# 32. SERCOM USART – SERCOM Universal Synchronous and Asynchronous Receiver and Transmitter

## 32.1. Overview

The Universal Synchronous and Asynchronous Receiver and Transmitter (USART) is one of the available modes in the Serial Communication Interface (SERCOM).

The USART uses the SERCOM transmitter and receiver, see Block Diagram. Labels in uppercase letters are synchronous to CLK_SERCOMx_APB and accessible for CPU. Labels in lowercase letters can be programmed to run on the internal generic clock or an external clock.

The transmitter consists of a single write buffer, a shift register, and control logic for different frame formats. The write buffer support data transmission without any delay between frames. The receiver consists of a two-level receive buffer and a shift register. Status information of the received data is available for error checking. Data and clock recovery units ensure robust synchronization and noise filtering during asynchronous data reception.

**Related Links**

SERCOM – Serial Communication Interface on page 515

## 32.2. USART Features

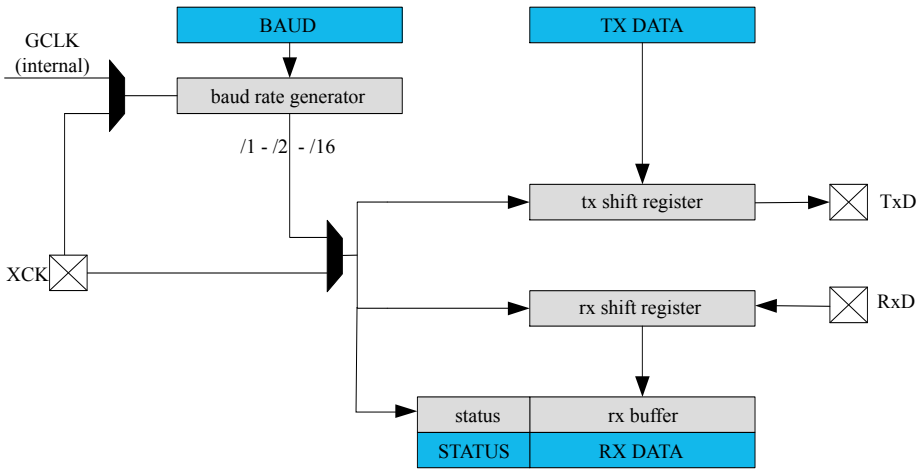
- Full-duplex operation
- Asynchronous (with clock reconstruction) or synchronous operation
- Internal or external clock source for asynchronous and synchronous operation
- Baud-rate generator
- Supports serial frames with 5, 6, 7, 8 or 9 data bits and 1 or 2 stop bits
- Odd or even parity generation and parity check
- Selectable LSB- or MSB-first data transfer
- Buffer overflow and frame error detection
- Noise filtering, including false start-bit detection and digital low-pass filter
- Collision detection
- Can operate in all sleep modes
- Operation at speeds up to half the system clock for internally generated clocks
- Operation at speeds up to the system clock for externally generated clocks
- RTS and CTS flow control
- IrDA modulation and demodulation up to 115.2kbps
- LIN master support
- LIN slave support
  - Auto-baud and break character detection
- RS485 Support
- Start-of-frame detection
- Can work with DMA

**Related Links**

Features on page 515

## 32.3. Block Diagram

**Figure 32-1. USART Block Diagram**




## 32.4. Signal Description

**Table 32-1. SERCOM USART Signals**

| Signal Name | Type | Description |
|---|---|---|
| PAD[3:0] | Digital I/O | General SERCOM pins |

One signal can be mapped to one of several pins.

**Related Links**

I/O Multiplexing and Considerations on page 29

## 32.5. Product Dependencies

In order to use this peripheral, other parts of the system must be configured correctly, as described below.

### 32.5.1. I/O Lines

Using the USART's I/O lines requires the I/O pins to be configured using the I/O Pin Controller (PORT).

When the SERCOM is used in USART mode, the SERCOM controls the direction and value of the I/O pins according to the table below. Both PORT control bits PINCFGn.PULLEN and PINCFGn.DRVSTR are still effective. If the receiver or transmitter is disabled, these pins can be used for other purposes.

**Table 32-2. USART Pin Configuration**

| Pin | Pin Configuration |
|---|---|
| TxD | Output |
| RxD | Input |
| XCK | Output or input |

The combined configuration of PORT and the Transmit Data Pinout and Receive Data Pinout bit fields in the Control A register (CTRLA.TXPO and CTRLA.RXPO, respectively) will define the physical position of the USART signals in Table 32-2 USART Pin Configuration.

**Related Links**

PORT: IO Pin Controller on page 460

### 32.5.2. Power Management

This peripheral can continue to operate in any sleep mode where its source clock is running. The interrupts can wake up the device from sleep modes. Refer to *PM – Power Manager* for details on the different sleep modes.

**Related Links**

PM – Power Manager on page 182

### 32.5.3. Clocks

The SERCOM bus clock (CLK_SERCOMx_APB) is enabled by default, and can be disabled and enabled in the Main Clock Controller. Refer to *Peripheral Clock Masking* for details.

A generic clock (GCLK_SERCOMx_CORE) is required to clock the SERCOMx_CORE. This clock must be configured and enabled in the Generic Clock Controller before using the SERCOMx_CORE. Refer to *GCLK - Generic Clock Controller* for details.

This generic clock is asynchronous to the bus clock (CLK_SERCOMx_APB). Therefore, writing to certain registers will require synchronization to the clock domains. Refer to Synchronizationfor further details.

**Related Links**

Peripheral Clock Masking on page 155

GCLK - Generic Clock Controller on page 130

### 32.5.4. DMA

The DMA request lines are connected to the DMA Controller (DMAC). In order to use DMA requests with this peripheral the DMAC must be configured first. Refer to *DMAC – Direct Memory Access Controller* for details.

**Related Links**

DMAC – Direct Memory Access Controller on page 351

### 32.5.5. Interrupts

The interrupt request line is connected to the Interrupt Controller. In order to use interrupt requests of this peripheral, the Interrupt Controller (NVIC) must be configured first. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

Nested Vector Interrupt Controller on page 44

### 32.5.6. Events

Not applicable.

### 32.5.7. Debug Operation

When the CPU is halted in debug mode, this peripheral will continue normal operation. If the peripheral is configured to require periodical service by the CPU through interrupts or similar, improper operation or data loss may result during debugging. This peripheral can be forced to halt operation during debugging - refer to the Debug Control (DBGCTRL) register for details.

### 32.5.8. Register Access Protection

Registers with write-access can be write-protected optionally by the peripheral access controller (PAC).

PAC Write-Protection is not available for the following registers:

- Interrupt Flag Clear and Status register (INTFLAG)
- Status register (STATUS)
- Data register (DATA)

Optional PAC Write-Protection is denoted by the "PAC Write-Protection" property in each individual register description.

Write-protection does not apply to accesses through an external debugger.

**Related Links**

PAC - Peripheral Access Controller on page 49

### 32.5.9. Analog Connections

Not applicable.

## 32.6. Functional Description

### 32.6.1. Principle of Operation

The USART uses the following lines for data transfer:

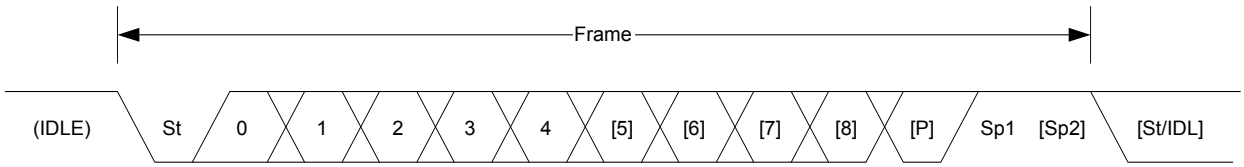
- RxD for receiving
- TxD for transmitting
- XCK for the transmission clock in synchronous operation

USART data transfer is frame based. A serial frame consists of:

- 1 start bit
- From 5 to 9 data bits (MSB or LSB first)
- No, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit followed by one character of data bits. If enabled, the parity bit is inserted after the data bits and before the first stop bit. After the stop bit(s) of a frame, either the next frame can follow immediately, or the communication line can return to the idle (high) state. The figure below illustrates the possible frame formats. Brackets denote optional bits.

**Figure 32-2. Frame Formats**




**St** Start bit. Signal is always low.

**n, [n]** Data bits. 0 to [5..9]

**[P]** Parity bit. Either odd or even.

**Sp, [Sp]** Stop bit. Signal is always high.

**IDLE** No frame is transferred on the communication line. Signal is always high in this state.

### 32.6.2. Basic Operation

#### 32.6.2.1. Initialization

The following registers are enable-protected, meaning they can only be written when the USART is disabled (CTRL.ENABLE=0):

- Control A register (CTRLA), except the Enable (ENABLE) and Software Reset (SWRST) bits.
- Control B register (CTRLB), except the Receiver Enable (RXEN) and Transmitter Enable (TXEN) bits.
- Baud register (BAUD)

Any writes to these registers when the USART is enabled or is being enabled (CTRL.ENABLE is one) will be discarded. Writes to these registers while the peripheral is being disabled, will be completed after the disabling is complete.

When the USART is enabled or is being enabled (CTRLA.ENABLE=1), any writing attempt to these registers will be discarded. If the peripheral is being disabled, writing to these registers will be executed after disabling is completed. Enable-protection is denoted by the "Enable-Protection" property in the register description.

Before the USART is enabled, it must be configured by these steps:

1. Select either external (0x0) or internal clock (0x1) by writing the Operating Mode value in the CTRLA register (CTRLA.MODE).
2. Select either asynchronous (0) or or synchronous (1) communication mode by writing the Communication Mode bit in the CTRLA register (CTRLA.CMODE).
3. Select pin for receive data by writing the Receive Data Pinout value in the CTRLA register (CTRLA.RXPO).
4. Select pads for the transmitter and external clock by writing the Transmit Data Pinout bit in the CTRLA register (CTRLA.TXPO).
5. Configure the Character Size field in the CTRLB register (CTRLB.CHSIZE) for character size.
6. Set the Data Order bit in the CTRLA register (CTRLA.DORD) to determine MSB- or LSB-first data transmission.
7. To use parity mode:
   7.1. Enable parity mode by writing 0x1 to the Frame Format field in the CTRLA register (CTRLA.FORM).
   7.2. Configure the Parity Mode bit in the CTRLB register (CTRLB.PMODE) for even or odd parity.
8. Configure the number of stop bits in the Stop Bit Mode bit in the CTRLB register (CTRLB.SBMODE).
9. When using an internal clock, write the Baud register (BAUD) to generate the desired baud rate.
10. Enable the transmitter and receiver by writing '1' to the Receiver Enable and Transmitter Enable bits in the CTRLB register (CTRLB.RXEN and CTRLB.TXEN).

#### 32.6.2.2. Enabling, Disabling, and Resetting

This peripheral is enabled by writing '1' to the Enable bit in the Control A register (CTRLA.ENABLE), and disabled by writing '0' to it.

Writing ‘1’ to the Software Reset bit in the Control A register (CTRLA.SWRST) will reset all registers of this peripheral to their initial states, except the DBGCTRL register, and the peripheral is disabled.

Refer to the CTRLA register description for details.

### 32.6.2.3. Clock Generation and Selection

For both synchronous and asynchronous modes, the clock used for shifting and sampling data can be generated internally by the SERCOM baud-rate generator or supplied externally through the XCK line.

The synchronous mode is selected by writing a '1' to the Communication Mode bit in the Control A register (CTRLA.CMODE), the asynchronous mode is selected by writing a zero to CTRLA.CMODE.

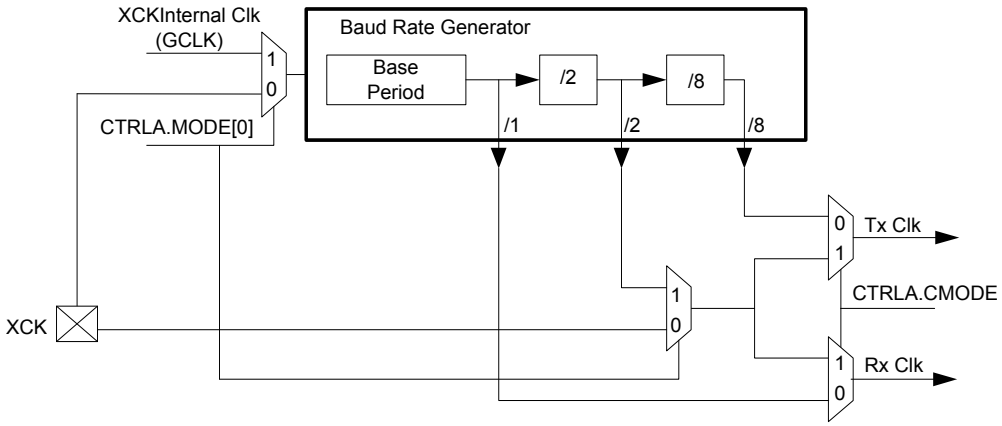
The internal clock source is selected by writing 0x1 to the Operation Mode bit field in the Control A register (CTRLA.MODE), the external clock source is selected by writing 0x0 to CTRLA.MODE.

The SERCOM baud-rate generator is configured as in the figure below.

In asynchronous mode (CTRLA.CMODE=0), the 16-bit Baud register value is used.

In synchronous mode (CTRLA.CMODE=1), the eight LSBs of the Baud register are used. Refer to *Clock Generation – Baud-Rate Generator* for details on configuring the baud rate.

**Figure 32-3. Clock Generation**




**Related Links**

Clock Generation – Baud-Rate Generator on page 519

Asynchronous Arithmetic Mode BAUD Value Selection on page 520

#### Synchronous Clock Operation

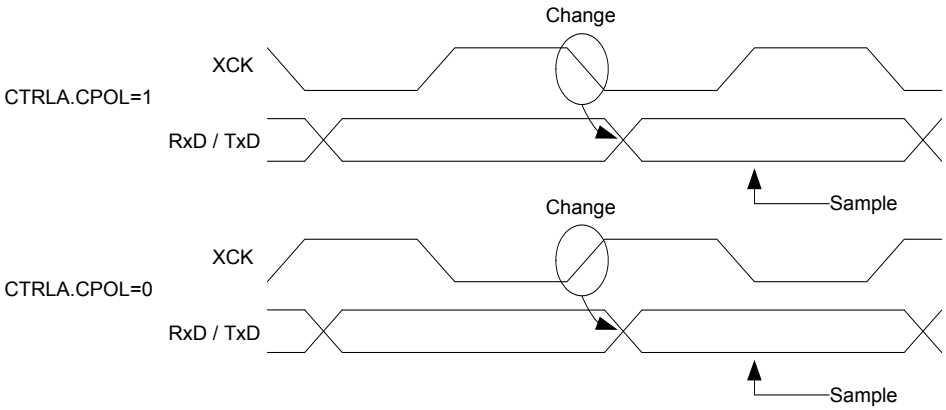
In synchronous mode, the CTRLA.MODE bit field determines whether the transmission clock line (XCK) serves either as input or output. The dependency between clock edges, data sampling, and data change is the same for internal and external clocks. Data input on the RxD pin is sampled at the opposite XCK clock edge when data is driven on the TxD pin.

The Clock Polarity bit in the Control A register (CTRLA.CPOL) selects which XCK clock edge is used for RxD sampling, and which is used for TxD change:

When CTRLA.CPOL is '0', the data will be changed on the rising edge of XCK, and sampled on the falling edge of XCK.

When CTRLA.CPOL is '1', the data will be changed on the falling edge of XCK, and sampled on the rising edge of XCK.

**Figure 32-4. Synchronous Mode XCK Timing**




When the clock is provided through XCK (CTRLA.MODE=0x0), the shift registers operate directly on the XCK clock. This means that XCK is not synchronized with the system clock and, therefore, can operate at frequencies up to the system frequency.

#### 32.6.2.4. Data Register

The USART Transmit Data register (TxDATA) and USART Receive Data register (RxDATA) share the same I/O address, referred to as the Data register (DATA). Writing the DATA register will update the TxDATA register. Reading the DATA register will return the contents of the RxDATA register.

#### 32.6.2.5. Data Transmission

Data transmission is initiated by writing the data to be sent into the DATA register. Then, the data in TxDATA will be moved to the shift register when the shift register is empty and ready to send a new frame. After the shift register is loaded with data, the data frame will be transmitted.

When the entire data frame including stop bit(s) has been transmitted and no new data was written to DATA, the Transmit Complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.TXC) will be set, and the optional interrupt will be generated.

The Data Register Empty flag in the Interrupt Flag Status and Clear register (INTFLAG.DRE) indicates that the register is empty and ready for new data. The DATA register should only be written to when INTFLAG.DRE is set.

**Disabling the Transmitter**

The transmitter is disabled by writing '0' to the Transmitter Enable bit in the CTRLB register (CTRLB.TXEN).

Disabling the transmitter will complete only after any ongoing and pending transmissions are completed, i.e., there is no data in the transmit shift register and TxDATA to transmit.

#### 32.6.2.6. Data Reception

The receiver accepts data when a valid start bit is detected. Each bit following the start bit will be sampled according to the baud rate or XCK clock, and shifted into the receive shift register until the first stop bit of a frame is received. The second stop bit will be ignored by the receiver.

When the first stop bit is received and a complete serial frame is present in the receive shift register, the contents of the shift register will be moved into the two-level receive buffer. Then, the Receive Complete interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG.RXC) will be set, and the optional interrupt will be generated.

The received data can be read from the DATA register when the Receive Complete interrupt flag is set.

### Disabling the Receiver

Writing '0' to the Receiver Enable bit in the CTRLB register (CTRLB.RXEN) will disable the receiver, flush the two-level receive buffer, and data from ongoing receptions will be lost.

### Error Bits

The USART receiver has three error bits in the Status (STATUS) register: Frame Error (FERR), Buffer Overflow (BUFOVF), and Parity Error (PERR). Once an error happens, the corresponding error bit will be set until it is cleared by writing '1' to it. These bits are also cleared automatically when the receiver is disabled.

There are two methods for buffer overflow notification, selected by the Immediate Buffer Overflow Notification bit in the Control A register (CTRLA.IBON):

When CTRLA.IBON=1, STATUS.BUFOVF is raised immediately upon buffer overflow. Software can then empty the receive FIFO by reading RxDATA, until the receiver complete interrupt flag (INTFLAG.RXC) is cleared.

When CTRLA.IBON=0, the buffer overflow condition is attending data through the receive FIFO. After the received data is read, STATUS.BUFOVF will be set along with INTFLAG.RXC.

### Asynchronous Data Reception

The USART includes a clock recovery and data recovery unit for handling asynchronous data reception.

The clock recovery logic can synchronize the incoming asynchronous serial frames at the RxD pin to the internally generated baud-rate clock.

The data recovery logic samples and applies a low-pass filter to each incoming bit, thereby improving the noise immunity of the receiver.

### Asynchronous Operational Range

The operational range of the asynchronous reception depends on the accuracy of the internal baud-rate clock, the rate of the incoming frames, and the frame size (in number of bits). In addition, the operational range of the receiver is depending on the difference between the received bit rate and the internally generated baud rate. If the baud rate of an external transmitter is too high or too low compared to the internally generated baud rate, the receiver will not be able to synchronize the frames to the start bit.

There are two possible sources for a mismatch in baud rate: First, the reference clock will always have some minor instability. Second, the baud-rate generator cannot always do an exact division of the reference clock frequency to get the baud rate desired. In this case, the BAUD register value should be set to give the lowest possible error. Refer to *Clock Generation – Baud-Rate Generator* for details.

Recommended maximum receiver baud-rate errors for various character sizes are shown in the table below.

**Table 32-3. Asynchronous Receiver Error for 16-fold Oversampling**

| D (Data bits+Parity) | $R_{SLOW}$ [%] | $R_{FAST}$ [%] | Max. total error [%] | Recommended max. Rx error [%] |
|---|---|---|---|---|
| 5 | 94.12 | 107.69 | +5.88/-7.69 | ±2.5 |
| 6 | 94.92 | 106.67 | +5.08/-6.67 | ±2.0 |
| 7 | 95.52 | 105.88 | +4.48/-5.88 | ±2.0 |
| 8 | 96.00 | 105.26 | +4.00/-5.26 | ±2.0 |
| 9 | 96.39 | 104.76 | +3.61/-4.76 | ±1.5 |
| 10 | 96.70 | 104.35 | +3.30/-4.35 | ±1.5 |

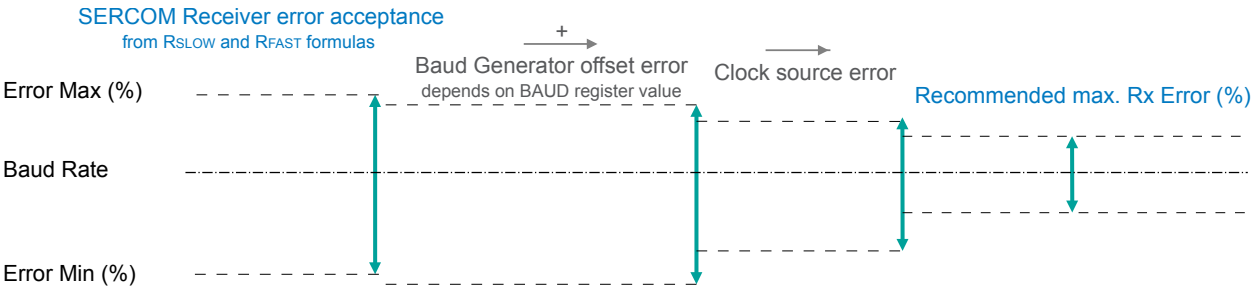
The following equations calculate the ratio of the incoming data rate and internal receiver baud rate:

$$R_{SLOW} = \frac{(D+1)S}{S-1+D\cdot S+S_F} \quad , \quad R_{FAST} = \frac{(D+2)S}{(D+1)S+S_M}$$

- $R_{SLOW}$ is the ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate
- $R_{FAST}$ is the ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate
- $D$ is the sum of character size and parity size ($D$ = 5 to 10 bits)
- $S$ is the number of samples per bit ($S$ = 16, 8 or 3)
- $S_F$ is the first sample number used for majority voting ($S_F$ = 7, 3, or 2) when CTRLA.SAMPA=0.
- $S_M$ is the middle sample number used for majority voting ($S_M$ = 8, 4, or 2) when CTRLA.SAMPA=0.

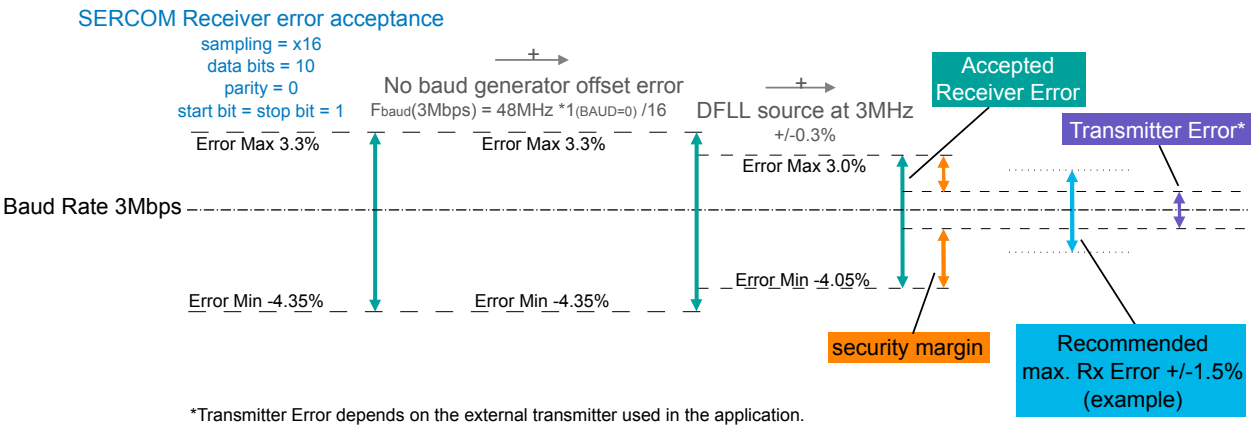
The recommended maximum Rx Error assumes that the receiver and transmitter equally divide the maximum total error. Its connection to the SERCOM Receiver error acceptance is depicted in this figure:

**Figure 32-5. USART Rx Error Calculation**




The recommendation values in the table above accommodate errors of the clock source and the baud generator. The following figure gives an example for a baud rate of 3Mbps:

**Figure 32-6. USART Rx Error Calculation Example**




*Transmitter Error depends on the external transmitter used in the application.
It is advised that it is within the Recommended max. Rx Error (+/-1.5% in this example).
Larger Transmitter Errors are acceptable but must lie within the Accepted Receiver Error.

**Related Links**

Clock Generation – Baud-Rate Generator on page 519
Asynchronous Arithmetic Mode BAUD Value Selection on page 520

### 32.6.3. Additional Features

#### 32.6.3.1. Parity

Even or odd parity can be selected for error checking by writing 0x1 to the Frame Format bit field in the Control A register (CTRLA.FORM).

If *even parity* is selected (CTRLB.PMODE=0), the parity bit of an outgoing frame is '1' if the data contains an odd number of bits that are '1', making the total number of '1' even.

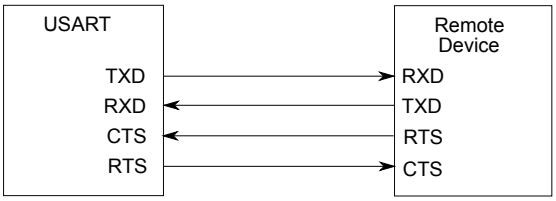
If *odd parity* is selected (CTRLB.PMODE=1), the parity bit of an outgoing frame is '1' if the data contains an even number of bits that are '0', making the total number of '1' odd.

When parity checking is enabled, the parity checker calculates the parity of the data bits in incoming frames and compares the result with the parity bit of the corresponding frame. If a parity error is detected, the Parity Error bit in the Status register (STATUS.PERR) is set.

### 32.6.3.2. Hardware Handshaking

The USART features an out-of-band hardware handshaking flow control mechanism, implemented by connecting the RTS and CTS pins with the remote device, as shown in the figure below.

**Figure 32-7. Connection with a Remote Device for Hardware Handshaking**

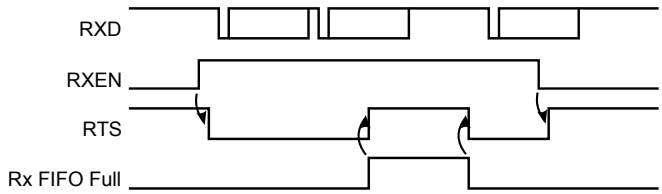



Hardware handshaking is only available in the following configuration:

- USART with internal clock (CTRLA.MODE=1),
- Asynchronous mode (CTRLA.CMODE=0),
- and Flow control pinout (CTRLA.TXPO=2).

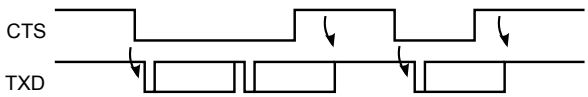
When the receiver is disabled or the receive FIFO is full, the receiver will drive the RTS pin high. This notifies the remote device to stop transfer after the ongoing transmission. Enabling and disabling the receiver by writing to CTRLB.RXEN will set/clear the RTS pin after a synchronization delay. When the receive FIFO goes full, RTS will be set immediately and the frame being received will be stored in the shift register until the receive FIFO is no longer full.

**Figure 32-8. Receiver Behavior when Operating with Hardware Handshaking**




The current CTS Status is in the STATUS register (STATUS.CTS). Character transmission will start only if STATUS.CTS=0. When CTS is set, the transmitter will complete the ongoing transmission and stop transmitting.

**Figure 32-9. Transmitter Behavior when Operating with Hardware Handshaking**




### 32.6.3.3. IrDA Modulation and Demodulation

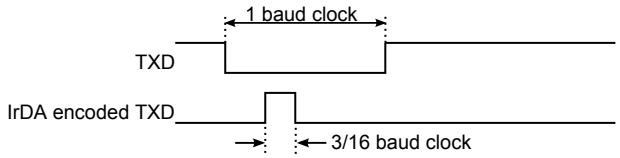
Transmission and reception can be encoded IrDA compliant up to 115.2 kb/s. IrDA modulation and demodulation work in the following configuration:

- IrDA encoding enabled (CTRLB.ENC=1),
- Asynchronous mode (CTRLA.CMODE=0),

- and 16x sample rate (CTRLA.SAMPR[0]=0).

During transmission, each low bit is transmitted as a high pulse. The pulse width is 3/16 of the baud rate period, as illustrated in the figure below.

**Figure 32-10. IrDA Transmit Encoding**




The reception decoder has two main functions.

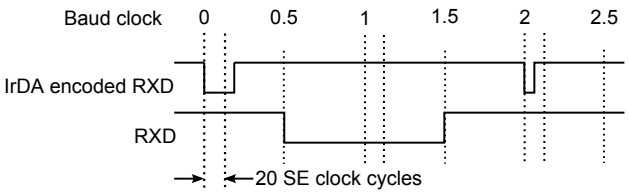
The first is to synchronize the incoming data to the IrDA baud rate counter. Synchronization is performed at the start of each zero pulse.

The second main function is to decode incoming Rx data. If a pulse width meets the minimum length set by configuration (RXPL.RXPL), it is accepted. When the baud rate counter reaches its middle value (1/2 bit length), it is transferred to the receiver.

**Note:** Note that the polarity of the transmitter and receiver are opposite: During transmission, a '0' bit is transmitted as a '1' pulse. During reception, an accepted '0' pulse is received as a '0' bit.

**Example:** The figure below illustrates reception where RXPL.RXPL is set to 19. This indicates that the pulse width should be at least 20 SE clock cycles. When using BAUD=0xE666 or 160 SE cycles per bit, this corresponds to 2/16 baud clock as minimum pulse width required. In this case the first bit is accepted as a '0', the second bit is a '1', and the third bit is also a '1'. A low pulse is rejected since it does not meet the minimum requirement of 2/16 baud clock.

**Figure 32-11. IrDA Receive Decoding**



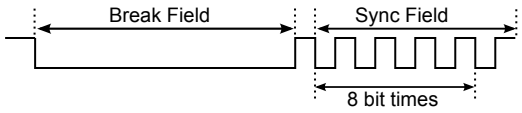

### 32.6.3.4. Break Character Detection and Auto-Baud

Break character detection and auto-baud are available in this configuration:

- Auto-baud frame format (CTRLA.FORM = 0x04 or 0x05),
- Asynchronous mode (CTRLA.CMODE = 0),
- and 16x sample rate using fractional baud rate generation (CTRLA.SAMPR = 1).

The auto-baud follows the LIN format. All LIN Frames start with a Break Field followed by a Sync Field. The USART uses a break detection threshold of greater than 11 nominal bit times at the configured baud rate. At any time, if more than 11 consecutive dominant bits are detected on the bus, the USART detects a Break Field. When a Break Field has been detected, the Receive Break interrupt flag (INTFLAG.RXBRK) is set and the USART expects the Sync Field character to be 0x55. This field is used to update the actual baud rate in order to stay synchronized. If the received Sync character is not 0x55, then the Inconsistent Sync Field error flag (STATUS.ISF) is set along with the Error interrupt flag (INTFLAG.ERROR), and the baud rate is unchanged.

**Figure 32-12. LIN Break and Sync Fields**




After a break field is detected and the start bit of the Sync Field is detected, a counter is started. The counter is then incremented for the next 8 bit times of the Sync Field. At the end of these 8 bit times, the counter is stopped. At this moment, the 13 most significant bits of the counter (value divided by 8) give the new clock divider (BAUD.BAUD), and the 3 least significant bits of this value (the remainder) give the new Fractional Part (BAUD.FP).

When the Sync Field has been received, the clock divider (BAUD.BAUD) and the Fractional Part (BAUD.FP) are updated after a synchronization delay. After the Break and Sync Fields are received, multiple characters of data can be received.

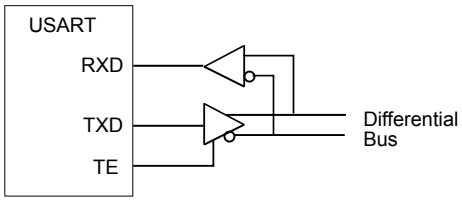
#### 32.6.3.5. RS485

RS485 is available with the following configuration:

- USART frame format (CTRLA.FORM = 0x00 or 0x01)
- RS485 pinout (CTRLA.TXPO=0x3).

The RS485 feature enables control of an external line driver as shown in the figure below. While operating in RS485 mode, the transmit enable pin (TE) is driven high when the transmitter is active.

**Figure 32-13. RS485 Bus Connection**




The TE pin will remain high for the complete frame including stop bit(s). If a Guard Time is programmed in the Control C register (CTRLC.GTIME), the line will remain driven after the last character completion. The following figure shows a transfer with one stop bit and CTRLC.GTIME=3.

**Figure 32-14. Example of TE Drive with Guard Time**

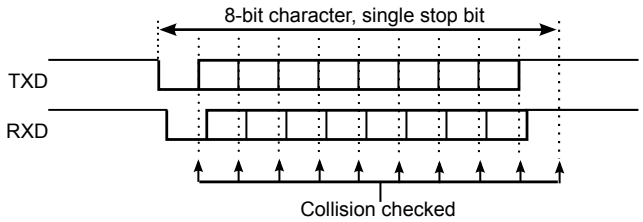

The Transmit Complete interrupt flag (INTFLAG.TXC) will be raised after the guard time is complete and TE goes low.

#### 32.6.3.6. Collision Detection

When the receiver and transmitter are connected either through pin configuration or externally, transmit collision can be detected after selecting the Collision Detection Enable bit in the CTRLB register (CTRLB.COLDEN=1). To detect collision, the receiver and transmitter must be enabled (CTRLB.RXEN=1 and CTRLB.TXEN=1).

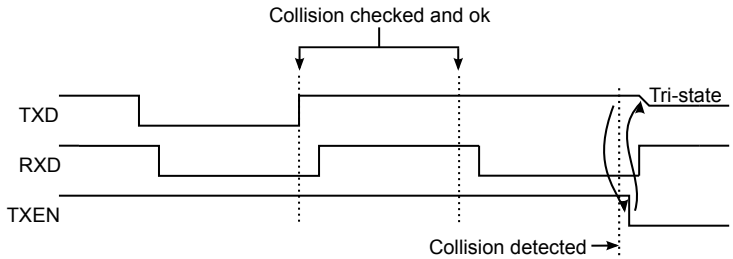
Collision detection is performed for each bit transmitted by comparing the received value with the transmit value, as shown in the figure below. While the transmitter is idle (no transmission in progress), characters can be received on RxD without triggering a collision.

**Figure 32-15. Collision Checking**




The next figure shows the conditions for a collision detection. In this case, the start bit and the first data bit are received with the same value as transmitted. The second received data bit is found to be different than the transmitted bit at the detection point, which indicates a collision.

**Figure 32-16. Collision Detected**




When a collision is detected, the USART follows this sequence:

1. Abort the current transfer.
2. Flush the transmit buffer.
3. Disable transmitter (CTRLB.TXEN=0)
   - This is done after a synchronization delay. The CTRLB Synchronization Busy bit (SYNCBUSY.CTRLB) will be set until this is complete.
   - After disabling, the TxD pin will be tri-stated.
4. Set the Collision Detected bit (STATUS.COLL) along with the Error interrupt flag (INTFLAG.ERROR).
5. Set the Transmit Complete interrupt flag (INTFLAG.TXC), since the transmit buffer no longer contains data.

After a collision, software must manually enable the transmitter again before continuing, after assuring that the CTRLB Synchronization Busy bit (SYNCBUSY.CTRLB) is not set.

#### 32.6.3.7. Loop-Back Mode

For loop-back mode, configure the Receive Data Pinout (CTRLA.RXPO) and Transmit Data Pinout (CTRLA.TXPO) to use the same data pins for transmit and receive. The loop-back is through the pad, so the signal is also available externally.

#### 32.6.3.8. Start-of-Frame Detection

The USART start-of-frame detector can wake up the CPU when it detects a start bit. In standby sleep mode, the internal fast startup oscillator must be selected as the GCLK_SERCOMx_CORE source.

When a 1-to-0 transition is detected on RxD, the 8MHz Internal Oscillator is powered up and the USART clock is enabled. After startup, the rest of the data frame can be received, provided that the baud rate is slow enough in relation to the fast startup internal oscillator start-up time. Refer to *Electrical Characteristics* for details. The start-up time of this oscillator varies with supply voltage and temperature.

The USART start-of-frame detection works both in asynchronous and synchronous modes. It is enabled by writing '1' to the Start of Frame Detection Enable bit in the Control B register (CTRLB.SFDE).

If the Receive Start Interrupt Enable bit in the Interrupt Enable Set register (INTENSET.RXS) is set, the Receive Start interrupt is generated immediately when a start is detected.

When using start-of-frame detection without the Receive Start interrupt, start detection will force the 8MHz Internal Oscillator and USART clock active while the frame is being received. In this case, the CPU will not wake up until the Receive Complete interrupt is generated.

**Related Links**

Electrical Characteristics 85°C on page 1111

#### 32.6.3.9. Sample Adjustment

In asynchronous mode (CTRLA.CMODE=0), three samples in the middle are used to determine the value based on majority voting. The three samples used for voting can be selected using the Sample Adjustment bit field in Control A register (CTRLA.SAMPA). When CTRLA.SAMPA=0, samples 7-8-9 are used for 16x oversampling, and samples 3-4-5 are used for 8x oversampling.

### 32.6.4. DMA, Interrupts and Events

**Table 32-4. Module Request for SERCOM USART**

| Condition | Request | | |
|---|---|---|---|
| | DMA | Interrupt | Event |
| Data Register Empty (DRE) | Yes<br>(request cleared when data is written) | Yes | NA |
| Receive Complete (RXC) | Yes<br>(request cleared when data is read) | Yes | |
| Transmit Complete (TXC) | NA | Yes | |
| Receive Start (RXS) | NA | Yes | |
| Clear to Send Input Change (CTSIC) | NA | Yes | |
| Receive Break (RXBRK) | NA | Yes | |
| Error (ERROR) | NA | Yes | |

#### 32.6.4.1. DMA Operation

The USART generates the following DMA requests:

- Data received (RX): The request is set when data is available in the receive FIFO. The request is cleared when DATA is read.
- Data transmit (TX): The request is set when the transmit buffer (TX DATA) is empty. The request is cleared when DATA is written.

#### 32.6.4.2. Interrupts

The USART has the following interrupt sources. These are asynchronous interrupts, and can wake up the device from any sleep mode:

- Data Register Empty (DRE)
- Receive Complete (RXC)
- Transmit Complete (TXC)
- Receive Start (RXS)
- Clear to Send Input Change (CTSIC)

- Received Break (RXBRK)
- Error (ERROR)

Each interrupt source has its own interrupt flag. The interrupt flag in the Interrupt Flag Status and Clear register (INTFLAG) will be set when the interrupt condition is met. Each interrupt can be individually enabled by writing '1' to the corresponding bit in the Interrupt Enable Set register (INTENSET), and disabled by writing '1' to the corresponding bit in the Interrupt Enable Clear register (INTENCLR).

An interrupt request is generated when the interrupt flag is set and if the corresponding interrupt is enabled. The interrupt request remains active until either the interrupt flag is cleared, the interrupt is disabled, or the USART is reset. For details on clearing interrupt flags, refer to the INTFLAG register description.

The USART has one common interrupt request line for all the interrupt sources. The value of INTFLAG indicates which interrupt is executed. Note that interrupts must be globally enabled for interrupt requests. Refer to *Nested Vector Interrupt Controller* for details.

**Related Links**

Nested Vector Interrupt Controller on page 44

#### 32.6.4.3. Events

Not applicable.

### 32.6.5. Sleep Mode Operation

The behavior in sleep mode is depending on the clock source and the Run In Standby bit in the Control A register (CTRLA.RUNSTDBY):

- Internal clocking, CTRLA.RUNSTDBY=1: GCLK_SERCOMx_CORE can be enabled in all sleep modes. Any interrupt can wake up the device.
- External clocking, CTRLA.RUNSTDBY=1: The Receive Start and the Receive Complete interrupt(s) can wake up the device.
- Internal clocking, CTRLA.RUNSTDBY=0: Internal clock will be disabled, after any ongoing transfer was completed. The Receive Start and the Receive Complete interrupt(s) can wake up the device.
- External clocking, CTRLA.RUNSTDBY=0: External clock will be disconnected, after any ongoing transfer was completed. All reception will be dropped.

### 32.6.6. Synchronization

Due to asynchronicity between the main clock domain and the peripheral clock domains, some registers need to be synchronized when written or read.

The following bits are synchronized when written:

- Software Reset bit in the CTRLA register (CTRLA.SWRST)
- Enable bit in the CTRLA register (CTRLA.ENABLE)
- Receiver Enable bit in the CTRLB register (CTRLB.RXEN)
- Transmitter Enable bit in the Control B register (CTRLB.TXEN)

**Note:** CTRLB.RXEN is write-synchronized somewhat differently. See also CTRLB for details.

Required write-synchronization is denoted by the "Write-Synchronized" property in the register description.

**Related Links**

Register Synchronization on page 126

## 32.7. Register Summary

| Offset | Name | Bit Pos. | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 0x00 | CTRLA | 7:0 | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| 0x01 | | 15:8 | SAMPR[2:0] | | | | | | | IBON |
| 0x02 | | 23:16 | SAMPA[1:0] | | RXPO[1:0] | | | | TXPO[1:0] | |
| 0x03 | | 31:24 | | DORD | CPOL | CMODE | FORM[3:0] | | | |
| 0x04 | CTRLB | 7:0 | | SBMODE | | | | CHSIZE[2:0] | | |
| 0x05 | | 15:8 | | | PMODE | | | ENC | SFDE | COLDEN |
| 0x06 | | 23:16 | | | | | | | RXEN | TXEN |
| 0x07 | | 31:24 | | | | | | | | |
| 0x08 | CTRLC | 7:0 | | | | | | GTIME[2:0] | | |
| 0x09 | | 15:8 | | | | | HDRDLY[1:0] | | BRKLEN[1:0] | |
| 0x0A | | 23:16 | | | | | | | | |
| 0x0B | | 31:24 | | | | | | | | |
| 0x0C | BAUD | 7:0 | BAUD[7:0] | | | | | | | |
| 0x0D | | 15:8 | BAUD[15:8] | | | | | | | |
| 0x0E | RXPL | 7:0 | RXPL[7:0] | | | | | | | |
| 0x0F ... 0x13 | Reserved | | | | | | | | | |
| 0x14 | INTENCLR | 7:0 | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| 0x15 | Reserved | | | | | | | | | |
| 0x16 | INTENSET | 7:0 | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| 0x17 | Reserved | | | | | | | | | |
| 0x18 | INTFLAG | 7:0 | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| 0x19 | Reserved | | | | | | | | | |
| 0x1A | STATUS | 7:0 | | TXE | COLL | ISF | CTS | BUFOVF | FERR | PERR |
| 0x1B | | 15:8 | | | | | | | | |
| 0x1C | SYNCBUSY | 7:0 | | | | | | CTRLB | ENABLE | SWRST |
| 0x1D | | 15:8 | | | | | | | | |
| 0x1E | | 23:16 | | | | | | | | |
| 0x1F | | 31:24 | | | | | | | | |
| 0x20 ... 0x27 | Reserved | | | | | | | | | |
| 0x28 | DATA | 7:0 | DATA[7:0] | | | | | | | |
| 0x29 | | 15:8 | | | | | | | | DATA[8:8] |
| 0x2A ... 0x2F | Reserved | | | | | | | | | |
| 0x30 | DBGCTRL | 7:0 | | | | | | | | DBGSTOP |

## 32.8. Register Description

Registers can be 8, 16, or 32 bits wide. Atomic 8-, 16-, and 32-bit accesses are supported. In addition, the 8-bit quarters and 16-bit halves of a 32-bit register, and the 8-bit halves of a 16-bit register can be accessed directly.

Some registers require synchronization when read and/or written. Synchronization is denoted by the "Read-Synchronized" and/or "Write-Synchronized" property in each individual register description.

Optional write-protection by the Peripheral Access Controller (PAC) is denoted by the "PAC Write-Protection" property in each individual register description.

Some registers are enable-protected, meaning they can only be written when the module is disabled. Enable-protection is denoted by the "Enable-Protected" property in each individual register description.

### 32.8.1. Control A

**Name:** CTRLA
**Offset:** 0x00
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | DORD | CPOL | CMODE | FORM[3:0] | | | |
| Access | | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | SAMPA[1:0] | | RXPO[1:0] | | | | TXPO[1:0] | |
| Access | R/W | R/W | R/W | R/W | | | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | SAMPR[2:0] | | | | | | | IBON |
| Access | R/W | R/W | R/W | | | | | R |
| Reset | 0 | 0 | 0 | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RUNSTDBY | | | MODE[2:0] | | | ENABLE | SWRST |
| Access | R/W | | | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | | 0 | 0 | 0 | 0 | 0 |

**Bit 30 – DORD: Data Order**
This bit selects the data order when a character is shifted out from the Data register.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | MSB is transmitted first. |
| 1 | LSB is transmitted first. |

**Bit 29 – CPOL: Clock Polarity**
This bit selects the relationship between data output change and data input sampling in synchronous mode.

This bit is not synchronized.

| CPOL | TxD Change | RxD Sample |
|---|---|---|
| 0x0 | Rising XCK edge | Falling XCK edge |
| 0x1 | Falling XCK edge | Rising XCK edge |

**Bit 28 – CMODE: Communication Mode**
This bit selects asynchronous or synchronous communication.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Asynchronous communication. |
| 1 | Synchronous communication. |

**Bits 27:24 – FORM[3:0]: Frame Format**
These bits define the frame format.

These bits are not synchronized.

| FORM[3:0] | Description |
|---|---|
| 0x0 | USART frame |
| 0x1 | USART frame with parity |
| 0x2-0x3 | Reserved |
| 0x4 | Auto-baud - break detection and auto-baud. |
| 0x5 | Auto-baud - break detection and auto-baud with parity |
| 0x6-0xF | Reserved |

**Bits 23:22 – SAMPA[1:0]: Sample Adjustment**
These bits define the sample adjustment.

These bits are not synchronized.

| SAMPA[1:0] | 16x Over-sampling (CTRLA.SAMPR=0 or 1) | 8x Over-sampling (CTRLA.SAMPR=2 or 3) |
|---|---|---|
| 0x0 | 7-8-9 | 3-4-5 |
| 0x1 | 9-10-11 | 4-5-6 |
| 0x2 | 11-12-13 | 5-6-7 |
| 0x3 | 13-14-15 | 6-7-8 |

**Bits 21:20 – RXPO[1:0]: Receive Data Pinout**
These bits define the receive data (RxD) pin configuration.

These bits are not synchronized.

| RXPO[1:0] | Name | Description |
|---|---|---|
| 0x0 | PAD[0] | SERCOM PAD[0] is used for data reception |
| 0x1 | PAD[1] | SERCOM PAD[1] is used for data reception |
| 0x2 | PAD[2] | SERCOM PAD[2] is used for data reception |
| 0x3 | PAD[3] | SERCOM PAD[3] is used for data reception |

**Bits 17:16 – TXPO[1:0]: Transmit Data Pinout**
These bits define the transmit data (TxD) and XCK pin configurations.

This bit is not synchronized.

| TXPO | TxD Pin Location | XCK Pin Location (When Applicable) | RTS | CTS |
|---|---|---|---|---|
| 0x0 | SERCOM PAD[0] | SERCOM PAD[1] | N/A | N/A |
| 0x1 | SERCOM PAD[2] | SERCOM PAD[3] | N/A | N/A |
| 0x2 | SERCOM PAD[0] | N/A | SERCOM PAD[2] | SERCOM PAD[3] |
| 0x3 | SERCOM_PAD[0] | SERCOM_PAD[1] | SERCOM_PAD[2] | N/A |

**Bits 15:13 – SAMPR[2:0]: Sample Rate**
These bits select the sample rate.

These bits are not synchronized.

| SAMPR[2:0] | Description |
|---|---|
| 0x0 | 16x over-sampling using arithmetic baud rate generation. |
| 0x1 | 16x over-sampling using fractional baud rate generation. |
| 0x2 | 8x over-sampling using arithmetic baud rate generation. |
| 0x3 | 8x over-sampling using fractional baud rate generation. |
| 0x4 | 3x over-sampling using arithmetic baud rate generation. |
| 0x5-0x7 | Reserved |

**Bit 8 – IBON: Immediate Buffer Overflow Notification**
This bit controls when the buffer overflow status bit (STATUS.BUFOVF) is asserted when a buffer overflow occurs.

| Value | Description |
|---|---|
| 0 | STATUS.BUFOVF is asserted when it occurs in the data stream. |
| 1 | STATUS.BUFOVF is asserted immediately upon buffer overflow. |

**Bit 7 – RUNSTDBY: Run In Standby**
This bit defines the functionality in standby sleep mode.

This bit is not synchronized.

| RUNSTDBY | External Clock | Internal Clock |
|---|---|---|
| 0x0 | External clock is disconnected when ongoing transfer is finished. All reception is dropped. | Generic clock is disabled when ongoing transfer is finished. The device can wake up on Receive Start or Transfer Complete interrupt. |
| 0x1 | Wake on Receive Start or Receive Complete interrupt. | Generic clock is enabled in all sleep modes. Any interrupt can wake up the device. |

**Bits 4:2 – MODE[2:0]: Operating Mode**
These bits select the USART serial communication interface of the SERCOM.

These bits are not synchronized.

| Value | Description |
|---|---|
| 0x0 | USART with external clock |
| 0x1 | USART with internal clock |

**Bit 1 – ENABLE: Enable**
Due to synchronization, there is delay from writing CTRLA.ENABLE until the peripheral is enabled/disabled. The value written to CTRLA.ENABLE will read back immediately and the Enable Synchronization Busy bit in the Synchronization Busy register (SYNCBUSY.ENABLE) will be set. SYNCBUSY.ENABLE is cleared when the operation is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The peripheral is disabled or being disabled. |
| 1 | The peripheral is enabled or being enabled. |

**Bit 0 – SWRST: Software Reset**
Writing '0' to this bit has no effect.

Writing '1' to this bit resets all registers in the SERCOM, except DBGCTRL, to their initial state, and the SERCOM will be disabled.

Writing '1' to CTRLA.SWRST will always take precedence, meaning that all other writes in the same write-operation will be discarded. Any register write access during the ongoing reset will result in an APB error. Reading any register will return the reset value of the register.

Due to synchronization, there is a delay from writing CTRLA.SWRST until the reset is complete. CTRLA.SWRST and SYNCBUSY.SWRST will both be cleared when the reset is complete.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | There is no reset operation ongoing. |
| 1 | The reset operation is ongoing. |

### 32.8.2. Control B

**Name:** CTRLB
**Offset:** 0x04
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected, Write-Synchronized

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | RXEN | TXEN |
| Access | | | | | | | R/W | R/W |
| Reset | | | | | | | 0 | 0 |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | PMODE | | | ENC | SFDE | COLDEN |
| Access | | | R/W | | | R/W | R/W | R/W |
| Reset | | | 0 | | | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | SBMODE | | | | CHSIZE[2:0] | | |
| Access | | R/W | | | | R/W | R/W | R/W |
| Reset | | 0 | | | | 0 | 0 | 0 |

**Bit 17 – RXEN: Receiver Enable**
Writing '0' to this bit will disable the USART receiver. Disabling the receiver will flush the receive buffer and clear the FERR, PERR and BUFOVF bits in the STATUS register.

Writing '1' to CTRLB.RXEN when the USART is disabled will set CTRLB.RXEN immediately. When the USART is enabled, CTRLB.RXEN will be cleared, and SYNCBUSY.CTRLB will be set and remain set until the receiver is enabled. When the receiver is enabled, CTRLB.RXEN will read back as '1'.

Writing '1' to CTRLB.RXEN when the USART is enabled will set SYNCBUSY.CTRLB, which will remain set until the receiver is enabled, and CTRLB.RXEN will read back as '1'.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The receiver is disabled or being enabled. |
| 1 | The receiver is enabled or will be enabled when the USART is enabled. |

**Bit 16 – TXEN: Transmitter Enable**
Writing '0' to this bit will disable the USART transmitter. Disabling the transmitter will not become effective until ongoing and pending transmissions are completed.

Writing '1' to CTRLB.TXEN when the USART is disabled will set CTRLB.TXEN immediately. When the USART is enabled, CTRLB.TXEN will be cleared, and SYNCBUSY.CTRLB will be set and remain set until the transmitter is enabled. When the transmitter is enabled, CTRLB.TXEN will read back as '1'.

Writing '1' to CTRLB.TXEN when the USART is enabled will set SYNCBUSY.CTRLB, which will remain set until the receiver is enabled, and CTRLB.TXEN will read back as '1'.

This bit is not enable-protected.

| Value | Description |
|---|---|
| 0 | The transmitter is disabled or being enabled. |
| 1 | The transmitter is enabled or will be enabled when the USART is enabled. |

**Bit 13 – PMODE: Parity Mode**
This bit selects the type of parity used when parity is enabled (CTRLA.FORM is '1'). The transmitter will automatically generate and send the parity of the transmitted data bits within each frame. The receiver will generate a parity value for the incoming data and parity bit, compare it to the parity mode and, if a mismatch is detected, STATUS.PERR will be set.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Even parity. |
| 1 | Odd parity. |

**Bit 10 – ENC: Encoding Format**
This bit selects the data encoding format.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Data is not encoded. |
| 1 | Data is IrDA encoded. |

**Bit 9 – SFDE: Start of Frame Detection Enable**
This bit controls whether the start-of-frame detector will wake up the device when a start bit is detected on the RxD line.

This bit is not synchronized.

| SFDE | INTENSET.RXS | INTENSET.RXC | Description |
|---|---|---|---|
| 0 | X | X | Start-of-frame detection disabled. |
| 1 | 0 | 0 | Reserved |
| 1 | 0 | 1 | Start-of-frame detection enabled. RXC wakes up the device from all sleep modes. |
| 1 | 1 | 0 | Start-of-frame detection enabled. RXS wakes up the device from all sleep modes. |
| 1 | 1 | 1 | Start-of-frame detection enabled. Both RXC and RXS wake up the device from all sleep modes. |

**Bit 8 – COLDEN: Collision Detection Enable**
This bit enables collision detection.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | Collision detection is not enabled. |
| 1 | Collision detection is enabled. |

**Bit 6 – SBMODE: Stop Bit Mode**
This bit selects the number of stop bits transmitted.

This bit is not synchronized.

| Value | Description |
|---|---|
| 0 | One stop bit. |
| 1 | Two stop bits. |

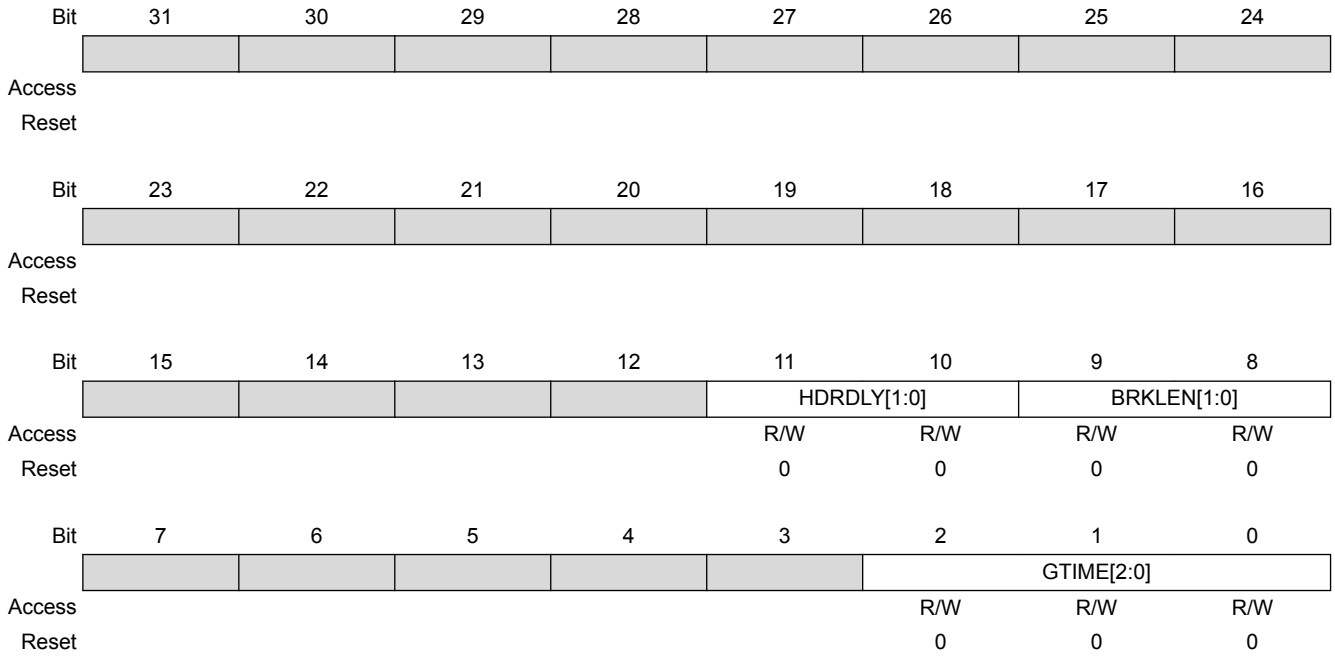
**Bits 2:0 – CHSIZE[2:0]: Character Size**
These bits select the number of bits in a character.

These bits are not synchronized.

| CHSIZE[2:0] | Description |
|---|---|
| 0x0 | 8 bits |
| 0x1 | 9 bits |
| 0x2-0x4 | Reserved |
| 0x5 | 5 bits |
| 0x6 | 6 bits |
| 0x7 | 7 bits |

### 32.8.3. Control C

**Name:** CTRLC
**Offset:** 0x08
**Reset:** 0x00000000
**Property:** PAC Write-Protection, Enable-Protected




**Bits 11:10 – HDRDLY[1:0]: LIN Master Header Delay**
These bits define the delay between break and sync transmission in addition to the delay between the sync and identifier (ID) fields when in LIN master mode (CTRLA.FORM=0x2).
This field is only valid when using the LIN header command (CTRLB.LINCMD=0x2).

| Value | Description |
|---|---|
| 0x0 | Delay between break and sync transmission is 1 bit time.<br>Delay between sync and ID transmission is 1 bit time. |
| 0x1 | Delay between break and sync transmission is 4 bit time.<br>Delay between sync and ID transmission is 4 bit time. |
| 0x2 | Delay between break and sync transmission is 8 bit time.<br>Delay between sync and ID transmission is 4 bit time. |
| 0x3 | Delay between break and sync transmission is 14 bit time.<br>Delay between sync and ID transmission is 4 bit time. |

**Bits 9:8 – BRKLEN[1:0]: LIN Master Break Length**
These bits define the length of the break field transmitted when in LIN master mode (CTRLA.FORM=0x2).

| Value | Description |
|---|---|
| 0x0 | Break field transmission is 13 bit times |
| 0x1 | Break field transmission is 17 bit times |
| 0x2 | Break field transmission is 21 bit times |
| 0x3 | Break field transmission is 26 bit times |

**Bits 2:0 – GTIME[2:0]: Guard Time**
These bits define the guard time when using RS485 mode (CTRLA.TXPO=0x3).

For RS485 mode, the guard time is programmable from 0-7 bit times and defines the time that the transmit enable pin (TE) remains high after the last stop bit is transmitted and there is no remaining data to be transmitted.

| Value | Description |
|---|---|
| 0 | The receiver is disabled or being enabled. |
| 1 | The receiver is enabled or will be enabled when the USART is enabled. |

### 32.8.4. Baud

**Name:** BAUD
**Offset:** 0x0C
**Reset:** 0x0000
**Property:** Enable-Protected, PAC Write-Protection

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | BAUD[15:8] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | BAUD[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 15:0 – BAUD[15:0]: Baud Value**
Arithmetic Baud Rate Generation (`CTRLA.SAMPR[0]=0`):

These bits control the clock generation, as described in the SERCOM Baud Rate section.

If Fractional Baud Rate Generation (`CTRLA.SAMPR[0]=1`) bit positions 15 to 13 are replaced by FP[2:0] Fractional Part:

- **Bits 15:13 - FP[2:0]: Fractional Part**

  These bits control the clock generation, as described in the *SERCOM Clock Generation – Baud-Rate Generator* section.
- **Bits 12:0 - BAUD[21:0]: Baud Value**

  These bits control the clock generation, as described in the *SERCOM Clock Generation – Baud-Rate Generator* section.

### 32.8.5. Receive Pulse Length Register

**Name:** RXPL
**Offset:** 0x0E
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | RXPL[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 7:0 – RXPL[7:0]: Receive Pulse Length**
When the encoding format is set to IrDA (CTRLB.ENC=1), these bits control the minimum pulse length that is required for a pulse to be accepted by the IrDA receiver with regards to the serial engine clock period $SE_{per}$.

$$PULSE \geq \left(\mathrm{RXPL} + 2\right) \cdot SE_{per}$$

### 32.8.6. Interrupt Enable Clear

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Set register (INTENSET).

**Name:** INTENCLR
**Offset:** 0x14
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR: Error Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Error Interrupt Enable bit, which disables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 5 – RXBRK: Receive Break Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Break Interrupt Enable bit, which disables the Receive Break interrupt.

| Value | Description |
|---|---|
| 0 | Receive Break interrupt is disabled. |
| 1 | Receive Break interrupt is enabled. |

**Bit 4 – CTSIC: Clear to Send Input Change Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Clear To Send Input Change Interrupt Enable bit, which disables the Clear To Send Input Change interrupt.

| Value | Description |
|---|---|
| 0 | Clear To Send Input Change interrupt is disabled. |
| 1 | Clear To Send Input Change interrupt is enabled. |

**Bit 3 – RXS: Receive Start Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Start Interrupt Enable bit, which disables the Receive Start interrupt.

| Value | Description |
|---|---|
| 0 | Receive Start interrupt is disabled. |
| 1 | Receive Start interrupt is enabled. |

**Bit 2 – RXC: Receive Complete Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Receive Complete interrupt is disabled. |
| 1 | Receive Complete interrupt is enabled. |

**Bit 1 – TXC: Transmit Complete Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Transmit Complete Interrupt Enable bit, which disables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Transmit Complete interrupt is disabled. |
| 1 | Transmit Complete interrupt is enabled. |

**Bit 0 – DRE: Data Register Empty Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Data Register Empty Interrupt Enable bit, which disables the Data Register Empty interrupt.

| Value | Description |
|---|---|
| 0 | Data Register Empty interrupt is disabled. |
| 1 | Data Register Empty interrupt is enabled. |

### 32.8.7. Interrupt Enable Set

This register allows the user to disable an interrupt without doing a read-modify-write operation. Changes in this register will also be reflected in the Interrupt Enable Clear register (INTENCLR).

**Name:** INTENSET
**Offset:** 0x16
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| Access | R/W | | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR: Error Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Error Interrupt Enable bit, which enables the Error interrupt.

| Value | Description |
|---|---|
| 0 | Error interrupt is disabled. |
| 1 | Error interrupt is enabled. |

**Bit 5 – RXBRK: Receive Break Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Receive Break Interrupt Enable bit, which enables the Receive Break interrupt.

| Value | Description |
|---|---|
| 0 | Receive Break interrupt is disabled. |
| 1 | Receive Break interrupt is enabled. |

**Bit 4 – CTSIC: Clear to Send Input Change Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Clear To Send Input Change Interrupt Enable bit, which enables the Clear To Send Input Change interrupt.

| Value | Description |
|---|---|
| 0 | Clear To Send Input Change interrupt is disabled. |
| 1 | Clear To Send Input Change interrupt is enabled. |

**Bit 3 – RXS: Receive Start Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Receive Start Interrupt Enable bit, which enables the Receive Start interrupt.

| Value | Description |
|---|---|
| 0 | Receive Start interrupt is disabled. |
| 1 | Receive Start interrupt is enabled. |

**Bit 2 – RXC: Receive Complete Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Receive Complete Interrupt Enable bit, which enables the Receive Complete interrupt.

| Value | Description |
|---|---|
| 0 | Receive Complete interrupt is disabled. |
| 1 | Receive Complete interrupt is enabled. |

**Bit 1 – TXC: Transmit Complete Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Transmit Complete Interrupt Enable bit, which enables the Transmit Complete interrupt.

| Value | Description |
|---|---|
| 0 | Transmit Complete interrupt is disabled. |
| 1 | Transmit Complete interrupt is enabled. |

**Bit 0 – DRE: Data Register Empty Interrupt Enable**
Writing '0' to this bit has no effect.

Writing '1' to this bit will set the Data Register Empty Interrupt Enable bit, which enables the Data Register Empty interrupt.

| Value | Description |
|---|---|
| 0 | Data Register Empty interrupt is disabled. |
| 1 | Data Register Empty interrupt is enabled. |

### 32.8.8. Interrupt Flag Status and Clear

**Name:** INTFLAG
**Offset:** 0x18
**Reset:** 0x00
**Property:** -

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | ERROR | | RXBRK | CTSIC | RXS | RXC | TXC | DRE |
| Access | R/W | | R/W | R/W | R/W | R | R/W | R |
| Reset | 0 | | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 7 – ERROR: Error**
This flag is cleared by writing '1' to it.

This bit is set when any error is detected. Errors that will set this flag have corresponding status flags in the STATUS register. Errors that will set this flag are COLL, ISF, BUFOVF, FERR, and PERR.Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 5 – RXBRK: Receive Break**
This flag is cleared by writing '1' to it.

This flag is set when auto-baud is enabled (CTRLA.FORM) and a break character is received.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 4 – CTSIC: Clear to Send Input Change**
This flag is cleared by writing a '1' to it.

This flag is set when a change is detected on the CTS pin.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 3 – RXS: Receive Start**
This flag is cleared by writing '1' to it.

This flag is set when a start condition is detected on the RxD line and start-of-frame detection is enabled (CTRLB.SFDE is '1').

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the Receive Start interrupt flag.

**Bit 2 – RXC: Receive Complete**
This flag is cleared by reading the Data register (DATA) or by disabling the receiver.

This flag is set when there are unread data in DATA.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

**Bit 1 – TXC: Transmit Complete**
This flag is cleared by writing '1' to it or by writing new data to DATA.

This flag is set when the entire frame in the transmit shift register has been shifted out and there are no new data in DATA.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear the flag.

**Bit 0 – DRE: Data Register Empty**
This flag is cleared by writing new data to DATA.

This flag is set when DATA is empty and ready to be written.

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

### 32.8.9. Status

**Name:** STATUS
**Offset:** 0x1A
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | TXE | COLL | ISF | CTS | BUFOVF | FERR | PERR |
| Access | | R/W | R/W | R/W | R | R/W | R/W | R/W |
| Reset | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bit 6 – TXE: Transmitter Empty**
When CTRLA.FORM is set to LIN master mode, this bit is set when any ongoing transmission is complete and TxDATA is empty. When CTRLA.FORM is not set to LIN master mode, this bit will always read back as zero.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 5 – COLL: Collision Detected**
This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when collision detection is enabled (CTRLB.COLDEN) and a collision is detected.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 4 – ISF: Inconsistent Sync Field**
This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when the frame format is set to auto-baud (CTRLA.FORM) and a sync field not equal to 0x55 is received.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 3 – CTS: Clear to Send**
This bit indicates the current level of the CTS pin when flow control is enabled (CTRLA.TXPO).

Writing '0' to this bit has no effect.

Writing '1' to this bit has no effect.

**Bit 2 – BUFOVF: Buffer Overflow**
Reading this bit before reading the Data register will indicate the error status of the next character to be read.

This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set when a buffer overflow condition is detected. A buffer overflow occurs when the receive buffer is full, there is a new character waiting in the receive shift register and a new start bit is detected.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 1 – FERR: Frame Error**
Reading this bit before reading the Data register will indicate the error status of the next character to be read.

This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set if the received character had a frame error, i.e., when the first stop bit is zero.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

**Bit 0 – PERR: Parity Error**
Reading this bit before reading the Data register will indicate the error status of the next character to be read.

This bit is cleared by writing '1' to the bit or by disabling the receiver.

This bit is set if parity checking is enabled (CTRLA.FORM is 0x1, 0x5) and a parity error is detected.

Writing '0' to this bit has no effect.

Writing '1' to this bit will clear it.

### 32.8.10. Synchronization Busy

**Name:** SYNCBUSY
**Offset:** 0x1C
**Reset:** 0x00000000
**Property:** -

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Access | | | | | | | | |
| Reset | | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CTRLB | ENABLE | SWRST |
| Access | | | | | | R | R | R |
| Reset | | | | | | 0 | 0 | 0 |

**Bit 2 – CTRLB: CTRLB Synchronization Busy**
Writing to the CTRLB register when the SERCOM is enabled requires synchronization. When writing to CTRLB the SYNCBUSY.CTRLB bit will be set until synchronization is complete. If CTRLB is written while SYNCBUSY.CTRLB is asserted, an APB error will be generated.

| Value | Description |
|---|---|
| 0 | CTRLB synchronization is not busy. |
| 1 | CTRLB synchronization is busy. |

**Bit 1 – ENABLE: SERCOM Enable Synchronization Busy**
Enabling and disabling the SERCOM (CTRLA.ENABLE) requires synchronization. When written, the SYNCBUSY.ENABLE bit will be set until synchronization is complete.

Writes to any register (except for CTRLA.SWRST) while enable synchronization is on-going will be discarded and an APB error will be generated.

| Value | Description |
|---|---|
| 0 | Enable synchronization is not busy. |
| 1 | Enable synchronization is busy. |

**Bit 0 – SWRST: Software Reset Synchronization Busy**
Resetting the SERCOM (CTRLA.SWRST) requires synchronization. When written, the SYNCBUSY.SWRST bit will be set until synchronization is complete.

Writes to any register while synchronization is on-going will be discarded and an APB error will be generated.

| Value | Description |
|---|---|
| 0 | SWRST synchronization is not busy. |
| 1 | SWRST synchronization is busy. |

### 32.8.11. Data

**Name:** DATA
**Offset:** 0x28
**Reset:** 0x0000
**Property:** -

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DATA[8:8] |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | DATA[7:0] | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Bits 8:0 – DATA[8:0]: Data**

Reading these bits will return the contents of the Receive Data register. The register should be read only when the Receive Complete Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.RXC) is set. The status bits in STATUS should be read before reading the DATA value in order to get any corresponding error.

Writing these bits will write the Transmit Data register. This register should be written only when the Data Register Empty Interrupt Flag bit in the Interrupt Flag Status and Clear register (INTFLAG.DRE) is set.

### 32.8.12. Debug Control

**Name:** DBGCTRL
**Offset:** 0x30
**Reset:** 0x00
**Property:** PAC Write-Protection

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | DBGSTOP |
| Access | | | | | | | | R/W |
| Reset | | | | | | | | 0 |

**Bit 0 – DBGSTOP: Debug Stop Mode**
This bit controls the baud-rate generator functionality when the CPU is halted by an external debugger.

| Value | Description |
|---|---|
| 0 | The baud-rate generator continues normal operation when the CPU is halted by an external debugger. |
| 1 | The baud-rate generator is halted when the CPU is halted by an external debugger. |