

USART

ATxmega128A1

Networks and Embedded Software

Module 4.3.5

by Wolfgang Neff

USART (1)

- Implementation

- Eight USART modules

- USARTC0 (RXD=PC2, TXD=PC3), USARTC1 (RXD=PC6, TXD=PC7)
- USARTE0 (RXD=PE2, TXD=PE3), USARTE1 (RXD=PE6, TXD=PE7)
- USARTF0 (RXD=PF2, TXD=PF3), USARTF1 (RXD=PF6, TXD=PF7)

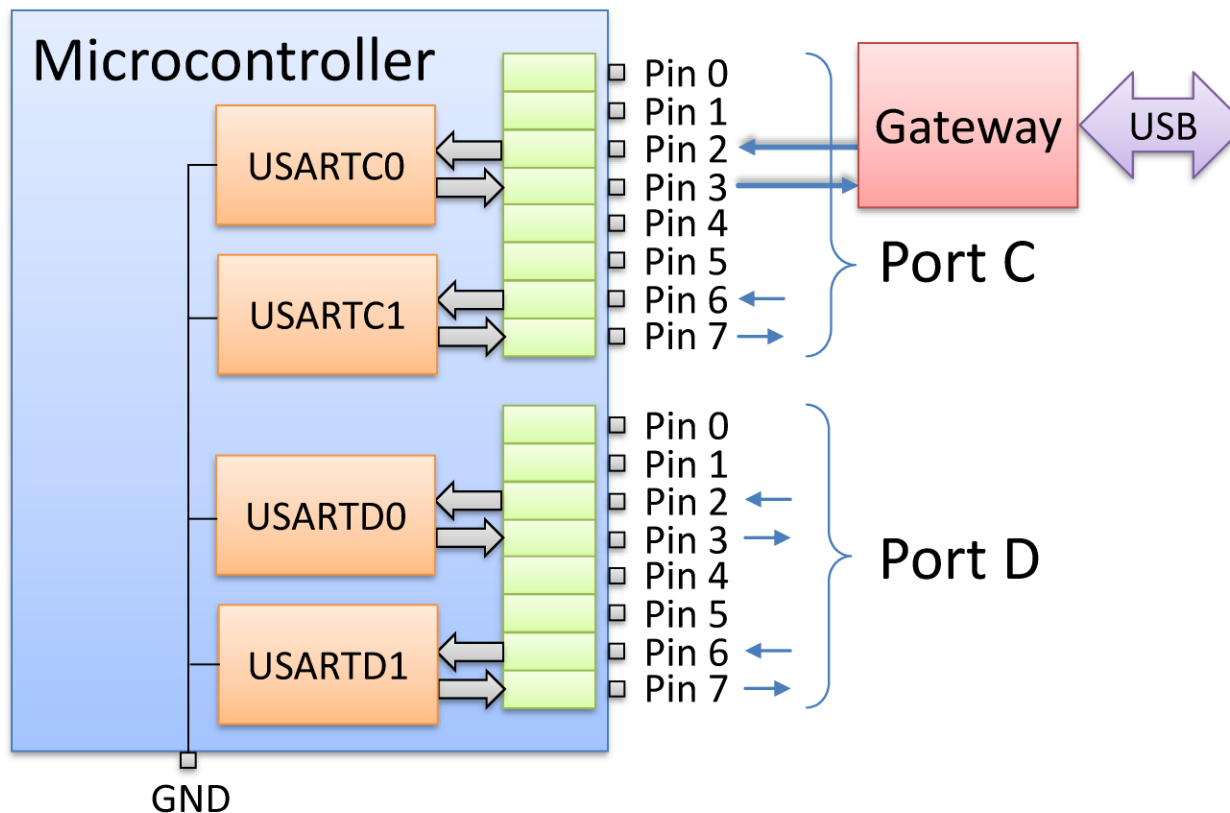
- USART-to-USB gateway

- Module: USARTC0
- Configuration: 115200/8N1



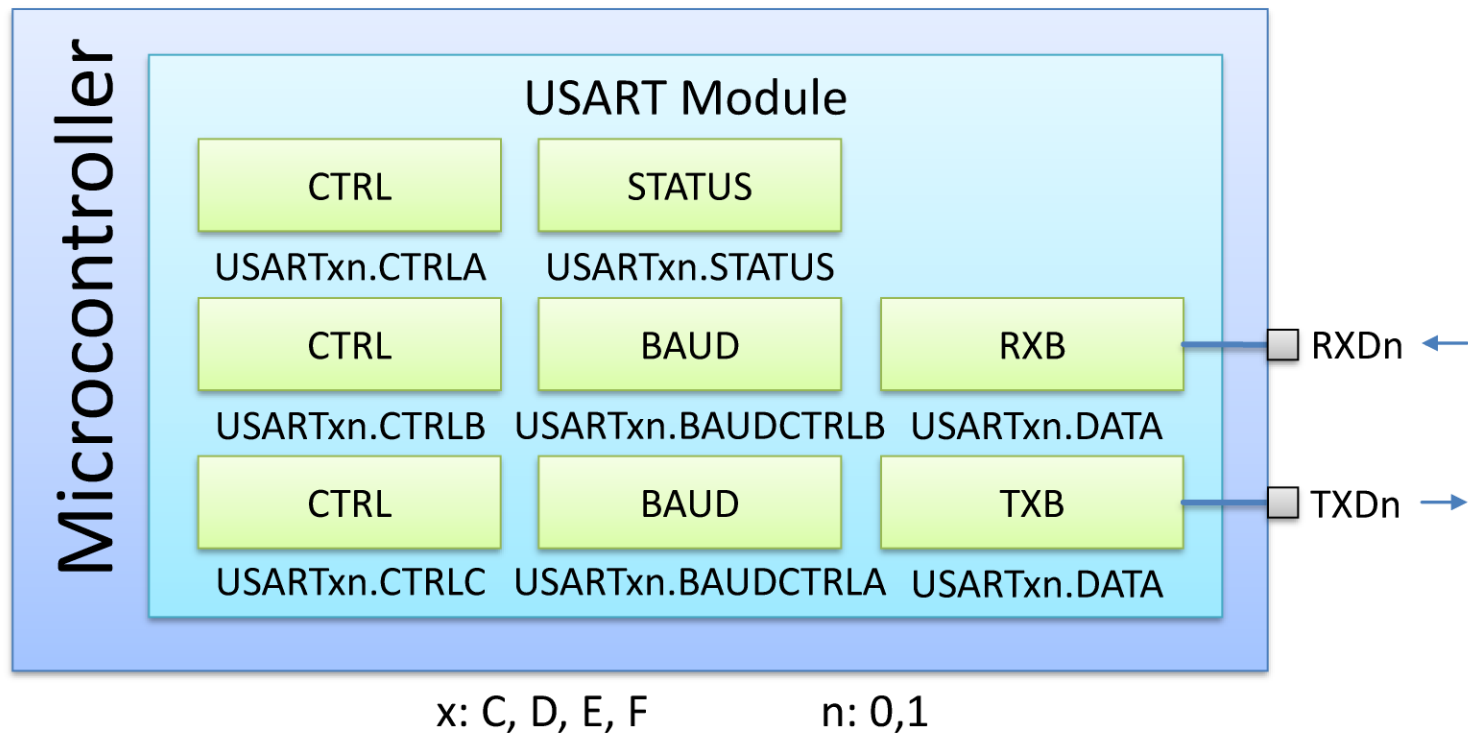
USART (2)

- Architecture



UART (3)

- Register Mapping



USART (4)

- Register Description
 - DATA: USART I/O Data Register
 - Writes to Transmit Data Buffer Register (TXB)
 - Reads from Receive Data Buffer Register (RXB)
 - STATUS: USART Status Register
 - Bit 7 – RXCIF: USART Receive Complete Interrupt Flag
 - Bit 6 – TXCIF: USART Transmit Complete Interrupt Flag
 - Bit 5 – DREIF: USART Data Register Empty Flag

USART (5)

- Register Description (continued)
 - CTRLA: USART Control Register A
 - Bit 5:4 – RXCINTLVL: Receive Complete Interrupt Level
 - Bit 3:2 – TXCINTLVL: Transmit Complete Interrupt Level
 - Bit 1:0 – DREINTLVL: Data Register Empty Interrupt Lev.
 - USART_DREINTLVL_OFF_gc
 - USART_DREINTLVL_LO_gc
 - USART_DREINTLVL_MED_gc
 - USART_DREINTLVL_HI_gc
 - Look at module 3.2 *Interrupts* for details

USART (6)

- Register Description (continued)
 - CTRLB: USART Control Register B
 - Bit 4 – RXEN: Receiver Enable
 - Bit 3 – TXEN: Transmitter Enable
 - CTRLC: USART Control Register C
 - Bits 7:6 – CMODE: USART Communication Mode
 - Usually `USART_CMODE_ASYNCHRONOUS_gc`
 - Consult datasheet for details

USART (7)

- Register Description (continued)
 - CTRLC: USART Control Register C (continued)
 - Bits 5:4 – PMODE: Parity Mode
 - USART_PMODE_DISABLED_gc: No parity
 - USART_PMODE_EVEN_gc: Even parity
 - USART_PMODE_ODD_gc: Odd parity
 - Bit 3 – SBMODE: Stop Bit Mode
 - 0: One stop bit
 - 1: Two stop bits

USART (8)

- Register Description (finished)
 - CTRLC: USART Control Register C (continued)
 - Bit 2:0 – CHSIZE: Character Size
 - Usually `USART_CHSIZE_8BIT_gc`
 - Consult datasheet for details
 - BAUDCTRLA: USART Baud Rate Register A
 - BAUDCTRLB: USART Baud Rate Register B
 - Bit 7:4 – BSCALE: USART Baud Rate Scale factor
 - -7 (1001) ... 7 (0111)

USART (9)

- Baud Rate Table
 - F_CPU = 2000000 (default frequency)

Speed	BSEL	BSCALE	Error
9600	12	0	0.16%
14400	123	-4	-0.08%
19200	11	-1	0.16%
38400	9	-2	0.16%
57600	75	-6	-0.08%
115200	11	-7	-0.08%

USART (10)

- Configuration Example (115200/8N1, no interrupts)

- Configure RXD and TXD pins

- `PORT.OUT |= PIN3_bm; // TXD - idle`
- `PORT.DIR |= PIN3_bm; // TXD - output`
- `PORT.DIR &= ~PIN2_bm; // RXD - input`

Pins are configured via the port module

- Configure 8N1 asynchronous mode

- `USART_CTRLC = (
 USART_CHSIZE_8BIT_gc |
 USART_PMODE_DISABLED_gc
);`

USART (11)

- Configuration Example (continued)
 - Set speed to 115200 Baud (see Baud rate table)
 - `USART.BAUDCTRLA = 11;`
 - `USART.BAUDCTRLB = (-7<<USART_BSCALE_gp);`
 - Enable receiver and transmitter
 - `USART.CTRLB = (
 USART_RXEN_bm |
 USART_TXEN_bm
);`

USART (12)

- Polling

- Synchronous Input/Output

- Blocking read

- `while (!(USART.STATUS & USART_RXCIF_bm));`
`data = USART.DATA; // read from USART`

- Blocking write

- `while (!(USART.STATUS & USART_DREIF_bm));`
`USART.DATA = data; // write to USART`

- Easy to implement but waste of performance



Active Wait.
Do Nothing

USART (13)

- Polling (continued)
 - Asynchronous Input/Output
 - Non-blocking read
 - `if (USART.STATUS & USART_RXCIF_bm) {`
 `data = USART.DATA;`
 `}`
 - Non-blocking write
 - `if (USART.STATUS & USART_DREIF_bm) {`
 `USART.DATA = data;`
 `}`
 - Performant but tricky. Good timing necessary.

USART (14)

- Register Summary

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DATA	DATA[7:0]							
STATUS	RXCIF	TXCIF	DREIF	-	-	-	-	-
CTRLA	-	-	RXCINTLVL[1:0]		TXCINTLVL[1:0]		DREINTLVL[1:0]	
CTRLB	-	-	-	RXEN	TXEN	CLK2X	MPCM	TXB8
CTRLC	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]		
BAUDCTRLA	BSEL[7:0]							
BAUDCTRLB	BSCALE[3:0]				BSEL[11:8]			

USART (15)

- Interrupt Summary

Source	Description
USARTxn_RXC_vect	USART receive complete interrupt vector
USARTxn_DRE_vect	USART data register empty interrupt vector
USARTxn_TXC_vect	USART transmit complete interrupt vector

