23. USART

23.1 Features

- Full-duplex operation
- Asynchronous or synchronous operation
  - Synchronous clock rates up to 1/2 of the device clock frequency
  - Asynchronous clock rates up to 1/8 of the device clock frequency
- Supports serial frames with 5, 6, 7, 8, or 9 data bits and 1 or 2 stop bits
- Fractional baud rate generator
  - Can generate desired baud rate from any system clock frequency
  - No need for external oscillator with certain frequencies
- Built-in error detection and correction schemes
  - Odd or even parity generation and parity check
  - Data overrun and framing error detection
  - Noise filtering includes false start bit detection and digital low-pass filter
- Separate interrupts for
  - Transmit complete
  - Transmit data register empty
  - Receive complete
- Multiprocessor communication mode
  - Addressing scheme to address a specific devices on a multi device bus
  - Enable unaddressed devices to automatically ignore all frames
- Master SPI mode
  - Double buffered operation
  - Configurable data order
  - Operation up to 1/2 of the peripheral clock frequency
- IRCOM module for IrDA compliant pulse modulation/demodulation

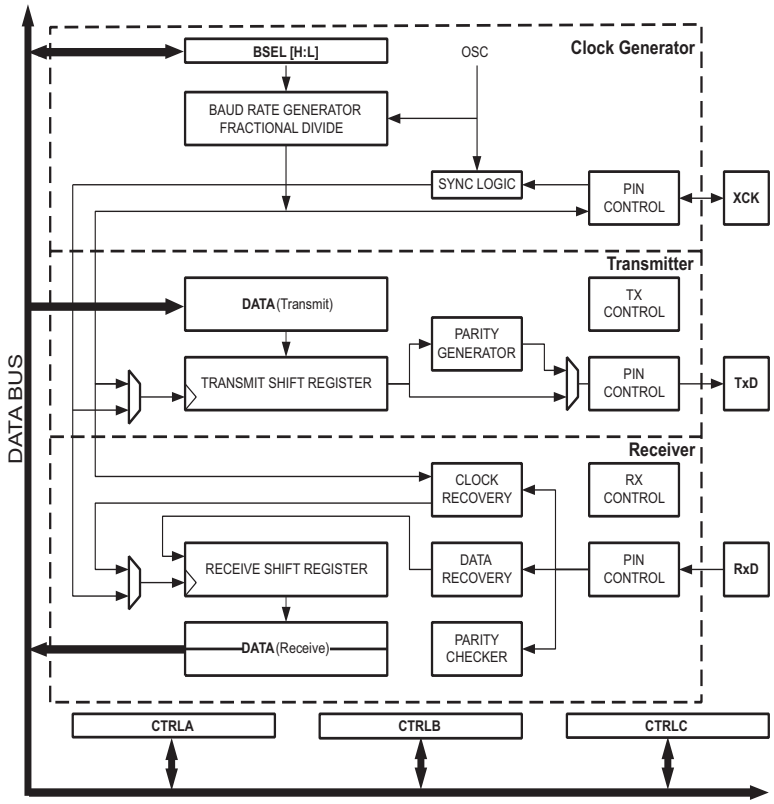
23.2 Overview

The universal synchronous and asynchronous serial receiver and transmitter (USART) is a fast and flexible serial communication module. The USART supports full-duplex communication and asynchronous and synchronous operation. The USART can be configured to operate in SPI master mode and used for SPI communication.

Communication is frame based, and the frame format can be customized to support a wide range of standards. The USART is buffered in both directions, enabling continued data transmission without any delay between frames. Separate interrupts for receive and transmit complete enable fully interrupt driven communication. Frame error and buffer overflow are detected in hardware and indicated with separate status flags. Even or odd parity generation and parity check can also be enabled.

A block diagram of the USART is shown in Figure 23-1 on page 281. The main functional blocks are the clock generator, the transmitter, and the receiver, which are indicated in dashed boxes.

Figure 23-1. USART block diagram.



The clock generator includes a fractional baud rate generator that is able to generate a wide range of USART baud rates from any system clock frequencies. This removes the need to use an external crystal oscillator with a specific frequency to achieve a required baud rate. It also supports external clock input in synchronous slave operation.

The transmitter consists of a single write buffer (DATA), a shift register, and a parity generator. The write buffer allows continuous data transmission without any delay between frames.

The receiver consists of a two-level receive buffer (DATA) and a shift register. Data and clock recovery units ensure robust synchronization and noise filtering during asynchronous data reception. It includes frame error, buffer overflow, and parity error detection.

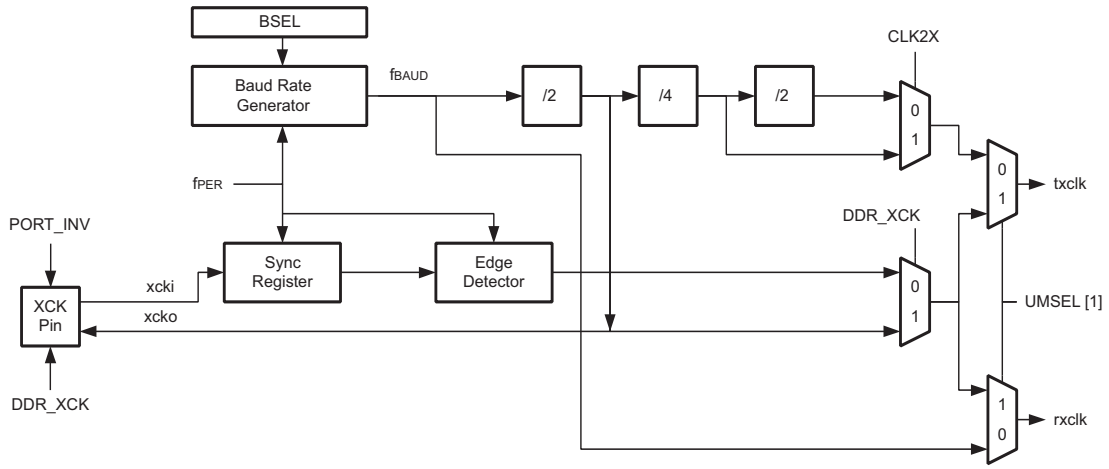
When the USART is set in master SPI mode, all USART-specific logic is disabled, leaving the transmit and receive buffers, shift registers, and baud rate generator enabled. Pin control and interrupt generation are identical in both modes. The registers are used in both modes, but their functionality differs for some control settings.

An IRCOM module can be enabled for one USART to support IrDA 1.4 physical compliant pulse modulation and demodulation for baud rates up to 115.2kbps. For details, refer to “IRCOM – IR Communication Module” on page 301.

23.3 Clock Generation

The clock used for baud rate generation and for shifting and sampling data bits is generated internally by the fractional baud rate generator or externally from the transfer clock (XCK) pin. Five modes of clock generation are supported: normal and double-speed asynchronous mode, master and slave synchronous mode, and master SPI mode.

Figure 23-2. Clock generation logic, block diagram.



23.3.1 Internal Clock Generation - The Fractional Baud Rate Generator

The fractional baud rate generator is used for internal clock generation for asynchronous modes, synchronous master mode, and master SPI mode operation. The output frequency generated (f_{BAUD}) is determined by the period setting (BSEL), an optional scale setting (BSCALE), and the peripheral clock frequency (f_{PER}). Table 23-1 on page 282 contains equations for calculating the baud rate (in bits per second) and for calculating the BSEL value for each mode of operation. It also shows the maximum baud rate versus peripheral clock frequency. BSEL can be set to any value between 0 and 4095. BSCALE can be set to any value between -7 and +7, and increases or decreases the baud rate slightly to provide the fractional baud rate scaling of the baud rate generator.

When BSEL is 0, BSCALE must also be 0. Also, the value $2^{ABS(BSCALE)}$ must at most be one half of the minimum number of clock cycles a frame requires. For more details, see "Fractional Baud Rate Generation" on page 289.

Table 23-1. Equations for calculating baud rate register settings.

Operating mode	Conditions	Baud rate[1] calculation	BSEL value calculation
Asynchronous normal speed mode (CLK2X = 0)	BSCALE ≥ 0 $f_{BAUD} \leq \frac{f_{PER}}{16}$	$f_{BAUD} = \frac{f_{PER}}{2^{BSCALE} \cdot 16(BSEL+1)}$	$BSEL = \frac{f_{PER}}{2^{BSCALE} \cdot 16 f_{BAUD}} - 1$
	BSCALE < 0 $f_{BAUD} \leq \frac{f_{PER}}{16}$	$f_{BAUD} = \frac{f_{PER}}{16((2^{BSCALE} \cdot BSEL)+1)}$	$BSEL = \frac{1}{2^{BSCALE}}\left(\frac{f_{PER}}{16 f_{BAUD}} - 1\right)$
Asynchronous double speed mode (CLK2X = 1)	BSCALE ≥ 0 $f_{BAUD} \leq \frac{f_{PER}}{8}$	$f_{BAUD} = \frac{f_{PER}}{2^{BSCALE} \cdot 8 \cdot (BSEL+1)}$	$BSEL = \frac{f_{PER}}{2^{BSCALE} \cdot 8 f_{BAUD}} - 1$
	BSCALE < 0 $f_{BAUD} \leq \frac{f_{PER}}{8}$	$f_{BAUD} = \frac{f_{PER}}{8((2^{BSCALE} \cdot BSEL)+1)}$	$BSEL = \frac{1}{2^{BSCALE}}\left(\frac{f_{PER}}{8 f_{BAUD}} - 1\right)$
Synchronous and master SPI mode	$f_{BAUD} < \frac{f_{PER}}{2}$	$f_{BAUD} = \frac{f_{PER}}{2 \cdot (BSEL+1)}$	$BSEL = \frac{f_{PER}}{2 f_{BAUD}} - 1$

Note: 1. The baud rate is defined to be the transfer rate in bits per second (bps)

For BSEL=0, all baud rates must be achieved by changing BSEL instead of setting BSCALE:

BSEL = ($2^{\text{desired BSCALE-1}}$)

BSCALE	BSEL		BSCALE	BSEL
1	0	→	0	1
2	0	→	0	3
3	0	→	0	7
4	0	→	0	15
5	0	→	0	31
6	0	→	0	63
7	0	→	0	127

23.3.2 External Clock

External clock (XCK) is used in synchronous slave mode operation. The XCK clock input is sampled on the peripheral clock frequency (f_{PER}), and the maximum XCK clock frequency (f_{XCK})is limited by the following:

$$f_{XCK} < \frac{f_{PER}}{4}$$

For each high and low period, XCK clock cycles must be sampled twice by the peripheral clock. If the XCK clock has jitter, or if the high/low period duty cycle is not 50/50, the maximum XCK clock speed must be reduced or the peripheral clock must be increased accordingly.

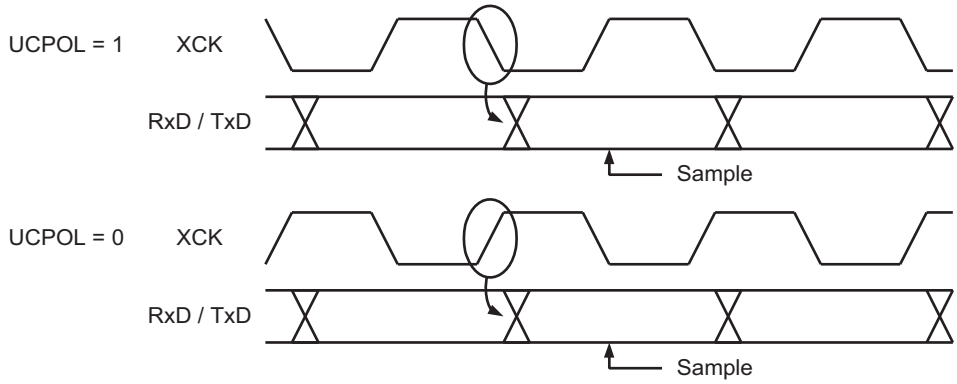
23.3.3 Double Speed Operation

Double speed operation allows for higher baud rates under asynchronous operation with lower peripheral clock frequencies. When this is enabled, the baud rate for a given asynchronous baud rate setting shown in Table 23-1 on page 282 will be doubled. In this mode, the receiver will use half the number of samples (reduced from 16 to 8) for data sampling and clock recovery. Due to the reduced sampling, a more accurate baud rate setting and peripheral clock are required. See “Asynchronous Data Reception” on page 287 for more details.

23.3.4 Synchronous Clock Operation

When synchronous mode is used, the XCK pin controls whether the transmission clock is input (slave mode) or output (master mode). The corresponding port pin must be set to output for master mode or to input for slave mode. The normal port operation of the XCK pin will be overridden. The dependency between the clock edges and data sampling or data change is the same. Data input (on RxD) is sampled at the XCK clock edge which is opposite the edge where data output (TxD) is changed.

Figure 23-3. Synchronous mode XCK timing.



Using the inverted I/O (INVEN) setting for the corresponding XCK port pin, the XCK clock edges used for data sampling and data change can be selected. If inverted I/O is disabled (INVEN=0), data will be changed at the rising XCK clock edge and sampled at the falling XCK clock edge. If inverted I/O is enabled (INVEN=1), data will be changed at the falling XCK clock edge and sampled at the rising XCK clock edge. For more details, see “I/O Ports” on page 139.

23.3.5 Master SPI Mode Clock Generation

For master SPI mode operation, only internal clock generation is supported. This is identical to the USART synchronous master mode, and the baud rate or BSEL setting is calculated using the same equations (see Table 23-1 on page 282).

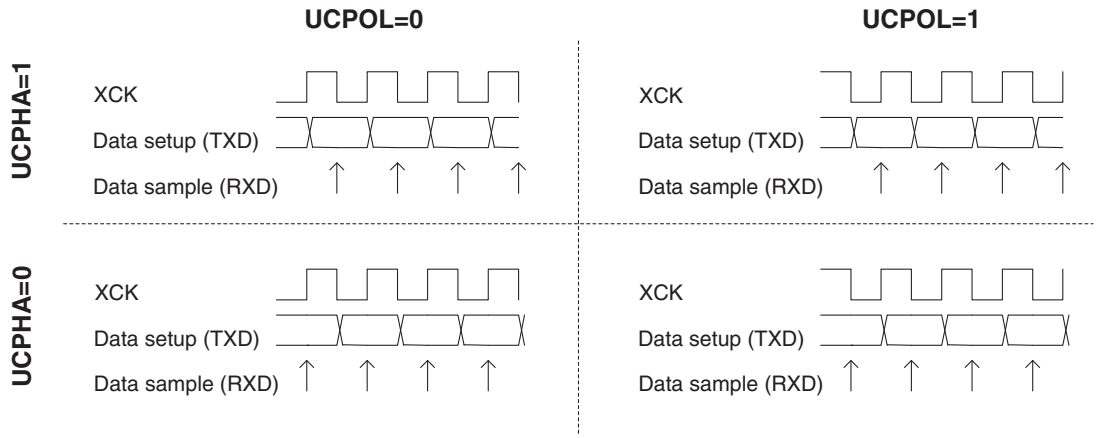
There are four combinations of the SPI clock (SCK) phase and polarity with respect to the serial data, and these are determined by the clock phase (UCPHA) control bit and the inverted I/O pin (INVEN) settings. The data transfer timing diagrams are shown in Figure 23-4 on page 284. Data bits are shifted out and latched in on opposite edges of the XCK signal, ensuring sufficient time for data signals to stabilize. The UCPHA and INVEN settings are summarized in Table 23-2 on page 284. Changing the setting of any of these bits during transmission will corrupt both the receiver and transmitter.

Table 23-2. INVEN and UCPHA functionality.

SPI Mode	INVEN	UCPHA	Leading edge	Trailing edge
0	0	0	Rising, sample	Falling, setup
1	0	1	Rising, setup	Falling, sample
2	1	0	Falling, sample	Rising, setup
3	1	1	Falling, setup	Rising, sample

The leading edge is the first clock edge of a clock cycle. The trailing edge is the last clock edge of a clock cycle.

Figure 23-4. UCPHA and INVEN data transfer timing diagrams.



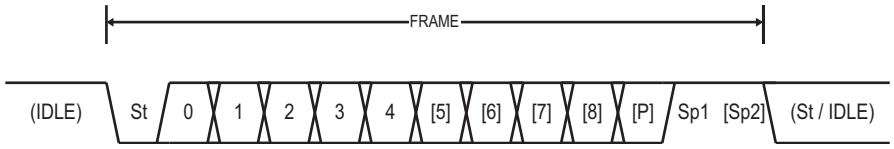
23.4 Frame Formats

Data transfer is frame based, where a serial frame consists of one character of data bits with synchronization bits (start and stop bits) and an optional parity bit for error checking. Note that this does not apply to master SPI operation (See “SPI Frame Formats” on page 285). The USART accepts all combinations of the following as valid frame formats:

- 1 start bit
- 5, 6, 7, 8, or 9 data bits
- no, even, or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit, followed by all the data bits (least-significant bit first and most-significant bit last). If enabled, the parity bit is inserted after the data bits, before the first stop bit. One frame can be directly followed by a start bit and a new frame, or the communication line can return to the idle (high) state. Figure 23-5 on page 285 illustrates the possible combinations of frame formats. Bits inside brackets are optional.

Figure 23-5. Frame formats.



St	:	Start bit, always low.
(n)	:	Data bits (0 to 8).
P	:	Parity bit, may be odd or even.
Sp	:	Stop bit, always high.
IDLE	:	No transfers on the communication line (RxD or TxD). The IDLE state is always high.

23.4.1 Parity Bit Calculation

Even or odd parity can be selected for error checking. If even parity is selected, the parity bit is set to one if the number of logical one data bits is odd (making the total number of ones even). If odd parity is selected, the parity bit is set to one if the number of logical one data bits is even (making the total number of ones odd).

23.4.2 SPI Frame Formats

The serial frame in SPI mode is defined to be one character of eight data bits. The USART in master SPI mode has two selectable frame formats:

- 8-bit data, msb first
- 8-bit data, lsb first

After a complete, 8-bit frame is transmitted, a new frame can directly follow it, or the communication line can return to the idle (high) state.

23.5 USART Initialization

USART initialization should use the following sequence:

1. Set the TxD pin value high, and optionally set the XCK pin low.
2. Set the TxD and optionally the XCK pin as output.
3. Set the baud rate and frame format.
4. Set the mode of operation (enables XCK pin output in synchronous mode).
5. Enable the transmitter or the receiver, depending on the usage.

For interrupt-driven USART operation, global interrupts should be disabled during the initialization.

Before doing a re-initialization with a changed baud rate or frame format, be sure that there are no ongoing transmissions while the registers are changed.

23.6 Data Transmission - The USART Transmitter

When the transmitter has been enabled, the normal port operation of the TxD pin is overridden by the USART and given the function as the transmitter's serial output. The direction of the pin must be set as output using the direction register for the corresponding port. For details on port pin control and output configuration, refer to “I/O Ports” on page 139.

23.6.1 Sending Frames

A data transmission is initiated by loading the transmit buffer (DATA) with the data to be sent. The data in the transmit buffer are moved to the shift register when the shift register is empty and ready to send a new frame. The shift register is loaded if it is in idle state (no ongoing transmission) or immediately after the last stop bit of the previous frame is transmitted. When the shift register is loaded with data, it will transfer one complete frame.

The transmit complete interrupt flag (TXCIF) is set and the optional interrupt is generated when the entire frame in the shift register has been shifted out and there are no new data present in the transmit buffer.

The transmit data register (DATA) can only be written when the data register empty flag (DREIF) is set, indicating that the register is empty and ready for new data.

When using frames with fewer than eight bits, the most-significant bits written to DATA are ignored. If 9-bit characters are used, the ninth bit must be written to the TXB8 bit before the low byte of the character is written to DATA.

23.6.2 Disabling the Transmitter

A disabling of the transmitter will not become effective until ongoing and pending transmissions are completed; i.e., when the transmit shift register and transmit buffer register do not contain data to be transmitted. When the transmitter is disabled, it will no longer override the TxDn pin, and the pin direction is set as input automatically by hardware, even if it was configured as output by the user.

23.7 Data Reception - The USART Receiver

When the receiver is enabled, the RxD pin functions as the receiver's serial input. The direction of the pin must be set as input, which is the default pin setting.

23.7.1 Receiving Frames

The receiver starts data reception when it detects a valid start bit. Each bit that follows the start bit will be sampled at the baud rate or XCK clock and shifted into the receive shift register until the first stop bit of a frame is received. A 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 receive buffer. The receive complete interrupt flag (RXCIF) is set, and the optional interrupt is generated.

The receiver buffer can be read by reading the data register (DATA) location. DATA should not be read unless the receive complete interrupt flag is set. When using frames with fewer than eight bits, the unused most-significant bits are read as zero. If 9-bit characters are used, the ninth bit must be read from the RXB8 bit before the low byte of the character is read from DATA.

23.7.2 Receiver Error Flags

The USART receiver has three error flags. The frame error (FERR), buffer overflow (BUFOVF) and parity error (PERR) flags are accessible from the status register. The error flags are located in the receive FIFO buffer together with their corresponding frame. Due to the buffering of the error flags, the status register must be read before the receive buffer (DATA), since reading the DATA location changes the FIFO buffer.

23.7.3 Parity Checker

When 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 flag is set.

23.7.4 Disabling the Receiver

A disabling of the receiver will be immediate. The receiver buffer will be flushed, and data from ongoing receptions will be lost.

23.7.5 Flushing the Receive Buffer

If the receive buffer has to be flushed during normal operation, read the DATA location until the receive complete interrupt flag is cleared.

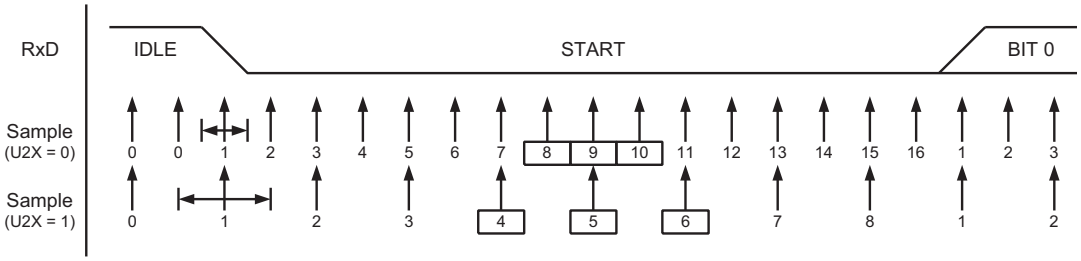
23.8 Asynchronous Data Reception

The USART includes a clock recovery and a data recovery unit for handling asynchronous data reception. The clock recovery unit is used for synchronizing the incoming asynchronous serial frames at the RxD pin to the internally generated baud rate clock. It samples and low-pass filters each incoming bit, thereby improving the noise immunity of the receiver. The asynchronous reception operational range depends on the accuracy of the internal baud rate clock, the rate of the incoming frames, and the frame size in number of bits.

23.8.1 Asynchronous Clock Recovery

The clock recovery unit synchronizes the internal clock to the incoming serial frames. Figure 23-6 on page 287 illustrates the sampling process for the start bit of an incoming frame. The sample rate is 16 times the baud rate for normal mode, and eight times the baud rate for double speed mode. The horizontal arrows illustrate the synchronization variation due to the sampling process. Note the larger time variation when using the double speed mode of operation. Samples denoted as zero are samples done when the RxD line is idle; i.e., when there is no communication activity.

Figure 23-6. Start bit sampling.

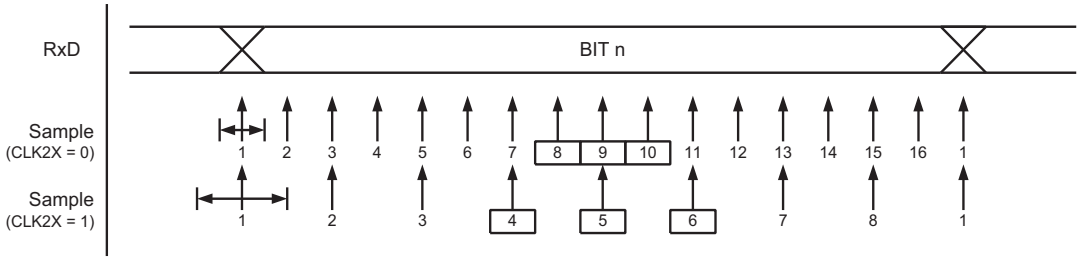


When the clock recovery logic detects a high (idle) to low (start) transition on the RxD line, the start bit detection sequence is initiated. Sample 1 denotes the first zero-sample, as shown in the figure. The clock recovery logic then uses samples 8, 9, and 10 for normal mode and samples 4, 5, and 6 for double speed mode to decide if a valid start bit is received. If two or three samples have a low level, the start bit is accepted. The clock recovery unit is synchronized, and the data recovery can begin. If two or three samples have a high level, the start bit is rejected as a noise spike, and the receiver looks for the next high-to-low transition. The process is repeated for each start bit.

23.8.2 Asynchronous Data Recovery

The data recovery unit uses sixteen samples in normal mode and eight samples in double speed mode for each bit. Figure 23-7 on page 287 shows the sampling process of data and parity bits.

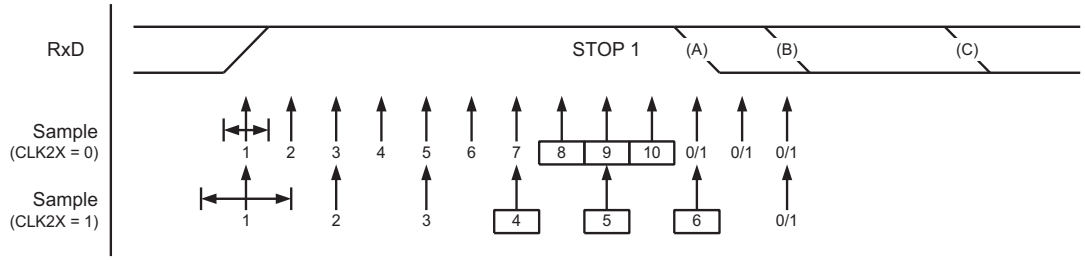
Figure 23-7. Sampling of data and parity bits.



As for start bit detection, an identical majority voting technique is used on the three center samples for deciding of the logic level of the received bit. The process is repeated for each bit until a complete frame is received. It includes the first stop bit, but excludes additional ones. If the sampled stop bit is a 0 value, the frame error (FERR) flag will be set.

Figure 23-8 on page 288 shows the sampling of the stop bit in relation to the earliest possible beginning of the next frame's start bit.

Figure 23-8. Stop bit and next start bit sampling.



A new high-to-low transition indicating the start bit of a new frame can come right after the last of the bits used for majority voting. For normal speed mode, the first low level sample can be at the point marked (A) in Stop Bit Sampling and Next Start Bit Sampling. For double speed mode, the first low level must be delayed to point (B). Point (C) marks a stop bit of full length at nominal baud rate. The early start bit detection influences the operational range of the receiver.

23.8.3 Asynchronous Operational Range

The operational range of the receiver is dependent on the mismatch between the received bit rate and the internally generated baud rate. If an external transmitter is sending using bit rates that are too fast or too slow, or if the internally generated baud rate of the receiver does not match the external source's base frequency, the receiver will not be able to synchronize the frames to the start bit.

The following equations can be used to calculate the ratio of the incoming data rate and internal receiver baud rate.

$$R_{slow} = \frac{(D+1)S}{S-1+D\cdot S+S_F} \qquad R_{fast} = \frac{(D+2)S}{(D+1)S+S_M}$$

D	:	Sum of character size and parity size (D = 5 to 10 bits).
S	:	Samples per bit. S = 16 for normal speed mode and S = 8 for double speed mode.
S_F	:	First sample number used for majority voting. S_F = 8 for normal speed mode and S_F = 4 for double speed mode.
S_M	:	Middle sample number used for majority voting. S_M = 9 for normal speed mode and S_M = 5 for double speed mode.
R_{slow}	:	The ratio of the slowest incoming data rate that can be accepted in relation to the receiver baud rate.
R_{fast}	:	The ratio of the fastest incoming data rate that can be accepted in relation to the receiver baud rate.

Table 23-3 and Table 23-4 on page 289 list the maximum receiver baud rate error that can be tolerated. Normal speed mode has higher tolerance of baud rate variations

Table 23-3. Recommended maximum receiver baud rate error for normal speed mode.

D #(Data + Parity Bit)	R_{slow} [%]	R_{fast} [%]	Max total error [%]	Recommended max receiver error [%]
5	93.20	106.67	+6.67/-6.80	± 3.0
6	94.12	105.79	+5.79/-5.88	± 2.5
7	94.81	105.11	+5.11/-5.19	± 2.0
8	95.36	104.58	+4.58/-4.54	± 2.0
9	95.81	104.14	+4.14/-4.19	± 1.5
10	96.17	103.78	+3.78/-3.83	± 1.5

Table 23-4. Recommended maximum receiver baud rate error for double speed mode.

D #(Data + Parity Bit)	R_{slow} [%]	R_{fast} [%]	Max Total Error [%]	Recommended Max Receiver Error [%]
5	94.12	105.66	+5.66/-5.88	± 2.5
6	94.92	104.92	+4.92/-5.08	± 2.0
7	95.52	104.35	+4.35/-4.48	± 1.5
8	96.00	103.90	+3.90/-4.00	± 1.5
9	96.39	103.53	+3.53/-3.61	± 1.5
10	96.70	103.23	+3.23/-3.30	± 1.0

The recommendations for the maximum receiver baud rate error assume that the receiver and transmitter equally divide the maximum total error.

23.9 Fractional Baud Rate Generation

Fractional baud rate generation is possible for asynchronous operation due to the relatively high number of clock cycles for each frame. Each bit is sampled sixteen times, but only the three middle samples are of importance. The total number of samples for one frame is also relatively high. Given a 1-start, 8-data, no-parity, and 1-stop-bit frame format, and assuming that normal speed mode is used, the total number of samples for a frame is (1+8+1)×16 or 160. As stated earlier, the UART can tolerate some variation in clock cycles for each sample. The critical factor is the time from the falling edge of the start bit (i.e., the clock synchronization) until the last bit's (i.e., the first stop bit's) value is recovered.

Standard baud rate generators have the unwanted property of having large frequency steps between high baud rate settings. The worst case is found between the BSEL values 0x000 and 0x001. Going from a BSEL value of 0x000, which has a 10-bit frame of 160 clock cycles, to a BSEL value of 0x001, with 320 clock cycles, gives a 50% change in frequency. Ideally, the step size should be small even between the fastest baud rates. This is where the advantage of the fractional baud rate generator emerges.

In principle, the fractional baud rate generator works by doing uneven counting and then distributing the error evenly over the entire frame. A typical count sequence for an ordinary baud rate generator is:

2, 1, 0, 2, 1, 0, 2, 1, 0, 2, …

which has an even period time. A baud rate clock ticks each time the counter reaches zero, and a sample of the signal received on RxD is taken for every 16th baud rate clock tick.

For the fractional baud rate generator, the count sequence can have an uneven period:

2, 1, 0, 2, 1-1, 0, 2, 1, 0, 2, 1-1, 0, ...

In this example, an extra cycle is added to every second baud clock. This gives a baud rate clock tick jitter, but the average period has been increased by a fraction of 0.5 clock cycles.

Figure 23-9 on page 290 shows an example of how BSEL and BSCALE can be used to achieve baud rates in between what is possible by just changing BSEL.

The impact of fractional baud rate generation is that the step size between baud rate settings has been reduced. Given a scale factor of -1, the worst-case step then becomes from 160 to 240 clock cycles per 10-bit frame, compared to the previous step of from 160 to 320. A higher negative scale factor gives even finer granularity. There is a limit however, to how high the scale factor can be. The value $2^{|BSCALE|}$ must be at most half the minimum number of clock cycles of a frame. For instance, for 10-bit frames, the minimum number of clock cycles is 160. This means that the highest applicable scale factor is -6 ($2^{|-6|}$ = 64 < (160/2) = 80).

For higher BSEL settings, the scale factor can be increased.

Table 23-5 on page 290 shows BSEL and BSCALE settings when using the internal oscillators to generate the most commonly used baud rates for asynchronous operation and how reducing the BSCALE can be used to reduce the baud rate error even further.

Figure 23-9. Fractional baud rate example.

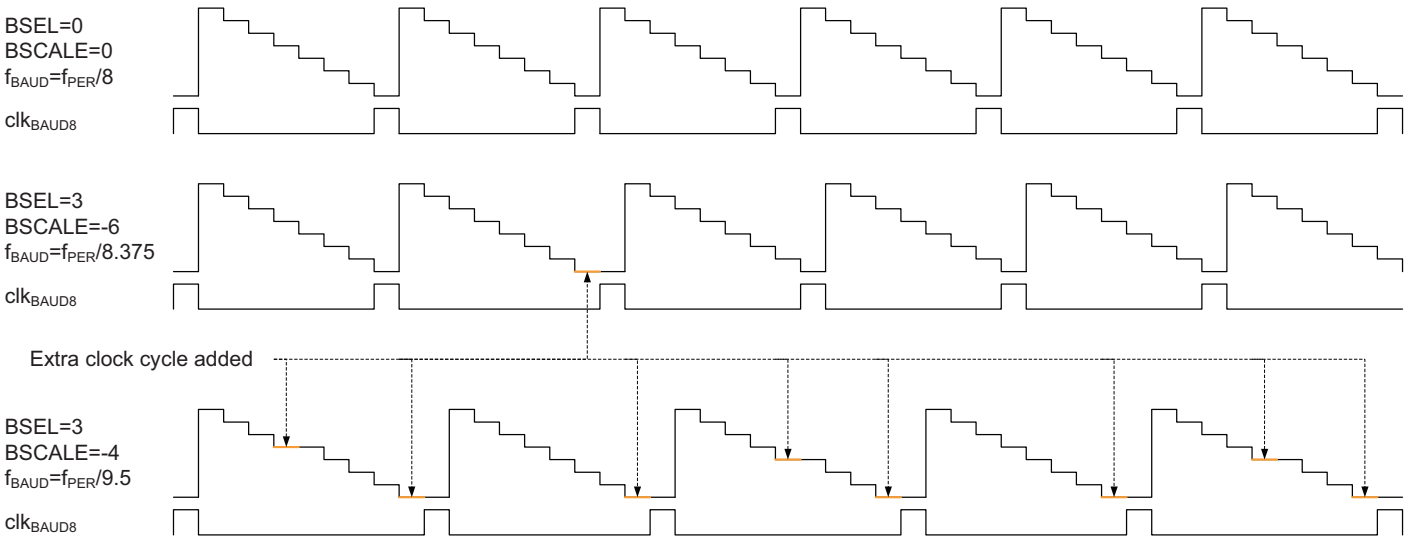


Table 23-5. USART baud rate.

Baud rate (bps)	f_{OSC} = 32.0000MHz					
	CLK2X = 0			CLK2X = 1		
	BSEL	BSCALE	Error [%]	BSEL	BSCALE	Error [%]
2400	12	6	0.2	12	7	0.2
4800	12	5	0.2	12	6	0.2
9600	12	4	0.2	12	5	0.2
14.4k	34	2	0.8	34	3	0.8
	138	0	-0.1	138	1	-0.1
19.2k	12	3	0.2	12	4	0.2

Baud	f_{OSC} = 32.0000MHz					
28.8k	34	1	-0.8	34	2	-0.8
	137	-1	-0.1	138	0	-0.1
38.4k	12	2	0.2	12	3	0.2
57.6k	34	0	-0.8	34	1	-0.8
	135	-2	-0.1	137	-1	-0.1
76.8k	12	1	0.2	12	2	0.2
115.2k	33	-1	-0.8	34	0	-0.8
	131	-3	-0.1	135	-2	-0.1
230.4k	31	-2	-0.8	33	-1	-0.8
	123	-4	-0.1	131	-3	-0.1
460.8k	27	-3	-0.8	31	-2	-0.8
	107	-5	-0.1	123	-4	-0.1
921.6k	19	-4	-0.8	27	-3	-0.8
	75	-6	-0.1	107	-5	-0.1
1.382M	7	-4	0.6	15	-3	0.6
	57	-7	0.1	121	-6	0.1
1.843M	3	-5	-0.8	19	-4	-0.8
	11	-7	-0.1	75	-6	-0.1
2.00M	0	0	0.0	1	0	0.0
2.304M	–	–	–	3	-2	-0.8
				47	-6	-0.1
2.5M	–	–	–	19	-4	0.4
				77	-7	-0.1
3.0M	–	–	–	11	-5	-0.8
				43	-7	-0.2
4.0M	–	–	–	0	0	0.0
Max	2.0Mbps			4.0Mbps		

23.10 USART in Master SPI Mode

Using the USART in master SPI mode requires the transmitter to be enabled. The receiver can optionally be enabled to serve as the serial input. The XCK pin will be used as the transfer clock.

As for the USART, a data transfer is initiated by writing to the DATA register. This is the case for both sending and receiving data, since the transmitter controls the transfer clock. The data written to DATA are moved from the transmit buffer to the shift register when the shift register is ready to send a new frame.

The transmitter and receiver interrupt flags and corresponding USART interrupts used in master SPI mode are identical in function to their use in normal USART operation. The receiver error status flags are not in use and are always read as zero.

Disabling of the USART transmitter or receiver in master SPI mode is identical to their disabling in normal USART operation.

23.11 USART SPI vs. SPI

The USART in master SPI mode is fully compatible with the standalone SPI module in that:

- Timing diagrams are the same
- UCPHA bit functionality is identical to that of the SPI CPHA bit
- UDORD bit functionality is identical to that of the SPI DORD bit

When the USART is set in master SPI mode, configuration and use are in some cases different from those of the standalone SPI module. In addition, the following differences exist:

- The USART transmitter in master SPI mode includes buffering, but the SPI module has no transmit buffer
- The USART receiver in master SPI mode includes an additional buffer level
- The USART in master SPI mode does not include the SPI write collision feature
- The USART in master SPI mode does not include the SPI double speed mode feature, but this can be achieved by configuring the baud rate generator accordingly
- Interrupt timing is not compatible
- Pin control differs due to the master-only operation of the USART in SPI master mode

A comparison of the USART in master SPI mode and the SPI pins is shown Table 23-6.

Table 23-6. Comparison of USART in master SPI mode and SPI pins.

USART	SPI	Comment
TxD	MOSI	Master out only
RxD	MISO	Master in only
XCK	SCK	Functionally identical
N/A	$\overline{SS}$	Not supported by USART in master SPI mode

23.12 Multiprocessor Communication Mode

The multiprocessor communication mode effectively reduces the number of incoming frames that have to be handled by the receiver in a system with multiple microcontrollers communicating via the same serial bus. In this mode, a dedicated bit in the frames is used to indicate whether the frame is an address or data frame type.

If the receiver is set up to receive frames that contain five to eight data bits, the first stop bit is used to indicate the frame type. If the receiver is set up for frames with nine data bits, the ninth bit is used. When the frame type bit is one, the frame contains an address. When the frame type bit is zero, the frame is a data frame. If 5-bit to 8-bit character frames are used, the transmitter must be set to use two stop bits, since the first stop bit is used for indicating the frame type.

If a particular slave MCU has been addressed, it will receive the following data frames as usual, while the other slave MCUs will ignore the frames until another address frame is received.

23.12.1 Using Multiprocessor Communication Mode

The following procedure should be used to exchange data in multiprocessor communication mode (MPCM):

1. All slave MCUs are in multiprocessor communication mode.
2. The master MCU sends an address frame, and all slaves receive and read this frame.
3. Each slave MCU determines if it has been selected.

4. The addressed MCU will disable MPCM and receive all data frames. The other slave MCUs will ignore the data frames.
5. When the addressed MCU has received the last data frame, it must enable MPCM again and wait for a new address frame from the master.

The process then repeats from step 2.

Using any of the 5-bit to 8-bit character frame formats is impractical, as the receiver must change between using n and n+1 character frame formats. This makes full-duplex operation difficult, since the transmitter and receiver must use the same character size setting.

23.13 IRCOM Mode of Operation

IRCOM mode can be enabled to use the IRCOM module with the USART. This enables IrDA 1.4 compliant modulation and demodulation for baud rates up to 115.2kbps. When IRCOM mode is enabled, double speed mode cannot be used for the USART.

For devices with more than one USART, IRCOM mode can be enabled for only one USART at a time. For details, refer to “IRCOM – IR Communication Module” on page 301.

23.14 DMA Support

DMA support is available on UART, USRT, and master SPI mode peripherals. For details on different USART DMA transfer triggers, refer to “Transfer Triggers” on page 55.

23.15 Register Description

23.15.1 DATA – Data register

Bit	7	6	5	4	3	2	1	0
+0x00	RXB[[7:0] / TXB[[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

The USART transmit data buffer register (TXB) and USART receive data buffer register (RXB) share the same I/O address and is referred to as USART data register (DATA). The TXB register is the destination for data written to the DATA register location. Reading the DATA register location returns the contents of the RXB register.

For 5-bit, 6-bit, or 7-bit characters, the upper unused bits will be ignored by the transmitter and set to zero by the receiver.

The transmit buffer can be written only when DREIF in the STATUS register is set. Data written to the DATA register when DREIF is not set will be ignored by the USART transmitter. When data are written to the transmit buffer and the transmitter is enabled, the transmitter will load the data into the transmit shift register when the shift register is empty. The data are then transmitted on the TxD pin.

The receive buffer consists of a two-level FIFO. Always read STATUS before DATA in order to get the correct status of the receive buffer.

23.15.2 STATUS – Status register

Bit	7	6	5	4	3	2	1	0
+0x01	RXCIF	TXCIF	DREIF	FERR	BUFOVF	PERR	–	RXB8
Read/Write	R	R/W	R	R	R	R	R	R/W
Initial Value	0	0	1	0	0	0	0	0

- **Bit 7 – RXCIF: Receive Complete Interrupt Flag**
  This flag is set when there are unread data in the receive buffer and cleared when the receive buffer is empty (i.e., does not contain any unread data). When the receiver is disabled, the receive buffer will be flushed, and consequently RXCIF will become zero.
  When interrupt-driven data reception is used, the receive complete interrupt routine must read the received data from DATA in order to clear RXCIF. If not, a new interrupt will occur directly after the return from the current interrupt. This flag can also be cleared by writing a one to its bit location.

- **Bit 6 – TXCIF: Transmit Complete Interrupt Flag**
  This flag is set when the entire frame in the transmit shift register has been shifted out and there are no new data in the transmit buffer (DATA). TXCIF is automatically cleared when the transmit complete interrupt vector is executed. The flag can also be cleared by writing a one to its bit location.

- **Bit 5 – DREIF: Data Register Empty Flag**
  This flag indicates whether the transmit buffer (DATA) is ready to receive new data. The flag is one when the transmit buffer is empty and zero when the transmit buffer contains data to be transmitted that has not yet been moved into the shift register. DREIF is set after a reset to indicate that the transmitter is ready. Always write this bit to zero when writing the STATUS register.
  DREIF is cleared by writing DATA. When interrupt-driven data transmission is used, the data register empty interrupt routine must either write new data to DATA in order to clear DREIF or disable the data register empty interrupt. If not, a new interrupt will occur directly after the return from the current interrupt.

- **Bit 4 – FERR: Frame Error**
  The FERR flag indicates the state of the first stop bit of the next readable frame stored in the receive buffer. The bit is set if the received character had a frame error, i.e., the first stop bit was zero, and cleared when the stop bit of the received data is one. This bit is valid until the receive buffer (DATA) is read. FERR is not affected by setting the number of stop bits used, as it always uses only the first stop bit. Always write this bit location to zero when writing the STATUS register.

  This flag is not used in master SPI mode operation.

- **Bit 3 – BUFOVF: Buffer Overflow**
  This flag indicates data loss due to a receiver buffer full condition. This flag is set if a buffer overflow condition is detected. A buffer overflow occurs when the receive buffer is full (two characters) with a new character waiting in the receive shift register and a new start bit is detected. This flag is valid until the receive buffer (DATA) is read. Always write this bit location to zero when writing the STATUS register.

  This flag is not used in master SPI mode operation.

- **Bit 2 – PERR: Parity Error**
  If parity checking is enabled and the next character in the receive buffer has a parity error, this flag is set. If parity check is not enabled, this flag will always be read as zero. This bit is valid until the receive buffer (DATA) is read. Always write this bit location to zero when writing the STATUS register. For details on parity calculation, refer to "Parity Bit Calculation" on page 285.

  This flag is not used in master SPI mode operation.

- **Bit 1 – Reserved**
  This bit is unused and reserved for future use. For compatibility with future devices, always write this bit to zero when this register is written.

- **Bit 0 – RXB8: Receive Bit 8**
  RXB8 is the ninth data bit of the received character when operating with serial frames with nine data bits. When used, this bit must be read before reading the low bits from DATA.

  This bit is unused in master SPI mode operation.

23.15.3 CTRLA – Control register A

Bit	7	6	5	4	3	2	1	0
+0x03	–	–	RXCINTLVL[1:0]		TXCINTLVL[1:0]		DREINTLVL[1:0]	
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:6 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written.

- **Bit 5:4 – RXCINTLVL[1:0]: Receive Complete Interrupt Level**
  These bits enable the receive complete interrupt and select the interrupt level, as described in "Interrupts and Programmable Multilevel Interrupt Controller" on page 131. The enabled interrupt will be triggered when the RXCIF flag in the STATUS register is set.

- **Bit 3:2 – TXCINTLVL[1:0]: Transmit Complete Interrupt Level**
  These bits enable the transmit complete interrupt and select the interrupt level, as described in "Interrupts and Programmable Multilevel Interrupt Controller" on page 131. The enabled interrupt will be triggered when the TXCIF flag in the STATUS register is set.

- **Bit 1:0 – DREINTLVL[1:0]: Data Register Empty Interrupt Level**
  These bits enable the data register empty interrupt and select the interrupt level, as described in “Interrupts and Programmable Multilevel Interrupt Controller” on page 131. The enabled interrupt will be triggered when the DREIF flag in the STATUS register is set.

23.15.4 CTRLB – Control register B

Bit	7	6	5	4	3	2	1	0
+0x04	–	–	–	RXEN	TXEN	CLK2X	MPCM	TXB8
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:5 – Reserved**
  These bits are unused and reserved for future use. For compatibility with future devices, always write these bits to zero when this register is written.

- **Bit 4 – RXEN: Receiver Enable**
  Setting this bit enables the USART receiver. The receiver will override normal port operation for the RxD pin, when enabled. Disabling the receiver will flush the receive buffer, invalidating the FERR, BUFOVF, and PERR flags.

- **Bit 3 – TXEN: Transmitter Enable**
  Setting this bit enables the USART transmitter. The transmitter will override normal port operation for the TxD pin, when enabled. Disabling the transmitter (writing TXEN to zero) will not become effective until ongoing and pending transmissions are completed; i.e., when the transmit shift register and transmit buffer register do not contain data to be transmitted. When disabled, the transmitter will no longer override the TxD port.

- **Bit 2 – CLK2X: Double Transmission Speed**
  Setting this bit will reduce the divisor of the baud rate divider from16 to 8, effectively doubling the transfer rate for asynchronous communication modes. For synchronous operation, this bit has no effect and should always be written to zero. This bit must be zero when the USART communication mode is configured to IRCOM.
  This bit is unused in master SPI mode operation.

- **Bit 1 – MPCM: Multiprocessor Communication Mode**
  This bit enables the multiprocessor communication mode. When the MPCM bit is written to one, the USART receiver ignores all the incoming frames that do not contain address information. The transmitter is unaffected by the MPCM setting. For more detailed information, see “Multiprocessor Communication Mode” on page 292.
  This bit is unused in master SPI mode operation.

- **Bit 0 – TXB8: Transmit Bit 8**
  TXB8 is the ninth data bit in the character to be transmitted when operating with serial frames with nine data bits. When used, this bit must be written before writing the low bits to DATA.
  This bit is unused in master SPI mode operation.

23.15.5 CTRLC – Control register C

Bit	7	6	5	4	3	2	1	0
+0x05	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]		
+0x05[1]	CMODE[1:0]		–	–	–	UDORD	UCPHA	–
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	1	1
	0	0	0	0	0	1	1	0

Note: 1. Master SPI mode

- **Bits 7:6 – CMODE[1:0]: Communication Mode**
  These bits select the mode of operation of the USART as shown in Table 23-7.

Table 23-7. CMODE bit settings.

CMODE[1:0]	Group configuration	Mode
00	ASYNCHRONOUS	Asynchronous USART
01	SYNCHRONOUS	Synchronous USART
10	IRCOM	IRCOM(1)
11	MSPI	Master SPI(2)

Notes: 1. See "IRCOM – IR Communication Module" on page 301 for full description on using IRCOM mode.
2. See "USART in Master SPI Mode" on page 291 for full description of the master SPI operation.

- **Bits 5:4 – PMODE[1:0]: Parity Mode**
  These bits enable and set the type of parity generation according to Table 23-8 on page 297. When enabled, 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 compare it to the PMODE setting, and if a mismatch is detected, the PERR flag in STATUS will be set.
  These bits are unused in master SPI mode operation.

Table 23-8. PMODE bit settings.

PMODE[1:0]	Group configuration	Parity Mode
00	DISABLED	Disabled
01	–	Reserved
10	EVEN	Enabled, even parity
11	ODD	Enabled, odd parity

- **Bit 3 – SBMODE: Stop Bit Mode**
  This bit selects the number of stop bits to be inserted by the transmitter according to Table 23-9 on page 297. The receiver ignores this setting.
  This bit is unused in master SPI mode operation.

Table 23-9. SBMODE bit settings.

SBMODE	Stop Bit(s)
0	1
1	2

- **Bit 2:0 – CHSIZE[2:0]: Character Size**
  The CHSIZE[2:0] bits set the number of data bits in a frame according to Table 23-10 on page 298. The receiver and transmitter use the same setting.

Table 23-10. CHSIZE bit settings.

CHSIZE[2:0]	Group configuration	Character size
000	5BIT	5-bit
001	6BIT	6-bit
010	7BIT	7-bit
011	8BIT	8-bit
100	–	Reserved
101	–	Reserved
110	–	Reserved
111	9BIT	9-bit

- **Bit 2 – UDORD: Data Order**
  This bit is only for master SPI mode, and this bit sets the frame format. When written to one, the lsb of the data word is transmitted first. When written to zero, the msb of the data word is transmitted first. The receiver and transmitter use the same setting. Changing the setting of UDORD will corrupt all ongoing communication for both receiver and transmitter.

- **Bit 1 – UCPHA: Clock Phase**
  This bit is only for master SPI mode, and the bit determine whether data are sampled on the leading (first) edge or tailing (last) edge of XCKn. Refer to the “Master SPI Mode Clock Generation” on page 284 for details.

23.15.6 BAUDCTRLA – Baud Rate register A

Bit	7	6	5	4	3	2	1	0
+0x06	BSEL[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:0 – BSEL[7:0]: Baud Rate bits**
  These are the lower 8 bits of the 12-bit BSEL value used for USART baud rate setting. BAUDCTRLB contains the four most-significant bits. Ongoing transmissions by the transmitter and receiver will be corrupted if the baud rate is changed. Writing BSEL will trigger an immediate update of the baud rate prescaler. See the equations in Table 23-1 on page 282.

23.15.7 BAUDCTRLB – Baud Rate register B

Bit	7	6	5	4	3	2	1	0
+0x07	BSCALE[3:0]				BSEL[11:8]			
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- **Bit 7:4 – BSCALE[3:0]: Baud Rate Scale factor**
  These bits select the baud rate generator scale factor. The scale factor is given in two's complement form from -7 (0b1001) to +7 (0b0111). The -8 (0b1000) setting is reserved. See the equations in Table 23-1 on page 282.

- **Bit 3:0 – BSEL[11:8]: Baud Rate bits**
  These are the upper 4 bits of the 12-bit value used for USART baud rate setting. BAUDCTRLA contains the eight least-significant bits. Ongoing transmissions by the transmitter and receiver will be corrupted if the baud rate is changed. Writing BAUDCTRLA will trigger an immediate update of the baud rate prescaler.

23.16 Register summary

23.16.1 Register description – USART

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
+0x00	DATA	DATA[7:0]								294
+0x01	STATUS	RXCIF	TXCIF	DREIF	FERR	BUFOVF	PERR	–	RXB8	294
+0x02	Reserved	–	–	–	–	–	–	–	–	
+0x03	CTRLA	–	–	RXCINTLVL[1:0]		TXCINTLVL[1:0]		DREINTLVL[1:0]		295
+0x04	CTRLB	–	–	–	RXEN	TXEN	CLK2X	MPCM	TXB8	296
+0x05	CTRLC	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]			296
+0x06	BAUDCTRLA	BSEL[7:0]								298
+0x07	BAUDCTRLB	BSCALE[3:0]				BSEL[11:8]				298

23.16.2 Register description – USART in SPI Master Mode

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
+0x00	DATA	DATA[7:0]								294
+0x01	STATUS	RXCIF	TXCIF	DREIF	–	–	–	–	–	294
+0x02	Reserved	–	–	–	–	–	–	–	–	
+0x03	CTRLA	–	–	RXCINTLVL[1:0]		TXCINTLVL[1:0]		DREINTLVL[1:0]		295
+0x04	CTRLB	–	–	–	RXEN	TXEN	–	–	–	296
+0x05	CTRLC	CMODE[1:0]		–	–	–	UDORD	UCPHA	–	296
+0x06	BAUDCTRLA	BSEL[7:0]								298
+0x07	BAUDCTRLB	BSCALE[3:0]				BSEL[11:8]				298

23.17 Interrupt vector summary

Table 23-11. USART interrupt vectors and their word offset address.

Offset	Source	Interrupt description
0x00	RXC_vect	USART receive complete interrupt vector
0x02	DRE_vect	USART data register empty interrupt vector
0x04	TXC_vect	USART transmit complete interrupt vector