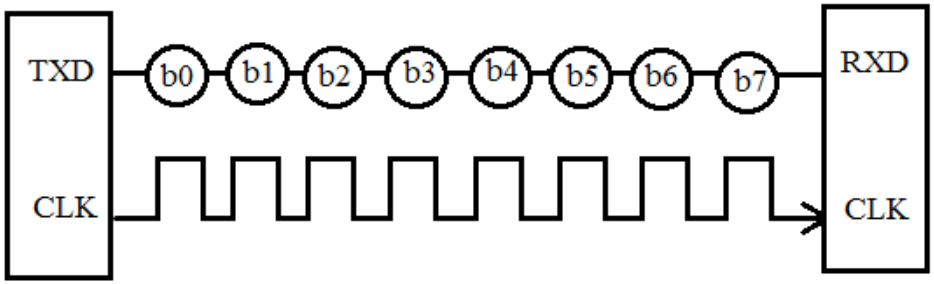
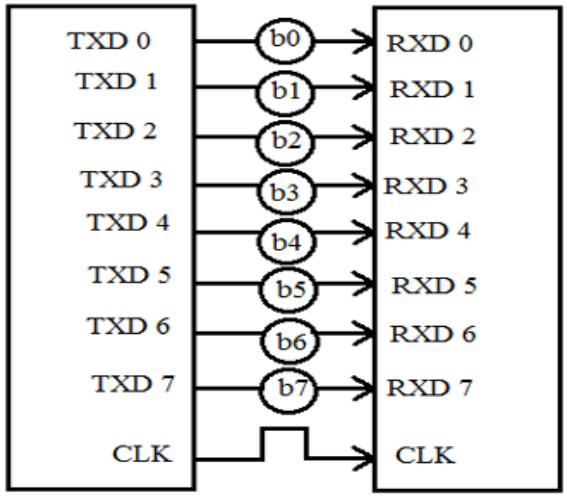
Serial Communication

Basics of Serial Communication

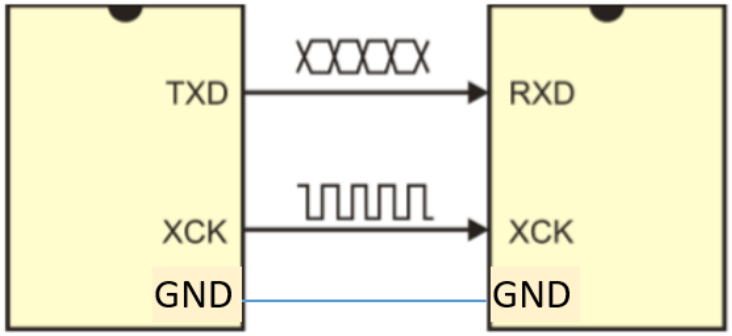
Serial	Parallel
1 bit a time	1 byte a time
Long range	Short range (into PC)
One wire	More tan one wire (Bus)
Synchronization→CLK*	
Protocolos: CAN, ETHERNET, I2C, SPI, RS232, USB, 1-Wire, SATA…	Protocolos: ISA, ATA, SCSI, PCI and IEEE-488...



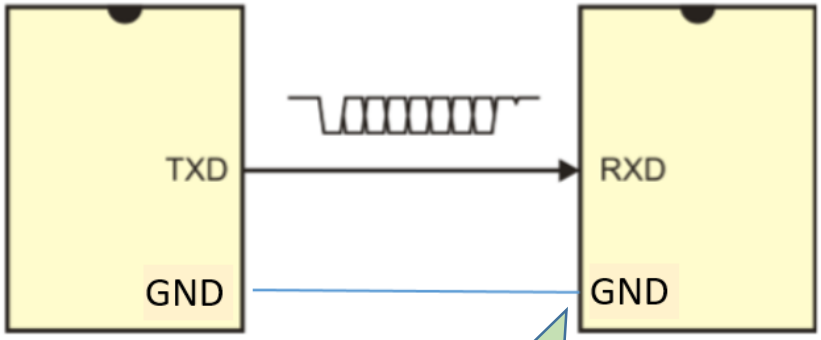


USART: Types of communication, standars

Synchronous



Asynchronous



I2C

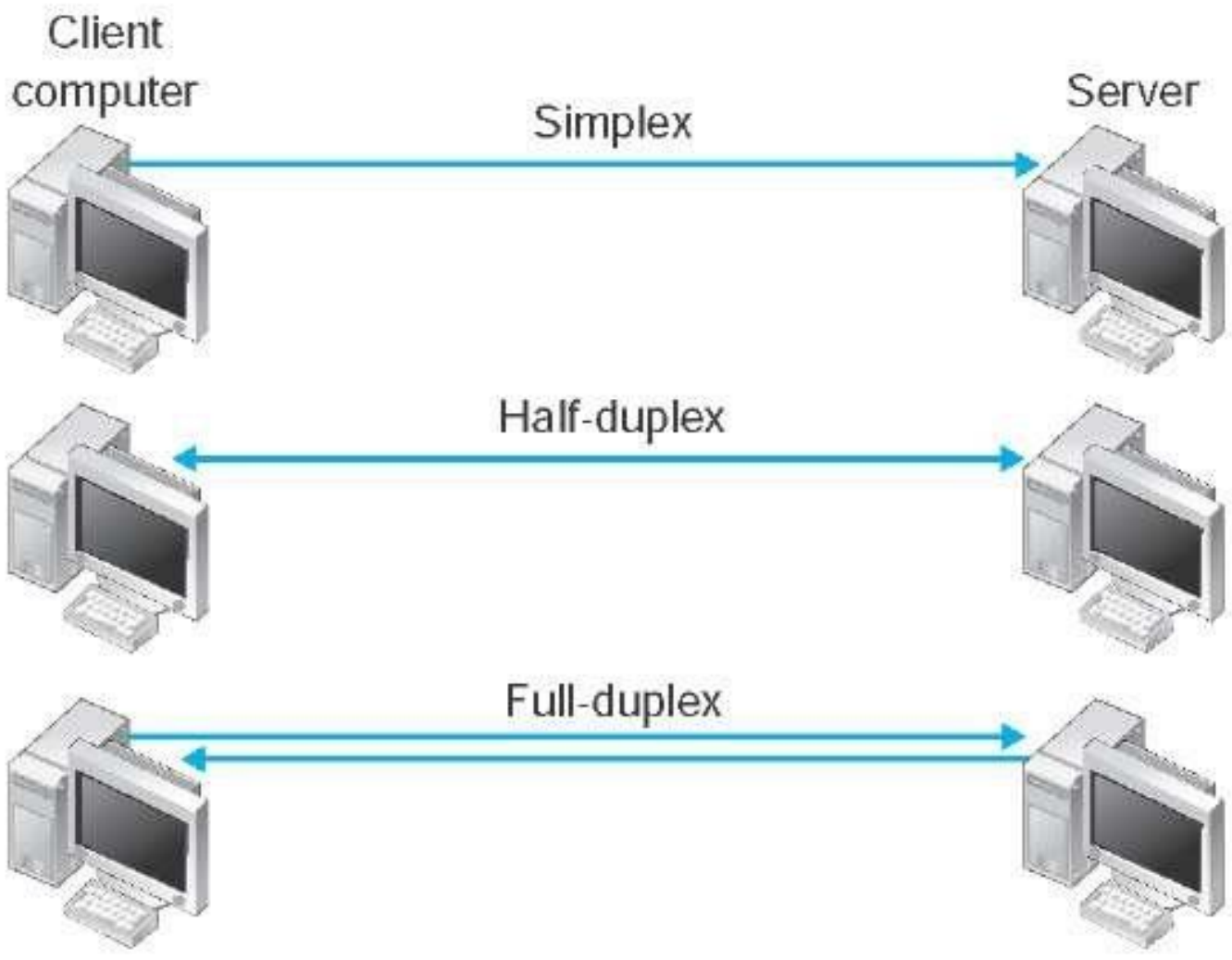
SPI

RS232

RS422

RS485

USART: Transmission Modes

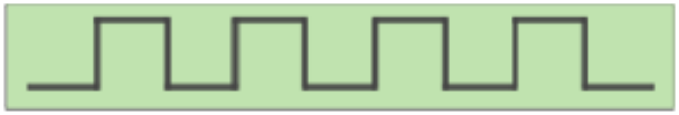


USART: Asynchronous Communication Rules

Not have an external clock signal, and it relies on four parameters namely

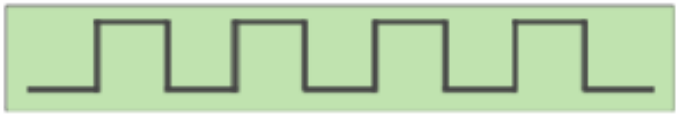
Rule 1: **Baud Rate**

No.of bits transmitted per second from sender to receiver.

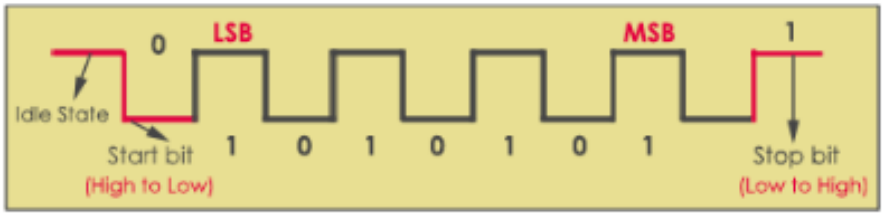


Rule 2: **Data length selection**

7 bit data sent from sender to receiver.



Rule 3: **Synchronization**



Start bit - Indicated by ZERO **Stop bit - Indicated by ONE**

Rule 4: **Error Checking**

Parity bit is '1' for even number of binary ones and '0' for odd number of binary ones. According to rule 3 it is set to 1.

Fuente: Codrey Electronics

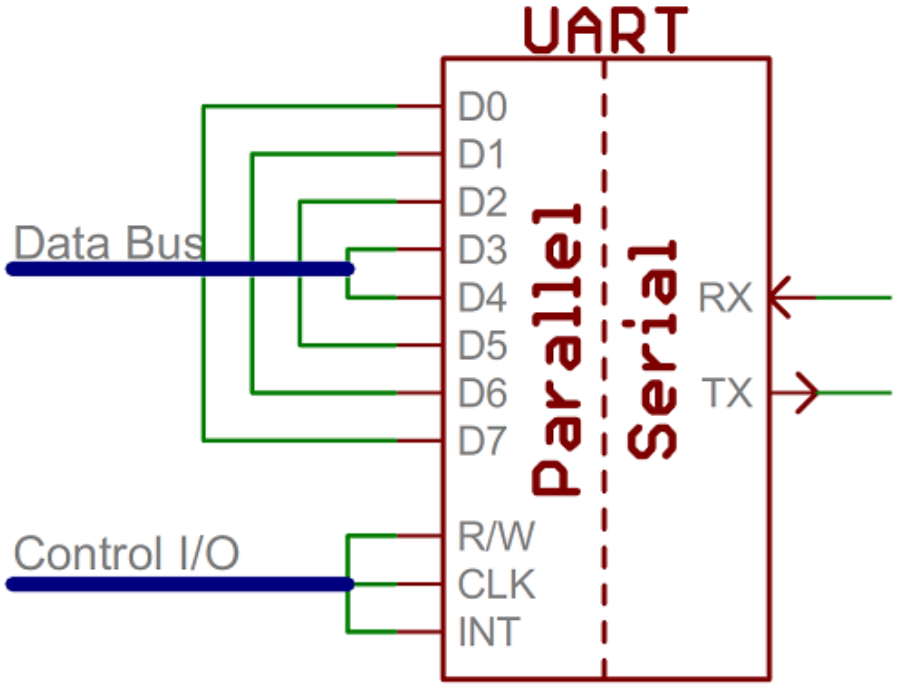
USART: Asynchronous Serial Frame

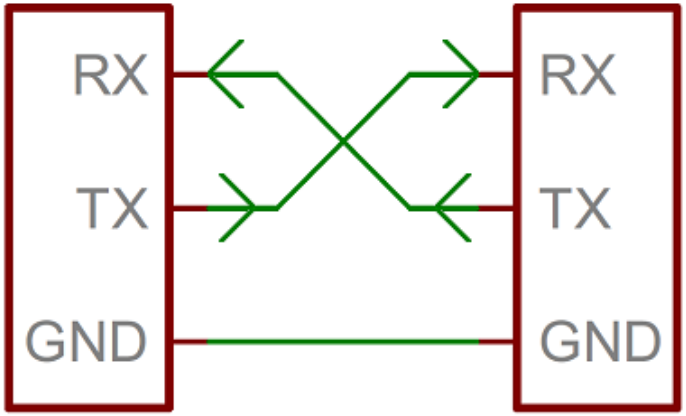
- For every byte of data transmitted, → 7-13 bits being sent
- 9600 bits per second / 10-bits frame →

  960 "8-bit data" per second.

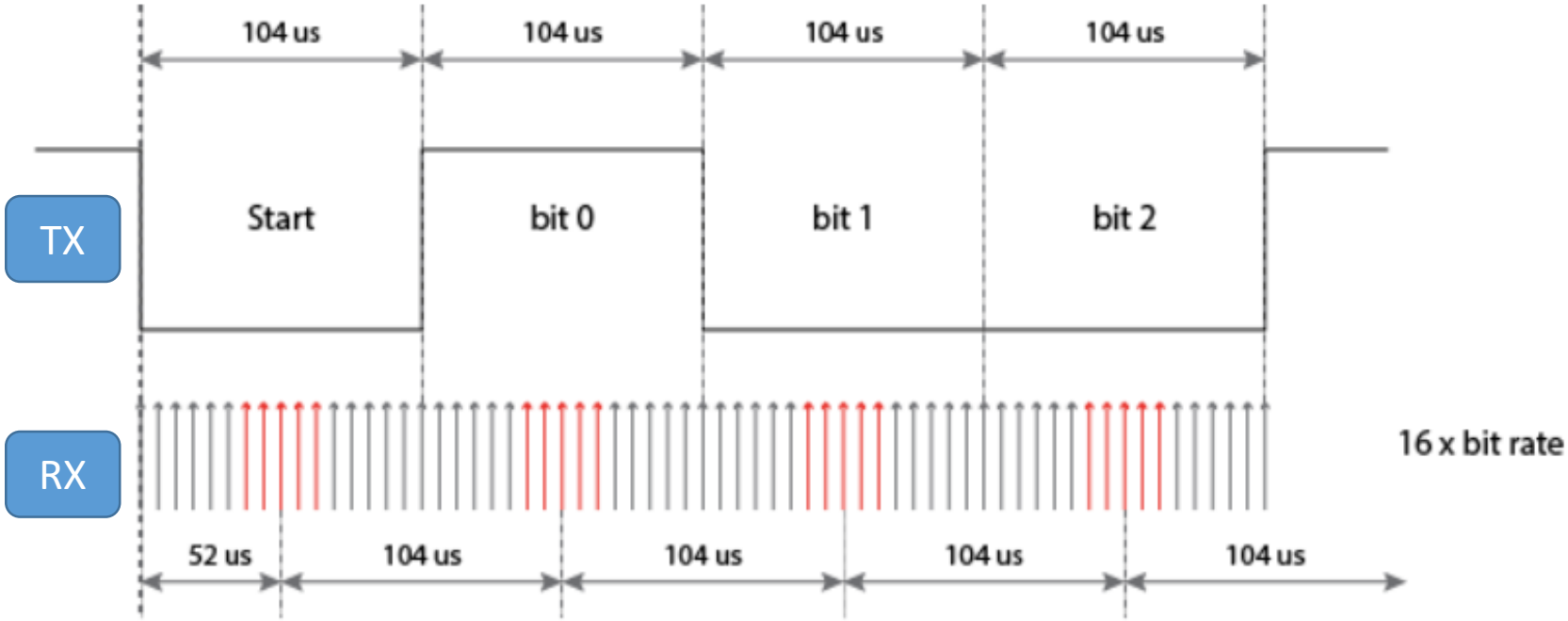
Author: Mirian Cifredo Chacón (Dpt. Ingeniería en Automática, Electrónica, Redes y Tecnología de Computadores)

USART: Asynchronous Serial Hardware (USART)





USART: Asynchronous Timing diagram (USART)



Example : 9.600 bit per second (bps) → 1 bit 104 us

USART ATMega 328p

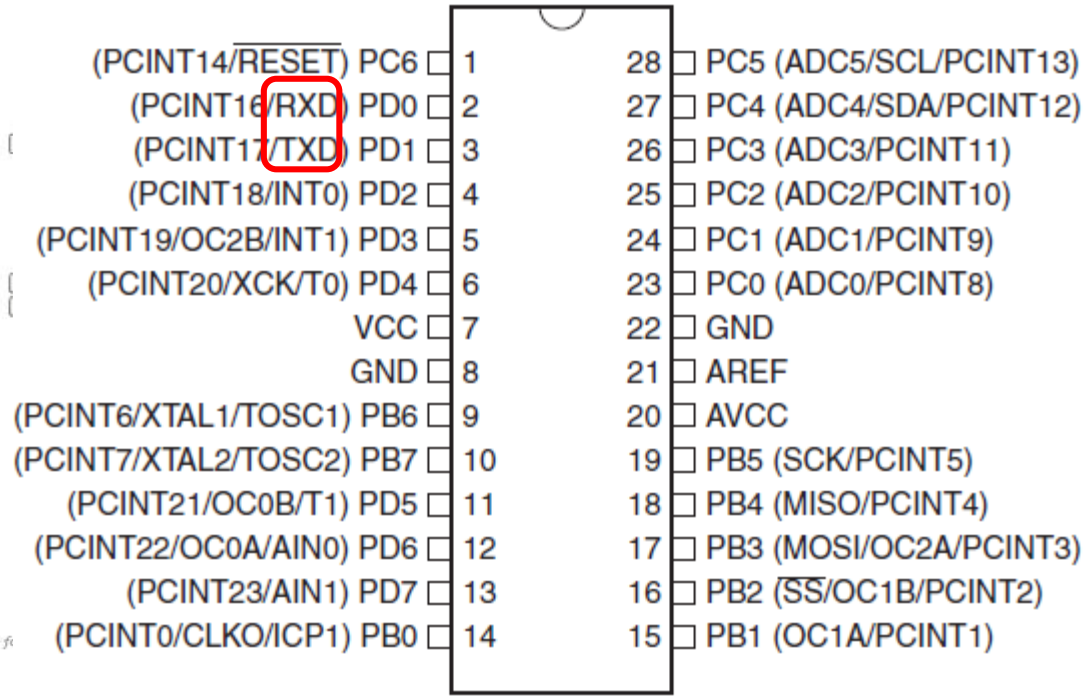
USART ATMega328: Features

20. USART0

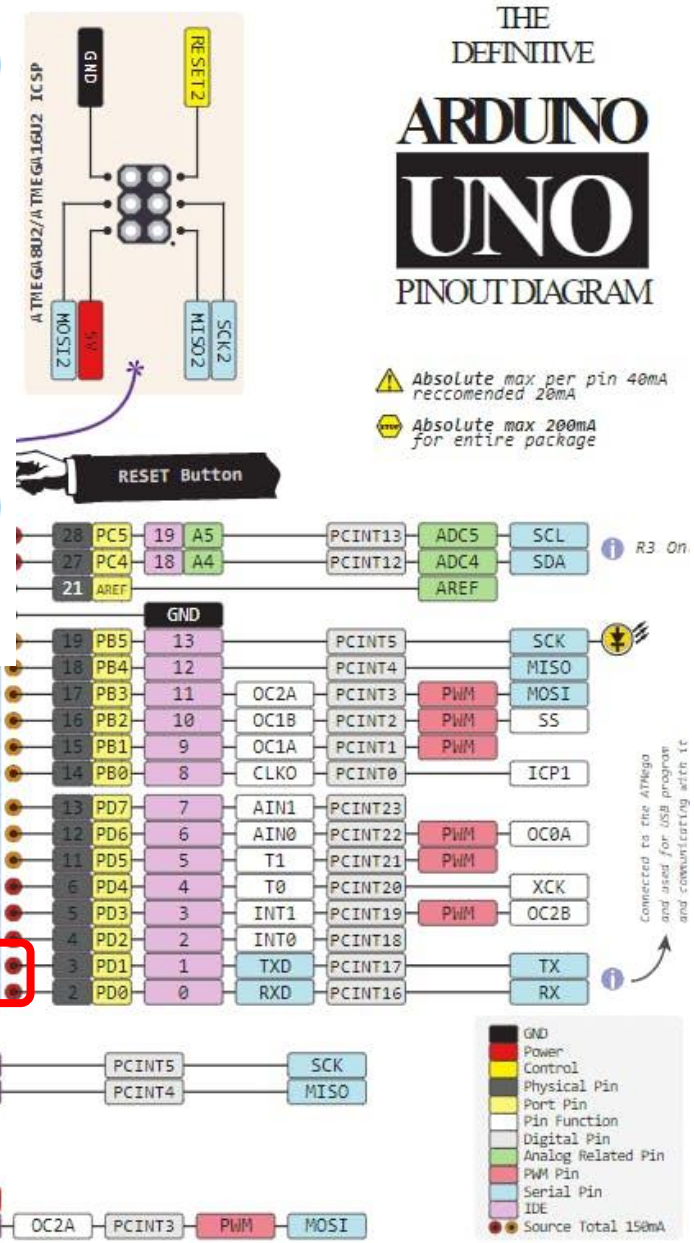
20.1 Features

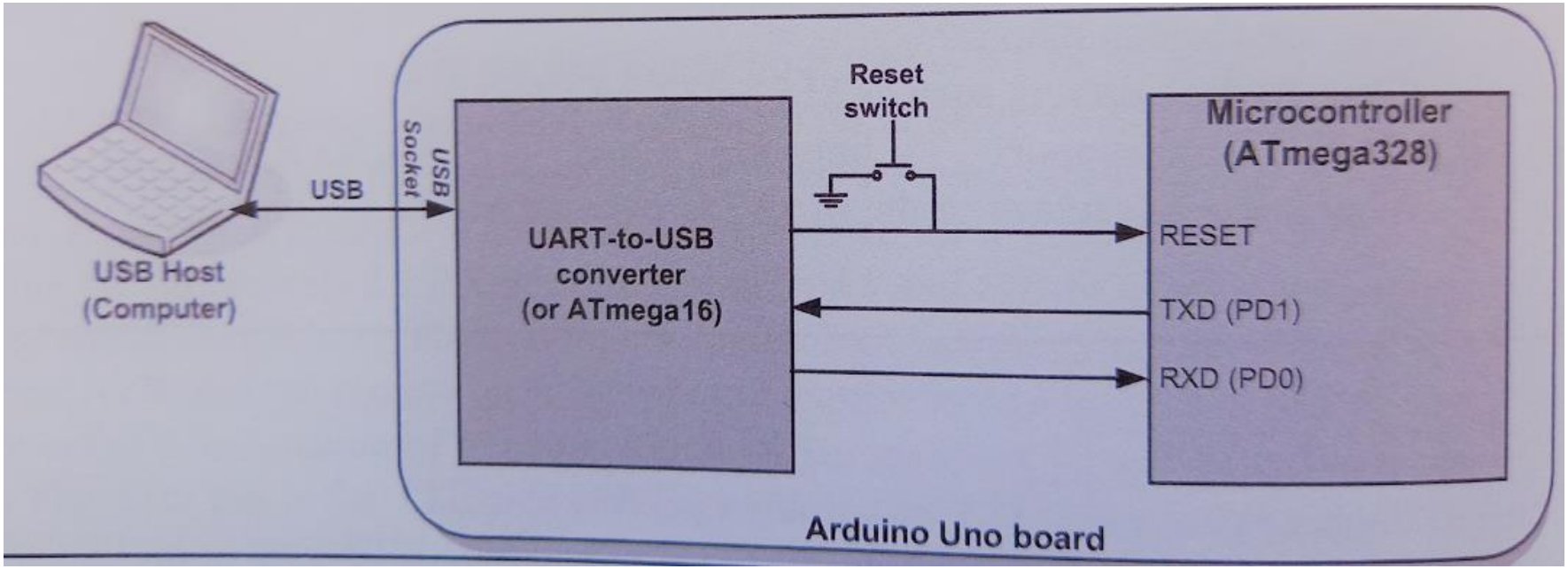
- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- Master or Slave Clocked Synchronous Operation
- High Resolution Baud Rate Generator
- Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Framing Error Detection
- Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter
- Three Separate Interrupts on TX Complete, TX Data Register Empty and RX Complete
- Multi-processor Communication Mode
- Double Speed Asynchronous Communication Mode

Atmega328


(PCINT14/RESET) PC6 1
(PCINT16/RXD) PD0 2
(PCINT17/TXD) PD1 3
(PCINT18/INT0) PD2 4
(PCINT19/OC2B/INT1) PD3 5
(PCINT20/XCK/T0) PD4 6
VCC 7
GND 8
(PCINT6/XTAL1/TOSC1) PB6 9
(PCINT7/XTAL2/TOSC2) PB7 10
(PCINT21/OC0B/T1) PD5 11
(PCINT22/OC0A/AIN0) PD6 12
(PCINT23/AIN1) PD7 13
(PCINT0/CLKO/ICP1) PB0 14
28 PC5 (ADC5/SCL/PCINT13)
27 PC4 (ADC4/SDA/PCINT12)
26 PC3 (ADC3/PCINT11)
25 PC2 (ADC2/PCINT10)
24 PC1 (ADC1/PCINT9)
23 PC0 (ADC0/PCINT8)
22 GND
21 AREF
20 AVCC
19 PB5 (SCK/PCINT5)
18 PB4 (MISO/PCINT4)
17 PB3 (MOSI/OC2A/PCINT3)
16 PB2 (SS/OC1B/PCINT2)
15 PB1 (OC1A/PCINT1)

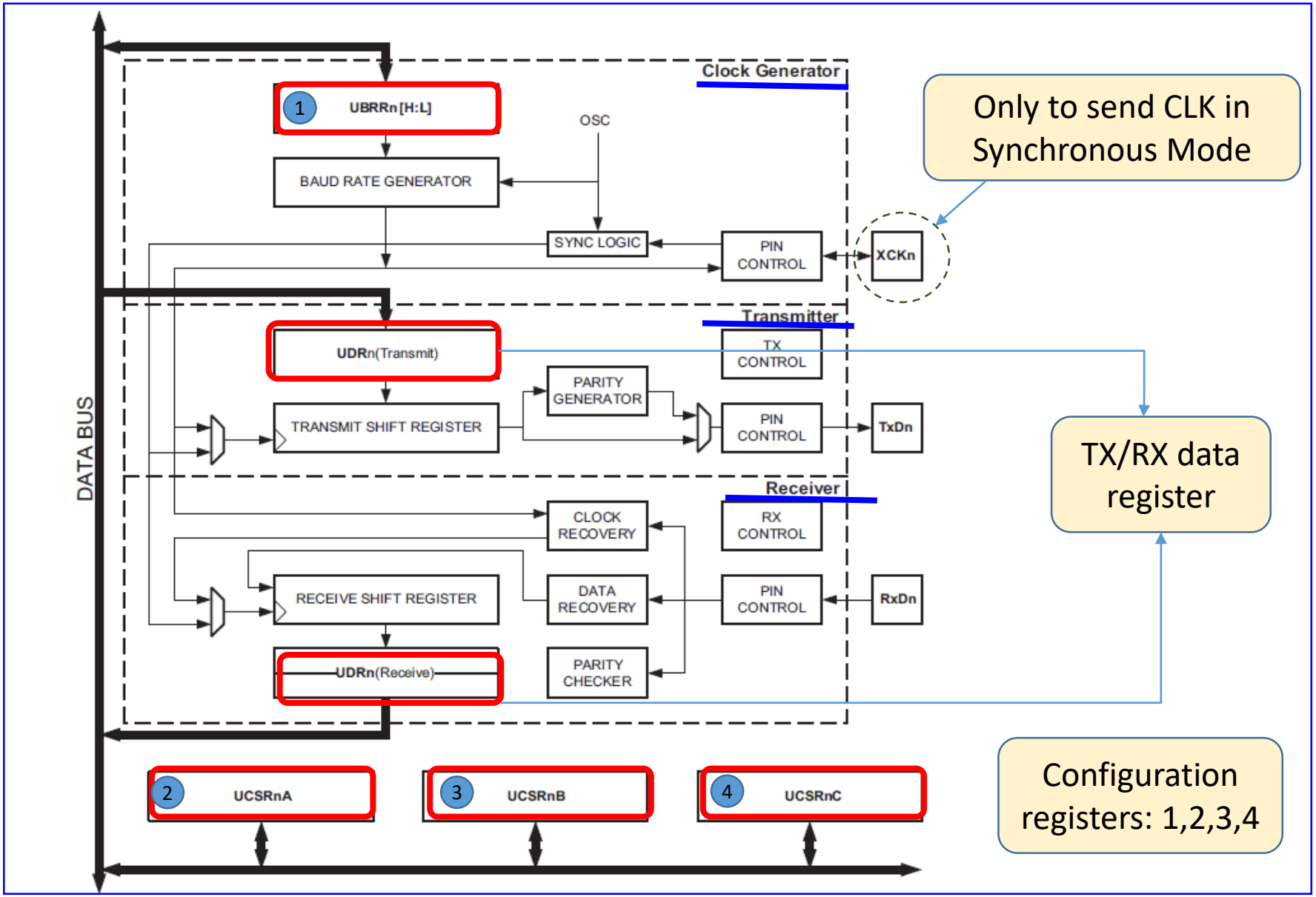
ATMEGA8U2/ATMEGA16U2 ICSP
GND
RESET2
MOSI2
5V
MISO2
SCK2


THE DEFINITIVE
ARDUINO
UNO
PINOUT DIAGRAM
Absolute max per pin 40mA
reccomended 20mA
Absolute max 200mA
for entire package
RESET Button
R3 On
The input voltage to the Arduino board when it is running from external power.
Not USB bus power.
Connected to the ATMega and used for USB program and communicating with it
www.pighixxx.com
18 FEB 2013
ver 2 rev 2 - 05.03.2013
GND
Power
Control
Physical Pin
Port Pin
Pin Function
Digital Pin
Analog Related Pin
PWM Pin
Serial Pin
IDE
Source Total 150mA



Source: “The AVR microcontroller and embedded system”. M.Ali Mazidi, S.Naimi

USART ATMega328: Block diagram



USART ATMega 328p: Initialization

USART ATMega328

1. Set the BAUDRATE (Register UBRR0)

UBRRnL and UBRRnH – USART Baud Rate Registers

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	UBRRn[11:8]				UBRRnH
	UBRRn[7:0]								UBRRnL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
	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	
	0	0	0	0	0	0	0	0	

- 12-bit register which contains the USART baud rate.

USART ATMega328

1. Set the BAUDRATE (Register UBRR0)

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	UBRRn[11:8]				UBRRnH
	UBRRn[7:0]								UBRRnL
	7	6	5	4	3	2	1	0	

CLK TX → 9.600 bps → 9.600 Hz

CLK RX → 16 samples for each bit → $\underbrace{9.600\text{Hz}}_{\text{BAUD}}$ * 16 = 153.600 Hz

To achieve 153.000 Hz from 16 MHz it must be divided by a **factor**…

Factor (UBRR) = 16.000.000Hz/153.000Hz = **104**

As the count start in 0, the value to be written in UBBR is 103.

Arduino Uno
Fosc = 16MHz

It is a frequency divider or prescaler

$$UBRRn = \frac{f_{OSC}}{16BAUD} - 1$$

USART ATMega328

Table 20-7. Examples of UBRRn Settings for Commonly Used Oscillator Frequencies (Continued)

Baud Rate (bps)	f_{osc} = 16.0000MHz				f_{osc} = 18.4320MHz				f_{osc} = 20.0000MHz			
	U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1		U2Xn = 0		U2Xn = 1	
	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error
2400	416	-0.1%	832	0.0%	479	0.0%	959	0.0%	520	0.0%	1041	0.0%
4800	207	0.2%	416	-0.1%	239	0.0%	479	0.0%	259	0.2%	520	0.0%
9600	103	0.2%	207	0.2%	119	0.0%	239	0.0%	129	0.2%	259	0.2%
14.4k	68	0.6%	138	-0.1%	79	0.0%	159	0.0%	86	-0.2%	173	-0.2%
19.2k	51	0.2%	103	0.2%	59	0.0%	119	0.0%	64	0.2%	129	0.2%
28.8k	34	-0.8%	68	0.6%	39	0.0%	79	0.0%	42	0.9%	86	-0.2%
38.4k	25	0.2%	51	0.2%	29	0.0%	59	0.0%	32	-1.4%	64	0.2%
57.6k	16	2.1%	34	-0.8%	19	0.0%	39	0.0%	21	-1.4%	42	0.9%
76.8k	12	0.2%	25	0.2%	14	0.0%	29	0.0%	15	1.7%	32	-1.4%
115.2k	8	-3.5%	16	2.1%	9	0.0%	19	0.0%	10	-1.4%	21	-1.4%
230.4k	3	8.5%	8	-3.5%	4	0.0%	9	0.0%	4	8.5%	10	-1.4%
250k	3	0.0%	7	0.0%	4	-7.8%	8	2.4%	4	0.0%	9	0.0%
0.5M	1	0.0%	3	0.0%	–	–	4	-7.8%	–	–	4	0.0%
1M	0	0.0%	1	0.0%	–	–	–	–	–	–	–	–
Max. (1)	1Mbps		2Mbps		1.152Mbps		2.304Mbps		1.25Mbps		2.5Mbps	

1. UBRRn = 0, Error = 0.0%

USART ATMega328

2. Enable receiver and transmitter (UCSR0B)

Bit	7	6	5	4	3	2	1	0	
	RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n	UCSRnB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Enable RX Interrupt

Enable TX Interrupt

Enable interrupt for Ready to send

Enable RX

Enable TX

Related to the register UCSR0C…

USART ATMega328

3. Set the mode (Register UCSR0C) and speed (Register UCSR0A)

- Decide mode:
  - Synchronous
  - Asynchronous:
    - Normal (Default)
    - Double speed.

Bit	7	6	5	4	3	2	1	0	
	RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn	UCSRnA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

Flag RX complete

Flag TX complete

Flag Ready to TX new data

Double speed (Async)

USART ATMega328

3. Modes (UCSR0C)

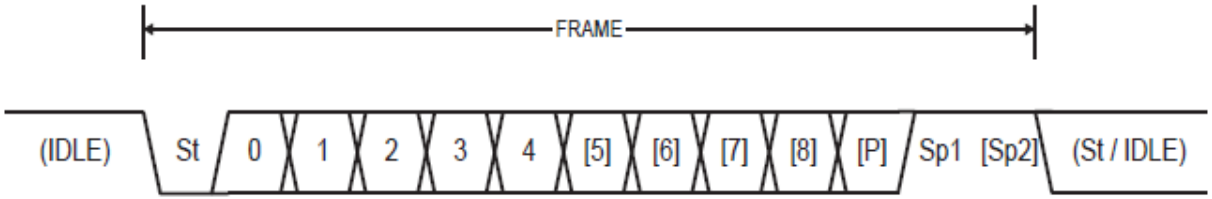
Bit	7	6	5	4	3	2	1	0	
	UMSELn1	UMSELn0	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn	UCSRnC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	1	1	0	

Table 20-8. UMSELn Bits Settings

UMSELn1	UMSELn0	Mode
0	0	Asynchronous USART
0	1	Synchronous USART
1	0	(Reserved)
1	1	Master SPI (MSPIM)[1]

4. Frame format (UCSR0C)

Figure 20-4. Frame Formats



St	Start bit, always low.
(n)	Data bits (0 to 8).
P	Parity bit. Can be odd or even.
Sp	Stop bit, always high.
IDLE	No transfers on the communication line (RxDn or TxDn). An IDLE line must be high.

4. Frame format (UCSR0C) - Parity

Bit	7	6	5	4	3	2	1	0	
	UMSELn1	UMSELn0	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn	UCSRnC
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	1	1	0	

Table 20-9. UPMn Bits Settings

UPMn1	UPMn0	Parity Mode
0	0	Disabled
0	1	Reserved
1	0	Enabled, Even Parity
1	1	Enabled, Odd Parity

4. Frame format (UCSR0C) – Data and bit-stop

Bit	7	6	5	4	3	2	1	0	
	UMSELn1	UMSELn0	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn	UCSRnC
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	1	1	0	

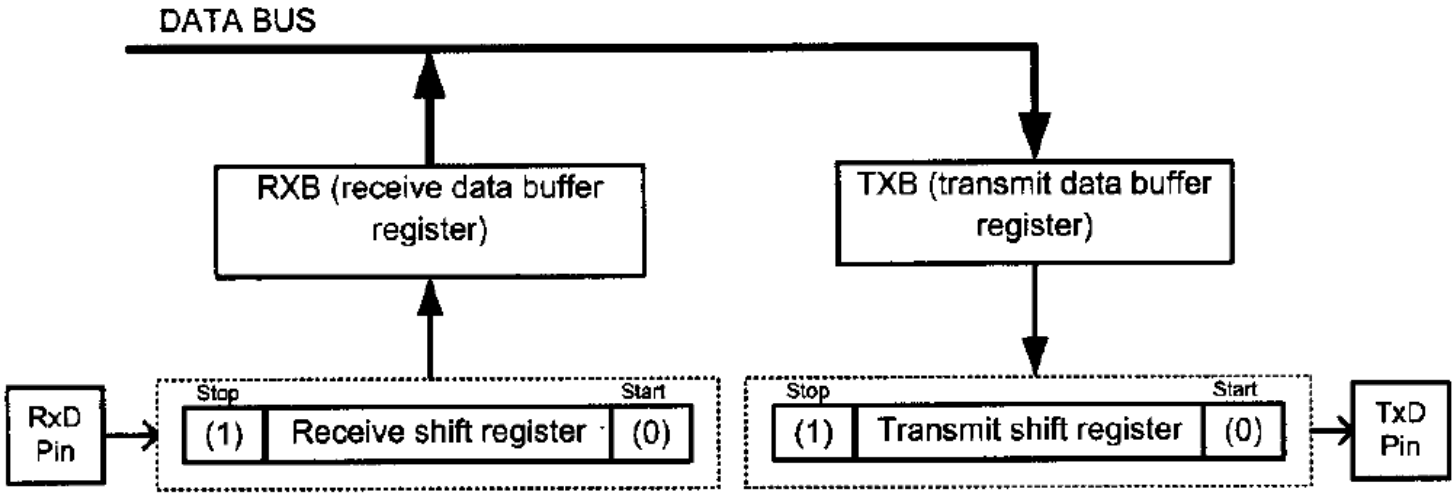
USBS	Stop Bit(s)
0	1-bit
1	2-bit

Table 20-11. UCSZn Bits Settings

Belongs to UCSR0B

UCSZn2	UCSZn1	UCSZn0	Character Size
0	0	0	5-bit
0	0	1	6-bit
0	1	0	7-bit
0	1	1	8-bit
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	9-bit

USART ATMega328: UDR Register



Source: The AVR microcontroller and embedded systems

USART ATMega328: Initialization

1.- Set baudrate (UBRR0H and UBRR0L)

103_{10}=0000	1100 1110
UBRR0-H	UBRR0-L

2.- Enable transmission (TXEN in UCSR0B)

3.- Set mode and speed (UCSR0A)

Asynchronous - normal

4.- Set data frame (UCSR0C)

9600 8N1

USART ATMega 328p: Data transmission by polling (TX)

USART ATMega328

TX/RX data register (UDR0)

Bit	7	6	5	4	3	2	1	0	
	RXB[7:0]								UDRn (Read)
	TXB[7:0]								UDRn (Write)
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	

- RX and TX data share the same register at the same address
- When the TX has finished (register empty) UDREn Flag in UCSRnA is set to ‘1’ (UCSR0A for ATMEGA328)

Bit	7	6	5	4	3	2	1	0	
	RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn	UCSRnA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

- If TX is enabled, the character written in the register is transmitted starting by the LSB.
- While the char is being transmited, UDREn Flag keeps to ‘0’.

USART ATMega328: Data transmission (Polling)

5.- If UDRE0='1' (UCSR0A) → Write data (UDR0)

Bit	7	6	5	4	3	2	1	0	
	RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn	UCSRnA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

```
void USART0_putchar (char data )
{
// 1.(Polling)Wait for empty transmit
register


// 2. Write "data" in register for sending a
character
}
```

USART ATMega 328p: Data reception by polling (RX)

USART ATMega328: Data reception (Polling)

Bit	7	6	5	4	3	2	1	0	
	RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn	UCSRnA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

RXC0 is asserted when the last bit of the character has been received in the UDR0 register. Once the register has been read, RXC0 is cleared.

```
char USART0_getchar( void )
{
// 1. Wait for data to be received
     -- Check if RXC0 bit is '1'
// 2. Get and return received data from
buffer
}
```

9600 8N1

USART ATMega 328p: Data transmission by interrupts (TX)

USART : Data transmission (Interrupt)

1.- Initialize USART

2.- Enable TX interrupt (UDRIE0 in UCSR0B)

Bit	7	6	5	4	3	2	1	0	
	RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n	UCSRnB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

3.- Enable global interrupts in main() → `sei ();`

```
void USART0_enaINT_TX()
{
// Enable transmission interrupt (UDRIE0),
        UCSR0B |= (1<<UDRIE0);
}
```

9600 8N1

n = 0
(ATmega328)

USART ATMega328: Data transmission (Interrupt-UDRE)

```
ISR (USART0_UDRE_vect)
{
    if (anything)
      UDR0 = data;
     …

    else
     <Disable Interrupt>
}
```

17	Timer/Counter0 Overflow	TIMER0_OVF_vect
18	SPI Serial Transfer Complete	SPI_STC_vect
19	USART Rx Complete	USART_RX_vect
20	USART Data Register Empty	USART_UDRE_vect

USART ATMega328: Data transmission (Interrupt-TXC)

```
ISR (USART0_TX_vect)
{
    if
      UDR0 = data;
     …

    else
     <Disable Interrupt>
}
```

19	USART Rx Complete	USART_RX_vect
20	USART Data Register Empty	USART_UDRE_vect
21	USART Tx Complete	USART_TX_vect

USART ATMega 328p: Data reception by interrupts (RX)

USART ATMega328: Data reception (Interrupt)

- Enable RX interrupt

Bit	7	6	5	4	3	2	1	0	
	RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n	UCSRnB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- Enable global interrupt → `sei();`

```
// Enable reception interrupt

ISR (USART0_RX_vect)
{
    uint_8 ReceivedByte;
    ReceivedByte = UDR0;
    // Echo back the received byte back to the computer
    UDR0 = ReceivedByte;
    // If TX is enabled, when UDR0 register is written,
    the character is sent.
}
```

USART ATMega328: More features…

- Parity
- Errors
- Synchronous
- Double speed
- Etc.

Datasheet (ATmega328)