworld on zoubworld\_Arduino\*

## Usage

- `#include <sam_ba_monitor.h>`

documentation under construction

title

title2

title3

## 2 Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">t_monitor_if</a>	<a href="#">3</a>
------------------------------	-------------------

## 3 File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

<a href="#">driver_usb.cpp</a>	<a href="#">5</a>
<a href="#">driver_usb.h</a>	<a href="#">11</a>
<a href="#">sam_ba_cdc.cpp</a>	<a href="#">15</a>
<a href="#">sam_ba_cdc.h</a>	<a href="#">16</a>
<a href="#">sam_ba_monitor.cpp</a>	<a href="#">16</a>
<a href="#">sam_ba_monitor.h</a>	<a href="#">22</a>
<a href="#">sam_ba_uart.cpp</a>	<a href="#">27</a>
<a href="#">sam_ba_uart.h</a>	<a href="#">39</a>
<a href="#">sam_ba_usb.cpp</a>	<a href="#">50</a>
<a href="#">sam_ba_usb.h</a>	<a href="#">51</a>
<a href="#">sam_ba_wire.cpp</a>	<a href="#">52</a>
<a href="#">sam_ba_wire.h</a>	<a href="#">63</a>
<a href="#">util.cpp</a>	<a href="#">74</a>

util.h

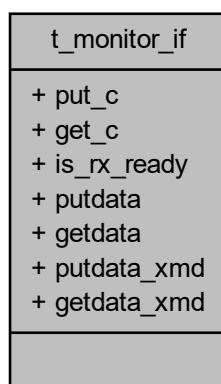
77

## 4 Data Structure Documentation

### 4.1 t\_monitor\_if Struct Reference

```
#include <sam_ba_monitor.h>
```

Collaboration diagram for t\_monitor\_if:



#### Data Fields

- int(\* [put\\_c](#) )(int value)
- int(\* [get\\_c](#) )(void)
- bool(\* [is\\_rx\\_ready](#) )(void)
- uint32\_t(\* [putdata](#) )(void const \*data, uint32\_t length)
- uint32\_t(\* [getdata](#) )(void \*data, uint32\_t length)
- uint32\_t(\* [putdata\\_xmd](#) )(void const \*data, uint32\_t length)
- uint32\_t(\* [getdata\\_xmd](#) )(void \*data, uint32\_t length)

#### 4.1.1 Detailed Description

Definition at line 56 of file sam\_ba\_monitor.h.

#### 4.1.2 Field Documentation

#### 4.1.2.1 `get_c`

```
int (* t_monitor_if::get_c) (void)
```

Definition at line 61 of file `sam_ba_monitor.h`.

#### 4.1.2.2 `getdata`

```
uint32_t (* t_monitor_if::getdata) (void *data, uint32_t length)
```

Definition at line 67 of file `sam_ba_monitor.h`.

#### 4.1.2.3 `getdata_xmd`

```
uint32_t (* t_monitor_if::getdata_xmd) (void *data, uint32_t length)
```

Definition at line 71 of file `sam_ba_monitor.h`.

#### 4.1.2.4 `is_rx_ready`

```
bool (* t_monitor_if::is_rx_ready) (void)
```

Definition at line 63 of file `sam_ba_monitor.h`.

#### 4.1.2.5 `put_c`

```
int (* t_monitor_if::put_c) (int value)
```

Definition at line 59 of file `sam_ba_monitor.h`.

#### 4.1.2.6 `putdata`

```
uint32_t (* t_monitor_if::putdata) (void const *data, uint32_t length)
```

Definition at line 65 of file `sam_ba_monitor.h`.

#### 4.1.2.7 `putdata_xmd`

```
uint32_t (* t_monitor_if::putdata_xmd) (void const *data, uint32_t length)
```

Definition at line 69 of file `sam_ba_monitor.h`.

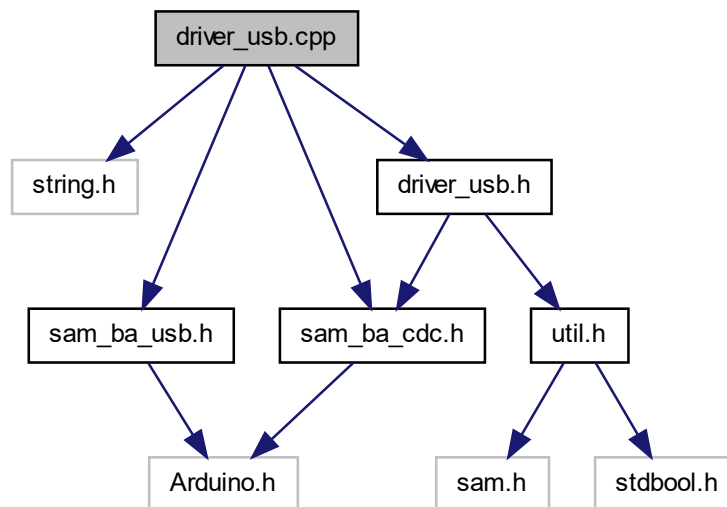
The documentation for this struct was generated from the following file:

- [sam\\_ba\\_monitor.h](#)

## 5 File Documentation

### 5.1 driver\_usb.cpp File Reference

```
#include <string.h>
#include "driver_usb.h"
#include "sam_ba_usb.h"
#include "sam_ba_cdc.h"
Include dependency graph for driver_usb.cpp:
```



#### Macros

- `#define USB_PAD_TRANSN_REG_POS` (6)
- `#define NVM_USB_PAD_TRANSN_POS` (45)
- `#define NVM_USB_PAD_TRANSN_SIZE` (5)
- `#define USB_PAD_TRANSP_REG_POS` (0)
- `#define NVM_USB_PAD_TRANSP_POS` (50)
- `#define NVM_USB_PAD_TRANSP_SIZE` (5)
- `#define USB_PAD_TRIM_REG_POS` (12)
- `#define NVM_USB_PAD_TRIM_POS` (55)
- `#define NVM_USB_PAD_TRIM_SIZE` (3)

#### Functions

- `__attribute__((__aligned__(4)))` UsbDeviceDescriptor `usb_endpoint_table`[MAX\_EP]
- `P_USB_CDC USB_Open` (P\_USB\_CDC pCdc, Usb \*pUsb)
- `void USB_Init` (void)
- `uint32_t USB_Write` (Usb \*pUsb, const char \*pData, uint32\_t length, uint8\_t ep\_num)
- `uint32_t USB_Read` (Usb \*pUsb, char \*pData, uint32\_t length)

- `uint32_t USB_Read_blocking` (Usb \*pUsb, char \*pData, `uint32_t length`)
- `uint8_t USB_IsConfigured` (P\_USB\_CDC pCdc)
- `void USB_SendStall` (Usb \*pUsb, bool direction\_in)
- `void USB_SendZip` (Usb \*pUsb)
- `void USB_SetAddress` (Usb \*pUsb, `uint16_t wValue`)
- `void USB_Configure` (Usb \*pUsb)

### 5.1.1 Macro Definition Documentation

#### 5.1.1.1 NVM\_USB\_PAD\_TRANSN\_POS

```
#define NVM_USB_PAD_TRANSN_POS (45)
```

Definition at line 26 of file `driver_usb.cpp`.

Referenced by `USB_Init()`.

#### 5.1.1.2 NVM\_USB\_PAD\_TRANSN\_SIZE

```
#define NVM_USB_PAD_TRANSN_SIZE (5)
```

Definition at line 27 of file `driver_usb.cpp`.

Referenced by `USB_Init()`.

#### 5.1.1.3 NVM\_USB\_PAD\_TRANSP\_POS

```
#define NVM_USB_PAD_TRANSP_POS (50)
```

Definition at line 29 of file `driver_usb.cpp`.

Referenced by `USB_Init()`.

#### 5.1.1.4 NVM\_USB\_PAD\_TRANSP\_SIZE

```
#define NVM_USB_PAD_TRANSP_SIZE (5)
```

Definition at line 30 of file `driver_usb.cpp`.

Referenced by `USB_Init()`.



#### 5.1.1.5 NVM\_USB\_PAD\_TRIM\_POS

```
#define NVM_USB_PAD_TRIM_POS (55)
```

Definition at line 32 of file driver\_usb.cpp.

Referenced by USB\_Init().

#### 5.1.1.6 NVM\_USB\_PAD\_TRIM\_SIZE

```
#define NVM_USB_PAD_TRIM_SIZE (3)
```

Definition at line 33 of file driver\_usb.cpp.

Referenced by USB\_Init().

#### 5.1.1.7 USB\_PAD\_TRANSN\_REG\_POS

```
#define USB_PAD_TRANSN_REG_POS (6)
```

Definition at line 25 of file driver\_usb.cpp.

Referenced by USB\_Init().

#### 5.1.1.8 USB\_PAD\_TRANSP\_REG\_POS

```
#define USB_PAD_TRANSP_REG_POS (0)
```

Definition at line 28 of file driver\_usb.cpp.

Referenced by USB\_Init().

#### 5.1.1.9 USB\_PAD\_TRIM\_REG\_POS

```
#define USB_PAD_TRIM_REG_POS (12)
```

Definition at line 31 of file driver\_usb.cpp.

Referenced by USB\_Init().

### 5.1.2 Function Documentation

### 5.1.2.1 `__attribute__()`

```
__attribute__ (
    (__aligned__(4)) )
```

### 5.1.2.2 `USB_Configure()`

```
void USB_Configure (
    Usb * pUsb )
```

Definition at line 349 of file `driver_usb.cpp`.

References `udd_ep_in_cache_buffer`, `udd_ep_out_cache_buffer`, and `usb_endpoint_table`.

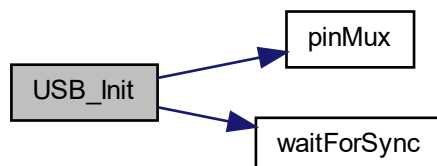
### 5.1.2.3 `USB_Init()`

```
void USB_Init (
    void )
```

Definition at line 71 of file `driver_usb.cpp`.

References `NVM_USB_PAD_TRANSN_POS`, `NVM_USB_PAD_TRANSN_SIZE`, `NVM_USB_PAD_TRANSP_POS`, `NVM_USB_PAD_TRANSP_SIZE`, `NVM_USB_PAD_TRIM_POS`, `NVM_USB_PAD_TRIM_SIZE`, `pinMux()`, `usb_endpoint_table`, `USB_PAD_TRANSN_REG_POS`, `USB_PAD_TRANSP_REG_POS`, `USB_PAD_TRIM_REG_POS`, and `waitForSync()`.

Here is the call graph for this function:



#### 5.1.2.4 USB\_IsConfigured()

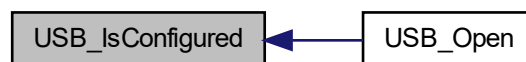
```
uint8_t USB_IsConfigured (
    P_USB_CDC pCdc )
```

Definition at line 262 of file driver\_usb.cpp.

References `udd_ep_in_cache_buffer`, `udd_ep_out_cache_buffer`, and `usb_endpoint_table`.

Referenced by `USB_Open()`.

Here is the caller graph for this function:



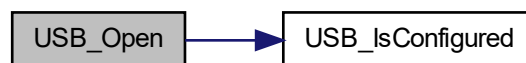
#### 5.1.2.5 USB\_Open()

```
P_USB_CDC USB_Open (
    P_USB_CDC pCdc,
    Usb * pUsb )
```

Definition at line 54 of file driver\_usb.cpp.

References `USB_IsConfigured()`.

Here is the call graph for this function:



#### 5.1.2.6 USB\_Read()

```
uint32_t USB_Read (
    Usb * pUsb,
    char * pData,
    uint32_t length )
```

Definition at line 199 of file driver\_usb.cpp.

#### 5.1.2.7 USB\_Read\_blocking()

```
uint32_t USB_Read_blocking (
    Usb * pUsb,
    char * pData,
    uint32_t length )
```

Definition at line 233 of file driver\_usb.cpp.

#### 5.1.2.8 USB\_SendStall()

```
void USB_SendStall (
    Usb * pUsb,
    bool direction_in )
```

Definition at line 308 of file driver\_usb.cpp.

#### 5.1.2.9 USB\_SendZlp()

```
void USB_SendZlp (
    Usb * pUsb )
```

Definition at line 326 of file driver\_usb.cpp.

References usb\_endpoint\_table.

#### 5.1.2.10 USB\_SetAddress()

```
void USB_SetAddress (
    Usb * pUsb,
    uint16_t wValue )
```

Definition at line 341 of file driver\_usb.cpp.

#### 5.1.2.11 USB\_Write()

```
uint32_t USB_Write (
    Usb * pUsb,
    const char * pData,
    uint32_t length,
    uint8_t ep_num )
```

Definition at line 155 of file driver\_usb.cpp.

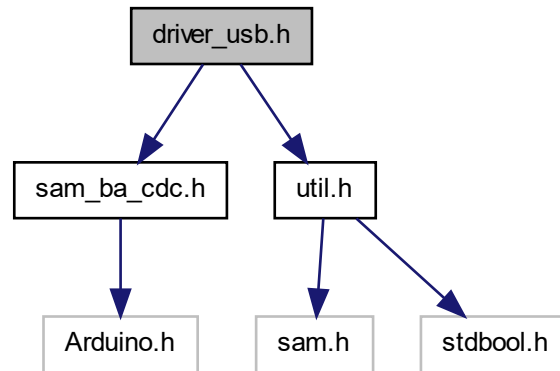
References length, udd\_ep\_in\_cache\_buffer, and usb\_endpoint\_table.

## 5.2 driver\_usb.h File Reference

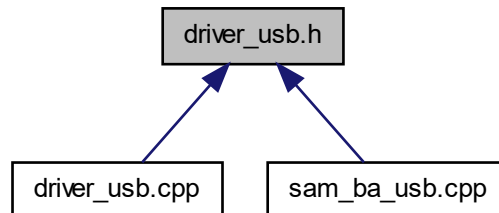
```
#include "sam_ba_cdc.h"
```

```
#include "util.h"
```

Include dependency graph for driver\_usb.h:



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



## Functions

- P\_USB\_CDC [USB\\_Open](#) (P\_USB\_CDC pCdc, Usb \*pUsb)
- void [USB\\_Init](#) (void)
- uint32\_t [USB\\_Write](#) (Usb \*pUsb, const char \*pData, uint32\_t [length](#), uint8\_t ep\_num)
- uint32\_t [USB\\_Read](#) (Usb \*pUsb, char \*pData, uint32\_t [length](#))
- uint32\_t [USB\\_Read\\_blocking](#) (Usb \*pUsb, char \*pData, uint32\_t [length](#))
- uint8\_t [USB\\_IsConfigured](#) (P\_USB\_CDC pCdc)
- void [USB\\_SendStall](#) (Usb \*pUsb, bool direction\_in)
- void [USB\\_SendZip](#) (Usb \*pUsb)
- void [USB\\_SetAddress](#) (Usb \*pUsb, uint16\_t wValue)
- void [USB\\_Configure](#) (Usb \*pUsb)

## Variables

- UsbDeviceDescriptor [usb\\_endpoint\\_table](#) [MAX\_EP]
- uint8\_t [udd\\_ep\\_out\\_cache\\_buffer](#) [2][64]
- uint8\_t [udd\\_ep\\_in\\_cache\\_buffer](#) [2][64]

## 5.2.1 Function Documentation

### 5.2.1.1 USB\_Configure()

```
void USB_Configure (
    Usb * pUsb )
```

Definition at line 349 of file driver\_usb.cpp.

References [udd\\_ep\\_in\\_cache\\_buffer](#), [udd\\_ep\\_out\\_cache\\_buffer](#), and [usb\\_endpoint\\_table](#).

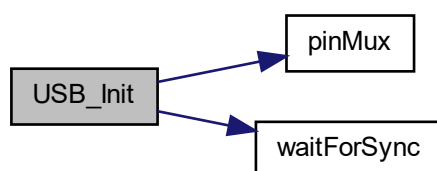
### 5.2.1.2 USB\_Init()

```
void USB_Init (
    void )
```

Definition at line 71 of file driver\_usb.cpp.

References [NVM\\_USB\\_PAD\\_TRANSN\\_POS](#), [NVM\\_USB\\_PAD\\_TRANSN\\_SIZE](#), [NVM\\_USB\\_PAD\\_TRANSP\\_POS](#), [NVM\\_USB\\_PAD\\_TRANSP\\_SIZE](#), [NVM\\_USB\\_PAD\\_TRIM\\_POS](#), [NVM\\_USB\\_PAD\\_TRIM\\_SIZE](#), [pinMux\(\)](#), [usb\\_endpoint\\_table](#), [USB\\_PAD\\_TRANSN\\_REG\\_POS](#), [USB\\_PAD\\_TRANSP\\_REG\\_POS](#), [USB\\_PAD\\_TRIM\\_REG\\_POS](#), and [waitForSync\(\)](#).

Here is the call graph for this function:



### 5.2.1.3 USB\_IsConfigured()

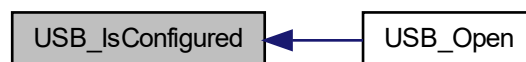
```
uint8_t USB_IsConfigured (
    P_USB_CDC pCdc )
```

Definition at line 262 of file driver\_usb.cpp.

References udd\_ep\_in\_cache\_buffer, udd\_ep\_out\_cache\_buffer, and usb\_endpoint\_table.

Referenced by USB\_Open().

Here is the caller graph for this function:



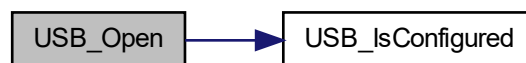
### 5.2.1.4 USB\_Open()

```
P_USB_CDC USB_Open (
    P_USB_CDC pCdc,
    Usb * pUsb )
```

Definition at line 54 of file driver\_usb.cpp.

References `USB_IsConfigured()`.

Here is the call graph for this function:



### 5.2.1.5 USB\_Read()

```
uint32_t USB_Read (
    Usb * pUsb,
    char * pData,
    uint32_t length )
```

Definition at line 199 of file driver\_usb.cpp.

#### 5.2.1.6 USB\_Read\_blocking()

```
uint32_t USB_Read_blocking (
    Usb * pUsb,
    char * pData,
    uint32_t length )
```

Definition at line 233 of file driver\_usb.cpp.

#### 5.2.1.7 USB\_SendStall()

```
void USB_SendStall (
    Usb * pUsb,
    bool direction_in )
```

Definition at line 308 of file driver\_usb.cpp.

#### 5.2.1.8 USB\_SendZlp()

```
void USB_SendZlp (
    Usb * pUsb )
```

Definition at line 326 of file driver\_usb.cpp.

References usb\_endpoint\_table.

#### 5.2.1.9 USB\_SetAddress()

```
void USB_SetAddress (
    Usb * pUsb,
    uint16_t wValue )
```

Definition at line 341 of file driver\_usb.cpp.

#### 5.2.1.10 USB\_Write()

```
uint32_t USB_Write (
    Usb * pUsb,
    const char * pData,
    uint32_t length,
    uint8_t ep_num )
```

Definition at line 155 of file driver\_usb.cpp.

References length, udd\_ep\_in\_cache\_buffer, and usb\_endpoint\_table.



## 5.2.2 Variable Documentation

### 5.2.2.1 udd\_ep\_in\_cache\_buffer

```
uint8_t udd_ep_in_cache_buffer[2][64]
```

Referenced by USB\_Configure(), USB\_IsConfigured(), and USB\_Write().

### 5.2.2.2 udd\_ep\_out\_cache\_buffer

```
uint8_t udd_ep_out_cache_buffer[2][64]
```

Referenced by USB\_Configure(), and USB\_IsConfigured().

### 5.2.2.3 usb\_endpoint\_table

```
UsbDeviceDescriptor usb_endpoint_table[MAX_EP]
```

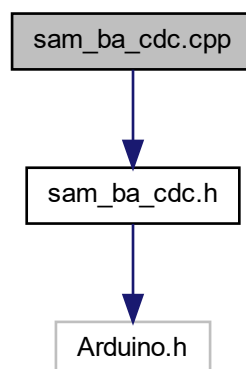
Referenced by USB\_Configure(), USB\_Init(), USB\_IsConfigured(), USB\_SendZlp(), and USB\_Write().

## 5.3 README.md File Reference

## 5.4 sam\_ba\_cdc.cpp File Reference

```
#include "sam_ba_cdc.h"
```

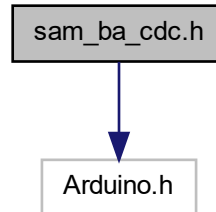
Include dependency graph for sam\_ba\_cdc.cpp:



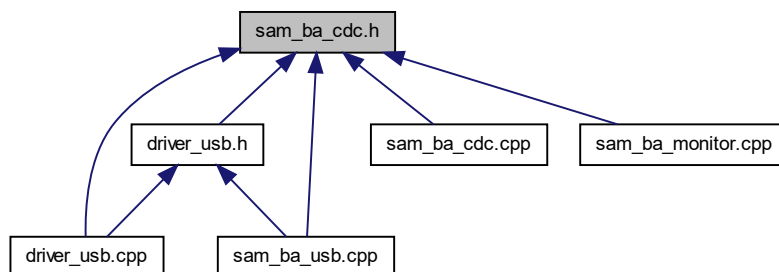
## 5.5 sam\_ba\_cdc.h File Reference

```
#include <Arduino.h>
```

Include dependency graph for sam\_ba\_cdc.h:



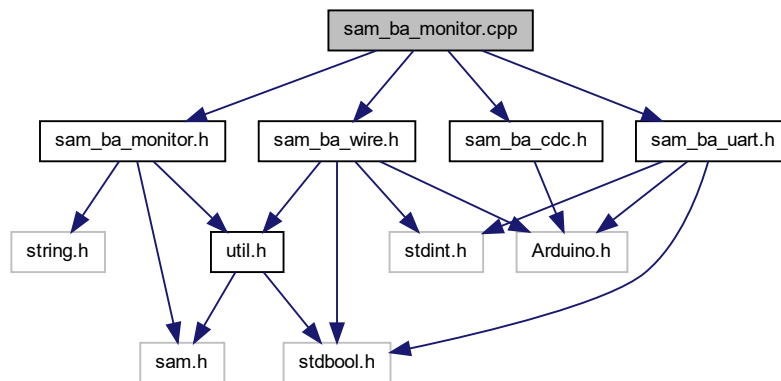
This graph shows which files directly or indirectly include this file:



## 5.6 sam\_ba\_monitor.cpp File Reference

```
#include "sam_ba_monitor.h"  
#include "sam_ba_uart.h"  
#include "sam_ba_wire.h"  
#include "sam_ba_cdc.h"
```

Include dependency graph for sam\_ba\_monitor.cpp:



## Macros

- `#define TX_RX_LED_PULSE_PERIOD 100`

## Functions

- void `sam_ba_monitor_init` (uint8\_t com\_interface)  
*Initialize the monitor.*
- void `sam_ba_putdata_term` (uint8\_t \*data, uint32\_t length)  
*This function allows data emission by USART.*
- void `call_applet` (uint32\_t address)
- void `sam_ba_monitor_sys_tick` (void)  
*System tick function of the SAM-BA Monitor.*
- void `sam_ba_monitor_run` (void)  
*This function starts the SAM-BA monitor.*

## Variables

- const char `RomBOOT_Version` [] = `SAM_BA_VERSION`
- t\_monitor\_if \* `ptr_monitor_if`
- volatile bool `b_sam_ba_interface_usart` = false
- volatile bool `b_sam_ba_interface_wire` = false
- volatile uint32\_t `sp`
- uint32\_t `current_number`
- uint32\_t `i`
- uint32\_t `length`
- uint8\_t `command`
- uint8\_t \* `ptr_data`
- uint8\_t \* `ptr`
- uint8\_t `data` [SIZEBUFMAX]
- uint8\_t `j`
- uint32\_t `u32tmp`

## 5.6.1 Macro Definition Documentation

### 5.6.1.1 TX\_RX\_LED\_PULSE\_PERIOD

```
#define TX_RX_LED_PULSE_PERIOD 100
```

Definition at line 47 of file sam\_ba\_monitor.cpp.

## 5.6.2 Function Documentation

### 5.6.2.1 call\_applet()

```
void call_applet (
    uint32_t address )
```

Definition at line 206 of file sam\_ba\_monitor.cpp.

References address, and sp.

### 5.6.2.2 sam\_ba\_monitor\_init()

```
void sam_ba_monitor_init (
    uint8_t com_interface )
```

Initialize the monitor.

Definition at line 55 of file sam\_ba\_monitor.cpp.

References b\_sam\_ba\_interface\_usart, b\_sam\_ba\_interface\_wire, ptr\_monitor\_if, SAM\_BA\_INTERFACE\_USART, SAM\_BA\_INTERFACE\_USBCDC, SAM\_BA\_INTERFACE\_WIRE, and uart\_if.

### 5.6.2.3 sam\_ba\_monitor\_run()

```
void sam_ba_monitor_run (
    void )
```

This function starts the SAM-BA monitor.

Main function of the SAM-BA Monitor.

Definition at line 494 of file sam\_ba\_monitor.cpp.

References command, and ptr\_data.

#### 5.6.2.4 sam\_ba\_monitor\_sys\_tick()

```
void sam_ba_monitor_sys_tick (
    void )
```

System tick function of the SAM-BA Monitor.

Definition at line 478 of file sam\_ba\_monitor.cpp.

#### 5.6.2.5 sam\_ba\_putdata\_term()

```
void sam_ba_putdata_term (
    uint8_t * data,
    uint32_t length )
```

This function allows data emission by USART.

**Parameters**

<i>*data</i>	Data pointer
<i>length</i>	Length of the data

Definition at line 163 of file sam\_ba\_monitor.cpp.

References data, i, and length.

**5.6.3 Variable Documentation****5.6.3.1 b\_sam\_ba\_interface\_usart**

```
volatile bool b_sam_ba_interface_usart = false
```

Definition at line 43 of file sam\_ba\_monitor.cpp.

Referenced by sam\_ba\_monitor\_init().

**5.6.3.2 b\_sam\_ba\_interface\_wire**

```
volatile bool b_sam_ba_interface_wire = false
```

Definition at line 44 of file sam\_ba\_monitor.cpp.

Referenced by sam\_ba\_monitor\_init().

**5.6.3.3 command**

```
uint8_t command
```

Definition at line 226 of file sam\_ba\_monitor.cpp.

Referenced by sam\_ba\_monitor\_run().

**5.6.3.4 current\_number**

```
uint32_t current_number
```

Definition at line 224 of file sam\_ba\_monitor.cpp.

#### 5.6.3.5 data

```
uint8_t data[SIZEBUFMAX]
```

Definition at line 226 of file sam\_ba\_monitor.cpp.

Referenced by sam\_ba\_putdata\_term(), uart\_getdata(), uart\_getdata\_xmd(), uart\_putdata(), uart\_putdata\_xmd(), wire\_getdata(), wire\_getdata\_xmd(), wire\_putdata(), and wire\_putdata\_xmd().

#### 5.6.3.6 i

```
uint32_t i
```

Definition at line 225 of file sam\_ba\_monitor.cpp.

Referenced by delayUs(), flashWrite(), sam\_ba\_putdata\_term(), uart\_putdata(), and wire\_putdata().

#### 5.6.3.7 j

```
uint8_t j
```

Definition at line 227 of file sam\_ba\_monitor.cpp.

#### 5.6.3.8 length

```
uint32_t length
```

Definition at line 225 of file sam\_ba\_monitor.cpp.

Referenced by sam\_ba\_putdata\_term(), uart\_getdata\_xmd(), uart\_putdata(), uart\_putdata\_xmd(), USB\_Write(), wire\_getdata\_xmd(), wire\_putdata(), and wire\_putdata\_xmd().

#### 5.6.3.9 ptr

```
uint8_t * ptr
```

Definition at line 226 of file sam\_ba\_monitor.cpp.

#### 5.6.3.10 ptr\_data

```
uint8_t * ptr_data
```

Definition at line 226 of file sam\_ba\_monitor.cpp.

Referenced by flashWrite(), sam\_ba\_monitor\_run(), uart\_getdata\_xmd(), uart\_putdata\_xmd(), wire\_getdata\_xmd(), and wire\_putdata\_xmd().

### 5.6.3.11 ptr\_monitor\_if

```
t_monitor_if* ptr_monitor_if
```

Definition at line 37 of file sam\_ba\_monitor.cpp.

Referenced by sam\_ba\_monitor\_init().

### 5.6.3.12 RomBOOT\_Version

```
const char RomBOOT_Version[] = SAM_BA_VERSION
```

Definition at line 26 of file sam\_ba\_monitor.cpp.

### 5.6.3.13 sp

```
volatile uint32_t sp
```

Definition at line 205 of file sam\_ba\_monitor.cpp.

Referenced by call\_applet().

### 5.6.3.14 u32tmp

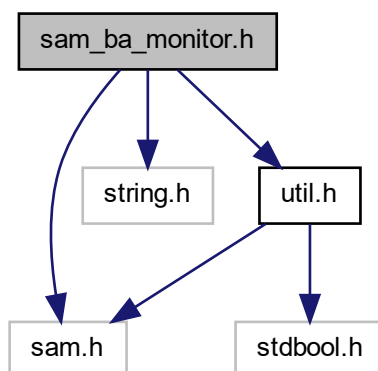
```
uint32_t u32tmp
```

Definition at line 228 of file sam\_ba\_monitor.cpp.

## 5.7 sam\_ba\_monitor.h File Reference

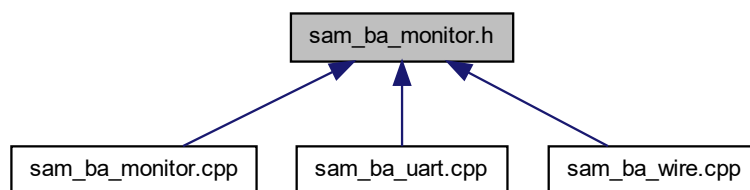
```
#include "sam.h"  
#include <string.h>  
#include "util.h"
```

Include dependency graph for sam\_ba\_monitor.h:





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



### Data Structures

- struct [t\\_monitor\\_if](#)

### Macros

- `#define` [\\_MONITOR\\_SAM\\_BA\\_H\\_](#)
- `#define` [SAM\\_BA\\_VERSION](#) "2.0"
- `#define` [SAM\\_BA\\_BOTH\\_INTERFACES](#) 0
- `#define` [SAM\\_BA\\_UART\\_ONLY](#) 1
- `#define` [SAM\\_BA\\_USBCDC\\_ONLY](#) 2
- `#define` [SAM\\_BA\\_NONE](#) 3
- `#define` [SAM\\_BA\\_INTERFACE\\_SAM\\_BA\\_UART\\_ONLY](#)
- `#define` [SAM\\_BA\\_INTERFACE\\_USBCDC](#) 0
- `#define` [SAM\\_BA\\_INTERFACE\\_USART](#) 1
- `#define` [SAM\\_BA\\_INTERFACE\\_WIRE](#) 2
- `#define` [SIZEBUFMAX](#) 64

### Functions

- void [sam\\_ba\\_monitor\\_init](#) (uint8\_t com\_interface)  
*Initialize the monitor.*
- void [sam\\_ba\\_monitor\\_sys\\_tick](#) (void)  
*System tick function of the SAM-BA Monitor.*
- void [sam\\_ba\\_monitor\\_run](#) (void)  
*Main function of the SAM-BA Monitor.*
- void [sam\\_ba\\_putdata\\_term](#) (uint8\_t \*data, uint32\_t length)  
*This function allows data emission by USART.*
- void [call\\_applet](#) (uint32\_t address)

### Variables

- [t\\_monitor\\_if](#) [uart\\_if](#)

### 5.7.1 Macro Definition Documentation

#### 5.7.1.1 `_MONITOR_SAM_BA_H_`

```
#define _MONITOR_SAM_BA_H_
```

Definition at line 28 of file `sam_ba_monitor.h`.

#### 5.7.1.2 `SAM_BA_BOTH_INTERFACES`

```
#define SAM_BA_BOTH_INTERFACES 0
```

Definition at line 33 of file `sam_ba_monitor.h`.

#### 5.7.1.3 `SAM_BA_INTERFACE`

```
#define SAM_BA_INTERFACE SAM\_BA\_UART\_ONLY
```

Definition at line 41 of file `sam_ba_monitor.h`.

#### 5.7.1.4 `SAM_BA_INTERFACE_USART`

```
#define SAM_BA_INTERFACE_USART 1
```

Definition at line 47 of file `sam_ba_monitor.h`.

Referenced by `sam_ba_monitor_init()`.

#### 5.7.1.5 `SAM_BA_INTERFACE_USBCDC`

```
#define SAM_BA_INTERFACE_USBCDC 0
```

Definition at line 45 of file `sam_ba_monitor.h`.

Referenced by `sam_ba_monitor_init()`.

#### 5.7.1.6 `SAM_BA_INTERFACE_WIRE`

```
#define SAM_BA_INTERFACE_WIRE 2
```

Definition at line 49 of file `sam_ba_monitor.h`.

Referenced by `sam_ba_monitor_init()`.

#### 5.7.1.7 SAM\_BA\_NONE

```
#define SAM_BA_NONE 3
```

Definition at line 37 of file sam\_ba\_monitor.h.

#### 5.7.1.8 SAM\_BA\_UART\_ONLY

```
#define SAM_BA_UART_ONLY 1
```

Definition at line 34 of file sam\_ba\_monitor.h.

#### 5.7.1.9 SAM\_BA\_USBCDC\_ONLY

```
#define SAM_BA_USBCDC_ONLY 2
```

Definition at line 35 of file sam\_ba\_monitor.h.

#### 5.7.1.10 SAM\_BA\_VERSION

```
#define SAM_BA_VERSION "2.0"
```

Definition at line 30 of file sam\_ba\_monitor.h.

#### 5.7.1.11 SIZEBUFMAX

```
#define SIZEBUFMAX 64
```

Definition at line 52 of file sam\_ba\_monitor.h.

### 5.7.2 Function Documentation

#### 5.7.2.1 call\_applet()

```
void call_applet (
    uint32_t address )
```

Definition at line 206 of file sam\_ba\_monitor.cpp.

References address, and sp.

### 5.7.2.2 `sam_ba_monitor_init()`

```
void sam_ba_monitor_init (
    uint8_t com_interface )
```

Initialize the monitor.

Definition at line 55 of file `sam_ba_monitor.cpp`.

References `b_sam_ba_interface_usart`, `b_sam_ba_interface_wire`, `ptr_monitor_if`, `SAM_BA_INTERFACE_USART`, `SAM_BA_INTERFACE_USBCDC`, `SAM_BA_INTERFACE_WIRE`, and `uart_if`.

### 5.7.2.3 `sam_ba_monitor_run()`

```
void sam_ba_monitor_run (
    void )
```

Main function of the SAM-BA Monitor.

Main function of the SAM-BA Monitor.

Definition at line 494 of file `sam_ba_monitor.cpp`.

References `command`, and `ptr_data`.

### 5.7.2.4 `sam_ba_monitor_sys_tick()`

```
void sam_ba_monitor_sys_tick (
    void )
```

System tick function of the SAM-BA Monitor.

Definition at line 478 of file `sam_ba_monitor.cpp`.

### 5.7.2.5 `sam_ba_putdata_term()`

```
void sam_ba_putdata_term (
    uint8_t * data,
    uint32_t length )
```

This function allows data emission by USART.

#### Parameters

<i>*data</i>	Data pointer
<i>length</i>	Length of the data

Definition at line 163 of file sam\_ba\_monitor.cpp.

References data, i, and length.

### 5.7.3 Variable Documentation

#### 5.7.3.1 uart\_if

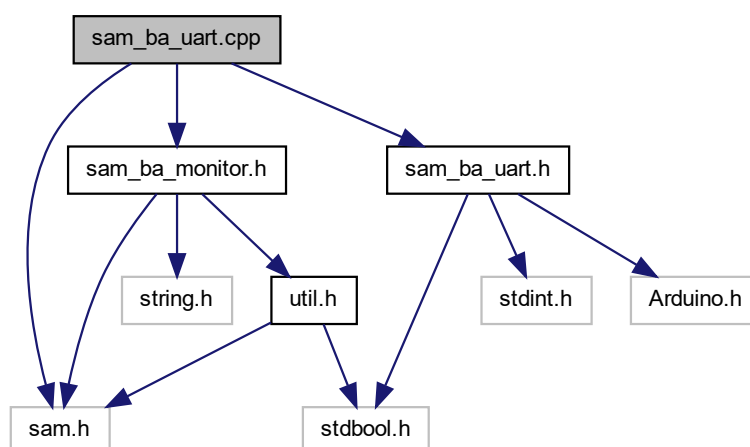
```
t_monitor_if uart_if
```

Definition at line 27 of file sam\_ba\_uart.cpp.

Referenced by sam\_ba\_monitor\_init().

### 5.8 sam\_ba\_uart.cpp File Reference

```
#include <sam.h>
#include "sam_ba_monitor.h"
#include "sam_ba_uart.h"
Include dependency graph for sam_ba_uart.cpp:
```



## Functions

- void `uart_setup` (Uart &Myserial)
- void `uart_open` (unsigned int fBaudSpeed)  
*Open the given USART.*
- void `uart_close` (void)  
*Close communication line.*
- int `uart_putc` (int value)  
*Puts a byte on usart line The type int is used to support printf redirection from compiler LIB.*
- int `uart_getc` (void)  
*Waits and gets a value on usart line.*
- int `uart_sharp_received` (void)  
*Returns true if the SAM-BA Uart received the sharp char.*
- bool `uart_is_rx_ready` (void)  
*This function checks if a character has been received on the usart line.*
- int `uart_readc` (void)  
*Gets a value on usart line.*
- uint32\_t `uart_putdata` (void const \*data, uint32\_t length)  
*Send buffer on usart line.*
- uint32\_t `uart_getdata` (void \*data, uint32\_t length)  
*Gets data from usart line.*
- unsigned short `uart_add_crc` (char ptr, unsigned short crc)  
*Compute the CRC.*
- uint32\_t `uart_putdata_xmd` (void const \*data, uint32\_t length)  
*Send buffer on usart line using Xmodem protocol.*
- uint32\_t `uart_getdata_xmd` (void \*data, uint32\_t length)  
*Gets data from usart line using Xmodem protocol.*

## Variables

- `t_monitor_if` `uart_if`
- Uart \* `serial`
- volatile uint8\_t `uart_b_sharp_received`
- volatile uint8\_t `buffer_rx_usart` [USART\_BUFFER\_SIZE]
- volatile uint8\_t `uart_idx_rx_read`
- volatile uint8\_t `uart_idx_rx_write`
- volatile uint8\_t `uart_buffer_tx_usart` [USART\_BUFFER\_SIZE]
- volatile uint8\_t `uart_idx_tx_read`
- volatile uint8\_t `uart_idx_tx_write`
- uint8\_t `uart_error_timeout`
- uint16\_t `uart_size_of_data`
- uint8\_t `uart_mode_of_transfer`

### 5.8.1 Function Documentation

#### 5.8.1.1 `uart_add_crc()`

```
unsigned short uart_add_crc (
    char c,
    unsigned short crc )
```

Compute the CRC.

**Parameters**

<i>Char</i>	to add to CRC
<i>Previous</i>	CRC

**Returns**

The new computed CRC

Definition at line 202 of file sam\_ba\_uart.cpp.

**5.8.1.2 uart\_close()**

```
void uart_close (  
    void )
```

Close communication line.

Stops the USART.

Definition at line 85 of file sam\_ba\_uart.cpp.

References serial.

**5.8.1.3 uart\_getc()**

```
int uart_getc (  
    void )
```

Waits and gets a value on usart line.

**Returns**

value read on usart line

Definition at line 104 of file sam\_ba\_uart.cpp.

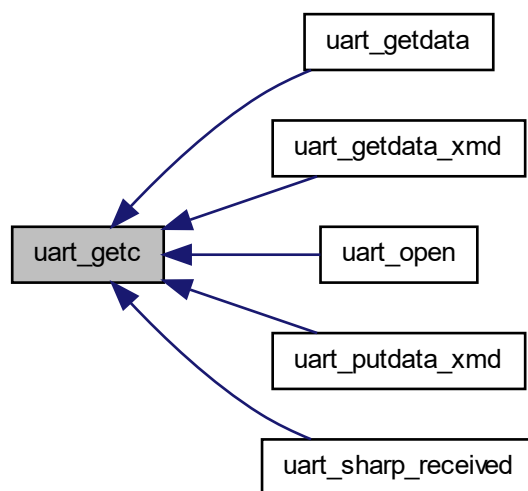
References serial, and uart\_is\_rx\_ready().

Referenced by uart\_getdata(), uart\_getdata\_xmd(), uart\_open(), uart\_putdata\_xmd(), and uart\_sharp\_received().

Here is the call graph for this function:



Here is the caller graph for this function:



#### 5.8.1.4 `uart_getdata()`

```
uint32_t uart_getdata (
    void * data,
    uint32_t length )
```

Gets data from usart line.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

##### Returns

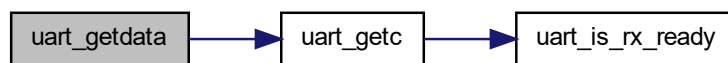
value read on usart line

Definition at line 155 of file `sam_ba_uart.cpp`.

References `data`, and `uart_getc()`.



Here is the call graph for this function:



#### 5.8.1.5 uart\_getdata\_xmd()

```

uint32_t uart_getdata_xmd (
    void * data,
    uint32_t length )
  
```

Gets data from usart line using Xmodem protocol.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

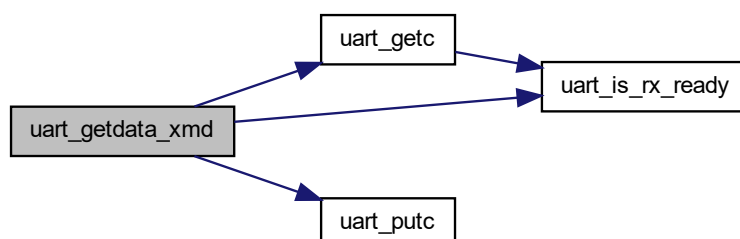
##### Returns

value read on usart line

Definition at line 410 of file sam\_ba\_uart.cpp.

References `data`, `length`, `ptr_data`, `SOH`, `uart_error_timeout`, `uart_getc()`, `uart_is_rx_ready()`, `uart_mode_of_↔` transfer, `uart_putc()`, and `uart_size_of_data`.

Here is the call graph for this function:



### 5.8.1.6 uart\_is\_rx\_ready()

```
bool uart_is_rx_ready (
    void )
```

This function checks if a character has been received on the usart line.

#### Returns

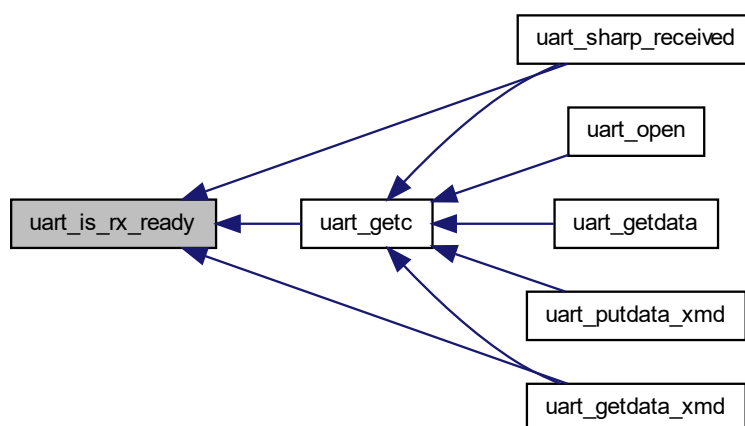
1 if a byte is ready to be read.

Definition at line 125 of file sam\_ba\_uart.cpp.

References serial.

Referenced by uart\_getc(), uart\_getdata\_xmd(), and uart\_sharp\_received().

Here is the caller graph for this function:



### 5.8.1.7 uart\_open()

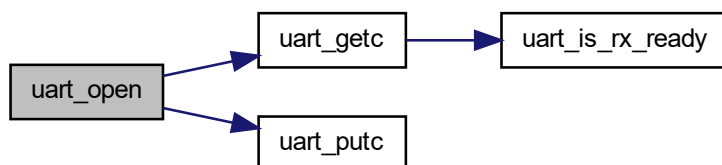
```
void uart_open (
    unsigned int fBaudSpeed )
```

Open the given USART.

Definition at line 72 of file sam\_ba\_uart.cpp.

References serial, uart\_error\_timeout, uart\_getc(), and uart\_putc().

Here is the call graph for this function:



#### 5.8.1.8 uart\_putc()

```
int uart_putc (
    int value )
```

Puts a byte on usart line The type int is used to support printf redirection from compiler LIB.

Puts a byte on usart line.

##### Parameters

<i>value</i>	Value to put
--------------	--------------

##### Returns

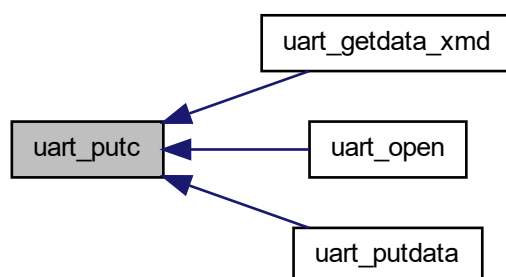
1 if function was successfully done, otherwise 0.

Definition at line 98 of file `sam_ba_uart.cpp`.

References `serial`.

Referenced by `uart_getdata_xmd()`, `uart_open()`, and `uart_putdata()`.

Here is the caller graph for this function:



#### 5.8.1.9 uart\_putdata()

```
uint32_t uart_putdata (
    void const * data,
    uint32_t length )
```

Send buffer on usart line.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to send

##### Returns

number of data sent

Definition at line 141 of file sam\_ba\_uart.cpp.

References data, i, length, and uart\_putc().

Here is the call graph for this function:



#### 5.8.1.10 uart\_putdata\_xmd()

```
uint32_t uart_putdata_xmd (
    void const * data,
    uint32_t length )
```

Send buffer on usart line using Xmodem protocol.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to send

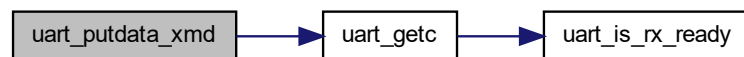
**Returns**

number of data sent

Definition at line 278 of file sam\_ba\_uart.cpp.

References data, length, NAK, PKTLEN\_128, ptr\_data, uart\_error\_timeout, uart\_getc(), uart\_mode\_of\_transfer, and uart\_size\_of\_data.

Here is the call graph for this function:

**5.8.1.11 uart\_readc()**

```
int uart_readc (
    void )
```

Gets a value on usart line.

**Returns**

value read on usart line

Definition at line 132 of file sam\_ba\_uart.cpp.

References buffer\_rx\_usart, uart\_idx\_rx\_read, and USART\_BUFFER\_SIZE.

**5.8.1.12 uart\_setup()**

```
void uart_setup (
    Uart & Myserial )
```

Definition at line 39 of file sam\_ba\_uart.cpp.

References serial.

### 5.8.1.13 `uart_sharp_received()`

```
int uart_sharp_received (  
    void )
```

Returns true if the SAM-BA Uart received the sharp char.

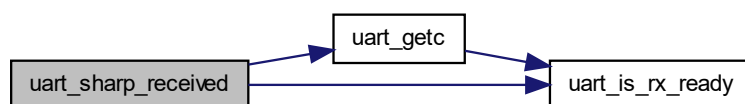
#### Returns

Returns true if the SAM-BA Uart received the sharp char

Definition at line 115 of file `sam_ba_uart.cpp`.

References `SHARP_CHARACTER`, `uart_getc()`, and `uart_is_rx_ready()`.

Here is the call graph for this function:



## 5.8.2 Variable Documentation

### 5.8.2.1 `buffer_rx_usart`

```
volatile uint8_t buffer_rx_usart[USART_BUFFER_SIZE]
```

Definition at line 51 of file `sam_ba_uart.cpp`.

Referenced by `uart_readc()`.

### 5.8.2.2 `serial`

```
Uart* serial
```

Definition at line 38 of file `sam_ba_uart.cpp`.

Referenced by `uart_close()`, `uart_getc()`, `uart_is_rx_ready()`, `uart_open()`, `uart_putc()`, and `uart_setup()`.

### 5.8.2.3 uart\_b\_sharp\_received

```
volatile uint8_t uart_b_sharp_received
```

Definition at line 48 of file sam\_ba\_uart.cpp.

### 5.8.2.4 uart\_buffer\_tx\_usart

```
volatile uint8_t uart_buffer_tx_usart[USART_BUFFER_SIZE]
```

Definition at line 56 of file sam\_ba\_uart.cpp.

### 5.8.2.5 uart\_error\_timeout

```
uint8_t uart_error_timeout
```

Definition at line 62 of file sam\_ba\_uart.cpp.

Referenced by `uart_getdata_xmd()`, `uart_open()`, and `uart_putdata_xmd()`.

### 5.8.2.6 uart\_idx\_rx\_read

```
volatile uint8_t uart_idx_rx_read
```

Definition at line 53 of file sam\_ba\_uart.cpp.

Referenced by `uart_readc()`.

### 5.8.2.7 uart\_idx\_rx\_write

```
volatile uint8_t uart_idx_rx_write
```

Definition at line 54 of file sam\_ba\_uart.cpp.

### 5.8.2.8 uart\_idx\_tx\_read

```
volatile uint8_t uart_idx_tx_read
```

Definition at line 58 of file sam\_ba\_uart.cpp.

#### 5.8.2.9 uart\_idx\_tx\_write

```
volatile uint8_t uart_idx_tx_write
```

Definition at line 59 of file sam\_ba\_uart.cpp.

#### 5.8.2.10 uart\_if

```
t_monitor_if uart_if
```

**Initial value:**

```
=  
{  
    .put_c =      uart_putc,  
    .get_c =      uart_getc,  
    .is_rx_ready = uart_is_rx_ready,  
    .putdata =     uart_putdata,  
    .getdata =     uart_getdata,  
    .putdata_xmd = uart_putdata_xmd,  
    .getdata_xmd = uart_getdata_xmd  
}
```

Definition at line 27 of file sam\_ba\_uart.cpp.

Referenced by sam\_ba\_monitor\_init().

#### 5.8.2.11 uart\_mode\_of\_transfer

```
uint8_t uart_mode_of_transfer
```

Definition at line 64 of file sam\_ba\_uart.cpp.

Referenced by uart\_getdata\_xmd(), and uart\_putdata\_xmd().

#### 5.8.2.12 uart\_size\_of\_data

```
uint16_t uart_size_of_data
```

Definition at line 63 of file sam\_ba\_uart.cpp.

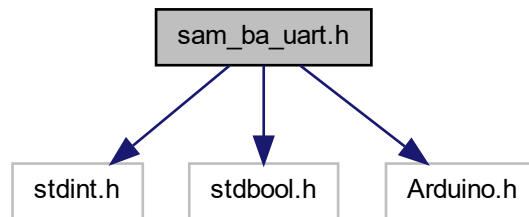
Referenced by uart\_getdata\_xmd(), and uart\_putdata\_xmd().



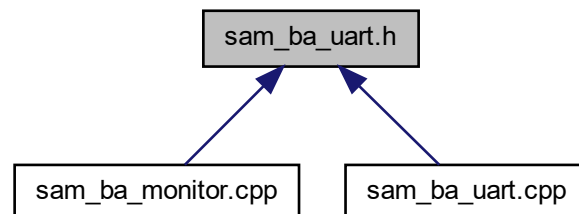
## 5.9 sam\_ba\_uart.h File Reference

```
#include <stdint.h>
#include <stdbool.h>
#include "Arduino.h"
```

Include dependency graph for sam\_ba\_uart.h:



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



### Macros

- #define [USART\\_BUFFER\\_SIZE](#) (128)
- #define [USART\\_DEFAULT\\_TIMEOUT](#) (1000)
- #define [CRC16POLY](#) (0x1021)
- #define [SHARP\\_CHARACTER](#) '#' /\* 0x23 : 35\*/
- #define [SOH](#) (0x01)
- #define [EOT](#) (0x04)
- #define [ACK](#) (0x06)
- #define [NAK](#) (0x15)
- #define [CAN](#) (0x18)
- #define [ESC](#) (0x1b)
- #define [PKTLEN\\_128](#) (128)

## Functions

- void `uart_setup` (Uart &Myserial)
- void `uart_open` (unsigned int fBaudSpeed)  
*Open the given USART.*
- void `uart_close` (void)  
*Stops the USART.*
- int `uart_putc` (int value)  
*Puts a byte on usart line.*
- int `uart_getc` (void)  
*Waits and gets a value on usart line.*
- int `uart_sharp_received` (void)  
*Returns true if the SAM-BA Uart received the sharp char.*
- bool `uart_is_rx_ready` (void)  
*This function checks if a character has been received on the usart line.*
- int `uart_readc` (void)  
*Gets a value on usart line.*
- uint32\_t `uart_putdata` (void const \*data, uint32\_t length)  
*Send buffer on usart line.*
- uint32\_t `uart_getdata` (void \*data, uint32\_t length)  
*Gets data from usart line.*
- uint32\_t `uart_putdata_xmd` (void const \*data, uint32\_t length)  
*Send buffer on usart line using Xmodem protocol.*
- uint32\_t `uart_getdata_xmd` (void \*data, uint32\_t length)  
*Gets data from usart line using Xmodem protocol.*
- unsigned short `uart_add_crc` (char c, unsigned short crc)  
*Compute the CRC.*

## 5.9.1 Macro Definition Documentation

### 5.9.1.1 ACK

```
#define ACK (0x06)
```

Definition at line 46 of file `sam_ba_uart.h`.

### 5.9.1.2 CAN

```
#define CAN (0x18)
```

Definition at line 48 of file `sam_ba_uart.h`.

#### 5.9.1.3 CRC16POLY

```
#define CRC16POLY (0x1021)
```

Definition at line 38 of file sam\_ba\_uart.h.

#### 5.9.1.4 EOT

```
#define EOT (0x04)
```

Definition at line 45 of file sam\_ba\_uart.h.

#### 5.9.1.5 ESC

```
#define ESC (0x1b)
```

Definition at line 49 of file sam\_ba\_uart.h.

#### 5.9.1.6 NAK

```
#define NAK (0x15)
```

Definition at line 47 of file sam\_ba\_uart.h.

Referenced by uart\_putdata\_xmd(), and wire\_putdata\_xmd().

#### 5.9.1.7 PKTLEN\_128

```
#define PKTLEN_128 (128)
```

Definition at line 51 of file sam\_ba\_uart.h.

Referenced by uart\_putdata\_xmd(), and wire\_putdata\_xmd().

#### 5.9.1.8 SHARP\_CHARACTER

```
#define SHARP_CHARACTER '#' /* 0x23 : 35*/
```

Definition at line 40 of file sam\_ba\_uart.h.

Referenced by uart\_sharp\_received(), and wire\_sharp\_received().

### 5.9.1.9 SOH

```
#define SOH (0x01)
```

Definition at line 43 of file `sam_ba_uart.h`.

Referenced by `uart_getdata_xmd()`, and `wire_getdata_xmd()`.

### 5.9.1.10 USART\_BUFFER\_SIZE

```
#define USART_BUFFER_SIZE (128)
```

Definition at line 31 of file `sam_ba_uart.h`.

Referenced by `uart_readc()`.

### 5.9.1.11 USART\_DEFAULT\_TIMEOUT

```
#define USART_DEFAULT_TIMEOUT (1000)
```

Definition at line 34 of file `sam_ba_uart.h`.

## 5.9.2 Function Documentation

### 5.9.2.1 uart\_add\_crc()

```
unsigned short uart_add_crc (  
    char c,  
    unsigned short crc )
```

Compute the CRC.

#### Parameters

<i>Char</i>	to add to CRC
<i>Previous</i>	CRC

#### Returns

The new computed CRC

Definition at line 202 of file `sam_ba_uart.cpp`.

### 5.9.2.2 uart\_close()

```
void uart_close (
    void )
```

Stops the USART.

Stops the USART.

Definition at line 85 of file sam\_ba\_uart.cpp.

References serial.

### 5.9.2.3 uart\_getc()

```
int uart_getc (
    void )
```

Waits and gets a value on usart line.

#### Returns

value read on usart line

Definition at line 104 of file sam\_ba\_uart.cpp.

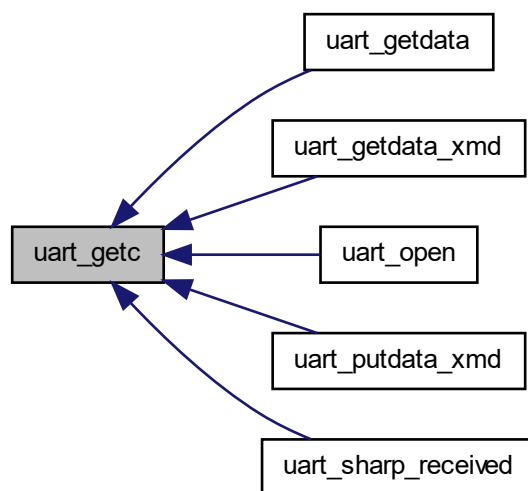
References serial, and uart\_is\_rx\_ready().

Referenced by uart\_getdata(), uart\_getdata\_xmd(), uart\_open(), uart\_putdata\_xmd(), and uart\_sharp\_received().

Here is the call graph for this function:



Here is the caller graph for this function:



#### 5.9.2.4 `uart_getdata()`

```
uint32_t uart_getdata (
    void * data,
    uint32_t length )
```

Gets data from usart line.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

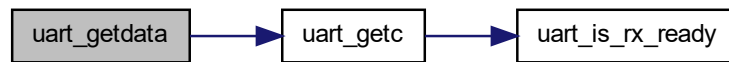
##### Returns

value read on usart line

Definition at line 155 of file `sam_ba_uart.cpp`.

References `data`, and `uart_getc()`.

Here is the call graph for this function:



#### 5.9.2.5 uart\_getdata\_xmd()

```

uint32_t uart_getdata_xmd (
    void * data,
    uint32_t length )
  
```

Gets data from usart line using Xmodem protocol.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

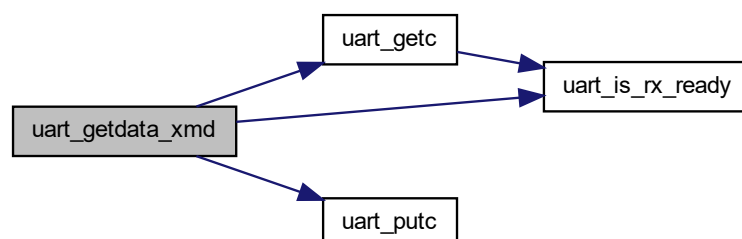
##### Returns

value read on usart line

Definition at line 410 of file sam\_ba\_uart.cpp.

References `data`, `length`, `ptr_data`, `SOH`, `uart_error_timeout`, `uart_getc()`, `uart_is_rx_ready()`, `uart_mode_of_↔` transfer, `uart_putc()`, and `uart_size_of_data`.

Here is the call graph for this function:



### 5.9.2.6 uart\_is\_rx\_ready()

```
bool uart_is_rx_ready (
    void )
```

This function checks if a character has been received on the usart line.

#### Returns

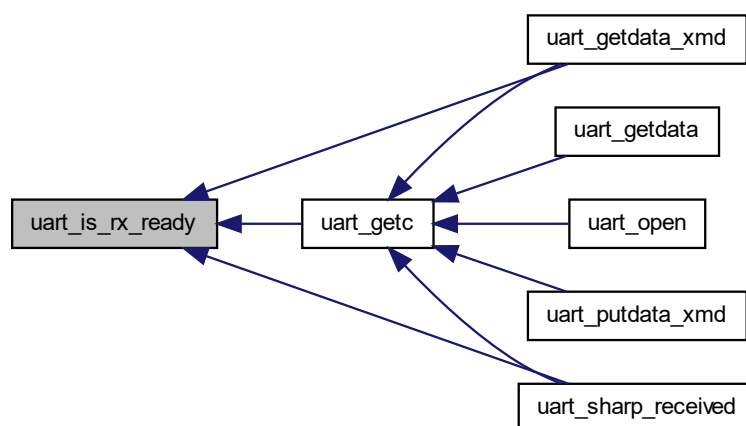
1 if a byte is ready to be read.

Definition at line 125 of file sam\_ba\_uart.cpp.

References serial.

Referenced by uart\_getc(), uart\_getdata\_xmd(), and uart\_sharp\_received().

Here is the caller graph for this function:



### 5.9.2.7 uart\_open()

```
void uart_open (
    unsigned int fBaudSpeed )
```

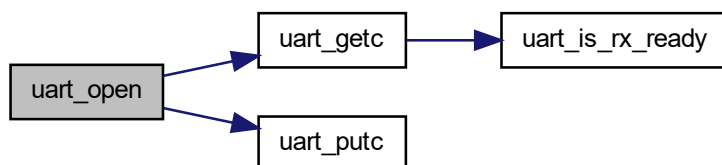
Open the given USART.

Definition at line 72 of file sam\_ba\_uart.cpp.

References serial, uart\_error\_timeout, uart\_getc(), and uart\_putc().



Here is the call graph for this function:



#### 5.9.2.8 uart\_putc()

```
int uart_putc (  
    int value )
```

Puts a byte on usart line.

##### Parameters

<i>value</i>	Value to put
--------------	--------------

##### Returns

1 if function was successfully done, otherwise 0.

Puts a byte on usart line.

##### Parameters

<i>value</i>	Value to put
--------------	--------------

##### Returns

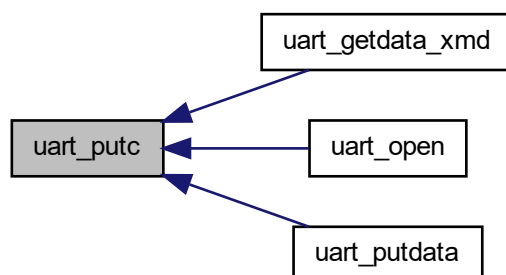
1 if function was successfully done, otherwise 0.

Definition at line 98 of file `sam_ba_uart.cpp`.

References `serial`.

Referenced by `uart_getdata_xmd()`, `uart_open()`, and `uart_putdata()`.

Here is the caller graph for this function:



#### 5.9.2.9 uart\_putdata()

```
uint32_t uart_putdata (
    void const * data,
    uint32_t length )
```

Send buffer on usart line.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to send

##### Returns

number of data sent

Definition at line 141 of file sam\_ba\_uart.cpp.

References data, i, length, and uart\_putc().

Here is the call graph for this function:



#### 5.9.2.10 uart\_putdata\_xmd()

```
uint32_t uart_putdata_xmd (
    void const * data,
    uint32_t length )
```

Send buffer on usart line using Xmodem protocol.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to send

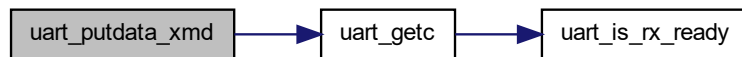
##### Returns

number of data sent

Definition at line 278 of file sam\_ba\_uart.cpp.

References `data`, `length`, `NAK`, `PKTLEN_128`, `ptr_data`, `uart_error_timeout`, `uart_getc()`, `uart_mode_of_transfer`, and `uart_size_of_data`.

Here is the call graph for this function:



#### 5.9.2.11 uart\_readc()

```
int uart_readc (
    void )
```

Gets a value on usart line.

##### Returns

value read on usart line

Definition at line 132 of file sam\_ba\_uart.cpp.

References `buffer_rx_usart`, `uart_idx_rx_read`, and `USART_BUFFER_SIZE`.

### 5.9.2.12 uart\_setup()

```
void uart_setup (
    Uart & Myserial )
```

Definition at line 39 of file sam\_ba\_uart.cpp.

References serial.

### 5.9.2.13 uart\_sharp\_received()

```
int uart_sharp_received (
    void )
```

Returns true if the SAM-BA Uart received the sharp char.

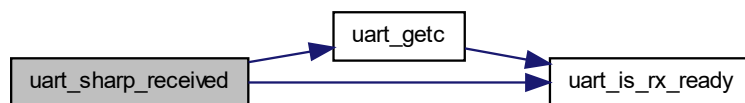
#### Returns

Returns true if the SAM-BA Uart received the sharp char

Definition at line 115 of file sam\_ba\_uart.cpp.

References SHARP\_CHARACTER, uart\_getc(), and uart\_is\_rx\_ready().

Here is the call graph for this function:

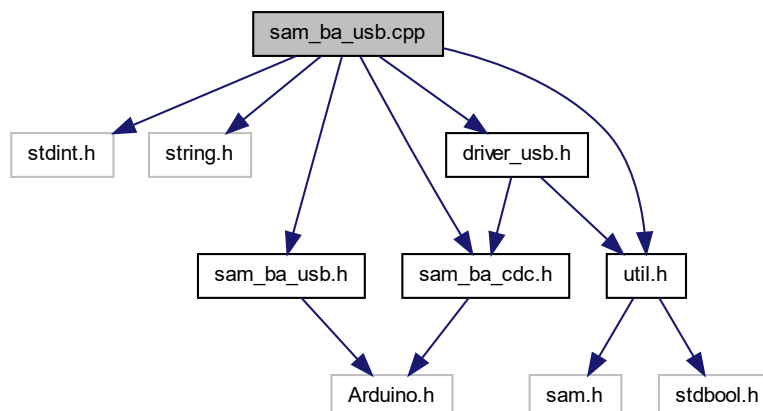


## 5.10 sam\_ba\_usb.cpp File Reference

```
#include <stdint.h>
#include <string.h>
#include "sam_ba_usb.h"
#include "driver_usb.h"
#include "sam_ba_cdc.h"
```

```
#include "util.h"
```

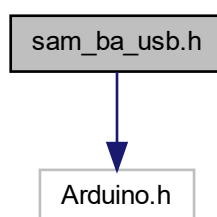
Include dependency graph for sam\_ba\_usb.cpp:



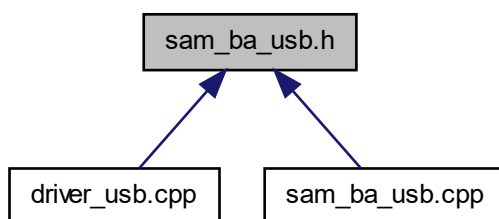
## 5.11 sam\_ba\_usb.h File Reference

```
#include <Arduino.h>
```

Include dependency graph for sam\_ba\_usb.h:

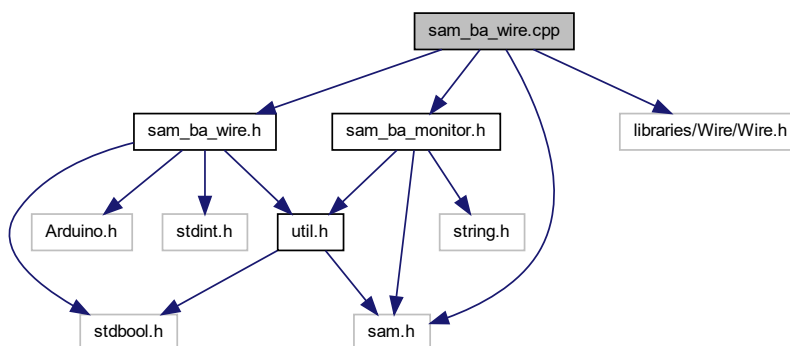


This graph shows which files directly or indirectly include this file:



## 5.12 sam\_ba\_wire.cpp File Reference

```
#include <sam.h>
#include "sam_ba_wire.h"
#include "sam_ba_monitor.h"
#include "libraries/Wire/Wire.h"
Include dependency graph for sam_ba_wire.cpp:
```



## Functions

- void [wire\\_setup](#) (TwoWire &Mywire, unsigned int Myaddress, unsigned int Myaddress\_Bossac)
- void [wire\\_open](#) (unsigned int fBaudSpeed)  
*Open the given USART.*
- void [wire\\_close](#) (void)  
*Close communication line.*
- int [wire\\_putc](#) (int value)  
*Puts a byte on usart line The type int is used to support printf redirection from compiler LIB.*
- int [wire\\_getc](#) (void)  
*Waits and gets a value on usart line.*

- int [wire\\_sharp\\_received](#) (void)  
*Returns true if the SAM-BA Uart received the sharp char.*
- bool [wire\\_is\\_rx\\_ready](#) (void)  
*This function checks if a character has been received on the usart line.*
- int [wire\\_readc](#) (void)  
*Gets a value on usart line.*
- uint32\_t [wire\\_putdata](#) (void const \*[data](#), uint32\_t [length](#))  
*Send buffer on usart line.*
- uint32\_t [wire\\_getdata](#) (void \*[data](#), uint32\_t [length](#))  
*Gets data from usart line.*
- unsigned short [wire\\_add\\_crc](#) (char [ptr](#), unsigned short [crc](#))  
*Compute the CRC.*
- uint32\_t [wire\\_putdata\\_xmd](#) (void const \*[data](#), uint32\_t [length](#))  
*Send buffer on usart line using Xmodem protocol.*
- uint32\_t [wire\\_getdata\\_xmd](#) (void \*[data](#), uint32\_t [length](#))  
*Gets data from usart line using Xmodem protocol.*

#### Variables

- volatile uint8\_t [b\\_sharp\\_received](#)
- volatile uint8\_t [buffer\\_rx\\_wire](#) [[WIRE\\_BUFFER\\_SIZE](#)]
- volatile uint8\_t [idx\\_rx\\_read](#)
- volatile uint8\_t [idx\\_rx\\_write](#)
- volatile uint8\_t [buffer\\_tx\\_wire](#) [[WIRE\\_BUFFER\\_SIZE](#)]
- volatile uint8\_t [idx\\_tx\\_read](#)
- volatile uint8\_t [idx\\_tx\\_write](#)
- uint8\_t [error\\_timeout](#)
- uint16\_t [size\\_of\\_data](#)
- uint8\_t [mode\\_of\\_transfer](#)
- unsigned int [address](#)
- unsigned int [address\\_Bossac](#)
- TwoWire \* [wire](#)

### 5.12.1 Function Documentation

#### 5.12.1.1 [wire\\_add\\_crc\(\)](#)

```
unsigned short wire_add_crc (
    char c,
    unsigned short crc )
```

Compute the CRC.

#### Parameters

<i>Char</i>	to add to CRC
<i>Previous</i>	CRC

**Returns**

The new computed CRC

Definition at line 217 of file sam\_ba\_wire.cpp.

**5.12.1.2 wire\_close()**

```
void wire_close (
    void )
```

Close communication line.

Stops the USART.

Definition at line 94 of file sam\_ba\_wire.cpp.

References wire.

**5.12.1.3 wire\_getc()**

```
int wire_getc (
    void )
```

Waits and gets a value on usart line.

**Returns**

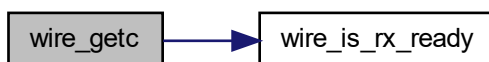
value read on usart line

Definition at line 119 of file sam\_ba\_wire.cpp.

References wire, and wire\_is\_rx\_ready().

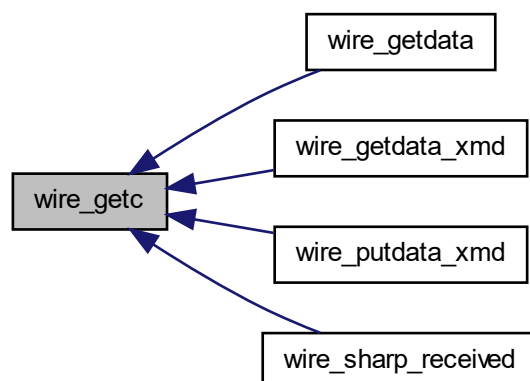
Referenced by wire\_getdata(), wire\_getdata\_xmd(), wire\_putdata\_xmd(), and wire\_sharp\_received().

Here is the call graph for this function:





Here is the caller graph for this function:



#### 5.12.1.4 wire\_getdata()

```

uint32_t wire_getdata (
    void * data,
    uint32_t length )
  
```

Gets data from usart line.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

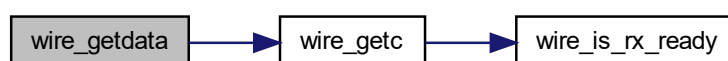
##### Returns

value read on usart line

Definition at line 170 of file sam\_ba\_wire.cpp.

References data, and wire\_getc().

Here is the call graph for this function:



### 5.12.1.5 wire\_getdata\_xmd()

```
uint32_t wire_getdata_xmd (
    void * data,
    uint32_t length )
```

Gets data from usart line using Xmodem protocol.

#### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

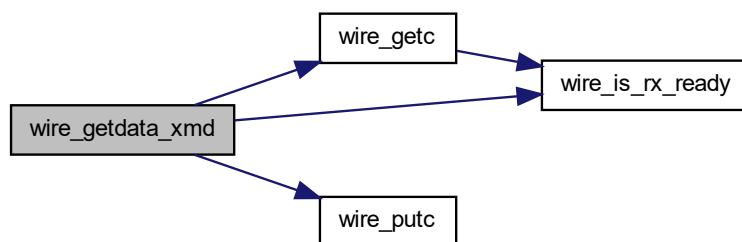
#### Returns

value read on usart line

Definition at line 424 of file sam\_ba\_wire.cpp.

References data, error\_timeout, length, mode\_of\_transfer, ptr\_data, size\_of\_data, SOH, wire\_getc(), wire\_is\_rx\_ready(), and wire\_putc().

Here is the call graph for this function:



### 5.12.1.6 wire\_is\_rx\_ready()

```
bool wire_is_rx_ready (
    void )
```

This function checks if a character has been received on the usart line.

**Returns**

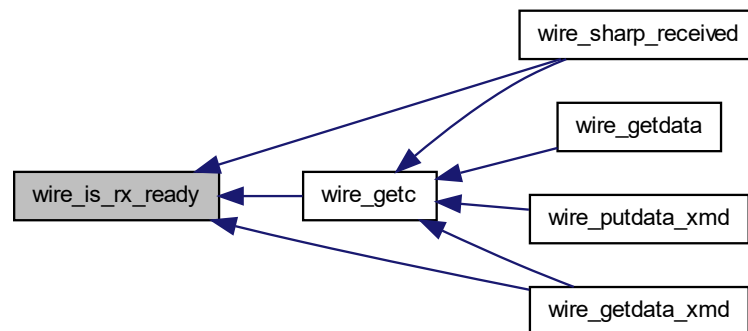
1 if a byte is ready to be read.

Definition at line 140 of file sam\_ba\_wire.cpp.

References wire.

Referenced by wire\_getc(), wire\_getdata\_xmd(), and wire\_sharp\_received().

Here is the caller graph for this function:

**5.12.1.7 wire\_open()**

```
void wire_open (
    unsigned int fBaudSpeed )
```

Open the given USART.

Definition at line 79 of file sam\_ba\_wire.cpp.

References address, and wire.

**5.12.1.8 wire\_putc()**

```
int wire_putc (
    int value )
```

Puts a byte on usart line The type int is used to support printf redirection from compiler LIB.

Puts a byte on usart line.

**Parameters**

<i>value</i>	Value to put
--------------	--------------

**Returns**

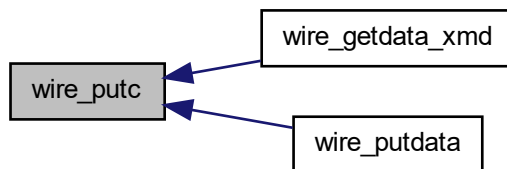
1 if function was successfully done, otherwise 0.

Definition at line 107 of file sam\_ba\_wire.cpp.

References address\_Bossac, and wire.

Referenced by wire\_getdata\_xmd(), and wire\_putdata().

Here is the caller graph for this function:

**5.12.1.9 wire\_putdata()**

```
uint32_t wire_putdata (
    void const * data,
    uint32_t length )
```

Send buffer on usart line.

**Parameters**

<i>data</i>	pointer
<i>number</i>	of data to send

**Returns**

number of data sent

Definition at line 156 of file sam\_ba\_wire.cpp.

References data, i, length, and wire\_putc().

Here is the call graph for this function:



#### 5.12.1.10 wire\_putdata\_xmd()

```
uint32_t wire_putdata_xmd (  
    void const * data,  
    uint32_t length )
```

Send buffer on usart line using Xmodem protocol.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to send

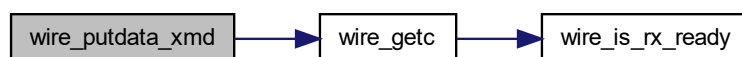
##### Returns

number of data sent

Definition at line 292 of file `sam_ba_wire.cpp`.

References `data`, `error_timeout`, `length`, `mode_of_transfer`, `NAK`, `PKTLEN_128`, `ptr_data`, `size_of_data`, and `wire_getc()`.

Here is the call graph for this function:



#### 5.12.1.11 wire\_readc()

```
int wire_readc (
    void )
```

Gets a value on usart line.

##### Returns

value read on usart line

Definition at line 147 of file sam\_ba\_wire.cpp.

References buffer\_rx\_wire, idx\_rx\_read, and WIRE\_BUFFER\_SIZE.

#### 5.12.1.12 wire\_setup()

```
void wire_setup (
    TwoWire & Mywire,
    unsigned int Myaddress,
    unsigned int Myaddress_Bossac )
```

Definition at line 70 of file sam\_ba\_wire.cpp.

References address, address\_Bossac, and wire.

#### 5.12.1.13 wire\_sharp\_received()

```
int wire_sharp_received (
    void )
```

Returns true if the SAM-BA Uart received the sharp char.

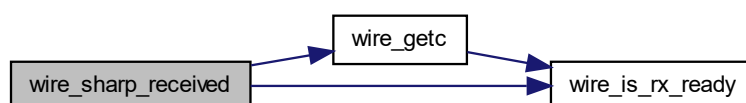
##### Returns

Returns true if the SAM-BA Uart received the sharp char

Definition at line 130 of file sam\_ba\_wire.cpp.

References SHARP\_CHARACTER, wire\_getc(), and wire\_is\_rx\_ready().

Here is the call graph for this function:



### 5.12.2 Variable Documentation

#### 5.12.2.1 address

```
unsigned int address
```

Definition at line 67 of file `sam_ba_wire.cpp`.

Referenced by `call_applet()`, `wire_open()`, and `wire_setup()`.

#### 5.12.2.2 address\_Bossac

```
unsigned int address_Bossac
```

Definition at line 68 of file `sam_ba_wire.cpp`.

Referenced by `wire_putc()`, and `wire_setup()`.

#### 5.12.2.3 b\_sharp\_received

```
volatile uint8_t b_sharp_received
```

Definition at line 44 of file `sam_ba_wire.cpp`.

#### 5.12.2.4 buffer\_rx\_wire

```
volatile uint8_t buffer_rx_wire[WIRE_BUFFER_SIZE]
```

Definition at line 47 of file `sam_ba_wire.cpp`.

Referenced by `wire_readc()`.

#### 5.12.2.5 buffer\_tx\_wire

```
volatile uint8_t buffer_tx_wire[WIRE_BUFFER_SIZE]
```

Definition at line 52 of file `sam_ba_wire.cpp`.

#### 5.12.2.6 error\_timeout

```
uint8_t error_timeout
```

Definition at line 58 of file sam\_ba\_wire.cpp.

Referenced by wire\_getdata\_xmd(), and wire\_putdata\_xmd().

#### 5.12.2.7 idx\_rx\_read

```
volatile uint8_t idx_rx_read
```

Definition at line 49 of file sam\_ba\_wire.cpp.

Referenced by wire\_readc().

#### 5.12.2.8 idx\_rx\_write

```
volatile uint8_t idx_rx_write
```

Definition at line 50 of file sam\_ba\_wire.cpp.

#### 5.12.2.9 idx\_tx\_read

```
volatile uint8_t idx_tx_read
```

Definition at line 54 of file sam\_ba\_wire.cpp.

#### 5.12.2.10 idx\_tx\_write

```
volatile uint8_t idx_tx_write
```

Definition at line 55 of file sam\_ba\_wire.cpp.

#### 5.12.2.11 mode\_of\_transfer

```
uint8_t mode_of_transfer
```

Definition at line 60 of file sam\_ba\_wire.cpp.

Referenced by wire\_getdata\_xmd(), and wire\_putdata\_xmd().



## 5.12.2.12 size\_of\_data

```
uint16_t size_of_data
```

Definition at line 59 of file sam\_ba\_wire.cpp.

Referenced by wire\_getdata\_xmd(), and wire\_putdata\_xmd().

## 5.12.2.13 wire

```
TwoWire* wire
```

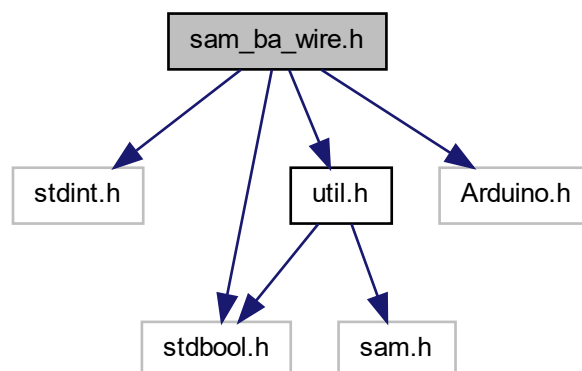
Definition at line 69 of file sam\_ba\_wire.cpp.

Referenced by wire\_close(), wire\_getc(), wire\_is\_rx\_ready(), wire\_open(), wire\_putc(), and wire\_setup().

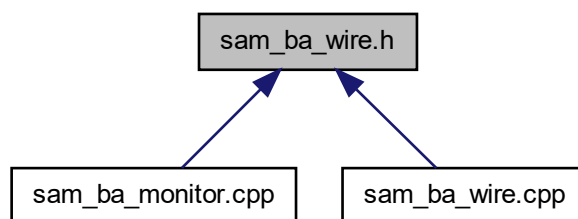
## 5.13 sam\_ba\_wire.h File Reference

```
#include <stdint.h>
#include <stdbool.h>
#include "util.h"
#include "Arduino.h"
```

Include dependency graph for sam\_ba\_wire.h:



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



## Macros

- #define [WIRE\\_BUFFER\\_SIZE](#) (128)
- #define [WIRE\\_DEFAULT\\_TIMEOUT](#) (1000)
- #define [CRC16POLY](#) (0x1021)
- #define [SHARP\\_CHARACTER](#) '#' /\* 0x23 : 35\*/
- #define [BOSSAC\\_ADDRESS](#) 0x23
- #define [SOH](#) (0x01)
- #define [EOT](#) (0x04)
- #define [ACK](#) (0x06)
- #define [NAK](#) (0x15)
- #define [CAN](#) (0x18)
- #define [ESC](#) (0x1b)
- #define [PKTLEN\\_128](#) (128)

## Functions

- void [wire\\_setup](#) (TwoWire &Mywire, unsigned int Myaddress, unsigned int Myaddress\_Bossac)
- void [wire\\_open](#) (unsigned int fBaudSpeed)  
*Open the given USART.*
- void [wire\\_close](#) (void)  
*Stops the USART.*
- int [wire\\_putc](#) (int value)  
*Puts a byte on usart line.*
- int [wire\\_getc](#) (void)  
*Waits and gets a value on usart line.*
- int [wire\\_sharp\\_received](#) (void)  
*Returns true if the SAM-BA Uart received the sharp char.*
- bool [wire\\_is\\_rx\\_ready](#) (void)  
*This function checks if a character has been received on the usart line.*
- int [wire\\_readc](#) (void)  
*Gets a value on usart line.*
- uint32\_t [wire\\_putdata](#) (void const \*data, uint32\_t length)  
*Send buffer on usart line.*
- uint32\_t [wire\\_getdata](#) (void \*data, uint32\_t length)

*Gets data from usart line.*

- uint32\_t [wire\\_putdata\\_xmd](#) (void const \*data, uint32\_t length)

*Send buffer on usart line using Xmodem protocol.*

- uint32\_t [wire\\_getdata\\_xmd](#) (void \*data, uint32\_t length)

*Gets data from usart line using Xmodem protocol.*

- unsigned short [wire\\_add\\_crc](#) (char c, unsigned short crc)

*Compute the CRC.*

### 5.13.1 Macro Definition Documentation

#### 5.13.1.1 ACK

```
#define ACK (0x06)
```

Definition at line 47 of file sam\_ba\_wire.h.

#### 5.13.1.2 BOSSAC\_ADDRESS

```
#define BOSSAC_ADDRESS 0x23
```

Definition at line 40 of file sam\_ba\_wire.h.

#### 5.13.1.3 CAN

```
#define CAN (0x18)
```

Definition at line 49 of file sam\_ba\_wire.h.

#### 5.13.1.4 CRC16POLY

```
#define CRC16POLY (0x1021)
```

Definition at line 37 of file sam\_ba\_wire.h.

#### 5.13.1.5 EOT

```
#define EOT (0x04)
```

Definition at line 46 of file sam\_ba\_wire.h.

#### 5.13.1.6 ESC

```
#define ESC (0x1b)
```

Definition at line 50 of file sam\_ba\_wire.h.

#### 5.13.1.7 NAK

```
#define NAK (0x15)
```

Definition at line 48 of file sam\_ba\_wire.h.

#### 5.13.1.8 PKTLEN\_128

```
#define PKTLEN_128 (128)
```

Definition at line 52 of file sam\_ba\_wire.h.

#### 5.13.1.9 SHARP\_CHARACTER

```
#define SHARP_CHARACTER '#' /* 0x23 : 35*/
```

Definition at line 39 of file sam\_ba\_wire.h.

#### 5.13.1.10 SOH

```
#define SOH (0x01)
```

Definition at line 44 of file sam\_ba\_wire.h.

#### 5.13.1.11 WIRE\_BUFFER\_SIZE

```
#define WIRE_BUFFER_SIZE (128)
```

Definition at line 30 of file sam\_ba\_wire.h.

Referenced by wire\_readc().

#### 5.13.1.12 WIRE\_DEFAULT\_TIMEOUT

```
#define WIRE_DEFAULT_TIMEOUT (1000)
```

Definition at line 33 of file sam\_ba\_wire.h.

### 5.13.2 Function Documentation

#### 5.13.2.1 wire\_add\_crc()

```
unsigned short wire_add_crc (  
    char c,  
    unsigned short crc )
```

Compute the CRC.

**Parameters**

<i>Char</i>	to add to CRC
<i>Previous</i>	CRC

**Returns**

The new computed CRC

Definition at line 217 of file sam\_ba\_wire.cpp.

**5.13.2.2 wire\_close()**

```
void wire_close (
    void )
```

Stops the USART.

Stops the USART.

Definition at line 94 of file sam\_ba\_wire.cpp.

References wire.

**5.13.2.3 wire\_getc()**

```
int wire_getc (
    void )
```

Waits and gets a value on usart line.

**Returns**

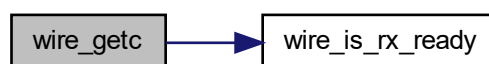
value read on usart line

Definition at line 119 of file sam\_ba\_wire.cpp.

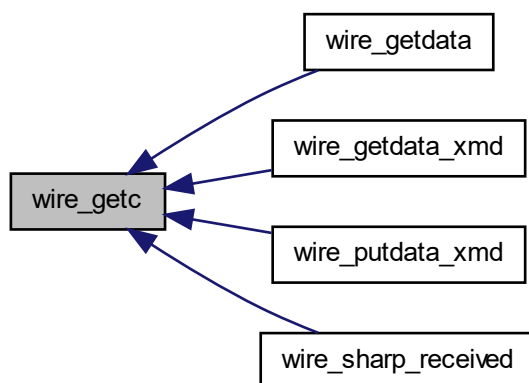
References wire, and wire\_is\_rx\_ready().

Referenced by wire\_getdata(), wire\_getdata\_xmd(), wire\_putdata\_xmd(), and wire\_sharp\_received().

Here is the call graph for this function:



Here is the caller graph for this function:



#### 5.13.2.4 wire\_getdata()

```

uint32_t wire_getdata (
    void * data,
    uint32_t length )
  
```

Gets data from usart line.

##### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

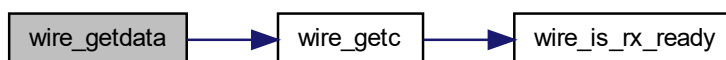
##### Returns

value read on usart line

Definition at line 170 of file sam\_ba\_wire.cpp.

References data, and wire\_getc().

Here is the call graph for this function:



### 5.13.2.5 wire\_getdata\_xmd()

```
uint32_t wire_getdata_xmd (
    void * data,
    uint32_t length )
```

Gets data from usart line using Xmodem protocol.

#### Parameters

<i>data</i>	pointer
<i>number</i>	of data to get

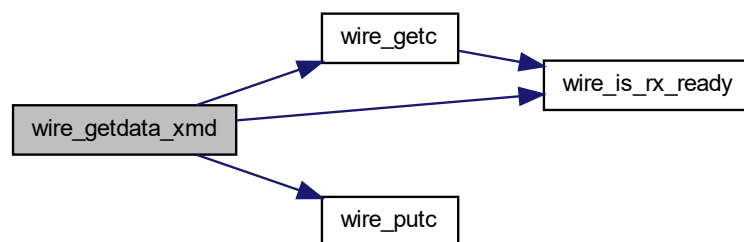
#### Returns

value read on usart line

Definition at line 424 of file sam\_ba\_wire.cpp.

References data, error\_timeout, length, mode\_of\_transfer, ptr\_data, size\_of\_data, SOH, wire\_getc(), wire\_is\_rx\_ready(), and wire\_putc().

Here is the call graph for this function:



### 5.13.2.6 wire\_is\_rx\_ready()

```
bool wire_is_rx_ready (
    void )
```

This function checks if a character has been received on the usart line.

**Returns**

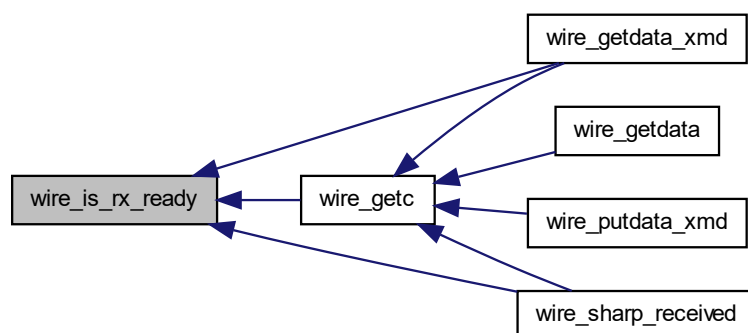
- 1 if a byte is ready to be read.

Definition at line 140 of file sam\_ba\_wire.cpp.

References wire.

Referenced by wire\_getc(), wire\_getdata\_xmd(), and wire\_sharp\_received().

Here is the caller graph for this function:

**5.13.2.7 wire\_open()**

```
void wire_open (
    unsigned int fBaudSpeed )
```

Open the given USART.

Definition at line 79 of file sam\_ba\_wire.cpp.

References address, and wire.

**5.13.2.8 wire\_putc()**

```
int wire_putc (
    int value )
```

Puts a byte on usart line.

**Parameters**

<i>value</i>	Value to put
--------------	--------------



**Returns**

1 if function was successfully done, otherwise 0.

Puts a byte on usart line.

**Parameters**

<i>value</i>	Value to put
--------------	--------------

**Returns**

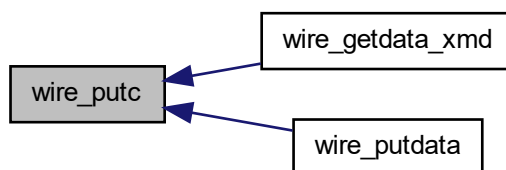
1 if function was successfully done, otherwise 0.

Definition at line 107 of file sam\_ba\_wire.cpp.

References address\_Bossac, and wire.

Referenced by wire\_getdata\_xmd(), and wire\_putdata().

Here is the caller graph for this function:

**5.13.2.9 wire\_putdata()**

```
uint32_t wire_putdata (
    void const * data,
    uint32_t length )
```

Send buffer on usart line.

**Parameters**

<i>data</i>	pointer
<i>number</i>	of data to send

**Returns**

number of data sent

Definition at line 156 of file sam\_ba\_wire.cpp.

References data, i, length, and wire\_putc().

Here is the call graph for this function:

**5.13.2.10 wire\_putdata\_xmd()**

```

uint32_t wire_putdata_xmd (
    void const * data,
    uint32_t length )
  
```

Send buffer on usart line using Xmodem protocol.

**Parameters**

<i>data</i>	pointer
<i>number</i>	of data to send

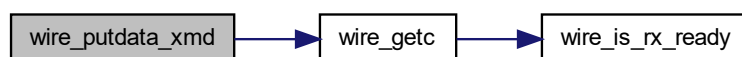
**Returns**

number of data sent

Definition at line 292 of file sam\_ba\_wire.cpp.

References data, error\_timeout, length, mode\_of\_transfer, NAK, PKTLEN\_128, ptr\_data, size\_of\_data, and wire\_getc().

Here is the call graph for this function:



#### 5.13.2.11 wire\_readc()

```
int wire_readc (
    void )
```

Gets a value on usart line.

##### Returns

value read on usart line

Definition at line 147 of file sam\_ba\_wire.cpp.

References `buffer_rx_wire`, `idx_rx_read`, and `WIRE_BUFFER_SIZE`.

#### 5.13.2.12 wire\_setup()

```
void wire_setup (
    TwoWire & Mywire,
    unsigned int Myaddress,
    unsigned int Myaddress_Bossac )
```

Definition at line 70 of file sam\_ba\_wire.cpp.

References `address`, `address_Bossac`, and `wire`.

#### 5.13.2.13 wire\_sharp\_received()

```
int wire_sharp_received (
    void )
```

Returns true if the SAM-BA Uart received the sharp char.

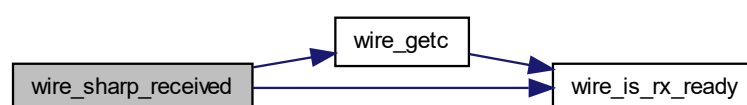
##### Returns

Returns true if the SAM-BA Uart received the sharp char

Definition at line 130 of file sam\_ba\_wire.cpp.

References `SHARP_CHARACTER`, `wire_getc()`, and `wire_is_rx_ready()`.

Here is the call graph for this function:

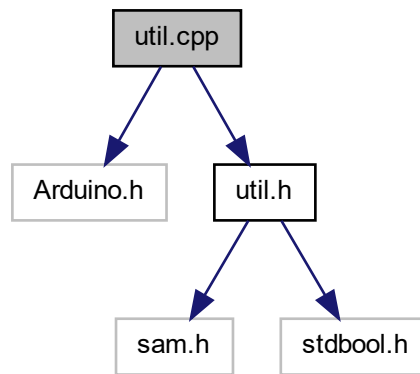


## 5.14 util.cpp File Reference

```
#include "Arduino.h"
```

```
#include "util.h"
```

Include dependency graph for util.cpp:



### Macros

- `#define Get\_sys\_count\(\) ( (SysTick->VAL) & SysTick_VAL_CURRENT_Msk )`

### Functions

- void [flashErase](#) (uint32\_t startAddress)
- void [flashWrite](#) (uint32\_t numBytes, uint32\_t \*buffer, uint32\_t \*[ptr\\_data](#))
- void [pinMux](#) (uint32\_t pinmux)
- void [pinConfig](#) (uint8\_t port, uint8\_t pin, uint8\_t config)
- bool [isPinActive](#) (uint8\_t port, uint8\_t pin, uint8\_t config)
- void [delayUs](#) (unsigned int delay)
- void [systemReset](#) (void)
- void [waitForSync](#) (void)

### Variables

- uint32\_t [\\_\\_sketch\\_vectors\\_ptr](#)

#### 5.14.1 Macro Definition Documentation

#### 5.14.1.1 Get\_sys\_count

```
#define Get_sys_count( ) ( (SysTick->VAL) & SysTick_VAL_CURRENT_Msk )
```

Definition at line 166 of file util.cpp.

### 5.14.2 Function Documentation

#### 5.14.2.1 delayUs()

```
void delayUs (
    unsigned int delay )
```

Definition at line 168 of file util.cpp.

References [i](#).

#### 5.14.2.2 flashErase()

```
void flashErase (
    uint32_t startAddress )
```

Definition at line 48 of file util.cpp.

#### 5.14.2.3 flashWrite()

```
void flashWrite (
    uint32_t numBytes,
    uint32_t * buffer,
    uint32_t * ptr_data )
```

Definition at line 70 of file util.cpp.

References [i](#), and [ptr\\_data](#).

#### 5.14.2.4 isPinActive()

```
bool isPinActive (
    uint8_t port,
    uint8_t pin,
    uint8_t config )
```

Definition at line 155 of file util.cpp.

References [PIN\\_POLARITY\\_ACTIVE\\_LOW](#).

#### 5.14.2.5 pinConfig()

```
void pinConfig (
    uint8_t port,
    uint8_t pin,
    uint8_t config )
```

Definition at line 130 of file util.cpp.

References INPUT, INPUT\_PULLDOWN, INPUT\_PULLUP, OUTPUT\_HIGH, and OUTPUT\_LOW.

#### 5.14.2.6 pinMux()

```
void pinMux (
    uint32_t pinmux )
```

Definition at line 114 of file util.cpp.

References PINMUX\_UNUSED.

Referenced by USB\_Init().

Here is the caller graph for this function:



#### 5.14.2.7 systemReset()

```
void systemReset (
    void )
```

Definition at line 261 of file util.cpp.

References SCB\_AIRCR\_VECTKEY\_Val.

## 5.14.2.8 waitForSync()

```
void waitForSync (
    void )
```

Definition at line 269 of file util.cpp.

Referenced by USB\_Init().

Here is the caller graph for this function:



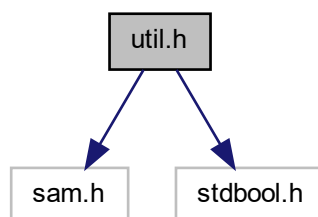
## 5.14.3 Variable Documentation

## 5.14.3.1 \_\_sketch\_vectors\_ptr

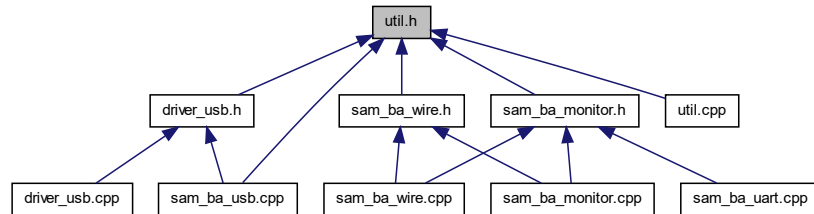
```
uint32_t __sketch_vectors_ptr
```

## 5.15 util.h File Reference

```
#include "sam.h"
#include <stdbool.h>
Include dependency graph for util.h:
```



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



## Macros

- `#define SYSTICK_NUMBER_CYCLE 1`
- `#define TRUE (1==1)`
- `#define FALSE (1==0)`
- `#define APP_START 0x00002000`
- `#define SCB_AIRCR_VECTKEY_Val 0x05FA`
- `#define INPUT (0x0)`
- `#define OUTPUT (0x1)`
- `#define INPUT_PULLUP (0x2)`
- `#define INPUT_PULLDOWN (0x3)`
- `#define OUTPUT_LOW (0x4)`
- `#define OUTPUT_HIGH (0x5)`
- `#define PINMUX_UNUSED 0xFFFFFFFF`
- `#define LED_POLARITY_LOW_ON 0`
- `#define LED_POLARITY_HIGH_ON 1`
- `#define PIN_POLARITY_ACTIVE_LOW 0`
- `#define PIN_POLARITY_ACTIVE_HIGH 1`
- `#define PIN_POLARITY_USBCDC_LOW 0`
- `#define PIN_POLARITY_USBCDC_HIGH 1`
- `#define USB_VID_HIGH 0x23`
- `#define USB_VID_LOW 0x41`
- `#define USB_PID_HIGH 0x00`
- `#define USB_PID_LOW 0x4D`

## Functions

- void `flashErase` (uint32\_t startAddress)
- void `flashWrite` (uint32\_t startAddress, uint32\_t \*buffer, uint32\_t \*ptr\_data)
- void `pinMux` (uint32\_t pinmux)
- void `systemReset` (void)
- void `pinConfig` (uint8\_t port, uint8\_t pin, uint8\_t config)
- bool `isPinActive` (uint8\_t port, uint8\_t pin, uint8\_t config)
- void `delayUs` (unsigned int delay)
- void `waitForSync` (void)

## Variables

- unsigned int `s_fcpu_hz`
- uint32\_t `__sketch_vectors_ptr`



### 5.15.1 Macro Definition Documentation

#### 5.15.1.1 APP\_START

```
#define APP_START 0x00002000
```

Definition at line 50 of file util.h.

#### 5.15.1.2 FALSE

```
#define FALSE (1==0)
```

Definition at line 44 of file util.h.

#### 5.15.1.3 INPUT

```
#define INPUT (0x0)
```

Definition at line 71 of file util.h.

Referenced by pinConfig().

#### 5.15.1.4 INPUT\_PULLDOWN

```
#define INPUT_PULLDOWN (0x3)
```

Definition at line 74 of file util.h.

Referenced by pinConfig().

#### 5.15.1.5 INPUT\_PULLUP

```
#define INPUT_PULLUP (0x2)
```

Definition at line 73 of file util.h.

Referenced by pinConfig().

#### 5.15.1.6 LED\_POLARITY\_HIGH\_ON

```
#define LED_POLARITY_HIGH_ON 1
```

Definition at line 81 of file util.h.

#### 5.15.1.7 LED\_POLARITY\_LOW\_ON

```
#define LED_POLARITY_LOW_ON 0
```

Definition at line 80 of file util.h.

#### 5.15.1.8 OUTPUT

```
#define OUTPUT (0x1)
```

Definition at line 72 of file util.h.

#### 5.15.1.9 OUTPUT\_HIGH

```
#define OUTPUT_HIGH (0x5)
```

Definition at line 76 of file util.h.

Referenced by pinConfig().

#### 5.15.1.10 OUTPUT\_LOW

```
#define OUTPUT_LOW (0x4)
```

Definition at line 75 of file util.h.

Referenced by pinConfig().

#### 5.15.1.11 PIN\_POLARITY\_ACTIVE\_HIGH

```
#define PIN_POLARITY_ACTIVE_HIGH 1
```

Definition at line 83 of file util.h.

#### 5.15.1.12 PIN\_POLARITY\_ACTIVE\_LOW

```
#define PIN_POLARITY_ACTIVE_LOW 0
```

Definition at line 82 of file util.h.

Referenced by isPinActive().

#### 5.15.1.13 PIN\_POLARITY\_USBCDC\_HIGH

```
#define PIN_POLARITY_USBCDC_HIGH 1
```

Definition at line 85 of file util.h.

#### 5.15.1.14 PIN\_POLARITY\_USBCDC\_LOW

```
#define PIN_POLARITY_USBCDC_LOW 0
```

Definition at line 84 of file util.h.

#### 5.15.1.15 PINMUX\_UNUSED

```
#define PINMUX_UNUSED 0xFFFFFFFF
```

Definition at line 78 of file util.h.

Referenced by pinMux().

#### 5.15.1.16 SCB\_AIRCR\_VECTKEY\_Val

```
#define SCB_AIRCR_VECTKEY_Val 0x05FA
```

Definition at line 59 of file util.h.

Referenced by systemReset().

#### 5.15.1.17 SYSTICK\_NUMBER\_CYCLE

```
#define SYSTICK_NUMBER_CYCLE 1
```

Definition at line 30 of file util.h.

**5.15.1.18 TRUE**

```
#define TRUE (1==1)
```

Definition at line 43 of file util.h.

**5.15.1.19 USB\_PID\_HIGH**

```
#define USB_PID_HIGH 0x00
```

Definition at line 89 of file util.h.

**5.15.1.20 USB\_PID\_LOW**

```
#define USB_PID_LOW 0x4D
```

Definition at line 90 of file util.h.

**5.15.1.21 USB\_VID\_HIGH**

```
#define USB_VID_HIGH 0x23
```

Definition at line 87 of file util.h.

**5.15.1.22 USB\_VID\_LOW**

```
#define USB_VID_LOW 0x41
```

Definition at line 88 of file util.h.

**5.15.2 Function Documentation****5.15.2.1 delayUs()**

```
void delayUs (
    unsigned int delay )
```

Definition at line 168 of file util.cpp.

References i.

#### 5.15.2.2 flashErase()

```
void flashErase (
    uint32_t startAddress )
```

Definition at line 48 of file util.cpp.

#### 5.15.2.3 flashWrite()

```
void flashWrite (
    uint32_t startAddress,
    uint32_t * buffer,
    uint32_t * ptr_data )
```

Definition at line 70 of file util.cpp.

References `i`, and `ptr_data`.

#### 5.15.2.4 isPinActive()

```
bool isPinActive (
    uint8_t port,
    uint8_t pin,
    uint8_t config )
```

Definition at line 155 of file util.cpp.

References `PIN_POLARITY_ACTIVE_LOW`.

#### 5.15.2.5 pinConfig()

```
void pinConfig (
    uint8_t port,
    uint8_t pin,
    uint8_t config )
```

Definition at line 130 of file util.cpp.

References `INPUT`, `INPUT_PULLDOWN`, `INPUT_PULLUP`, `OUTPUT_HIGH`, and `OUTPUT_LOW`.

#### 5.15.2.6 pinMux()

```
void pinMux (
    uint32_t pinmux )
```

Definition at line 114 of file util.cpp.

References PINMUX\_UNUSED.

Referenced by USB\_Init().

Here is the caller graph for this function:



#### 5.15.2.7 systemReset()

```
void systemReset (
    void )
```

Definition at line 261 of file util.cpp.

References SCB\_AIRCR\_VECTKEY\_Val.

#### 5.15.2.8 waitForSync()

```
void waitForSync (
    void )
```

Definition at line 269 of file util.cpp.

Referenced by USB\_Init().

Here is the caller graph for this function:



### 5.15.3 Variable Documentation

#### 5.15.3.1 \_\_sketch\_vectors\_ptr

uint32\_t \_\_sketch\_vectors\_ptr

#### 5.15.3.2 s\_fcpu\_hz

unsigned int s\_fcpu\_hz





## Index

- `_MONITOR_SAM_BA_H_`
    - `sam_ba_monitor.h`, [24](#)
  - `__attribute__`
    - `driver_usb.cpp`, [7](#)
  - `__sketch_vectors_ptr`
    - `util.cpp`, [77](#)
    - `util.h`, [85](#)
- ACK
  - `sam_ba_uart.h`, [40](#)
  - `sam_ba_wire.h`, [65](#)
- APP\_START
  - `util.h`, [79](#)
- address
  - `sam_ba_wire.cpp`, [61](#)
- `address_Bossac`
  - `sam_ba_wire.cpp`, [61](#)
- `b_sam_ba_interface_usart`
  - `sam_ba_monitor.cpp`, [20](#)
- `b_sam_ba_interface_wire`
  - `sam_ba_monitor.cpp`, [20](#)
- `b_sharp_received`
  - `sam_ba_wire.cpp`, [61](#)
- BOSSAC\_ADDRESS
  - `sam_ba_wire.h`, [65](#)
- `buffer_rx_usart`
  - `sam_ba_uart.cpp`, [36](#)
- `buffer_rx_wire`
  - `sam_ba_wire.cpp`, [61](#)
- `buffer_tx_wire`
  - `sam_ba_wire.cpp`, [61](#)
- CAN
  - `sam_ba_uart.h`, [40](#)
  - `sam_ba_wire.h`, [65](#)
- CRC16POLY
  - `sam_ba_uart.h`, [40](#)
  - `sam_ba_wire.h`, [65](#)
- `call_applet`
  - `sam_ba_monitor.cpp`, [18](#)
  - `sam_ba_monitor.h`, [25](#)
- `command`
  - `sam_ba_monitor.cpp`, [20](#)
- `current_number`
  - `sam_ba_monitor.cpp`, [20](#)
- data
  - `sam_ba_monitor.cpp`, [20](#)
- `delayUs`
  - `util.cpp`, [75](#)
  - `util.h`, [82](#)
- `driver_usb.cpp`, [5](#)
  - `__attribute__`, [7](#)
  - `NVM_USB_PAD_TRANSN_POS`, [6](#)
  - `NVM_USB_PAD_TRANSN_SIZE`, [6](#)
  - `NVM_USB_PAD_TRANSP_POS`, [6](#)
  - `NVM_USB_PAD_TRANSP_SIZE`, [6](#)
  - `NVM_USB_PAD_TRIM_POS`, [6](#)
  - `NVM_USB_PAD_TRIM_SIZE`, [7](#)
  - `USB_Configure`, [8](#)
  - `USB_Init`, [8](#)
  - `USB_IsConfigured`, [8](#)
  - `USB_Open`, [9](#)
  - `USB_PAD_TRANSN_REG_POS`, [7](#)
  - `USB_PAD_TRANSP_REG_POS`, [7](#)
  - `USB_PAD_TRIM_REG_POS`, [7](#)
  - `USB_Read`, [9](#)
  - `USB_Read_blocking`, [9](#)
  - `USB_SendStall`, [10](#)
  - `USB_SendZlp`, [10](#)
  - `USB_SetAddress`, [10](#)
  - `USB_Write`, [10](#)
- `driver_usb.h`, [11](#)
  - `USB_Configure`, [12](#)
  - `USB_Init`, [12](#)
  - `USB_IsConfigured`, [12](#)
  - `USB_Open`, [13](#)
  - `USB_Read`, [13](#)
  - `USB_Read_blocking`, [13](#)
  - `USB_SendStall`, [14](#)
  - `USB_SendZlp`, [14](#)
  - `USB_SetAddress`, [14](#)
  - `USB_Write`, [14](#)
  - `udd_ep_in_cache_buffer`, [15](#)
  - `udd_ep_out_cache_buffer`, [15](#)
  - `usb_endpoint_table`, [15](#)
- EOT
  - `sam_ba_uart.h`, [41](#)
  - `sam_ba_wire.h`, [65](#)
- ESC
  - `sam_ba_uart.h`, [41](#)
  - `sam_ba_wire.h`, [65](#)
- `error_timeout`
  - `sam_ba_wire.cpp`, [61](#)
- FALSE
  - `util.h`, [79](#)
- `flashErase`
  - `util.cpp`, [75](#)
  - `util.h`, [82](#)
- `flashWrite`
  - `util.cpp`, [75](#)
  - `util.h`, [83](#)
- `get_c`
  - `t_monitor_if`, [3](#)
- `Get_sys_count`
  - `util.cpp`, [74](#)
- `getdata`
  - `t_monitor_if`, [4](#)

- getdata\_xmd
  - t\_monitor\_if, 4
- i
  - sam\_ba\_monitor.cpp, 21
- INPUT\_PULLDOWN
  - util.h, 79
- INPUT\_PULLUP
  - util.h, 79
- INPUT
  - util.h, 79
- idx\_rx\_read
  - sam\_ba\_wire.cpp, 62
- idx\_rx\_write
  - sam\_ba\_wire.cpp, 62
- idx\_tx\_read
  - sam\_ba\_wire.cpp, 62
- idx\_tx\_write
  - sam\_ba\_wire.cpp, 62
- is\_rx\_ready
  - t\_monitor\_if, 4
- isPinActive
  - util.cpp, 75
  - util.h, 83
- j
  - sam\_ba\_monitor.cpp, 21
- LED\_POLARITY\_HIGH\_ON
  - util.h, 79
- LED\_POLARITY\_LOW\_ON
  - util.h, 80
- length
  - sam\_ba\_monitor.cpp, 21
- mode\_of\_transfer
  - sam\_ba\_wire.cpp, 62
- NAK
  - sam\_ba\_uart.h, 41
  - sam\_ba\_wire.h, 66
- NVM\_USB\_PAD\_TRANSN\_POS
  - driver\_usb.cpp, 6
- NVM\_USB\_PAD\_TRANSN\_SIZE
  - driver\_usb.cpp, 6
- NVM\_USB\_PAD\_TRANSP\_POS
  - driver\_usb.cpp, 6
- NVM\_USB\_PAD\_TRANSP\_SIZE
  - driver\_usb.cpp, 6
- NVM\_USB\_PAD\_TRIM\_POS
  - driver\_usb.cpp, 6
- NVM\_USB\_PAD\_TRIM\_SIZE
  - driver\_usb.cpp, 7
- OUTPUT\_HIGH
  - util.h, 80
- OUTPUT\_LOW
  - util.h, 80
- OUTPUT
  - util.h, 80
- PIN\_POLARITY\_ACTIVE\_HIGH
  - util.h, 80
- PIN\_POLARITY\_ACTIVE\_LOW
  - util.h, 80
- PIN\_POLARITY\_USBCDC\_HIGH
  - util.h, 81
- PIN\_POLARITY\_USBCDC\_LOW
  - util.h, 81
- PINMUX\_UNUSED
  - util.h, 81
- PKTLEN\_128
  - sam\_ba\_uart.h, 41
  - sam\_ba\_wire.h, 66
- pinConfig
  - util.cpp, 75
  - util.h, 83
- pinMux
  - util.cpp, 76
  - util.h, 83
- ptr
  - sam\_ba\_monitor.cpp, 21
- ptr\_data
  - sam\_ba\_monitor.cpp, 21
- ptr\_monitor\_if
  - sam\_ba\_monitor.cpp, 21
- put\_c
  - t\_monitor\_if, 4
- putdata
  - t\_monitor\_if, 4
- putdata\_xmd
  - t\_monitor\_if, 4
- README.md, 15
- RomBOOT\_Version
  - sam\_ba\_monitor.cpp, 22
- s\_fcpu\_hz
  - util.h, 85
- SAM\_BA\_BOTH\_INTERFACES
  - sam\_ba\_monitor.h, 24
- SAM\_BA\_INTERFACE\_USART
  - sam\_ba\_monitor.h, 24
- SAM\_BA\_INTERFACE\_USBCDC
  - sam\_ba\_monitor.h, 24
- SAM\_BA\_INTERFACE\_WIRE
  - sam\_ba\_monitor.h, 24
- SAM\_BA\_INTERFACE
  - sam\_ba\_monitor.h, 24
- SAM\_BA\_NONE
  - sam\_ba\_monitor.h, 24
- SAM\_BA\_UART\_ONLY
  - sam\_ba\_monitor.h, 25
- SAM\_BA\_USBCDC\_ONLY
  - sam\_ba\_monitor.h, 25
- SAM\_BA\_VERSION
  - sam\_ba\_monitor.h, 25
- SCB\_AIRCR\_VECTKEY\_Val
  - util.h, 81
- SHARP\_CHARACTER

- sam\_ba\_uart.h, 41
- sam\_ba\_wire.h, 66
- SIZEBUFMAX
  - sam\_ba\_monitor.h, 25
- SOH
  - sam\_ba\_uart.h, 41
  - sam\_ba\_wire.h, 66
- SYSTICK\_NUMBER\_CYCLE
  - util.h, 81
- sam\_ba\_cdc.cpp, 15
- sam\_ba\_cdc.h, 16
- sam\_ba\_monitor.cpp, 16
  - b\_sam\_ba\_interface\_usart, 20
  - b\_sam\_ba\_interface\_wire, 20
  - call\_applet, 18
  - command, 20
  - current\_number, 20
  - data, 20
  - i, 21
  - j, 21
  - length, 21
  - ptr, 21
  - ptr\_data, 21
  - ptr\_monitor\_if, 21
  - RomBOOT\_Version, 22
  - sam\_ba\_monitor\_init, 18
  - sam\_ba\_monitor\_run, 18
  - sam\_ba\_monitor\_sys\_tick, 18
  - sam\_ba\_putdata\_term, 19
  - sp, 22
  - TX\_RX\_LED\_PULSE\_PERIOD, 18
  - u32tmp, 22
- sam\_ba\_monitor.h, 22
  - \_MONITOR\_SAM\_BA\_H\_, 24
  - call\_applet, 25
  - SAM\_BA\_BOTH\_INTERFACES, 24
  - SAM\_BA\_INTERFACE\_USART, 24
  - SAM\_BA\_INTERFACE\_USBCDC, 24
  - SAM\_BA\_INTERFACE\_WIRE, 24
  - SAM\_BA\_INTERFACE, 24
  - SAM\_BA\_NONE, 24
  - SAM\_BA\_UART\_ONLY, 25
  - SAM\_BA\_USBCDC\_ONLY, 25
  - SAM\_BA\_VERSION, 25
  - SIZEBUFMAX, 25
  - sam\_ba\_monitor\_init, 25
  - sam\_ba\_monitor\_run, 26
  - sam\_ba\_monitor\_sys\_tick, 26
  - sam\_ba\_putdata\_term, 26
  - uart\_if, 27
- sam\_ba\_monitor\_init
  - sam\_ba\_monitor.cpp, 18
  - sam\_ba\_monitor.h, 25
- sam\_ba\_monitor\_run
  - sam\_ba\_monitor.cpp, 18
  - sam\_ba\_monitor.h, 26
- sam\_ba\_monitor\_sys\_tick
  - sam\_ba\_monitor.cpp, 18
- sam\_ba\_monitor.h, 26
- sam\_ba\_putdata\_term
  - sam\_ba\_monitor.cpp, 19
  - sam\_ba\_monitor.h, 26
- sam\_ba\_uart.cpp, 27
  - buffer\_rx\_usart, 36
  - serial, 36
  - uart\_add\_crc, 28
  - uart\_b\_sharp\_received, 36
  - uart\_buffer\_tx\_usart, 37
  - uart\_close, 29
  - uart\_error\_timeout, 37
  - uart\_getc, 29
  - uart\_getdata, 30
  - uart\_getdata\_xmd, 31
  - uart\_idx\_rx\_read, 37
  - uart\_idx\_rx\_write, 37
  - uart\_idx\_tx\_read, 37
  - uart\_idx\_tx\_write, 37
  - uart\_if, 38
  - uart\_is\_rx\_ready, 31
  - uart\_mode\_of\_transfer, 38
  - uart\_open, 32
  - uart\_putc, 33
  - uart\_putdata, 34
  - uart\_putdata\_xmd, 34
  - uart\_readc, 35
  - uart\_setup, 35
  - uart\_sharp\_received, 35
  - uart\_size\_of\_data, 38
- sam\_ba\_uart.h, 39
  - ACK, 40
  - CAN, 40
  - CRC16POLY, 40
  - EOT, 41
  - ESC, 41
  - NAK, 41
  - PKTLEN\_128, 41
  - SHARP\_CHARACTER, 41
  - SOH, 41
  - USART\_BUFFER\_SIZE, 42
  - USART\_DEFAULT\_TIMEOUT, 42
  - uart\_add\_crc, 42
  - uart\_close, 42
  - uart\_getc, 43
  - uart\_getdata, 44
  - uart\_getdata\_xmd, 45
  - uart\_is\_rx\_ready, 45
  - uart\_open, 46
  - uart\_putc, 47
  - uart\_putdata, 48
  - uart\_putdata\_xmd, 48
  - uart\_readc, 49
  - uart\_setup, 49
  - uart\_sharp\_received, 50
- sam\_ba\_usb.cpp, 50
- sam\_ba\_usb.h, 51
- sam\_ba\_wire.cpp, 52

- address, 61
- address\_Bossac, 61
- b\_sharp\_received, 61
- buffer\_rx\_wire, 61
- buffer\_tx\_wire, 61
- error\_timeout, 61
- idx\_rx\_read, 62
- idx\_rx\_write, 62
- idx\_tx\_read, 62
- idx\_tx\_write, 62
- mode\_of\_transfer, 62
- size\_of\_data, 62
- wire, 63
- wire\_add\_crc, 53
- wire\_close, 54
- wire\_getc, 54
- wire\_getdata, 55
- wire\_getdata\_xmd, 56
- wire\_is\_rx\_ready, 56
- wire\_open, 57
- wire\_putc, 57
- wire\_putdata, 58
- wire\_putdata\_xmd, 59
- wire\_readc, 59
- wire\_setup, 60
- wire\_sharp\_received, 60
- sam\_ba\_wire.h, 63
- ACK, 65
- BOSSAC\_ADDRESS, 65
- CAN, 65
- CRC16POLY, 65
- EOT, 65
- ESC, 65
- NAK, 66
- PKTLEN\_128, 66
- SHARP\_CHARACTER, 66
- SOH, 66
- WIRE\_BUFFER\_SIZE, 66
- WIRE\_DEFAULT\_TIMEOUT, 66
- wire\_add\_crc, 66
- wire\_close, 67
- wire\_getc, 67
- wire\_getdata, 68
- wire\_getdata\_xmd, 69
- wire\_is\_rx\_ready, 69
- wire\_open, 70
- wire\_putc, 70
- wire\_putdata, 71
- wire\_putdata\_xmd, 72
- wire\_readc, 72
- wire\_setup, 73
- wire\_sharp\_received, 73
- serial
  - sam\_ba\_uart.cpp, 36
- size\_of\_data
  - sam\_ba\_wire.cpp, 62
- sp
  - sam\_ba\_monitor.cpp, 22
- systemReset
  - util.cpp, 76
  - util.h, 84
- t\_monitor\_if, 3
  - get\_c, 3
  - getdata, 4
  - getdata\_xmd, 4
  - is\_rx\_ready, 4
  - put\_c, 4
  - putdata, 4
  - putdata\_xmd, 4
- TRUE
  - util.h, 81
- TX\_RX\_LED\_PULSE\_PERIOD
  - sam\_ba\_monitor.cpp, 18
- u32tmp
  - sam\_ba\_monitor.cpp, 22
- USART\_BUFFER\_SIZE
  - sam\_ba\_uart.h, 42
- USART\_DEFAULT\_TIMEOUT
  - sam\_ba\_uart.h, 42
- USB\_Configure
  - driver\_usb.cpp, 8
  - driver\_usb.h, 12
- USB\_Init
  - driver\_usb.cpp, 8
  - driver\_usb.h, 12
- USB\_IsConfigured
  - driver\_usb.cpp, 8
  - driver\_usb.h, 12
- USB\_Open
  - driver\_usb.cpp, 9
  - driver\_usb.h, 13
- USB\_PAD\_TRANSN\_REG\_POS
  - driver\_usb.cpp, 7
- USB\_PAD TRANSP\_REG\_POS
  - driver\_usb.cpp, 7
- USB\_PAD\_TRIM\_REG\_POS
  - driver\_usb.cpp, 7
- USB\_PID\_HIGH
  - util.h, 82
- USB\_PID\_LOW
  - util.h, 82
- USB\_Read
  - driver\_usb.cpp, 9
  - driver\_usb.h, 13
- USB\_Read\_blocking
  - driver\_usb.cpp, 9
  - driver\_usb.h, 13
- USB\_SendStall
  - driver\_usb.cpp, 10
  - driver\_usb.h, 14
- USB\_SendZlp
  - driver\_usb.cpp, 10
  - driver\_usb.h, 14
- USB\_SetAddress
  - driver\_usb.cpp, 10

- driver\_usb.h, 14
- USB\_VID\_HIGH
  - util.h, 82
- USB\_VID\_LOW
  - util.h, 82
- USB\_Write
  - driver\_usb.cpp, 10
  - driver\_usb.h, 14
- uart\_add\_crc
  - sam\_ba\_uart.cpp, 28
  - sam\_ba\_uart.h, 42
- uart\_b\_sharp\_received
  - sam\_ba\_uart.cpp, 36
- uart\_buffer\_tx\_usart
  - sam\_ba\_uart.cpp, 37
- uart\_close
  - sam\_ba\_uart.cpp, 29
  - sam\_ba\_uart.h, 42
- uart\_error\_timeout
  - sam\_ba\_uart.cpp, 37
- uart\_getc
  - sam\_ba\_uart.cpp, 29
  - sam\_ba\_uart.h, 43
- uart\_getdata
  - sam\_ba\_uart.cpp, 30
  - sam\_ba\_uart.h, 44
- uart\_getdata\_xmd
  - sam\_ba\_uart.cpp, 31
  - sam\_ba\_uart.h, 45
- uart\_idx\_rx\_read
  - sam\_ba\_uart.cpp, 37
- uart\_idx\_rx\_write
  - sam\_ba\_uart.cpp, 37
- uart\_idx\_tx\_read
  - sam\_ba\_uart.cpp, 37
- uart\_idx\_tx\_write
  - sam\_ba\_uart.cpp, 37
- uart\_if
  - sam\_ba\_monitor.h, 27
  - sam\_ba\_uart.cpp, 38
- uart\_is\_rx\_ready
  - sam\_ba\_uart.cpp, 31
  - sam\_ba\_uart.h, 45
- uart\_mode\_of\_transfer
  - sam\_ba\_uart.cpp, 38
- uart\_open
  - sam\_ba\_uart.cpp, 32
  - sam\_ba\_uart.h, 46
- uart\_putc
  - sam\_ba\_uart.cpp, 33
  - sam\_ba\_uart.h, 47
- uart\_putdata
  - sam\_ba\_uart.cpp, 34
  - sam\_ba\_uart.h, 48
- uart\_putdata\_xmd
  - sam\_ba\_uart.cpp, 34
  - sam\_ba\_uart.h, 48
- uart\_readc
  - sam\_ba\_uart.cpp, 35
  - sam\_ba\_uart.h, 49
- uart\_setup
  - sam\_ba\_uart.cpp, 35
  - sam\_ba\_uart.h, 49
- uart\_sharp\_received
  - sam\_ba\_uart.cpp, 35
  - sam\_ba\_uart.h, 50
- uart\_size\_of\_data
  - sam\_ba\_uart.cpp, 38
- udd\_ep\_in\_cache\_buffer
  - driver\_usb.h, 15
- udd\_ep\_out\_cache\_buffer
  - driver\_usb.h, 15
- usb\_endpoint\_table
  - driver\_usb.h, 15
- util.cpp, 74
  - \_\_sketch\_vectors\_ptr, 77
  - delayUs, 75
  - flashErase, 75
  - flashWrite, 75
  - Get\_sys\_count, 74
  - isPinActive, 75
  - pinConfig, 75
  - pinMux, 76
  - systemReset, 76
  - waitForSync, 76
- util.h, 77
  - \_\_sketch\_vectors\_ptr, 85
  - APP\_START, 79
  - delayUs, 82
  - FALSE, 79
  - flashErase, 82
  - flashWrite, 83
  - INPUT\_PULLDOWN, 79
  - INPUT\_PULLUP, 79
  - INPUT, 79
  - isPinActive, 83
  - LED\_POLARITY\_HIGH\_ON, 79
  - LED\_POLARITY\_LOW\_ON, 80
  - OUTPUT\_HIGH, 80
  - OUTPUT\_LOW, 80
  - OUTPUT, 80
  - PIN\_POLARITY\_ACTIVE\_HIGH, 80
  - PIN\_POLARITY\_ACTIVE\_LOW, 80
  - PIN\_POLARITY\_USBCDC\_HIGH, 81
  - PIN\_POLARITY\_USBCDC\_LOW, 81
  - PINMUX\_UNUSED, 81
  - pinConfig, 83
  - pinMux, 83
  - s\_fcpu\_hz, 85
  - SCB\_AIRCR\_VECTKEY\_Val, 81
  - SYSTICK\_NUMBER\_CYCLE, 81
  - systemReset, 84
  - TRUE, 81
  - USB\_PID\_HIGH, 82
  - USB\_PID\_LOW, 82
  - USB\_VID\_HIGH, 82

- USB\_VID\_LOW, [82](#)
- waitForSync, [84](#)
- WIRE\_BUFFER\_SIZE
  - sam\_ba\_wire.h, [66](#)
- WIRE\_DEFAULT\_TIMEOUT
  - sam\_ba\_wire.h, [66](#)
- waitForSync
  - util.cpp, [76](#)
  - util.h, [84](#)
- wire
  - sam\_ba\_wire.cpp, [63](#)
- wire\_add\_crc
  - sam\_ba\_wire.cpp, [53](#)
  - sam\_ba\_wire.h, [66](#)
- wire\_close
  - sam\_ba\_wire.cpp, [54](#)
  - sam\_ba\_wire.h, [67](#)
- wire\_getc
  - sam\_ba\_wire.cpp, [54](#)
  - sam\_ba\_wire.h, [67](#)
- wire\_getdata
  - sam\_ba\_wire.cpp, [55](#)
  - sam\_ba\_wire.h, [68](#)
- wire\_getdata\_xmd
  - sam\_ba\_wire.cpp, [56](#)
  - sam\_ba\_wire.h, [69](#)
- wire\_is\_rx\_ready
  - sam\_ba\_wire.cpp, [56](#)
  - sam\_ba\_wire.h, [69](#)
- wire\_open
  - sam\_ba\_wire.cpp, [57](#)
  - sam\_ba\_wire.h, [70](#)
- wire\_putc
  - sam\_ba\_wire.cpp, [57](#)
  - sam\_ba\_wire.h, [70](#)
- wire\_putdata
  - sam\_ba\_wire.cpp, [58](#)
  - sam\_ba\_wire.h, [71](#)
- wire\_putdata\_xmd
  - sam\_ba\_wire.cpp, [59](#)
  - sam\_ba\_wire.h, [72](#)
- wire\_readc
  - sam\_ba\_wire.cpp, [59](#)
  - sam\_ba\_wire.h, [72](#)
- wire\_setup
  - sam\_ba\_wire.cpp, [60](#)
  - sam\_ba\_wire.h, [73](#)
- wire\_sharp\_received
  - sam\_ba\_wire.cpp, [60](#)
  - sam\_ba\_wire.h, [73](#)