ZSamba Library

Generated by Doxygen 1.8.14

Contents

1 SAMBA

Arduino Library for samba boot loader compatible with ATSAMDx/ATSAMCx/ATSAMLx product familly 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:

3 File Index

3.1 File List

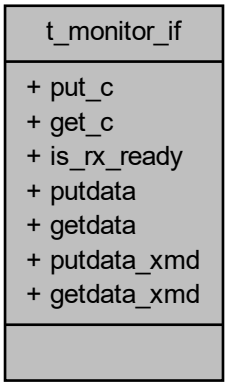
Here is a list of all files with brief descriptions:

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:

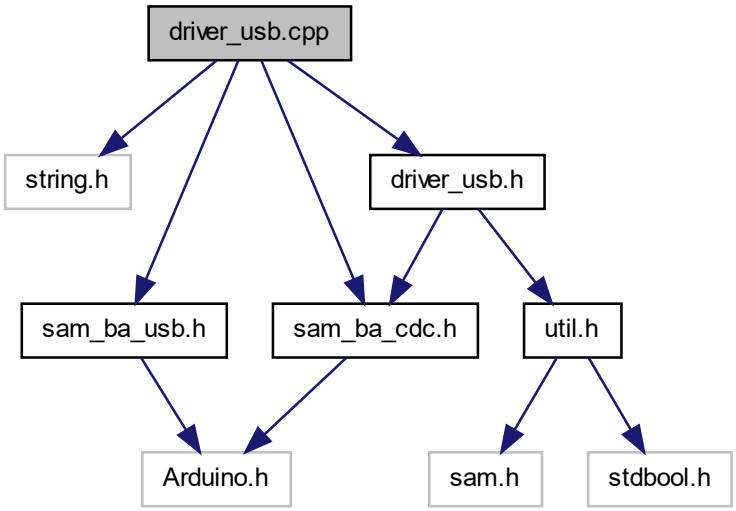
- sam_ba_monitor.h

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_SendZlp (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.

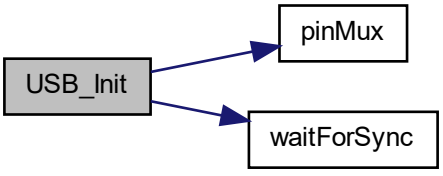
5.1.2.3 USB_Init()

```
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_RE↩ G_POS, and waitForSync().

Here is the call graph for this function:



5.1.2.4 USB_IsConfigured()

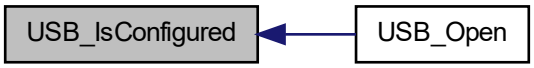
```
uint8_t USB_IsConfigured (
            P_USB_CDC pCdc )
```

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:



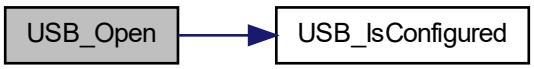
5.1.2.5 USB_Open()

```
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 )
```

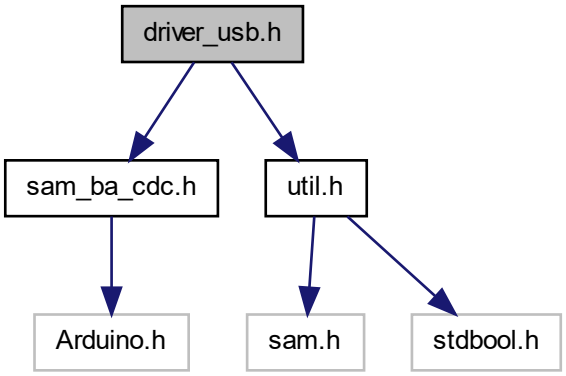
Definition at line 155 of file driver_usb.cpp.

References length, udd_ep_in_cache_buffer, and usb_endpoint_table.

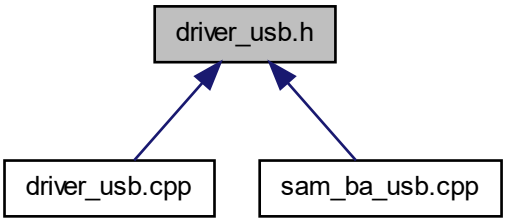
5.2 driver_usb.h File Reference

```
#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_SendZlp (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.

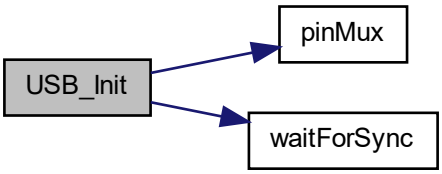
5.2.1.2 USB_Init()

```
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_RE↩ G_POS, and waitForSync().

Here is the call graph for this function:



5.2.1.3 USB_IsConfigured()

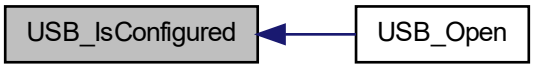
```
uint8_t USB_IsConfigured (
            P_USB_CDC pCdc )
```

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:



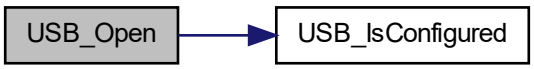
5.2.1.4 USB_Open()

```
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]
```

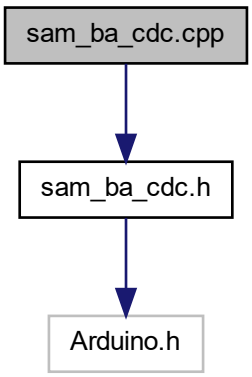
Referenced by USB_Configure(), USB_Init(), USB_IsConfigured(), USB_SendZlp(), and USB_Write().

5.3 README.md File Reference

5.4 sam_ba_cdc.cpp File Reference

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

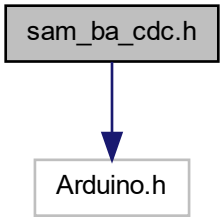
Include dependency graph for sam_ba_cdc.cpp:



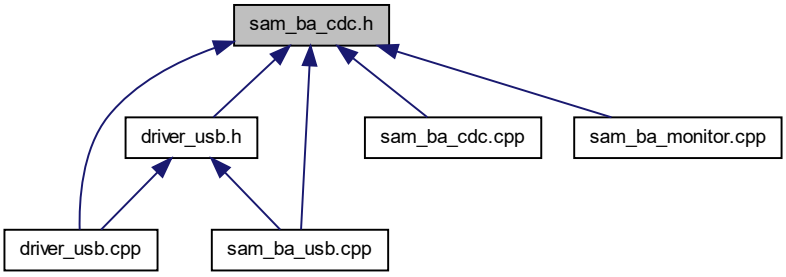
5.5 sam_ba_cdc.h File Reference

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

Include dependency graph for sam_ba_cdc.h:



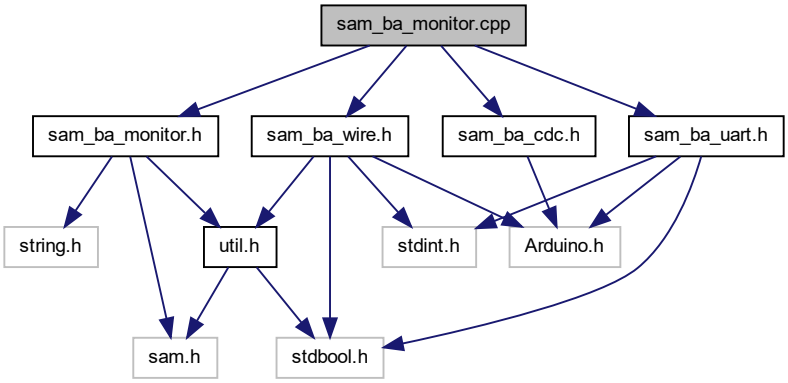
This graph shows which files directly or indirectly include this file:



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

∗data	Data pointer
length	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

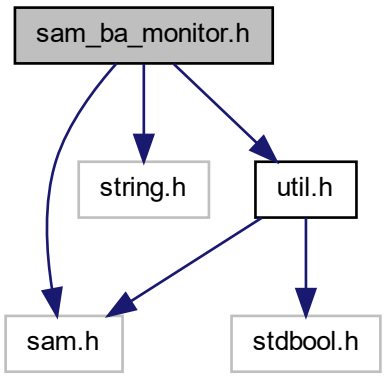
```
uint32_t u32tmp
```

Definition at line 228 of file sam_ba_monitor.cpp.

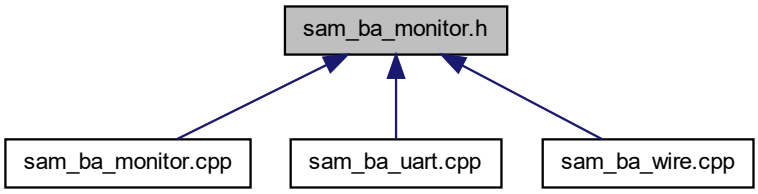
5.7 sam_ba_monitor.h File Reference

```
#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

∗data	Data pointer
length	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
```

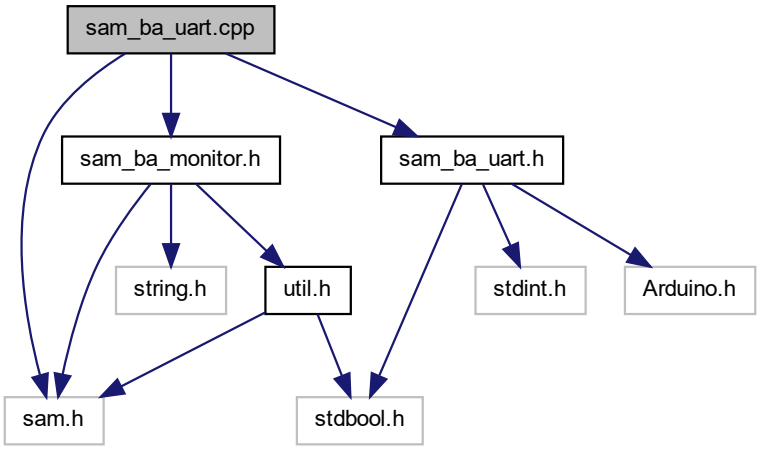
Definition at line 27 of file sam_ba_uart.cpp.

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

Char	to add to CRC
Previous	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

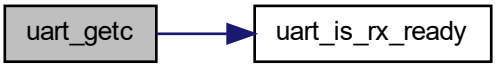
value read on usart line

Definition at line 104 of file sam_ba_uart.cpp.

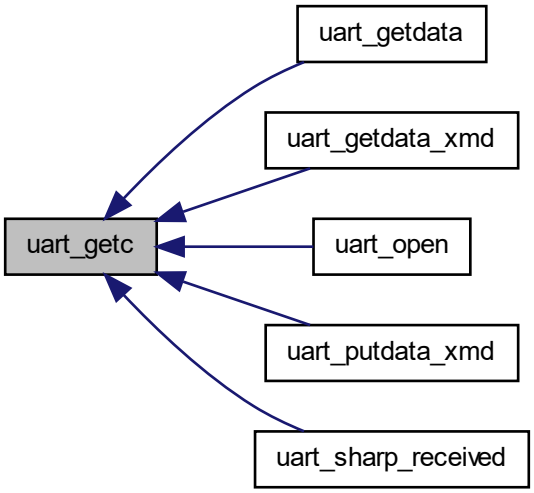
References serial, and uart_is_rx_ready().

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

data	pointer
number	of data to get

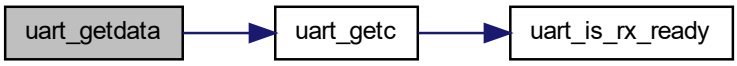
Returns

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

data	pointer
number	of data to get

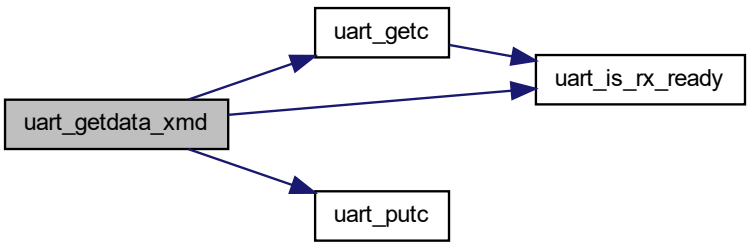
Returns

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

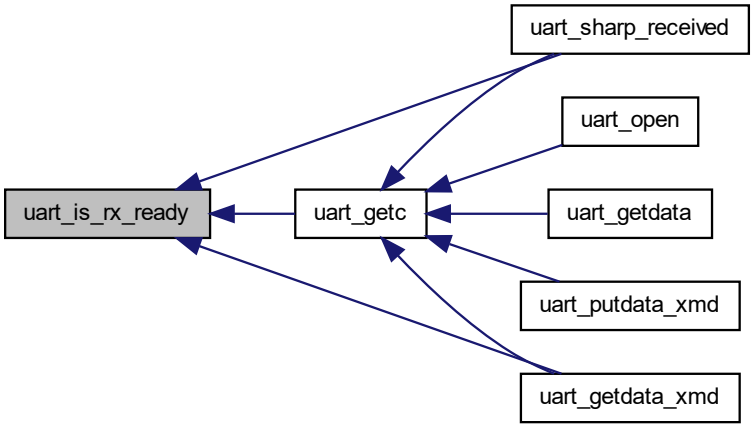
`1` if a byte is ready to be read.

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()

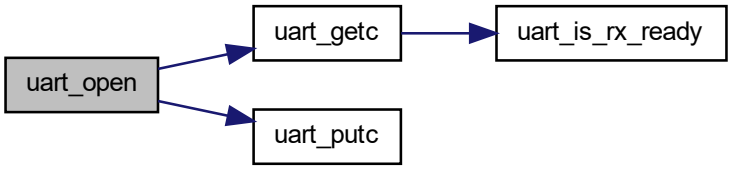
```
void uart_open (
            unsigned int fBaudSpeed )
```

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

value	Value to put

Returns

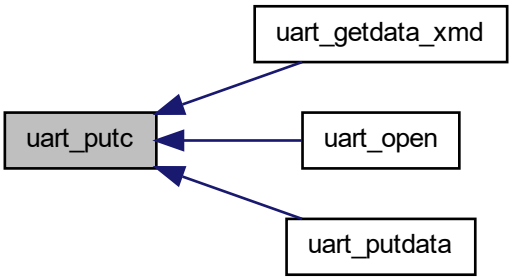
`1` if function was successfully done, otherwise `0`.

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

data	pointer
number	of data to send

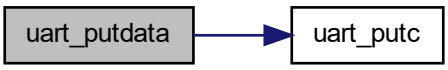
Returns

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

data	pointer
number	of data to send

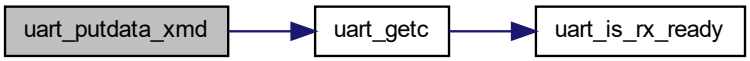
Returns

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.

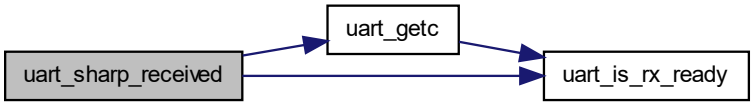
Returns

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
```

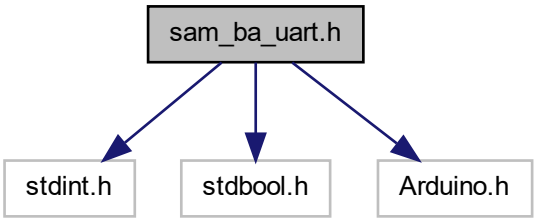
Definition at line 63 of file sam_ba_uart.cpp.

Referenced by uart_getdata_xmd(), and uart_putdata_xmd().

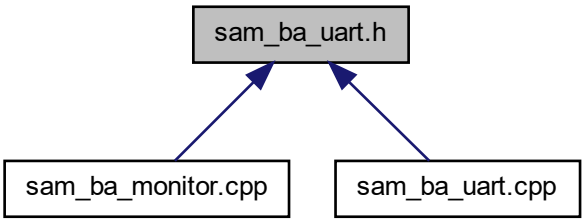
5.9 sam_ba_uart.h File Reference

```
#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

Char	to add to CRC
Previous	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

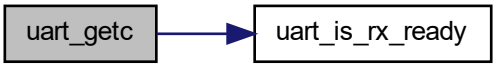
value read on usart line

Definition at line 104 of file sam_ba_uart.cpp.

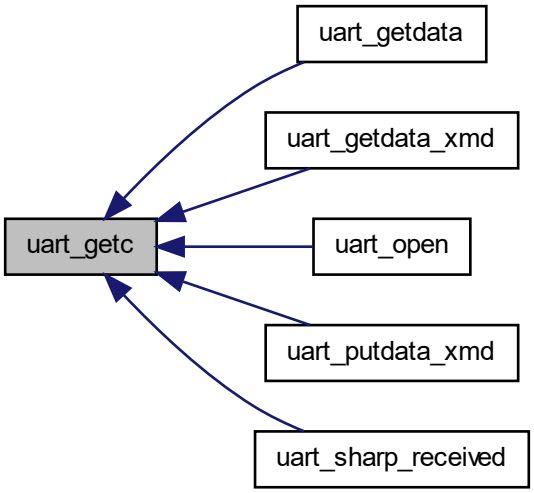
References serial, and uart_is_rx_ready().

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

data	pointer
number	of data to get

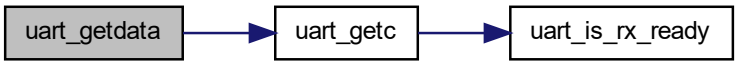
Returns

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

data	pointer
number	of data to get

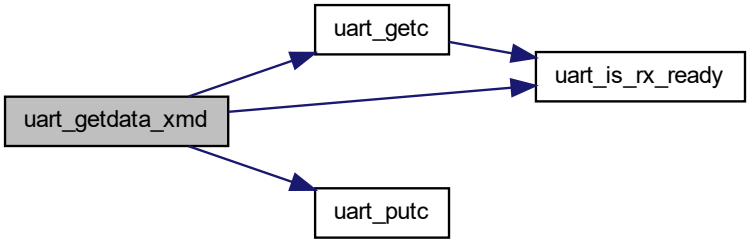
Returns

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

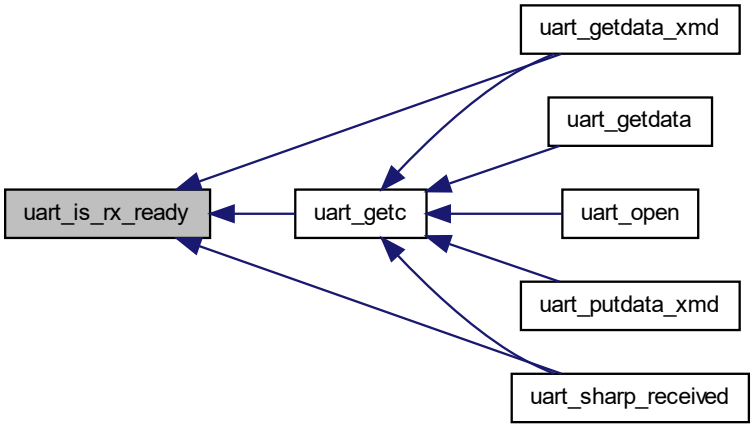
`1` if a byte is ready to be read.

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()

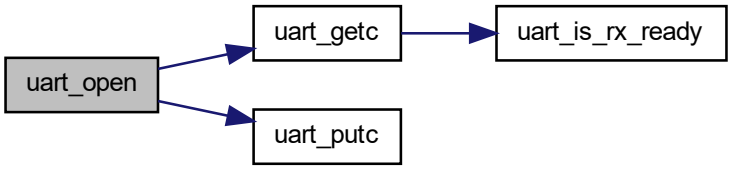
```
void uart_open (
            unsigned int fBaudSpeed )
```

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

value	Value to put

Returns

`1` if function was successfully done, otherwise `0`.

Puts a byte on usart line.

Parameters

value	Value to put

Returns

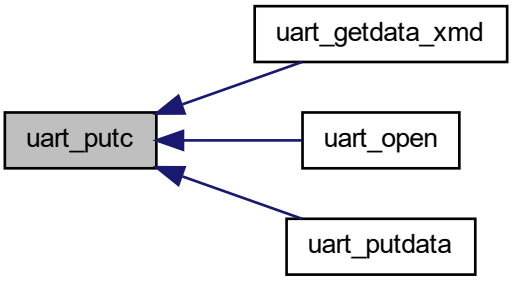
`1` if function was successfully done, otherwise `0`.

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

data	pointer
number	of data to send

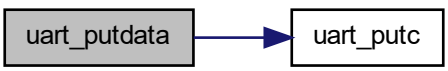
Returns

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

data	pointer
number	of data to send

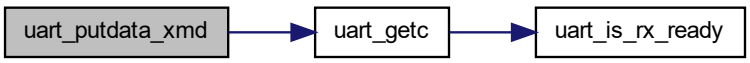
Returns

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.

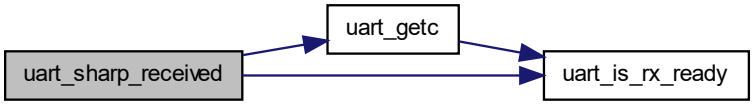
Returns

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:

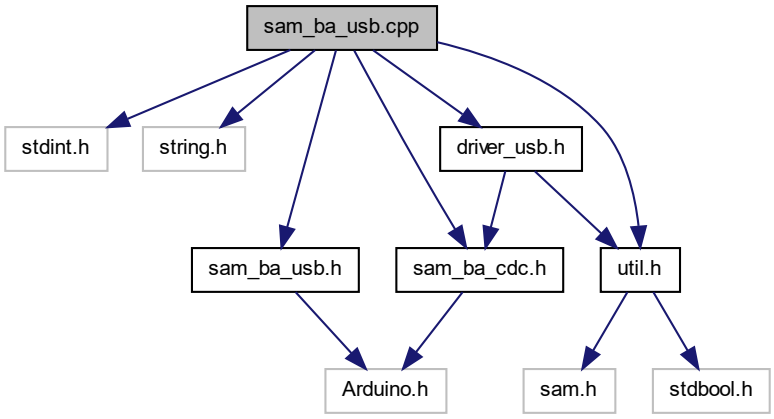


5.10 sam_ba_usb.cpp File Reference

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

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

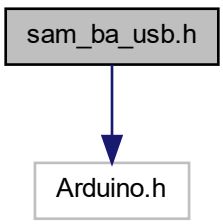
Include dependency graph for sam_ba_usb.cpp:



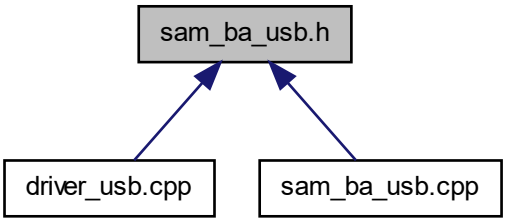
5.11 sam_ba_usb.h File Reference

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

Include dependency graph for sam_ba_usb.h:



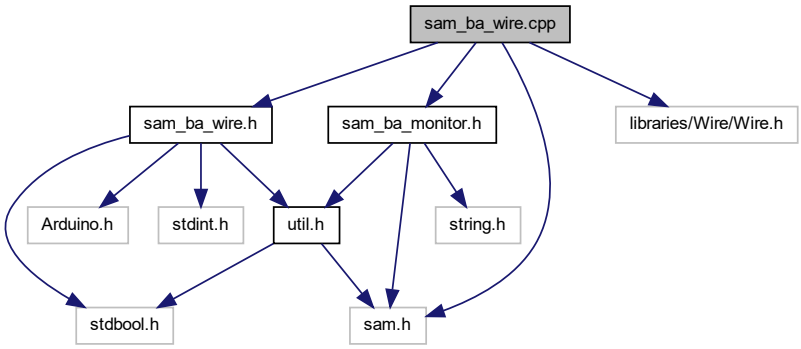
This graph shows which files directly or indirectly include this file:



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

Char	to add to CRC
Previous	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

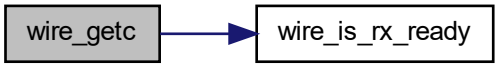
value read on usart line

Definition at line 119 of file sam_ba_wire.cpp.

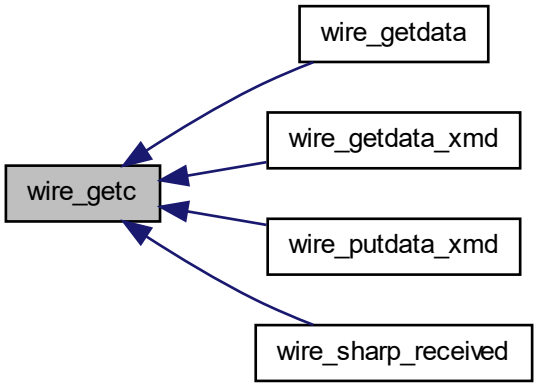
References wire, and wire_is_rx_ready().

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

data	pointer
number	of data to get

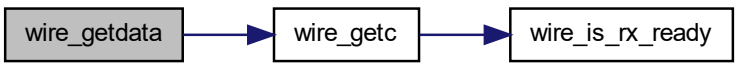
Returns

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

data	pointer
number	of data to get

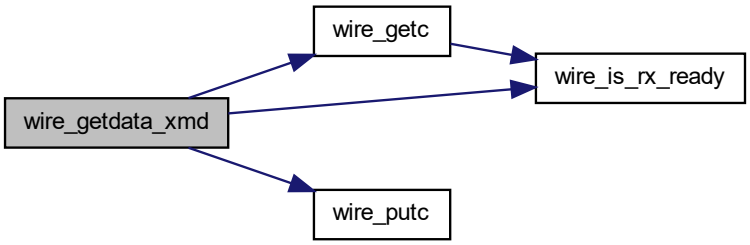
Returns

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

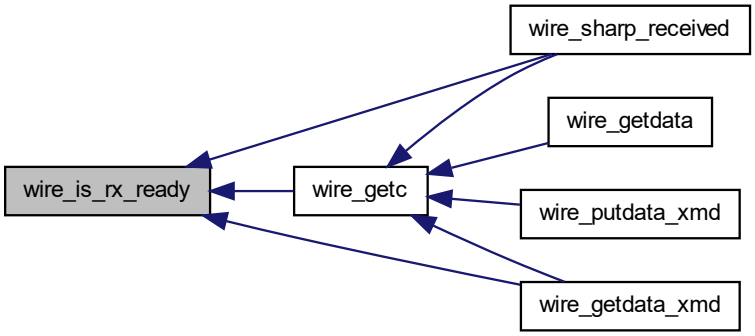
1 if a byte is ready to be read.

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

value	Value to put

Returns

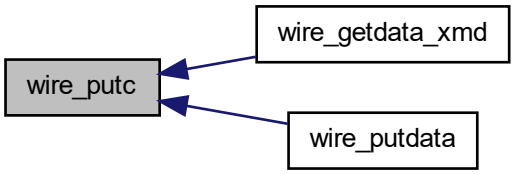
`1` if function was successfully done, otherwise `0`.

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

data	pointer
number	of data to send

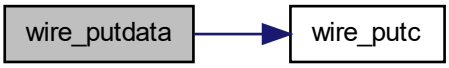
Returns

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

data	pointer
number	of data to send

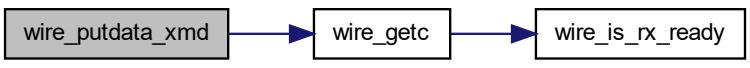
Returns

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.

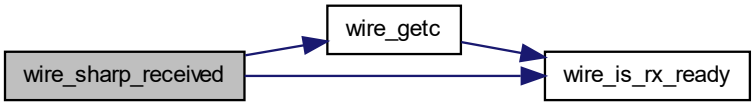
Returns

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
```

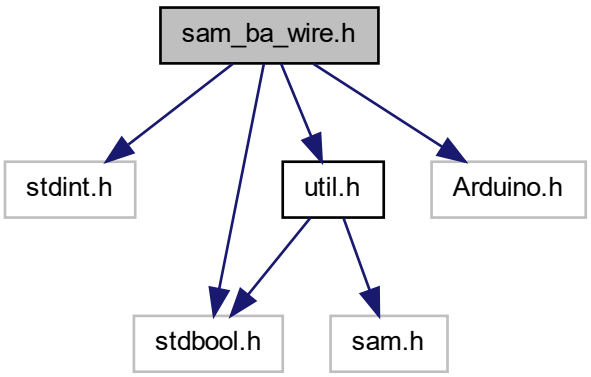
Definition at line 69 of file sam_ba_wire.cpp.

Referenced by wire_close(), wire_getc(), wire_is_rx_ready(), wire_open(), wire_putc(), and wire_setup().

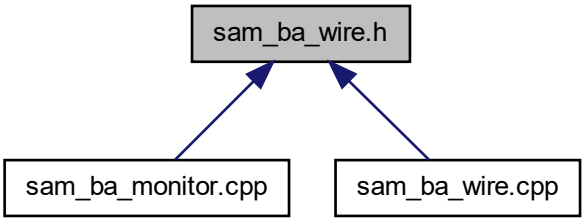
5.13 sam_ba_wire.h File Reference

```
#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

Char	to add to CRC
Previous	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

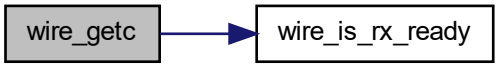
value read on usart line

Definition at line 119 of file sam_ba_wire.cpp.

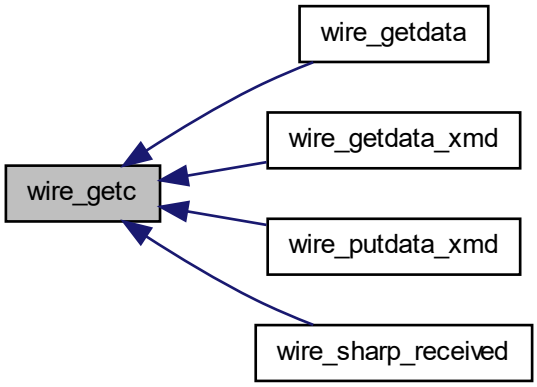
References wire, and wire_is_rx_ready().

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

data	pointer
number	of data to get

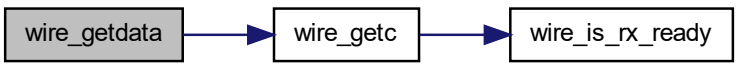
Returns

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

data	pointer
number	of data to get

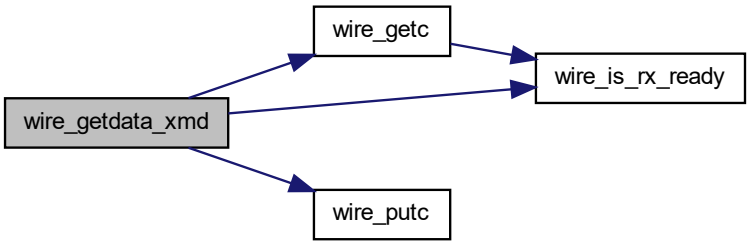
Returns

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

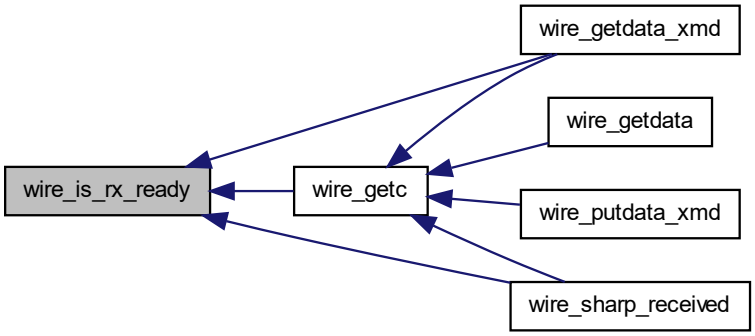
1 if a byte is ready to be read.

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

value	Value to put

Returns

`1` if function was successfully done, otherwise `0`.

Puts a byte on usart line.

Parameters

value	Value to put

Returns

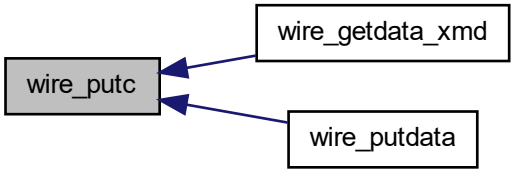
`1` if function was successfully done, otherwise `0`.

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

data	pointer
number	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

data	pointer
number	of data to send

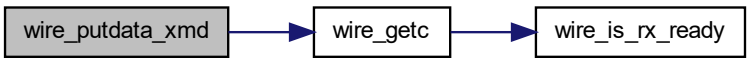
Returns

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.

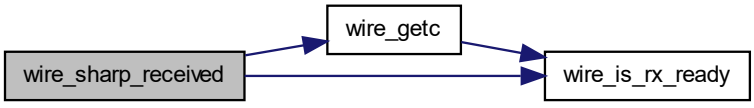
Returns

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().

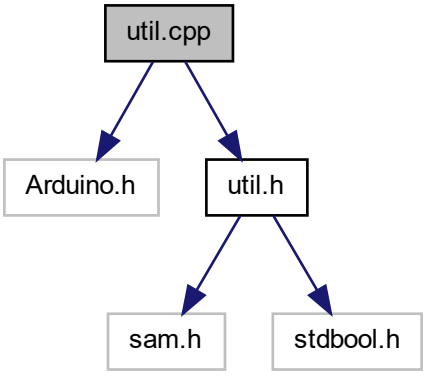
Here is the call graph for this function:



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.

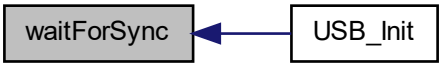
5.14.2.8 waitForSync()

```
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

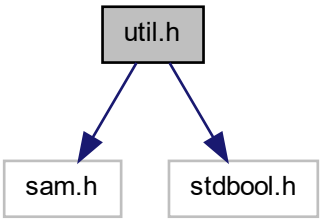
5.14.3.1 __sketch_vectors_ptr

```
uint32_t __sketch_vectors_ptr
```

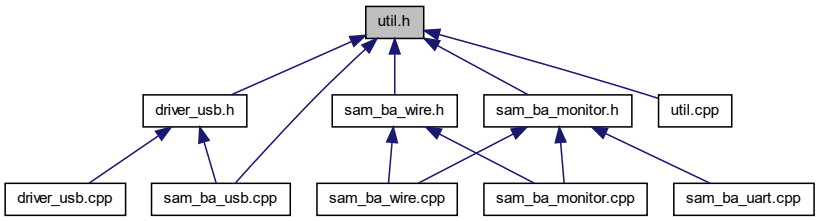
5.15 util.h File Reference

```
#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.

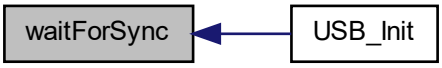
5.15.2.8 waitForSync()

```
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