Microprocessor Applications

XMEGA: USART

UF UNIVERSITY *of* FLORIDA

Dr. Eric Schwartz
Christopher Crary
Wesley Piard

1

USART on the ATxmega128A1U

❖ The XMEGA has a total of eight Universal Synchronous and Asynchronous Receiver/Transmitter (**USART**) modules.

❖ Each module can be configured to perform either synchronous or asynchronous serial communication.

❖ In this presentation, the synchronous portion will not be covered. More details about it can be found in the XMEGA AU (doc8331) manual.

2

XMEGA UART Specifications

Ports C, D, E, and F all have two USART modules available.

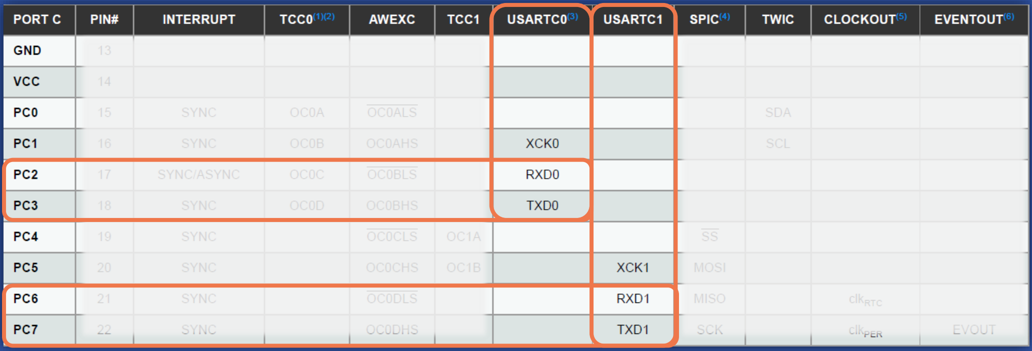
- ❖ A USART module on the XMEGA consists of two pins:
  - ➢ Rx (receive)
  - ➢ Tx (transmit)
- ❖ The naming convention for labeling each individual module is USART*p*#, where *p* and # are the following:
  - ➢ *p* : {C, D, E, F}
  - ➢ # : {0, 1}
  - ➢ For example, Port C has two UART modules: USARTC0 and USARTC1.

3

Alternate Pin Function Table

- ❖ Section 33.2 of doc8385 has a table of alternate pin functions for each port.
- ❖ The table for Port C is shown here. See the two columns labeled USARTC0 and USARTC1.

doc8385
Table 33-3



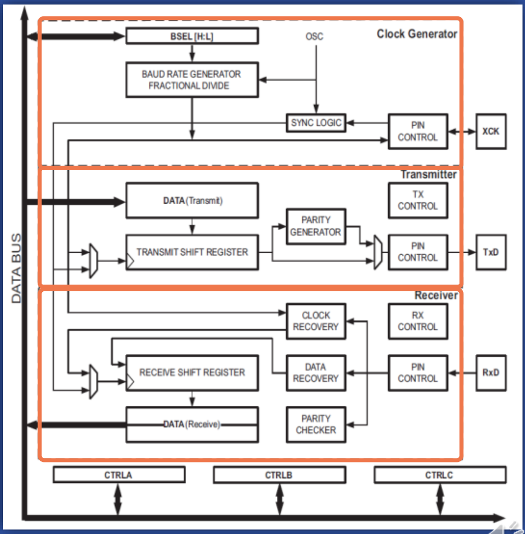
PORT C	PIN#	INTERRUPT	TCC0(1)(2)	AWEXC	TCC1	USARTC0(3)	USARTC1	SPIC(4)	TWIC	CLOCKOUT(5)	EVENTOUT(6)
GND	13										
VCC	14										
PC0	15	SYNC	OC0A	$\overline{\text{OC0ALS}}$					SDA		
PC1	16	SYNC	OC0B	OC0AHS		XCK0			SCL		
PC2	17	SYNC/ASYNC	OC0C	$\overline{\text{OC0BLS}}$		RXD0					
PC3	18	SYNC	OC0D	OC0BHS		TXD0					
PC4	19	SYNC		$\overline{\text{OC0CLS}}$	OC1A			$\overline{\text{SS}}$			
PC5	20	SYNC		OC0CHS	OC1B		XCK1	MOSI			
PC6	21	SYNC		$\overline{\text{OC0DLS}}$			RXD1	MISO		clk_{RTC}	
PC7	22	SYNC		OC0DHS			TXD1	SCK		clk_{PER}	EVOUT

4

Block Diagram

- Here is the block diagram for each UART module. There are three major sections.
- The **Clock Generator** section generates the clock signals required for the *Transmitter* and *Receiver* sections to function properly.
- The **Transmitter** section shifts data out via the TxD (transmit) pin at the baud rate.
- The **Receiver** section samples data via the RxD (receive) pin.

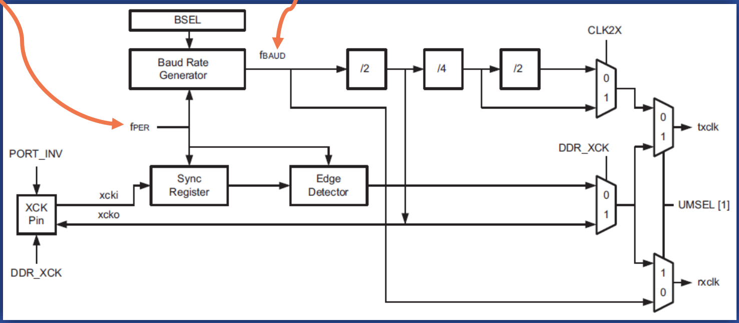
doc8331
Figure 23-1



UART Clock Generation

- The *Clock Generation* section contains a fractional baud rate generator that uses the XMEGA's peripheral clock frequency, f_{PER}, to generate the baud rate, f_{BAUD}.
- f_{PER} is the frequency of the peripheral clock, clk_{PER}, which is derived from the overall system clock, clk_{SYS}.

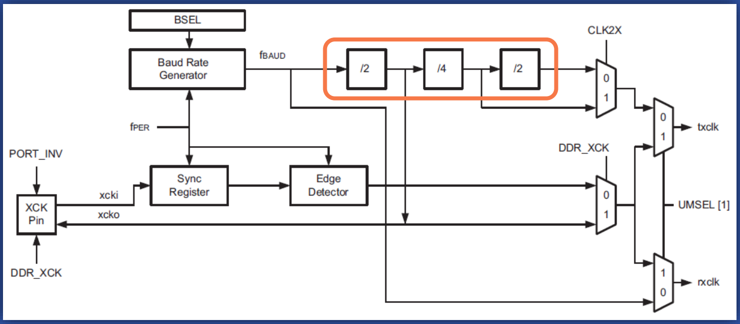
doc8331
Figure 23-2



UART Clock Generator

- The UART clock generator can generate a wide range of baud rates such that it is possible to interface with external UARTs.
- Note that the receiver's clock is 16 times faster than the transmitter's clock.
- More details about why this is necessary can be found in section 23.8 of the doc8331 manual.

doc8331
Figure 23-2



UART Clock Generator – Baud Rate Equations

- To configure a UART for a specific baud rate, the equations below are utilized.
- The three parameters that can be modified are f_{PER}, $BSEL$, and $BSCALE$.
- The equations that calculate f_{BAUD} (in the third column) are equivalent to the equations used to calculate $BSEL$ (in the fourth column). The only difference is which parameter is solved for.
- Note that the CLK2X bit can be set to allow for higher baud rates. See the rest of Table 23-1 in the doc8331 manual for more details.

doc8331
Table 23-1

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 $f_{BAUD} \leq \frac{f_{PER}}{16}$	$f_{BAUD} = \frac{f_{PER}}{16((2^{BSCALE} \cdot BSEL)+1)}$	$BSEL = \frac{1}{2^{BSCALE}}\left(\frac{f_{PER}}{16 f_{BAUD}} - 1\right)$

UART Clock Generator – Baud Rate Equations

- The baud rate generated, f_{BAUD}, is determined by the period setting, *BSEL*, an optional scale setting *BSCALE*, and the peripheral clock frequency f_{PER}.
- Generally, the first step when configuring UART is to determine the baud rate. The two UARTs that are communicating must have identical, or as close to identical as possible, baud rates.

doc8331
Table 23-1

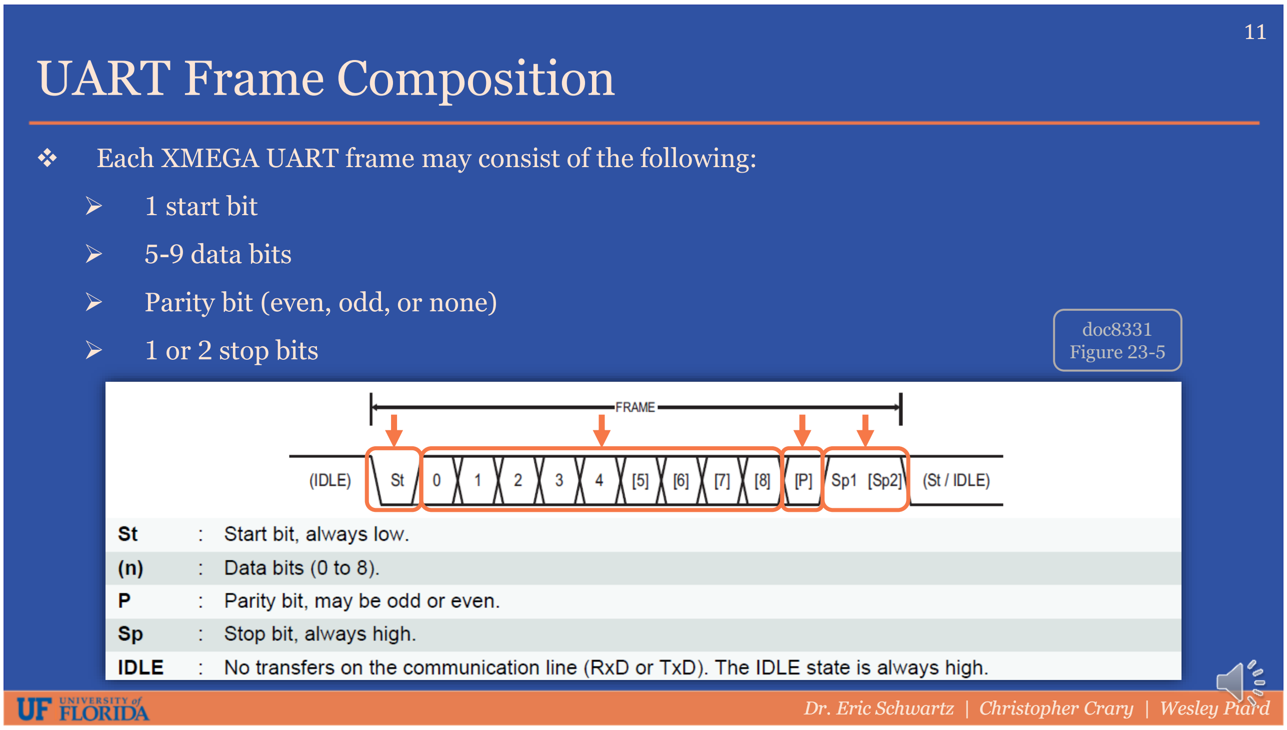
Operating mode	Conditions	Baud rate(1) calculation	BSEL value calculation
Asynchronous normal speed mode (CLK2X = 0)	BSCALE ≥ 0 $f_{BAUD} \le \frac{f_{PER}}{16}$	$f_{BAUD} = \frac{f_{PER}}{2^{BSCALE} \cdot 16(BSEL + 1)}$	$BSEL = \frac{f_{PER}}{2^{BSCALE} \cdot 16 f_{BAUD}} - 1$
	BSCALE < 0 $f_{BAUD} \le \frac{f_{PER}}{16}$	$f_{BAUD} = \frac{f_{PER}}{16((2^{BSCALE} \cdot BSEL) + 1)}$	$BSEL = \frac{1}{2^{BSCALE}}\left(\frac{f_{PER}}{16 f_{BAUD}} - 1\right)$

UART Clock Generator – Baud Rate Equations

- For a given f_{BAUD}, the following values must be chosen or calculated:
  - f_{PER} : 2 MHz by default for the XMEGA; derived from the system clock
  - *BSEL* : Period value between 0 and 4095
  - *BSCALE* : A value in the range [-7, +7] that is used to fine tune f_{BAUD}

doc8331
Table 23-1

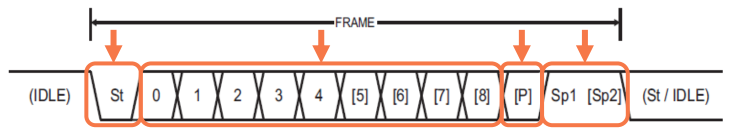
Operating mode	Conditions	Baud rate(1) calculation	BSEL value calculation
Asynchronous normal speed mode (CLK2X = 0)	BSCALE ≥ 0 $f_{BAUD} \le \frac{f_{PER}}{16}$	$f_{BAUD} = \frac{f_{PER}}{2^{BSCALE} \cdot 16(BSEL + 1)}$	$BSEL = \frac{f_{PER}}{2^{BSCALE} \cdot 16 f_{BAUD}} - 1$
	BSCALE < 0 $f_{BAUD} \le \frac{f_{PER}}{16}$	$f_{BAUD} = \frac{f_{PER}}{16((2^{BSCALE} \cdot BSEL) + 1)}$	$BSEL = \frac{1}{2^{BSCALE}}\left(\frac{f_{PER}}{16 f_{BAUD}} - 1\right)$



UART Frame Composition

- Each XMEGA UART frame may consist of the following:
  - 1 start bit
  - 5-9 data bits
  - Parity bit (even, odd, or none)
  - 1 or 2 stop bits

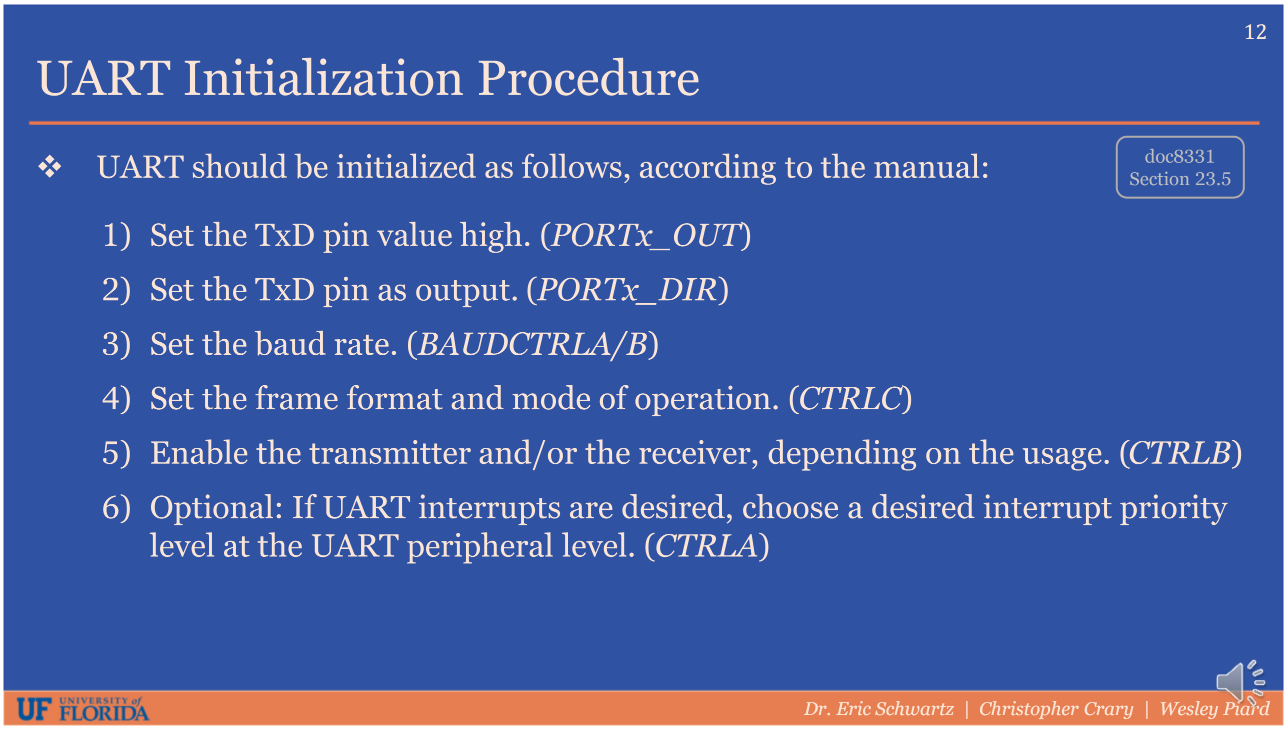
doc8331
Figure 23-5



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.

UF UNIVERSITY of FLORIDA — *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

11



UART Initialization Procedure

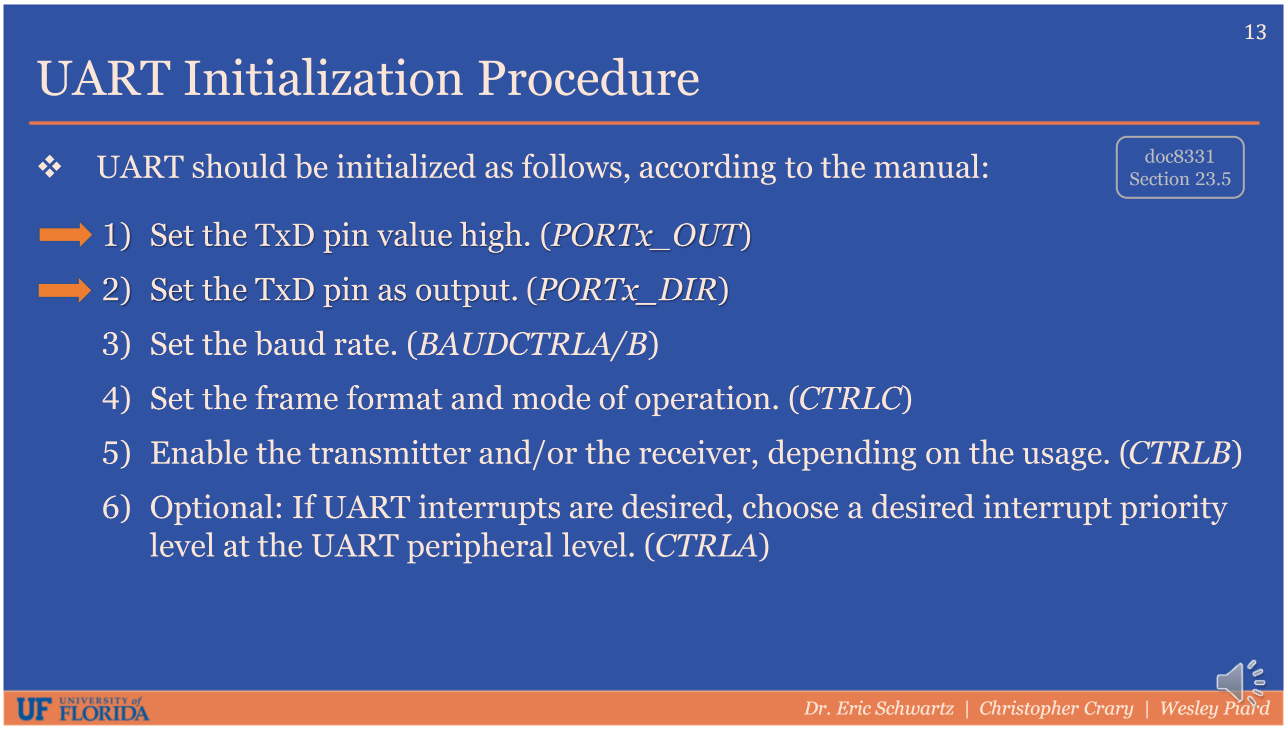
- UART should be initialized as follows, according to the manual:

doc8331
Section 23.5

1) Set the TxD pin value high. (*PORTx_OUT*)
2) Set the TxD pin as output. (*PORTx_DIR*)
3) Set the baud rate. (*BAUDCTRLA/B*)
4) Set the frame format and mode of operation. (*CTRLC*)
5) Enable the transmitter and/or the receiver, depending on the usage. (*CTRLB*)
6) Optional: If UART interrupts are desired, choose a desired interrupt priority level at the UART peripheral level. (*CTRLA*)

UF UNIVERSITY of FLORIDA — *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

12



UART Initialization Procedure

doc8331
Section 23.5

❖ UART should be initialized as follows, according to the manual:

➡ 1) Set the TxD pin value high. (*PORTx_OUT*)

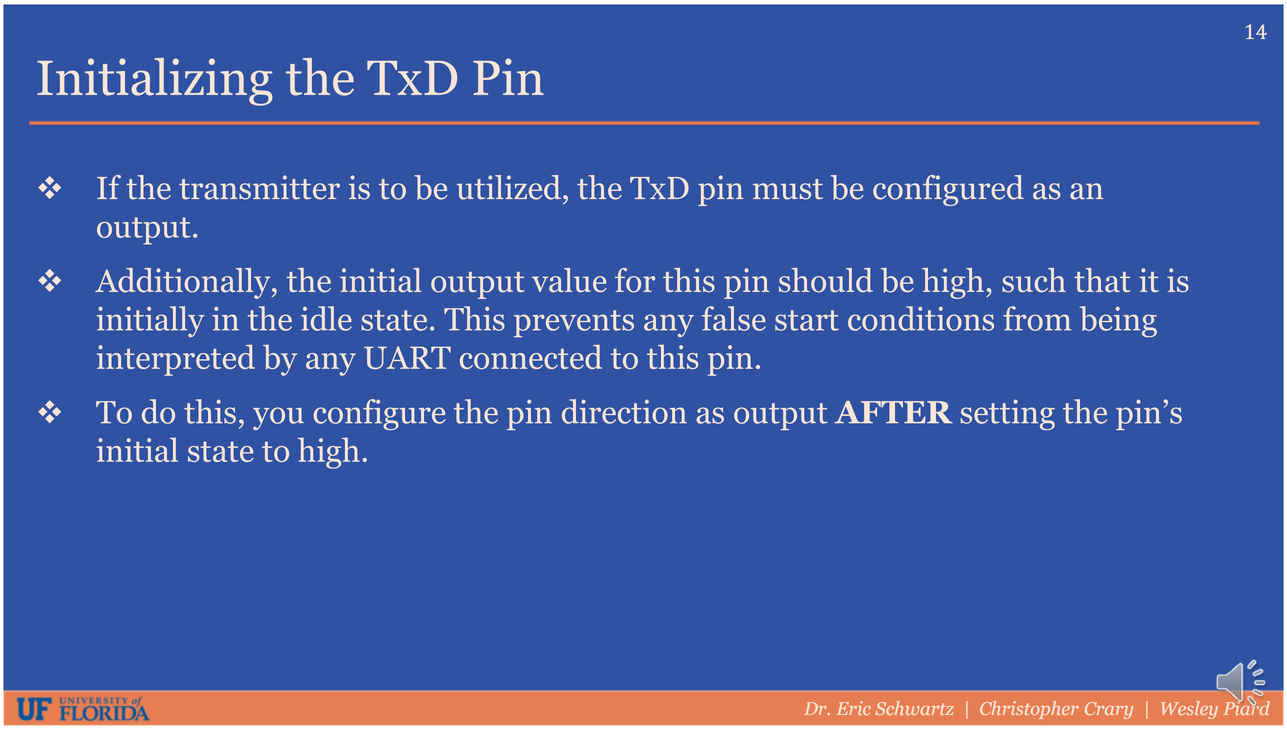
➡ 2) Set the TxD pin as output. (*PORTx_DIR*)

3) Set the baud rate. (*BAUDCTRLA/B*)

4) Set the frame format and mode of operation. (*CTRLC*)

5) Enable the transmitter and/or the receiver, depending on the usage. (*CTRLB*)

6) Optional: If UART interrupts are desired, choose a desired interrupt priority level at the UART peripheral level. (*CTRLA*)



Initializing the TxD Pin

❖ If the transmitter is to be utilized, the TxD pin must be configured as an output.

❖ Additionally, the initial output value for this pin should be high, such that it is initially in the idle state. This prevents any false start conditions from being interpreted by any UART connected to this pin.

❖ To do this, you configure the pin direction as output **AFTER** setting the pin's initial state to high.

UART Initialization Procedure

doc8331
Section 23.5

❖ UART should be initialized as follows, according to the manual:

~~1) Set the TxD pin value high. (*PORTx_OUT*)~~

~~2) Set the TxD pin as output. (*PORTx_DIR*)~~

➡ 3) Set the baud rate. (*BAUDCTRLA/B*)

4) Set the frame format and mode of operation. (*CTRLC*)

5) Enable the transmitter and/or the receiver, depending on the usage. (*CTRLB*)

6) Optional: If UART interrupts are desired, choose a desired interrupt priority level at the UART peripheral level. (*CTRLA*)

Initializing the Baud Rate

❖ To configure the baud rate, the *BSEL*[11:0] and *BSCALE*[3:0] bits must be initialized.

❖ These bitfields are within the *BAUDCTRLA* and *BAUDCTRLB* registers, as shown below.

BAUDCTRLA

Bit	7	6	5	4	3	2	1	0
+0x06	BSEL[7:0]							

doc8331
Section 23.15.6

BAUDCTRLB

Bit	7	6	5	4	3	2	1	0
+0x07	BSCALE[3:0]				BSEL[11:8]			

doc8331
Section 23.15.7

Initializing the Baud Rate - Example

- ❖ As a simple example, we will determine the *BSEL* value that will yield a baud rate of approximately 9,600 bps. Assume *BSCALE* = 0 and f_{PER} = *2 MHz*.

$$BSEL = \frac{f_{PER}}{2^{BSCALE} \cdot 16 f_{BAUD}} - 1$$

$$BSEL = \frac{2{,}000{,}000}{2^0 \cdot 16 \cdot 9{,}600} - 1$$

$$BSEL = 12.02 \approx 12$$

- ❖ If we plug *BSEL* = 12 back into this equation, we get f_{BAUD} = 9,615 bps.
- ❖ This would be an error of about 0.16% which is generally acceptable.
- ❖ The amount of error that is acceptable is dependent on the application. Testing should be performed to rule out the possibility of errors occurring.

UF UNIVERSITY of FLORIDA *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

Initializing the Baud Rate - Example

- ❖ Since we determined *BSEL* = 12, and *BSCALE* = 0, we would configure the BAUDCTRL registers as follows:
  - ➢ BAUDCTRLA = BSEL[7:0] = 12 = 0b00001100
  - ➢ BAUDCTRLB = 0

BAUDCTRLA

Bit	7	6	5	4	3	2	1	0
+0x06	BSEL[7:0]							

doc8331
Section 23.15.6

BAUDCTRLB

Bit	7	6	5	4	3	2	1	0
+0x07	BSCALE[3:0]				BSEL[11:8]			

doc8331
Section 23.15.7

UF UNIVERSITY of FLORIDA *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

19

UART Initialization Procedure

doc8331
Section 23.5

- UART should be initialized as follows, according to the manual:

~~1) Set the TxD pin value high. (*PORTx_OUT*)~~

~~2) Set the TxD pin as output. (*PORTx_DIR*)~~

~~3) Set the baud rate. (*BAUDCTRLA/B*)~~

➡ 4) Set the frame format and mode of operation. (*CTRLC*)

5) Enable the transmitter and/or the receiver, depending on the usage. (*CTRLB*)

6) Optional: If UART interrupts are desired, choose a desired interrupt priority level at the UART peripheral level. (*CTRLA*)

19

20

Initializing the Operation Mode

- The operation mode for a UART module is configured by the CMODE[1:0] bitfield.
- For this course, we will generally use the **asynchronous** mode, thus CMODE[1:0] = 0b00.

doc8331
Table 23-7

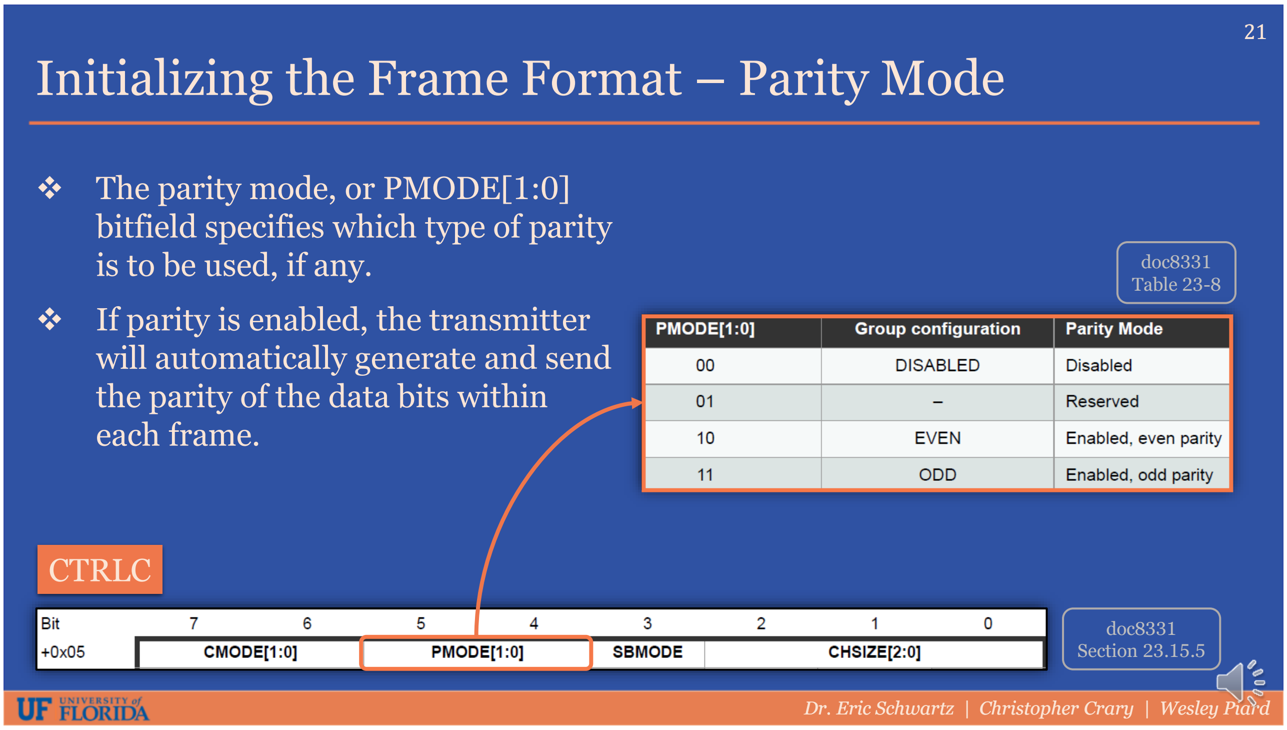
CMODE[1:0]	Group configuration	Mode
00	ASYNCHRONOUS	Asynchronous USART
01	SYNCHRONOUS	Synchronous USART
10	IRCOM	IRCOM[1]
11	MSPI	Master SPI[2]

CTRLC

Bit	7	6	5	4	3	2	1	0
+0x05	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]		

doc8331
Section 23.15.5

20



Initializing the Frame Format – Parity Mode

- ❖ The parity mode, or PMODE[1:0] bitfield specifies which type of parity is to be used, if any.
- ❖ If parity is enabled, the transmitter will automatically generate and send the parity of the data bits within each frame.

doc8331
Table 23-8

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

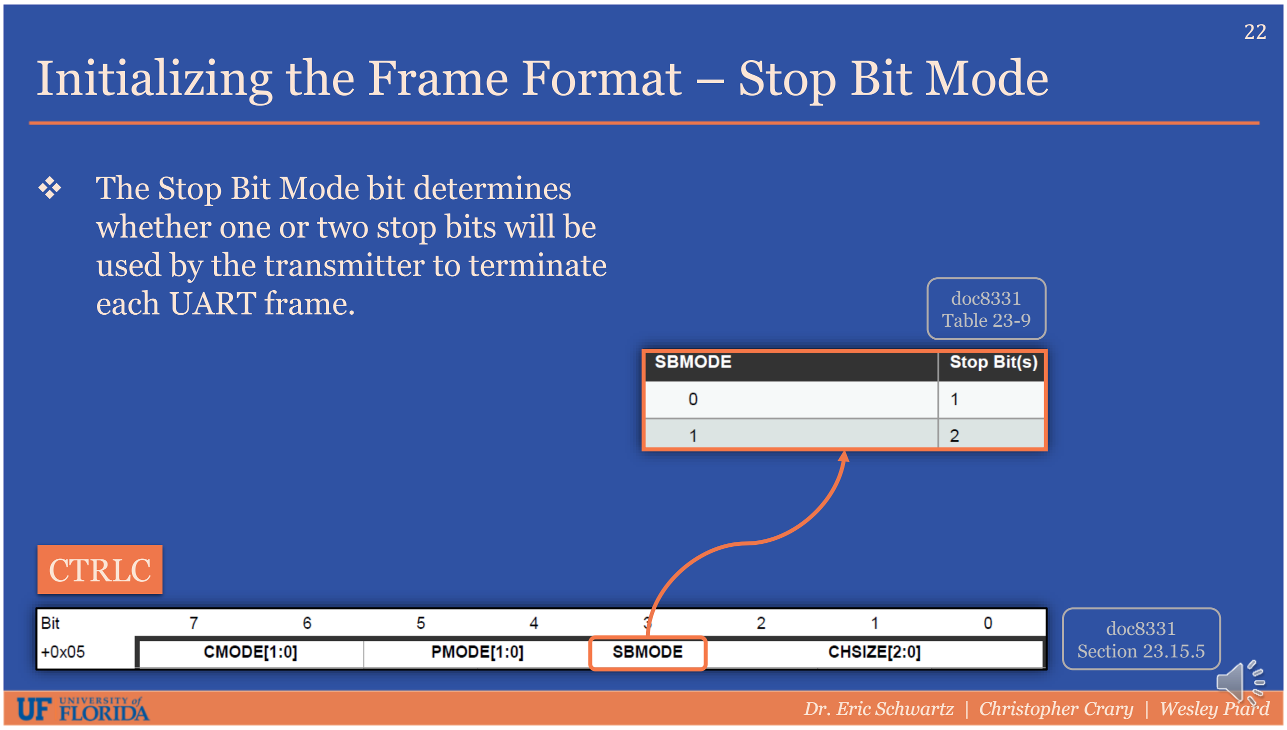
CTRLC

Bit	7	6	5	4	3	2	1	0
+0x05	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]		

doc8331
Section 23.15.5

UF UNIVERSITY of FLORIDA *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

21



Initializing the Frame Format – Stop Bit Mode

- ❖ The Stop Bit Mode bit determines whether one or two stop bits will be used by the transmitter to terminate each UART frame.

doc8331
Table 23-9

SBMODE	Stop Bit(s)
0	1
1	2

CTRLC

Bit	7	6	5	4	3	2	1	0
+0x05	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]		

doc8331
Section 23.15.5

UF UNIVERSITY of FLORIDA *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

22

Initializing the Frame Format – Character Size

- ❖ The Character Size bitfield determines the number of data bits contained within each frame.
- ❖ Note that the transmitter and receiver must have the same configuration.

doc8331
Table 23-10

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

CTRLC

Bit	7	6	5	4	3	2	1	0
+0x05	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]		

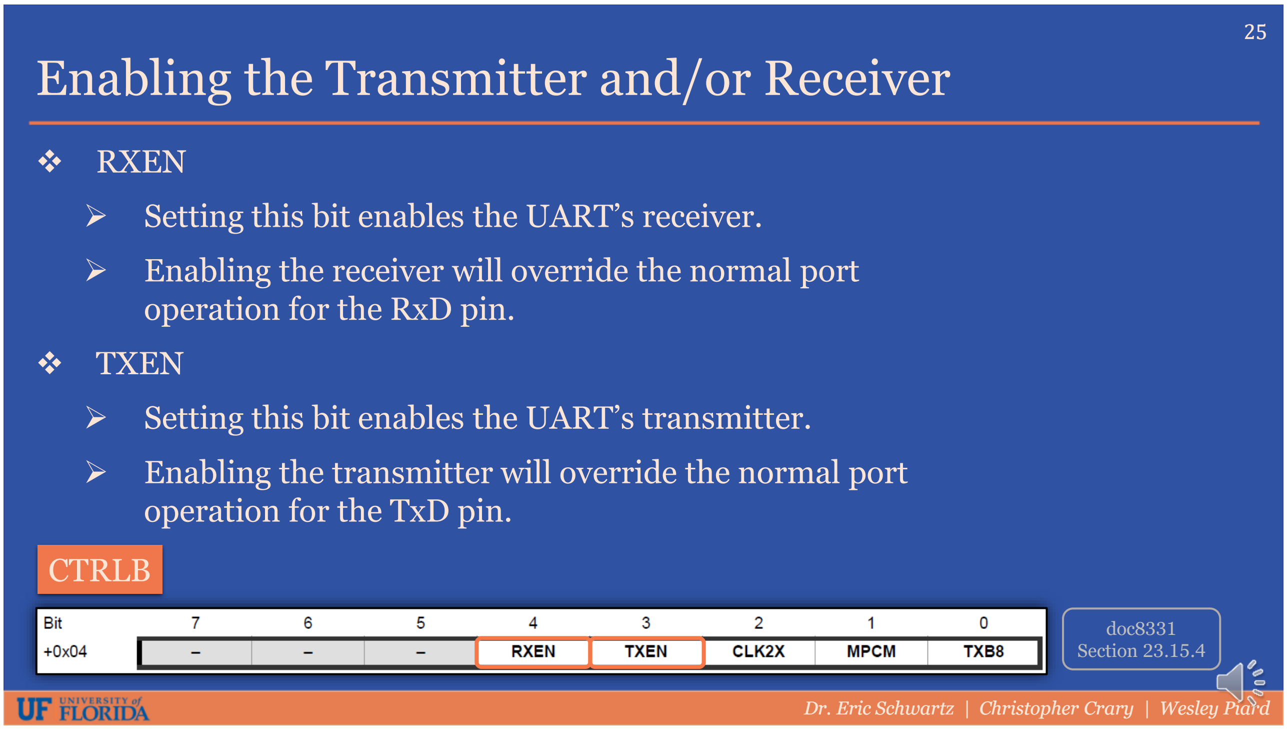
doc8331
Section 23.15.5

UART Initialization Procedure

- ❖ UART should be initialized as follows, according to the manual:

doc8331
Section 23.5

1) ~~Set the TxD pin value high. (*PORTx_OUT*)~~
2) ~~Set the TxD pin as output. (*PORTx_DIR*)~~
3) ~~Set the baud rate. (*BAUDCTRLA/B*)~~
4) ~~Set the frame format and mode of operation. (*CTRLC*)~~
5) Enable the transmitter and/or the receiver, depending on the usage. (*CTRLB*)
6) Optional: If UART interrupts are desired, choose a desired interrupt priority level at the UART peripheral level. (*CTRLA*)



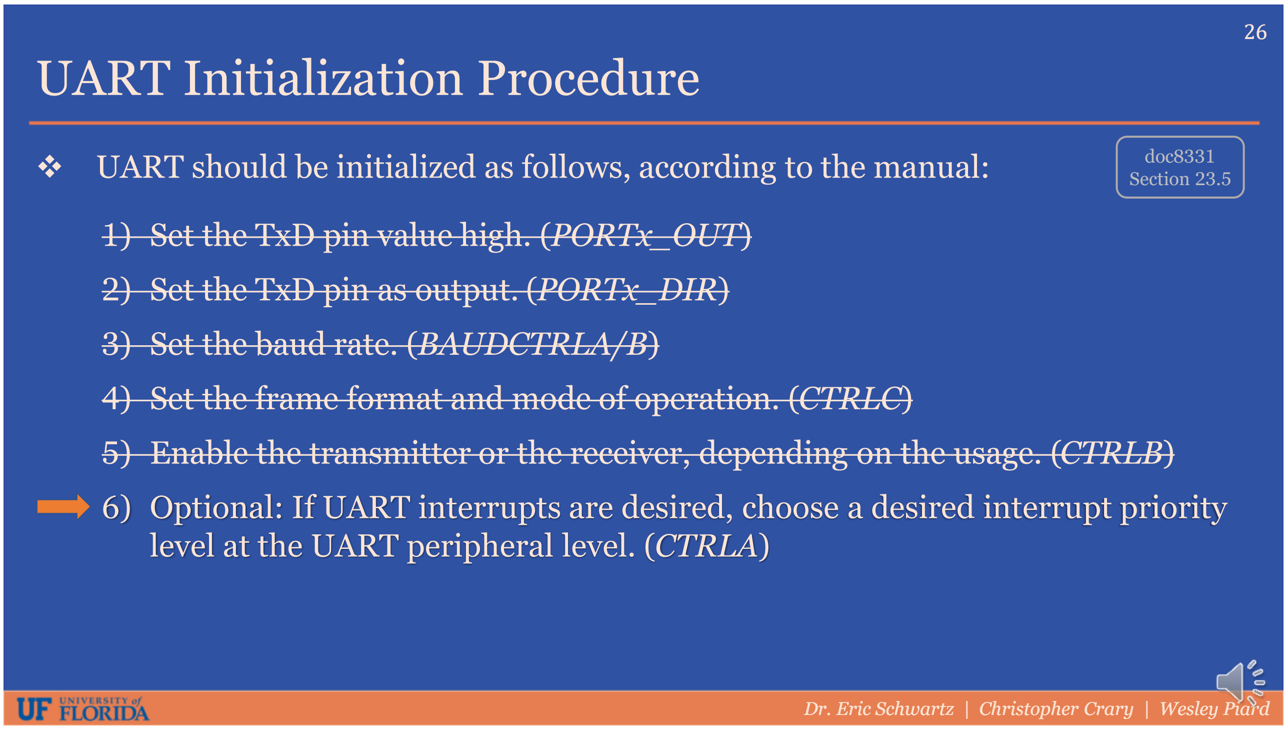
Enabling the Transmitter and/or Receiver

- ❖ RXEN
  - ➢ Setting this bit enables the UART's receiver.
  - ➢ Enabling the receiver will override the normal port operation for the RxD pin.
- ❖ TXEN
  - ➢ Setting this bit enables the UART's transmitter.
  - ➢ Enabling the transmitter will override the normal port operation for the TxD pin.

CTRLB

Bit	7	6	5	4	3	2	1	0
+0x04	–	–	–	RXEN	TXEN	CLK2X	MPCM	TXB8

doc8331
Section 23.15.4



UART Initialization Procedure

doc8331
Section 23.5

- ❖ UART should be initialized as follows, according to the manual:

~~1) Set the TxD pin value high. (*PORTx_OUT*)~~

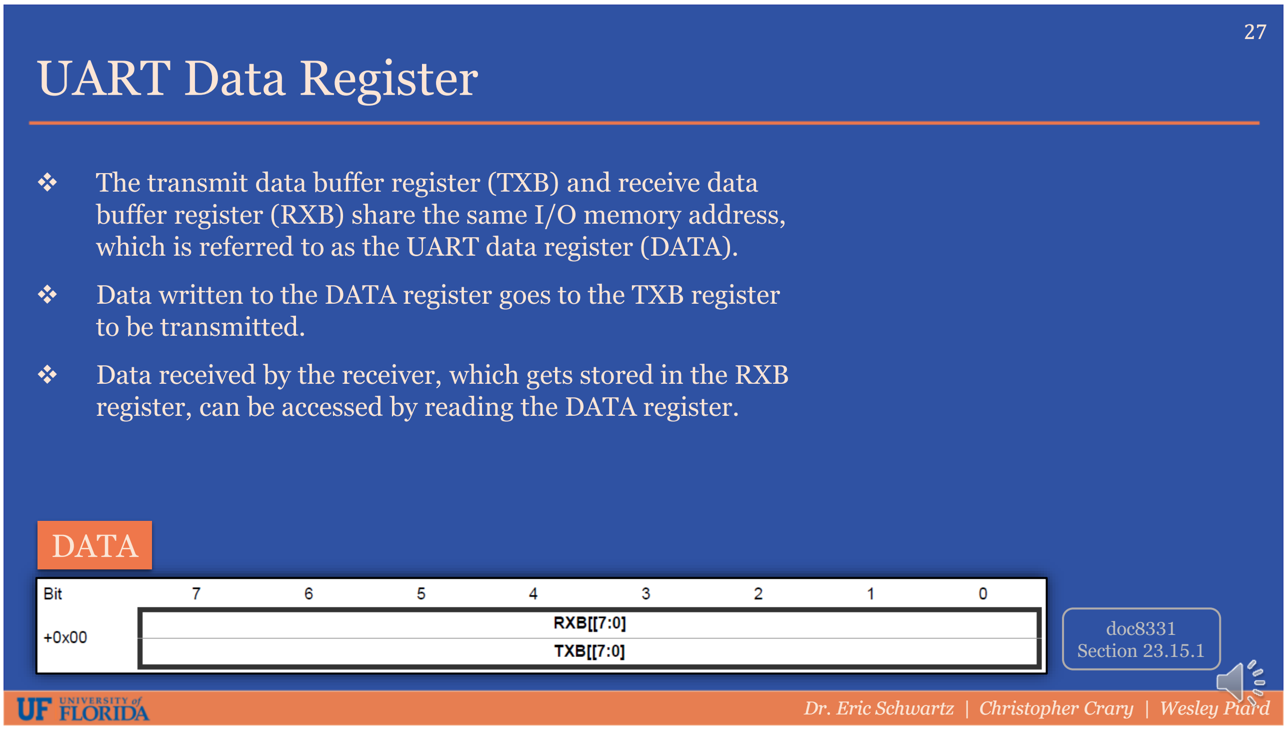
~~2) Set the TxD pin as output. (*PORTx_DIR*)~~

~~3) Set the baud rate. (*BAUDCTRLA/B*)~~

~~4) Set the frame format and mode of operation. (*CTRLC*)~~

~~5) Enable the transmitter or the receiver, depending on the usage. (*CTRLB*)~~

➡ 6) Optional: If UART interrupts are desired, choose a desired interrupt priority level at the UART peripheral level. (*CTRLA*)



27

UART Data Register

- The transmit data buffer register (TXB) and receive data buffer register (RXB) share the same I/O memory address, which is referred to as the UART data register (DATA).
- Data written to the DATA register goes to the TXB register to be transmitted.
- Data received by the receiver, which gets stored in the RXB register, can be accessed by reading the DATA register.

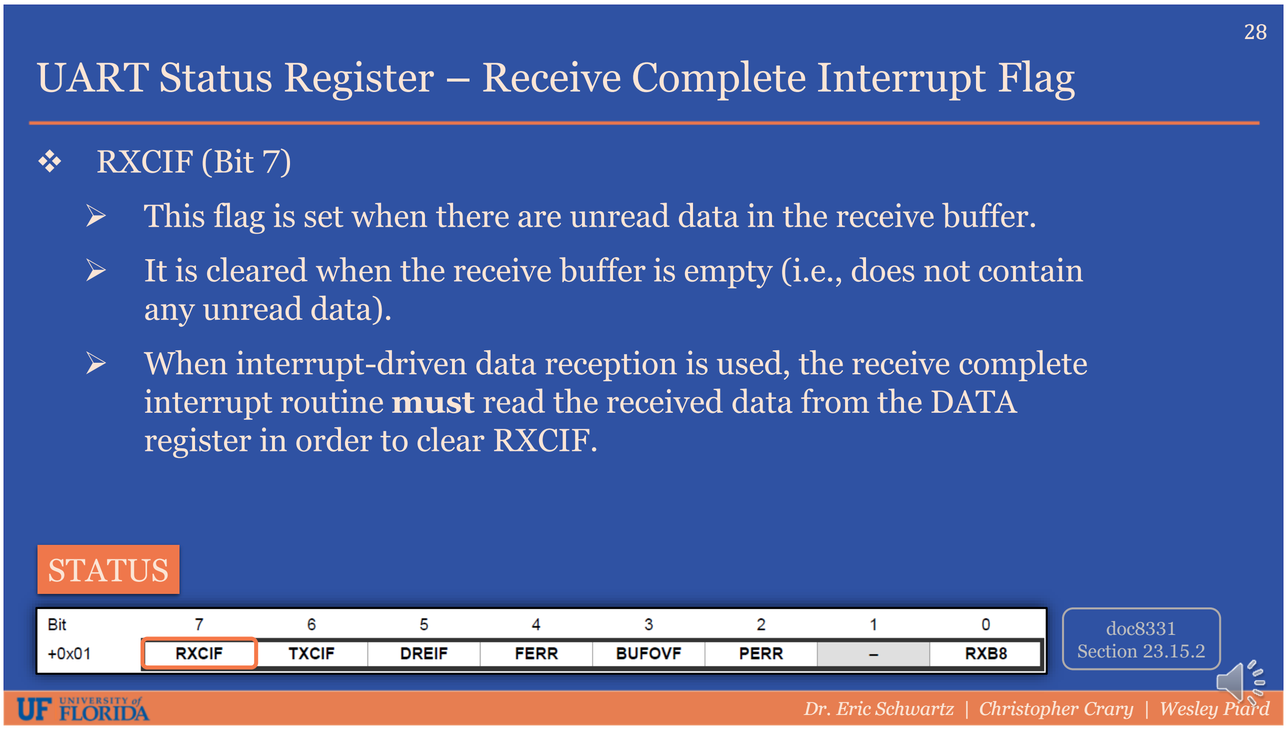
DATA

Bit	7	6	5	4	3	2	1	0
+0x00	RXB[[7:0]							
	TXB[[7:0]							

doc8331
Section 23.15.1

UF UNIVERSITY of FLORIDA *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

27



28

UART Status Register – Receive Complete Interrupt Flag

- RXCIF (Bit 7)
  - This flag is set when there are unread data in the receive buffer.
  - It is cleared when the receive buffer is empty (i.e., does not contain any unread data).
  - When interrupt-driven data reception is used, the receive complete interrupt routine **must** read the received data from the DATA register in order to clear RXCIF.

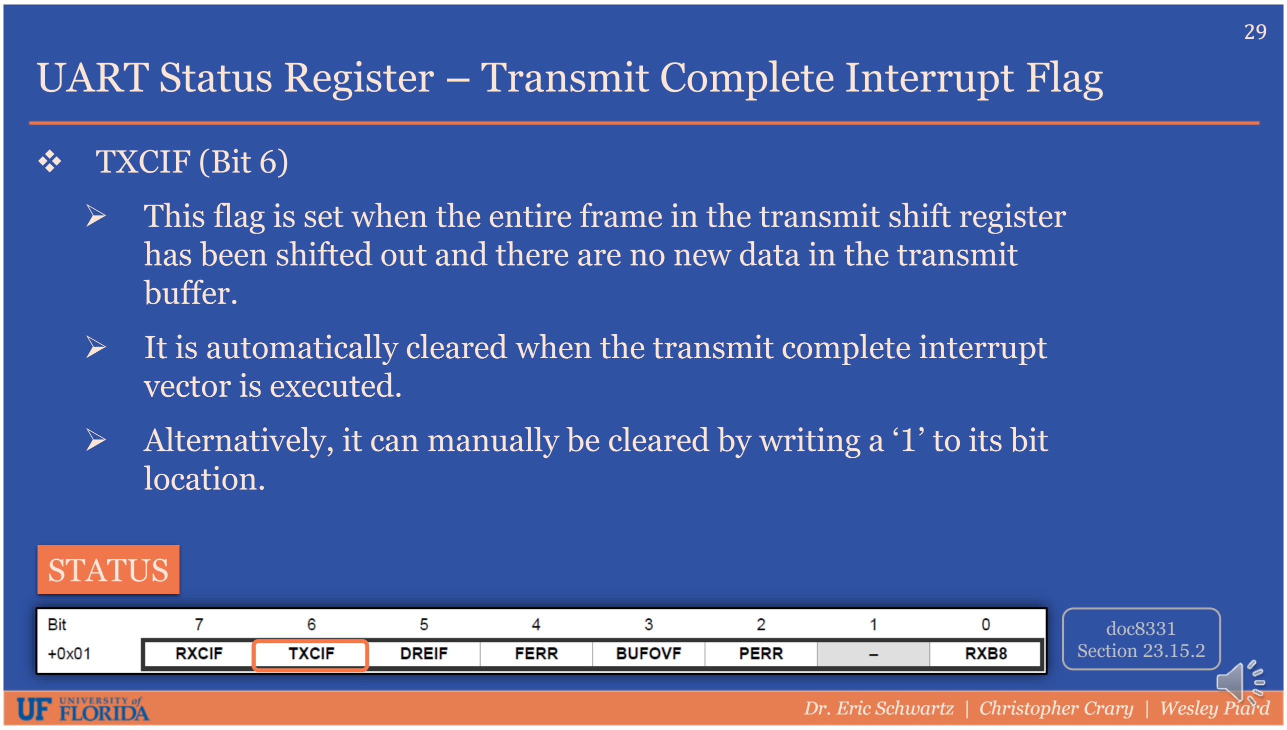
STATUS

Bit	7	6	5	4	3	2	1	0
+0x01	RXCIF	TXCIF	DREIF	FERR	BUFOVF	PERR	–	RXB8

doc8331
Section 23.15.2

UF UNIVERSITY of FLORIDA *Dr. Eric Schwartz | Christopher Crary | Wesley Piard*

28



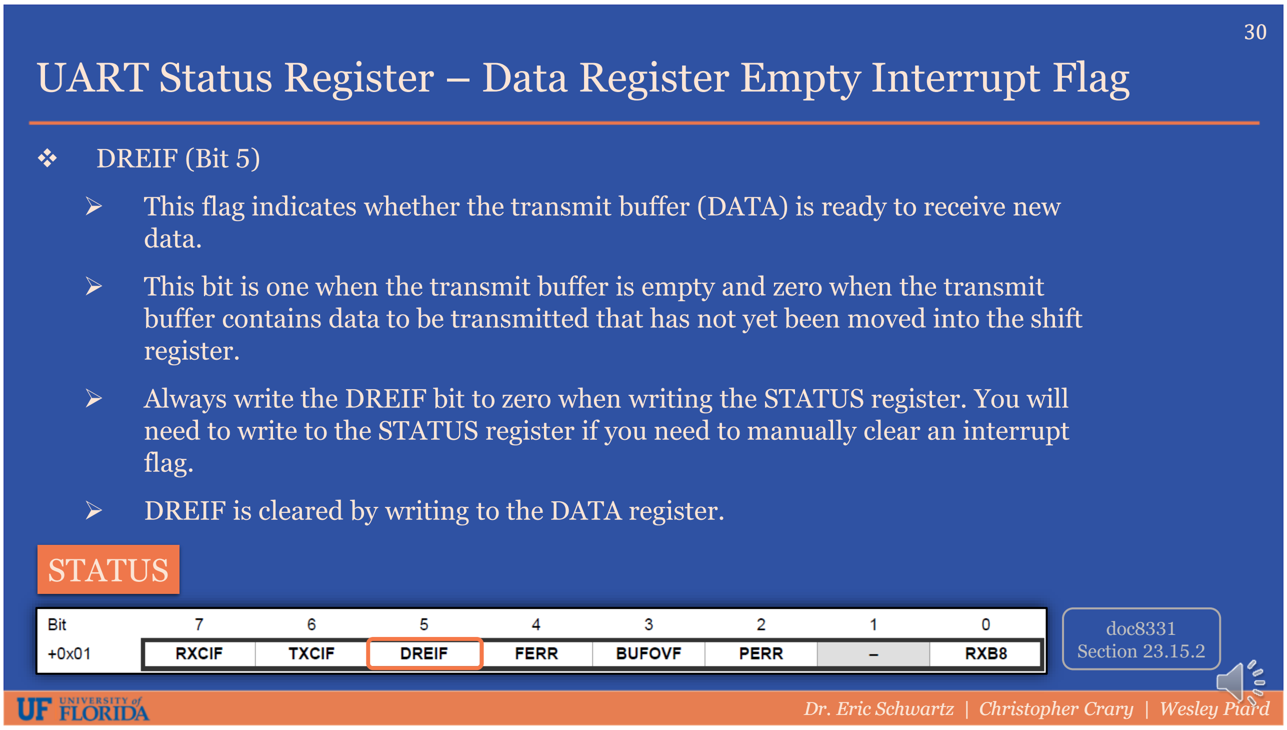
UART Status Register – Transmit Complete Interrupt Flag

- ❖ TXCIF (Bit 6)
  - ➢ 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.
  - ➢ It is automatically cleared when the transmit complete interrupt vector is executed.
  - ➢ Alternatively, it can manually be cleared by writing a ‘1’ to its bit location.

STATUS

Bit	7	6	5	4	3	2	1	0
+0x01	RXCIF	TXCIF	DREIF	FERR	BUFOVF	PERR	–	RXB8

doc8331
Section 23.15.2



UART Status Register – Data Register Empty Interrupt Flag

- ❖ DREIF (Bit 5)
  - ➢ This flag indicates whether the transmit buffer (DATA) is ready to receive new data.
  - ➢ This bit 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.
  - ➢ Always write the DREIF bit to zero when writing the STATUS register. You will need to write to the STATUS register if you need to manually clear an interrupt flag.
  - ➢ DREIF is cleared by writing to the DATA register.

STATUS

Bit	7	6	5	4	3	2	1	0
+0x01	RXCIF	TXCIF	DREIF	FERR	BUFOVF	PERR	–	RXB8

doc8331
Section 23.15.2

Enabling UART Interrupts

❖ Each of the three UART interrupts (RXC, TXC, and DRE) can be enabled with different priority levels, as per the PMIC.

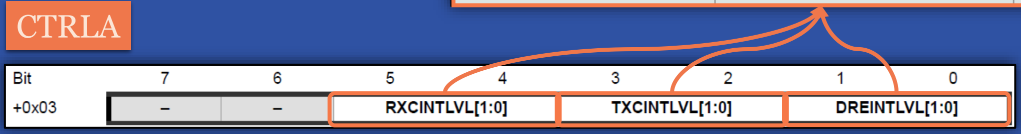
❖ Each of these interrupts is triggered by its respective interrupt flag in the USART STATUS register (RXCIF, TXCIF, and DREIF).

doc8331
Table 12-1

```
; USARTC0 interrupt vectors
.equ USARTC0_RXC_vect = 50
.equ USARTC0_DRE_vect = 52
.equ USARTC0_TXC_vect = 54
```

Interrupt level configuration	Group configuration	Description
00	OFF	Interrupt disabled.
01	LO	Low-level interrupt
10	MED	Medium-level interrupt
11	HI	High-level interrupt

CTRLA



doc8331
Section 23.15.3

31

UART Initialization Procedure

❖ UART should be initialized as follows, according to the manual:

doc8331
Section 23.5

~~1) Set the TxD pin value high. (*PORTx_OUT*)~~

~~2) Set the TxD pin as output. (*PORTx_DIR*)~~

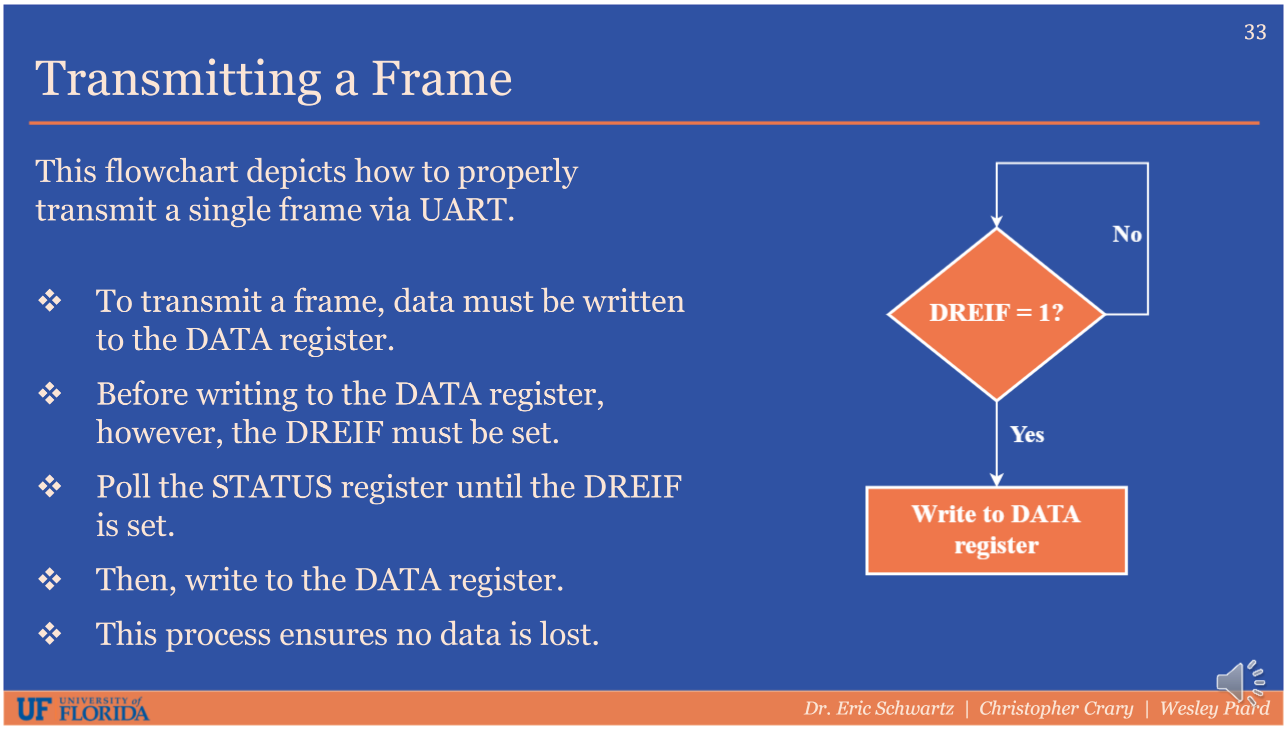
~~3) Set the baud rate. (*BAUDCTRLA/B*)~~

~~4) Set the frame format and mode of operation. (*CTRLC*)~~

~~5) Enable the transmitter and/or the receiver, depending on the usage. (*CTRLB*)~~

~~6) Optional: If UART interrupts are desired, choose a desired interrupt priority level at the UART peripheral level. (*CTRLA*)~~

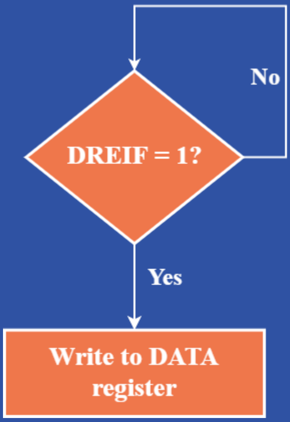
32

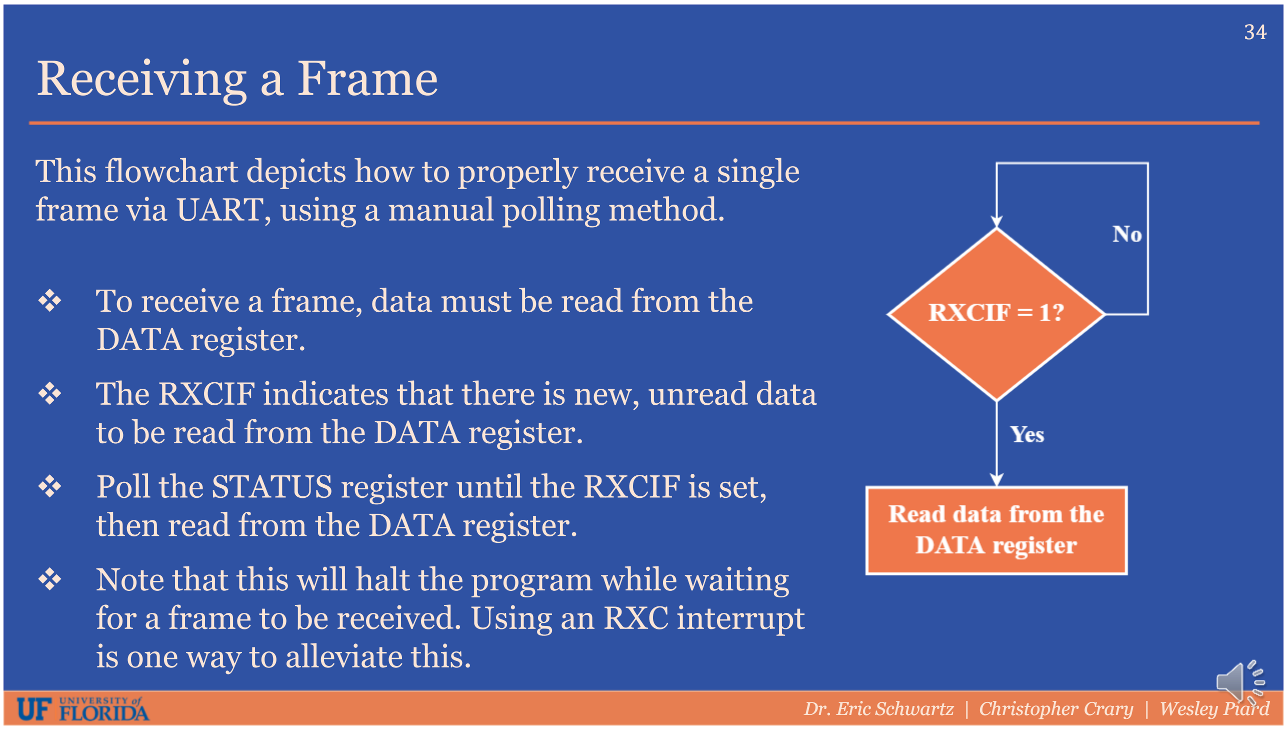


Transmitting a Frame

This flowchart depicts how to properly transmit a single frame via UART.

- ❖ To transmit a frame, data must be written to the DATA register.
- ❖ Before writing to the DATA register, however, the DREIF must be set.
- ❖ Poll the STATUS register until the DREIF is set.
- ❖ Then, write to the DATA register.
- ❖ This process ensures no data is lost.

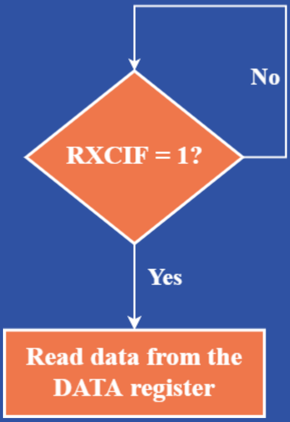


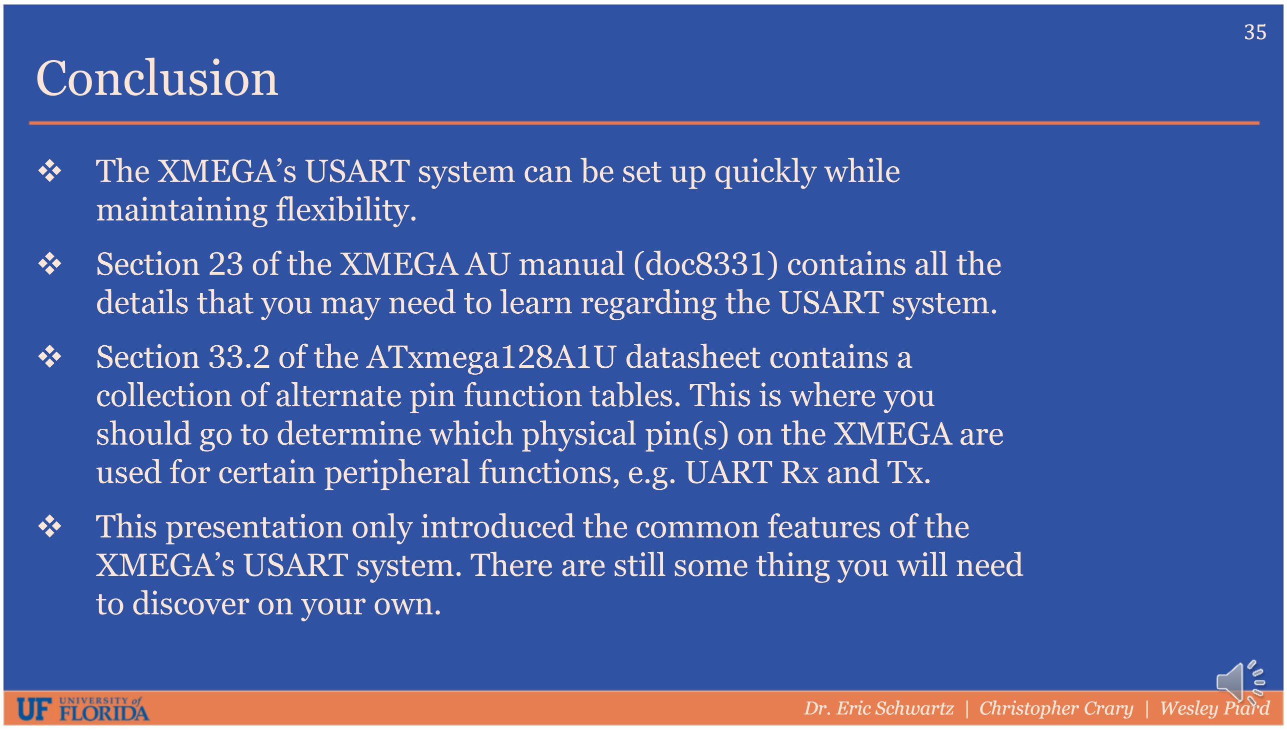


Receiving a Frame

This flowchart depicts how to properly receive a single frame via UART, using a manual polling method.

- ❖ To receive a frame, data must be read from the DATA register.
- ❖ The RXCIF indicates that there is new, unread data to be read from the DATA register.
- ❖ Poll the STATUS register until the RXCIF is set, then read from the DATA register.
- ❖ Note that this will halt the program while waiting for a frame to be received. Using an RXC interrupt is one way to alleviate this.





Conclusion

- The XMEGA's USART system can be set up quickly while maintaining flexibility.
- Section 23 of the XMEGA AU manual (doc8331) contains all the details that you may need to learn regarding the USART system.
- Section 33.2 of the ATxmega128A1U datasheet contains a collection of alternate pin function tables. This is where you should go to determine which physical pin(s) on the XMEGA are used for certain peripheral functions, e.g. UART Rx and Tx.
- This presentation only introduced the common features of the XMEGA's USART system. There are still some thing you will need to discover on your own.

35