

USART library for AVR uC

Version : 1.0

Generated by Doxygen 1.7.4

Sat Jul 30 2011 17:49:23

Contents

1	File Documentation	1
1.1	usart.h File Reference	1
1.1.1	Detailed Description	2
1.1.2	License	3
1.1.3	Specifications	3
1.1.4	More informations	4
1.1.5	Function Documentation	4

1 File Documentation

1.1 usart.h File Reference

USART functions for serial communication for AVR uC.

```
#include <stdbool.h>
#include <avr/io.h>
#include <avr/pgmspace.h>
#include "ascii/ascii.h"
```

Defines

For `usart_Setup()`

- `#define usart_BAUD_2400` 2400
- `#define usart_BAUD_4800` 4800
- `#define usart_BAUD_9600` 9600
- `#define usart_BAUD_14400` 14400
- `#define usart_BAUD_19200` 19200
- `#define usart_BAUD_2880` 28800
- `#define usart_BAUD_38400` 38400
- `#define usart_BAUD_57600` 57600
- `#define usart_TX_ENABLE` true
- `#define usart_TX_DISABLE` !usart_TX_ENABLE
- `#define usart_RX_ENABLE` true
- `#define usart_RX_DISABLE` !usart_RX_ENABLE
- `#define usart_PARITY_NONE` 'n'
- `#define usart_PARITY_ODD` 'o'
- `#define usart_PARITY_EVEN` 'e'
- `#define usart_DATA_5` 5

- #define **usart_DATA_6** 6
- #define **usart_DATA_7** 7
- #define **usart_DATA_8** 8
- #define **usart_DATA_9** 9
- #define **usart_STOP_1** 1
- #define **usart_STOP_2** 2

Functions

- void **usart_Byte** (uint8_t byte)
Transmit the binary representation of a byte one character at a time.
- void **usart_DoubleByte** (uint16_t byte_long)
Transmit the binary representation of a double byte one character at a time.
- void **usart_FlushBuffer** (void)
Empty the receive buffer.
- bool **usart_IsCharacter** (void)
Check if a character is ready to be read.
- char **usart_ReceiveCharacter** (void)
Return the character in the USART.
- void **usart_SendCharacter** (char c)
Send character c to the USART.
- void **usart_SendString** (const char *data)
Send a string to the UART.
- void **usart_SendString_P** (const char *data)
Send a PROGMEM string to the UART.
- void **usart_SendStringReturn** (const char *data)
Send a string to the UART with CR-LF at the end.
- void **usart_SendStringReturn_P** (const char *data)
Send a PROGMEM string to the UART with CR-LF at the end.
- void **usart_Setup** (uint16_t baud, uint8_t length, char parity, uint8_t stop_bits, uint8_t tx, uint8_t rx)
Setup the USART.

1.1.1 Detailed Description

USART functions for serial communication for AVR uC.

1.1.2 License

Copyright (c) 2009, 2010, 2011 Patrice Nadeau All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of Patrice Nadeau nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL Patrice Nadeau BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.1.3 Specifications

Serial functions to use the USART as a serial port

1.1.3.1 Language

- C (c99)

1.1.3.2 Target

- ATmega48
- ATmega88
- ATmega168 (Tested)
- ATmega328
- ATmega8515

1.1.4 More informations

Datasheet

ATmega48-88-168-328 & 8515

Author

Patrice Nadeau

email : patricen@telwarwick.net

www : <http://nadeaup.homeip.net:8080/>

Note

Use the F_CPU variable of the compiler to determine the speed of the AVR

Clock must be a multiple of 1.8432 MHz to have 0.0% error

Tested at 9600,8,N,1 (External 8MHz) and 4800,8,N,1 (Internal 1MHz)

Warning

Works only for the first 128 bits of the ASCII code

Data length of 9 bits is not implemented yet

Remarks

The #define usart_AVR is used to determine for which AVR to compile the code (via preprocessor directives) It does not show in the doxygen documentation

1.1.5 Function Documentation

1.1.5.1 void usart_Byte (uint8_t *byte*)

Transmit the binary representation of a byte one character at a time.

Parameters

in	<i>byte</i>	A byte
----	-------------	--------

1.1.5.2 void usart_DoubleByte (uint16_t *byte_long*)

Transmit the binary representation of a double byte one character at a time.

Parameters

in	<i>byte_long</i>	A double byte
----	------------------	---------------

1.1.5.3 void usart_FlushBuffer (void)

Empty the receive buffer.

Note

Low level function. Need to be modified to run on another AVR than those listed in the documentation

1.1.5.4 bool usart_IsCharacter (void)

Check if a character is ready to be read.

Return values

<i>true</i> <i>false</i>	A character ready to be read
----------------------------	------------------------------

Note

Low level function. Need to be modified to run on another AVR than those listed in the documentation

1.1.5.5 char usart_ReceiveCharacter (void)

Return the character in the USART.

Returns

A character

Note

Low level function. Need to be modified to run on another AVR than those listed in the documentation

1.1.5.6 void usart_SendCharacter (char c)

Send character *c* to the USART.

Parameters

<i>in</i>	<i>c</i>	Character to send
-----------	----------	-------------------

Note

Low level function. Need to be modified to run on another AVR than those listed in the documentation

1.1.5.7 void usart_SendString (const char * *data*)

Send a string to the UART.

Parameters

in	* <i>data</i>	Pointer to a string
----	---------------	---------------------

Example

```
usart_SendString("Hello World");
```

1.1.5.8 void usart_SendString_P (const char * *data*)

Send a PROGMEM string to the UART.

Parameters

in	* <i>data</i>	Pointer to a PROGMEM string
----	---------------	-----------------------------

Example

```
usart_SendString_P(PSTR("Hello World"));
```

Warning

Data must not be a single character or a #define

1.1.5.9 void usart_SendStringReturn (const char * *data*)

Send a string to the UART with CR-LF at the end.

Parameters

in	* <i>data</i>	Pointer to a string *
----	---------------	-----------------------

Example

```
usart_SendStringReturn("Hello World");
```

1.1.5.10 void usart_SendStringReturn_P (const char * *data*)

Send a PROGMEM string to the UART with CR-LF at the end.

Parameters

in	* <i>data</i>	Pointer to a PROGMEM string
----	---------------	-----------------------------

Example :

```
usart_SendString_P(PSTR("Hello World"));
```

Warning

Data must not be a single character or a #define

1.1.5.11 void usart_Setup (uint16_t *baud*, uint8_t *length*, char *parity*, uint8_t *stop_bits*, uint8_t *tx*, uint8_t *rx*)

Setup the USART.

Parameters

in	<i>baud</i>	Baud rate. Possible values : <ul style="list-style-type: none"> • usart_BAUD_2400 • usart_BAUD_4800 • usart_BAUD_9600 • usart_BAUD_14400 • usart_BAUD_19200 • usart_BAUD_28800 • usart_BAUD_38400 • usart_BAUD_57600
in	<i>length</i>	Length of the data bit. Possible values : <ul style="list-style-type: none"> • usart_DATA_5 • usart_DATA_6 • usart_DATA_7 • usart_DATA_8 • usart_DATA_9
in	<i>parity</i>	Parity <ul style="list-style-type: none"> • usart_PARITY_NONE • usart_PARITY_ODD • usart_PARITY_EVEN
in	<i>stop_bits</i>	Number of stop bit(s). Possible values : <ul style="list-style-type: none"> • usart_STOP_1 • usart_STOP_2

in	tx	Transmit enable/disable. Possible values : <ul style="list-style-type: none">• usart_TX_ENABLE• usart_TX_DISABLE
in	rx	Receive enable/disable. Possible values : <ul style="list-style-type: none">• usart_RX_ENABLE• usart_RX_DISABLE

Example

Setup the usart for 9600bps, 8 data bits, parity none, 1 stop bit, enable RX and TX

```
usart_Setup(usart_BAUD_9600, usart_DATA_8, usart_PARITY_NONE, usart_STOP_1, usart_TX_ENABLE, usart_RX_ENABLE);
```

Note

Low level function. Need to be modified to run on another AVR than those listed in the documentation

Index

- usart.h, [1](#)
 - usart_Byte, [4](#)
 - usart_DoubleByte, [4](#)
 - usart_FlushBuffer, [4](#)
 - usart_IsCharacter, [4](#)
 - usart_ReceiveCharacter, [5](#)
 - usart_SendCharacter, [5](#)
 - usart_SendString, [5](#)
 - usart_SendString_P, [5](#)
 - usart_SendStringReturn, [6](#)
 - usart_SendStringReturn_P, [6](#)
 - usart_Setup, [6](#)
- usart_Byte
 - usart.h, [4](#)
- usart_DoubleByte
 - usart.h, [4](#)
- usart_FlushBuffer
 - usart.h, [4](#)
- usart_IsCharacter
 - usart.h, [4](#)
- usart_ReceiveCharacter
 - usart.h, [5](#)
- usart_SendCharacter
 - usart.h, [5](#)
- usart_SendString
 - usart.h, [5](#)
- usart_SendString_P
 - usart.h, [5](#)
- usart_SendStringReturn
 - usart.h, [6](#)
- usart_SendStringReturn_P
 - usart.h, [6](#)
- usart_Setup
 - usart.h, [6](#)