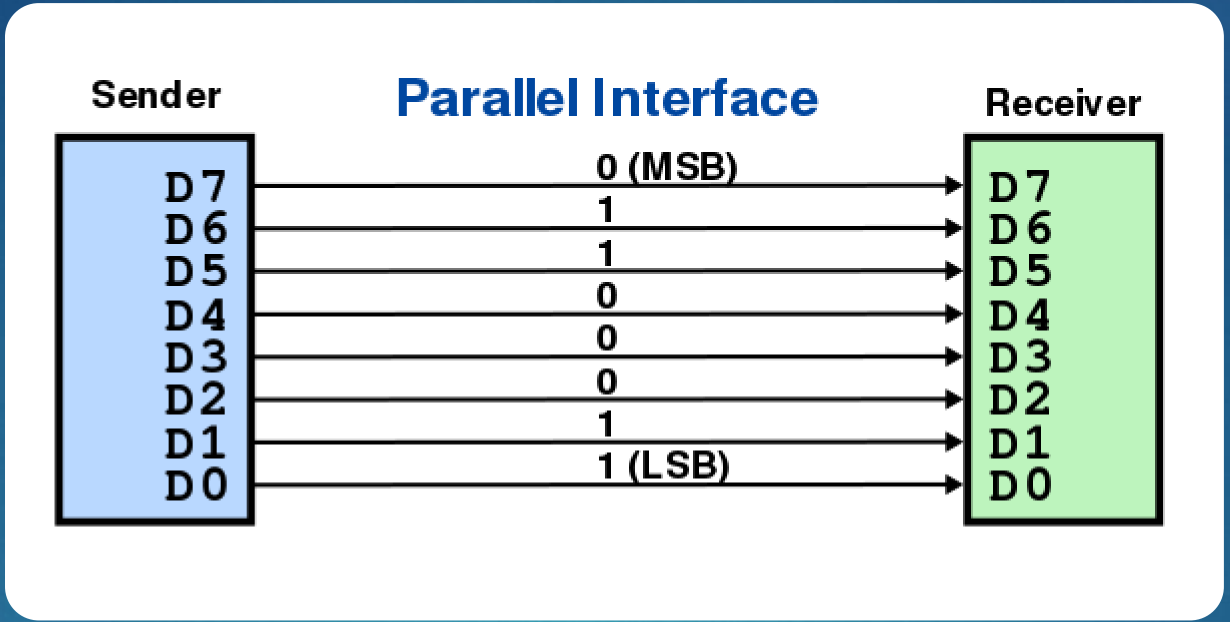
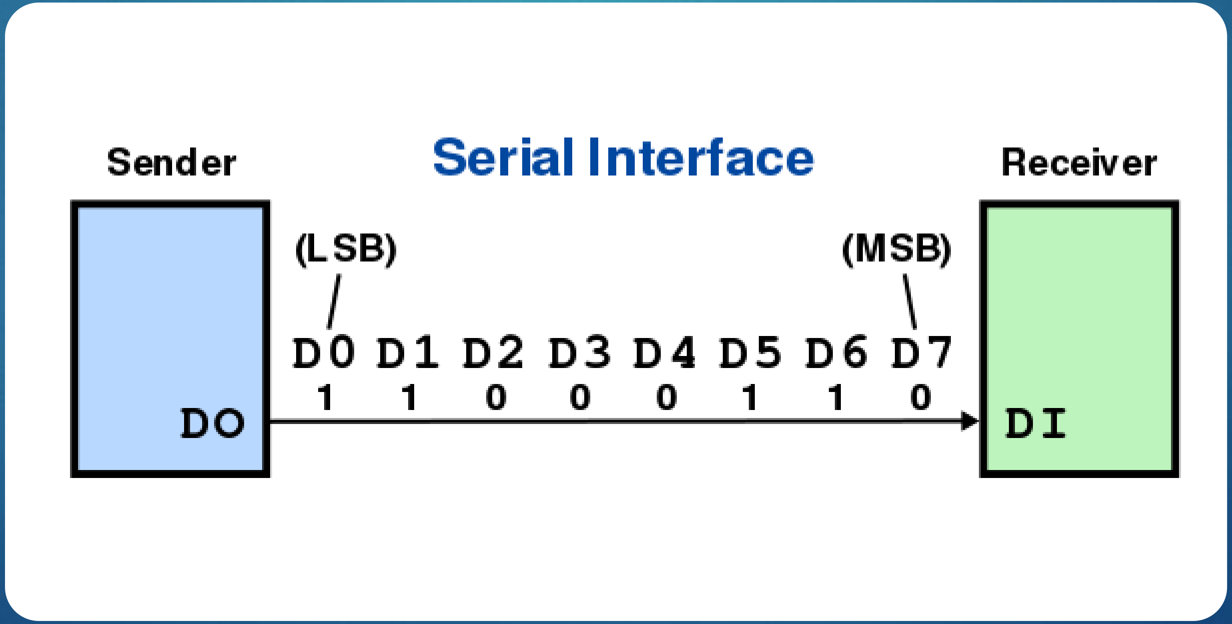
Libro página 186

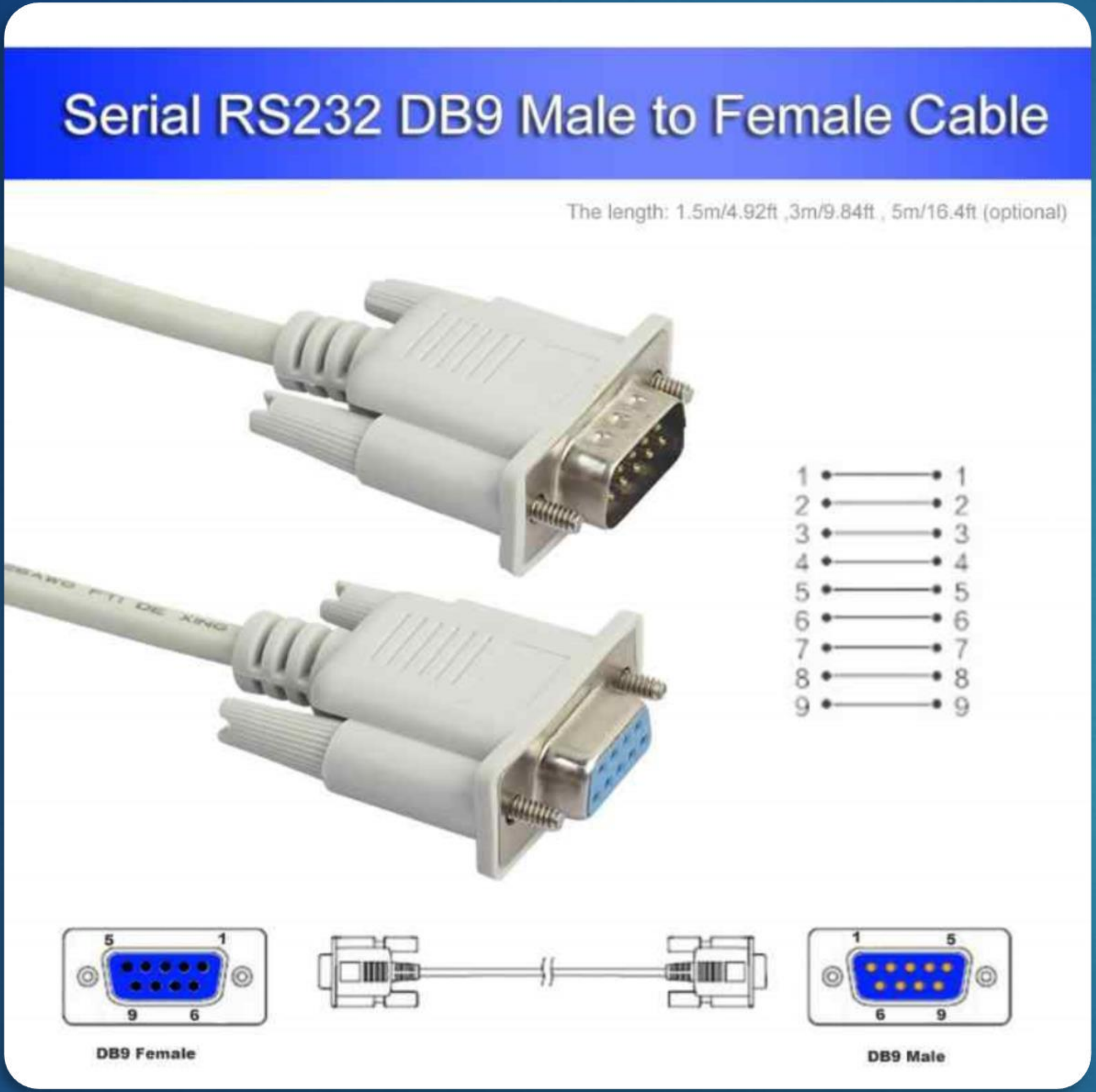
Fundamentos de Comunicación serial

Teresa Orvañanos Guerrero

01100011


Sender
Parallel Interface
Receiver
D7
D6
D5
D4
D3
D2
D1
D0
0 (MSB)
1
1
0
0
0
1
1 (LSB)
D7
D6
D5
D4
D3
D2
D1
D0


Sender
Serial Interface
Receiver
(LSB)
(MSB)
D0 D1 D2 D3 D4 D5 D6 D7
1 1 0 0 0 1 1 0
DO
DI

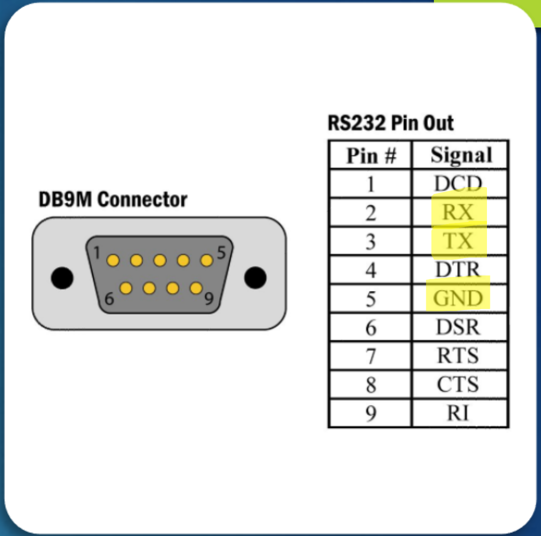


Serial RS232 DB9 Male to Female Cable

The length: 1.5m/4.92ft ,3m/9.84ft , 5m/16.4ft (optional)

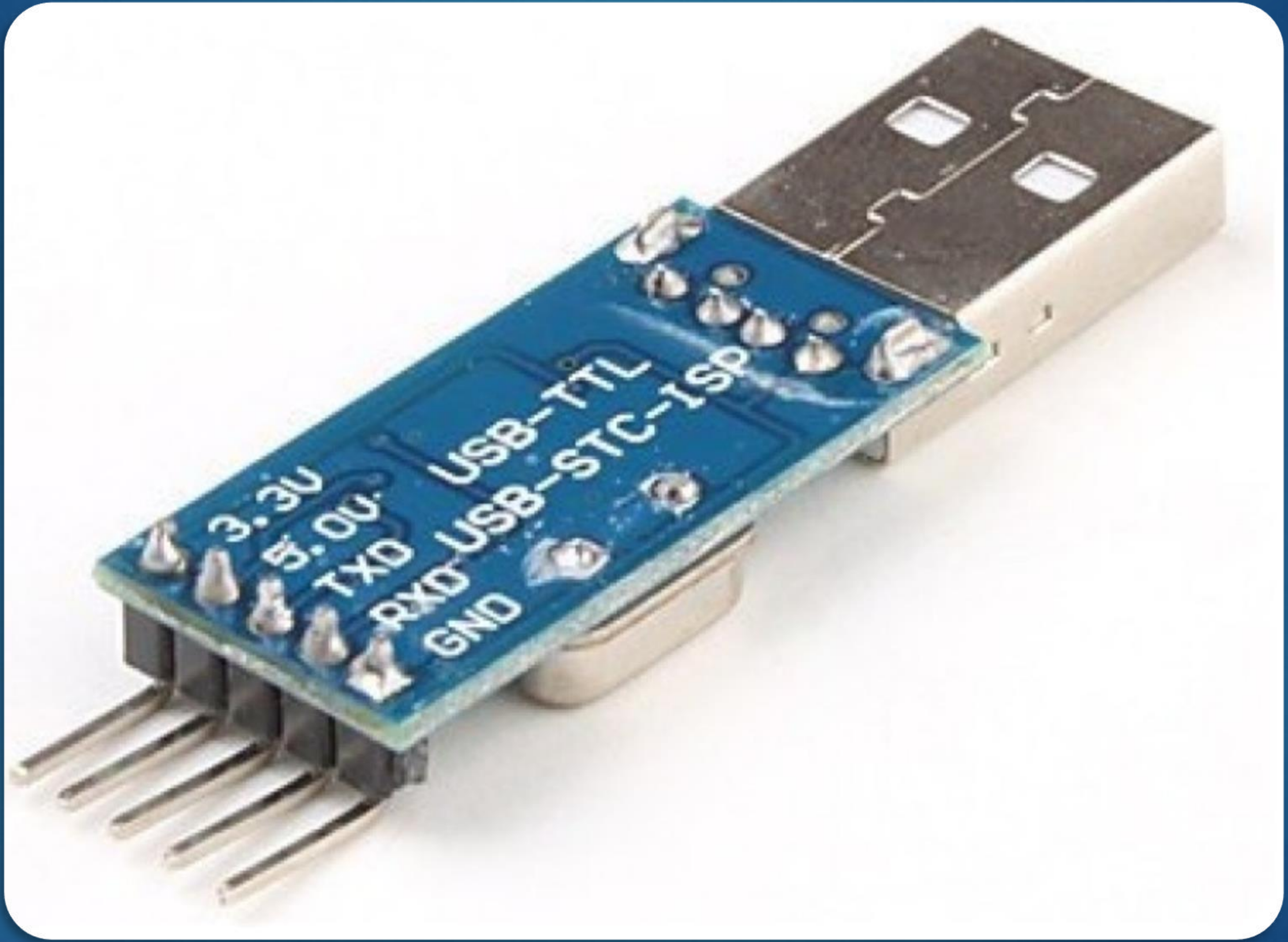
DB9 Female

DB9 Male



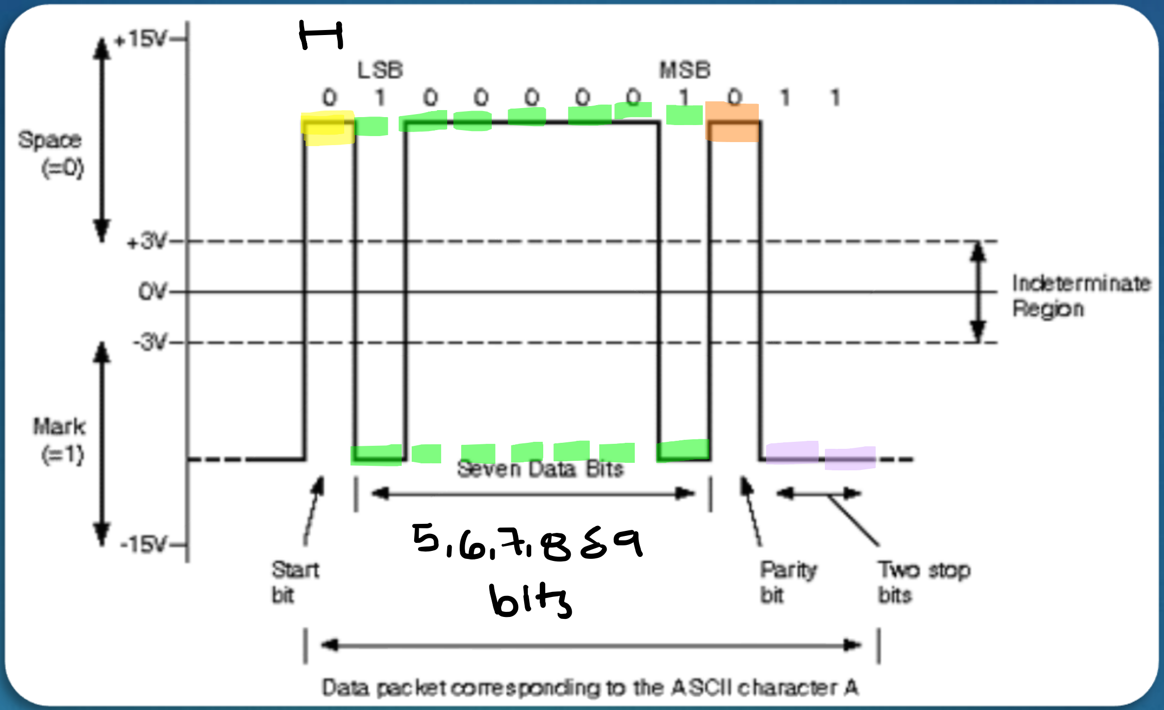
RS232 Pin Out

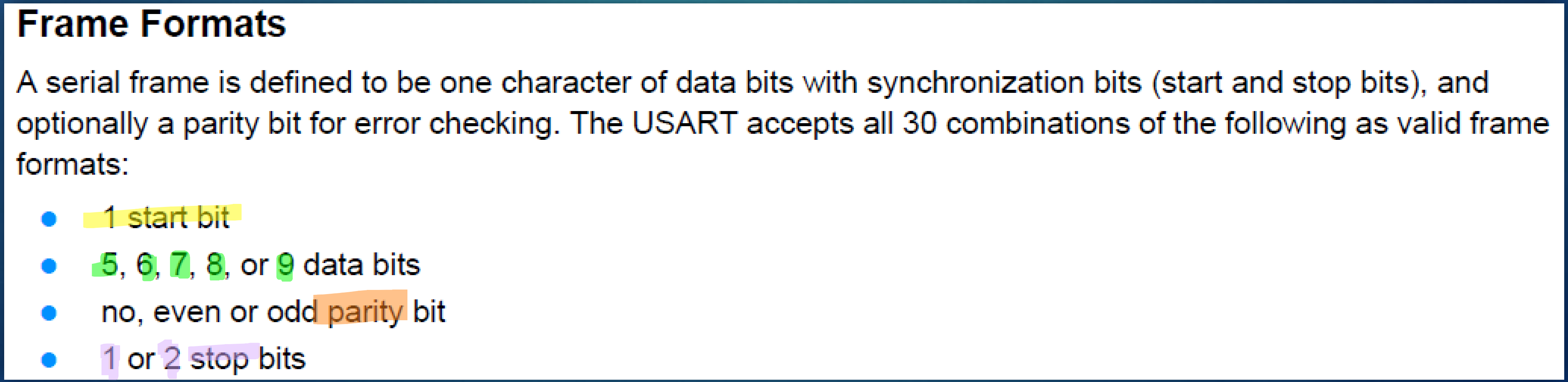
Pin #	Signal
1	DCD
2	RX
3	TX
4	DTR
5	GND
6	DSR
7	RTS
8	CTS
9	RI


USB-TTL
USB-STC-ISP
3.3V
5.0V
TXD
RXD
GND

frame
paquete

baud
rate

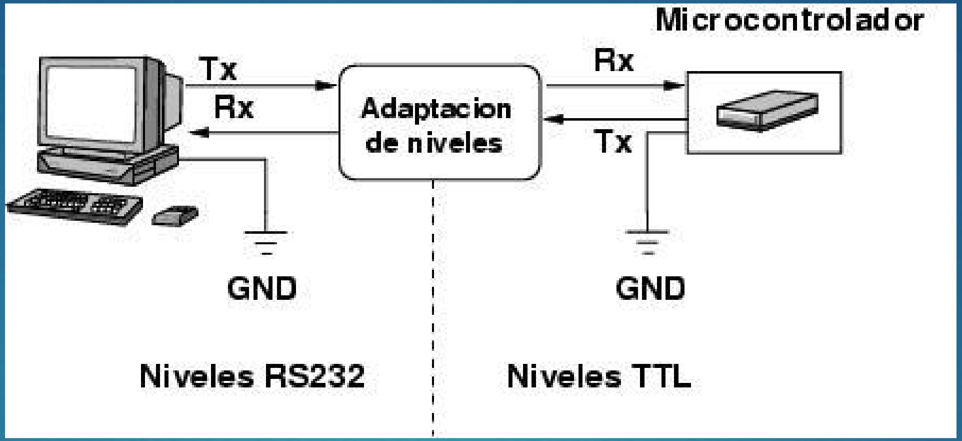


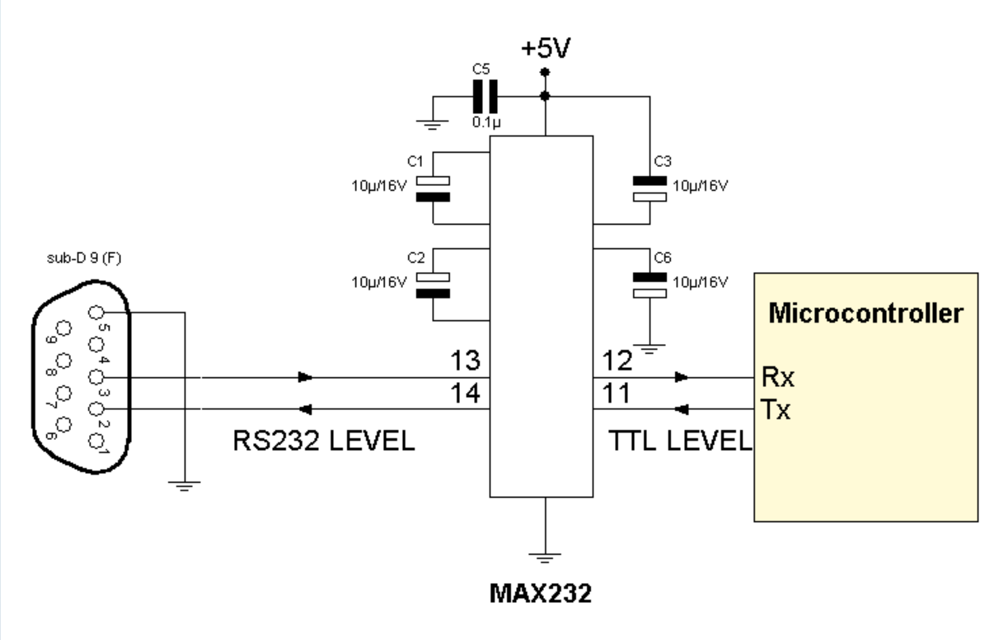


Frame Formats

A serial frame is defined to be one character of data bits with synchronization bits (start and stop bits), and optionally a parity bit for error checking. The USART accepts all 30 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


Microcontrolador
Tx
Rx
Adaptacion de niveles
Rx
Tx
GND
GND
Niveles RS232
Niveles TTL


+5V
C5
0.1µ
C1
10µ/16V
C3
10µ/16V
C2
10µ/16V
C6
10µ/16V
sub-D 9 (F)
5
4
3
2
1
9
8
7
6
13
14
12
11
RS232 LEVEL
TTL LEVEL
Microcontroller
Rx
Tx
MAX232

Datasheet página 140 / Libro página 187

USART (Universal Synchronous and Asynchronous serial receiver and transmitter)

Teresa Orvañanos Guerrero

El el ATmega16A el USART puede trabajar en modos:

⇨ - Normal asincrono

- Asincrono de doble vel

} Asíncrono

- síncrono maestro

- síncrono esclavo

} síncrono

UCSRC bit UMSEL

⇨ 0= com. asíncrona

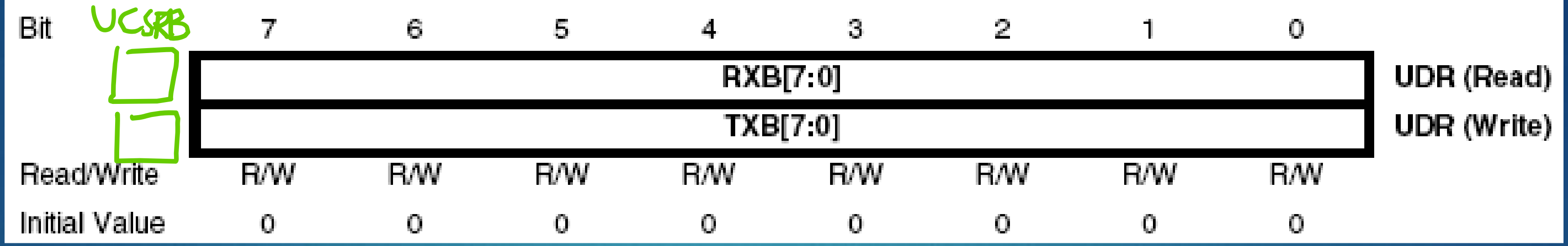
1= síncrona

dato → transmicion

dato → recepción

```
UDR = 0b10101010 ;
uint8_t dato = UDR;
```

UDR



Bit	UCSRB	7	6	5	4	3	2	1	0	
	□	RXB[7:0]								UDR (Read)
	□	TXB[7:0]								UDR (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	

Buffer de transmisión o recepción

```
X while (cero_en_bit(&UDR, 2)) {} //NOOOOO
```

!!! Primero leer UDR en variable

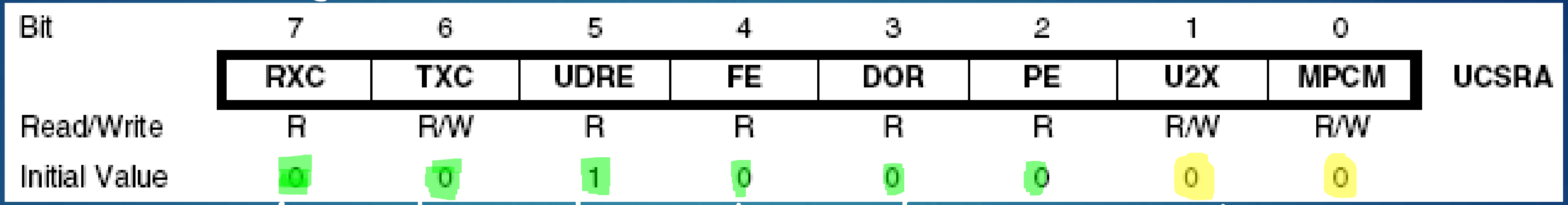
```
!  UDR |= (1<<2);
   UDR |= (1<<3);
```

UCSRA

control int.: flag flag flag

indican error

siempre ∅



Bit	7	6	5	4	3	2	1	0	
	RXC	TXC	UDRE	FE	DOR	PE	U2X	MPCM	UCSRA
Read/Write	R	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	1	0	0	0	0	0	

! RXC

1 – Hay datos sin leer

∅ – El buffer vacío (lectura)

TXC

Terminó

1 – Ya se transmit. todos los datos

∅ – "Limpio" en automático cuando entra a la int. correspon.

UDRE

∅ – No listo para transmisión

1 – Buffer vacío listo para transm.

FE

∅ – Ponerlo! *

1 – Hay un error en el frame

*válido antes de leer al UDR

DOR

∅ – Ponerlo! *

1 – Hay un dato over run (recepción)

*válido antes de leer UDR

PE

∅ – Ponerlo *

1 – Hay error de paridad

*valido antes de leer al UDR

U2X

si síncrona = ∅

si asíncrona

∅ – normal

1 – doble vel.

MPCM

∅ – normal

1 – ignora datos recibidos sin info de la dir

* Errores en Recepción

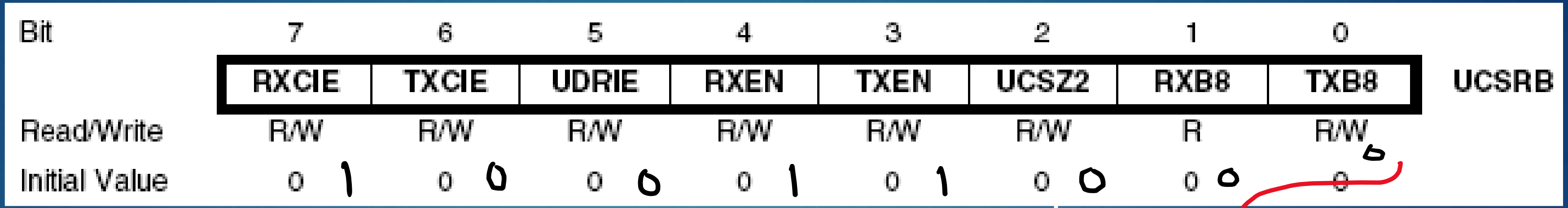
UCSRB

Interruption enable

~EN Enable

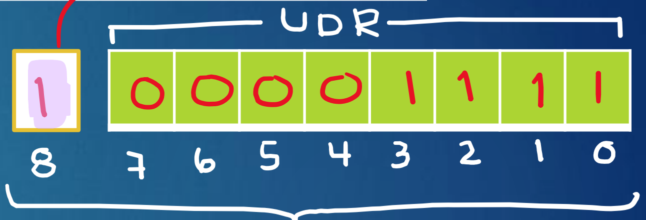
#bits

el 9no bit de UDR



Bit	7	6	5	4	3	2	1	0	
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	UCSZ2	RXB8	TXB8	UCSRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0 1	0 0	0 0	0 1	0 1	0 0	0 0	~~0~~ 0	

UDR = 0b00000111



¡ 9 bits en total !

Leer o escribir ANTES de leer o escribir al UDR

RXCIE 1 – Int. cuando reciba un dato (!!!)
0 – No int

TXCIE 1 – Int. cuando terminó la trans. ←
0 – No int

UDRIE – 1 – Int. cuando listo para trans. ←
0 – No. int

RXEN — 0 – Deshabilita Recepción
1 – Habilitar Recepción

TX EN — 0 – Deshabilita Transmision
1 – Habilitar transm...

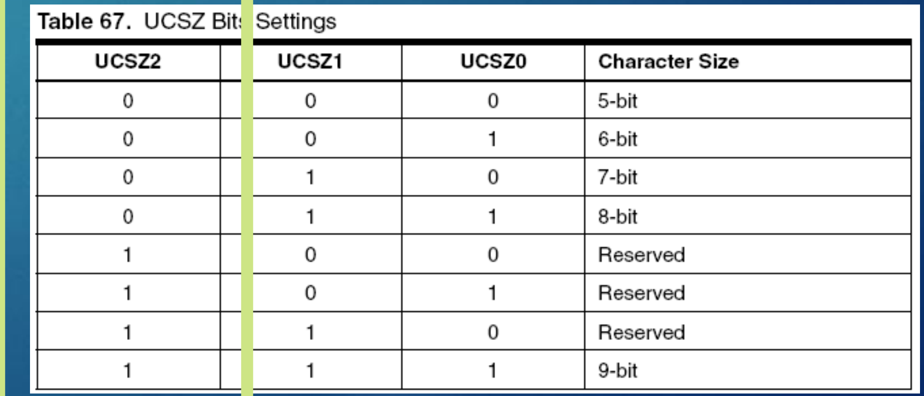


Table 67. UCSZ Bits Settings

UCSZ2	UCSZ1	UCSZ0	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

¿ configuré p/ 9 bits *

TRANSMITIR 9 BITS

$$\left(0b\ \overset{\text{UCSRB en bit } \emptyset\ ①}{\underset{0}{1}}\ \underbrace{00001111}_{\text{UDR } ②}\right)$$

- ① UCSRB |= (1<< ∅); *
- ② UDR = 0b00001111;

UCSRB |= (0<<0); X MAL

* $\boxed{\text{UCSRB \&= \textasciitilde(1<<0);}}$ ✓

RECIBIR 9 BITS

$$\left(0b\ \overset{\text{UCSRB en bit1 } ①}{\underset{0}{1}}\ \underbrace{0000\,1111}_{\text{UDR } ②}\right)$$

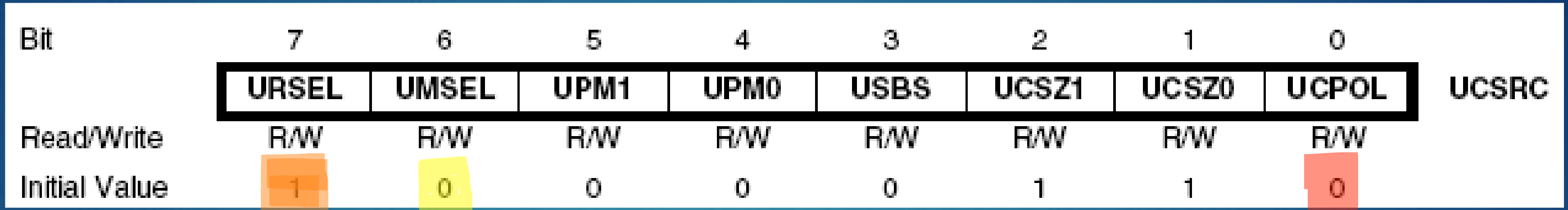
```
uint16_t dato=0;
 if (uno_en_bit(&UCSRB, 1)) { dato |= (1<<8) }

uint8_t temp;
 temp = UDR;
 dato |= temp;
```

$$
\begin{array}{rl}
\text{dato} = & \overset{0}{1}\ 0000\ 0000 \\
 & \ 0000\ 1111 \\
\hline
 & \underset{0}{1}\ 0000\ 1111
\end{array}
$$

tamano dato

en com sincrona . . . .

UcSRc



Bit	7	6	5	4	3	2	1	0	
	URSEL	UMSEL	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL	UCSRC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	1	0	0	0	0	1	1	0	

1 1

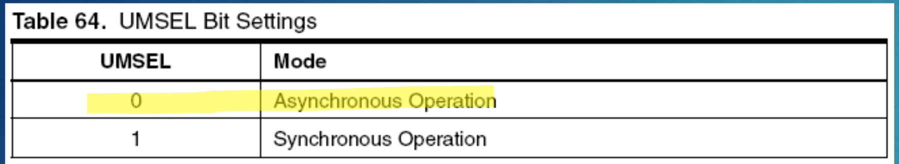


Table 64. UMSEL Bit Settings

UMSEL	Mode
0	Asynchronous Operation
1	Synchronous Operation

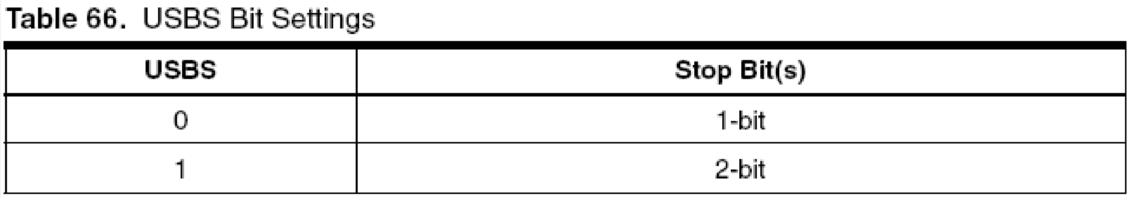


Table 66. USBS Bit Settings

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

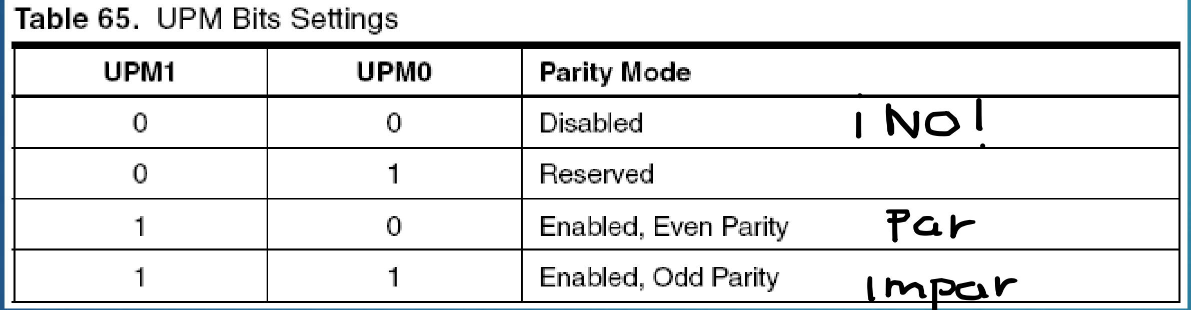


Table 65. UPM Bits Settings

UPM1	UPM0	Parity Mode	
0	0	Disabled	¡NO!
0	1	Reserved	
1	0	Enabled, Even Parity	par
1	1	Enabled, Odd Parity	impar

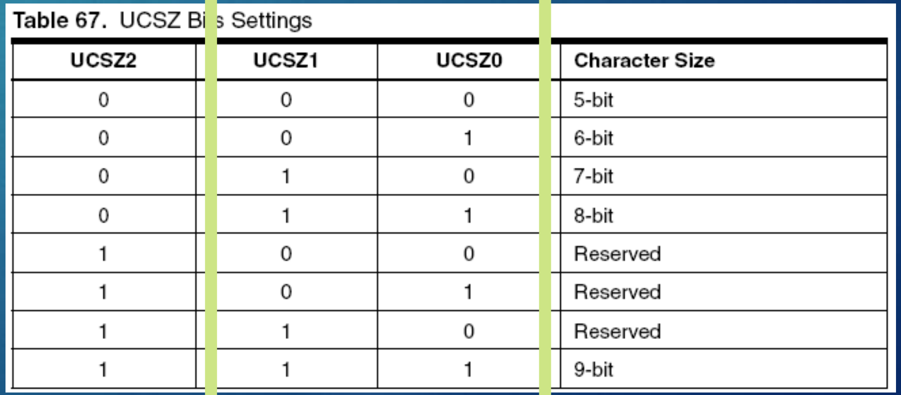
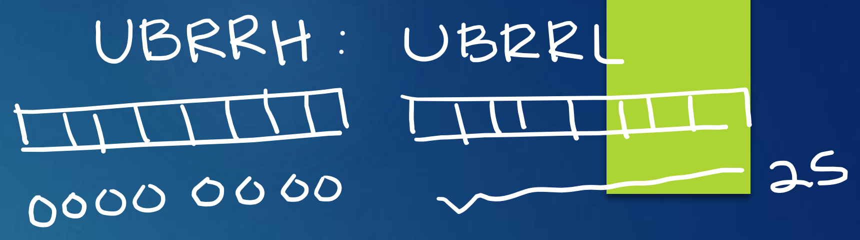


Table 67. UCSZ Bits Settings

UCSZ2	UCSZ1	UCSZ0	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

UBRR 16bits → UBRRH : UBRRL

para velocidad de transmisión



UBRR=12;

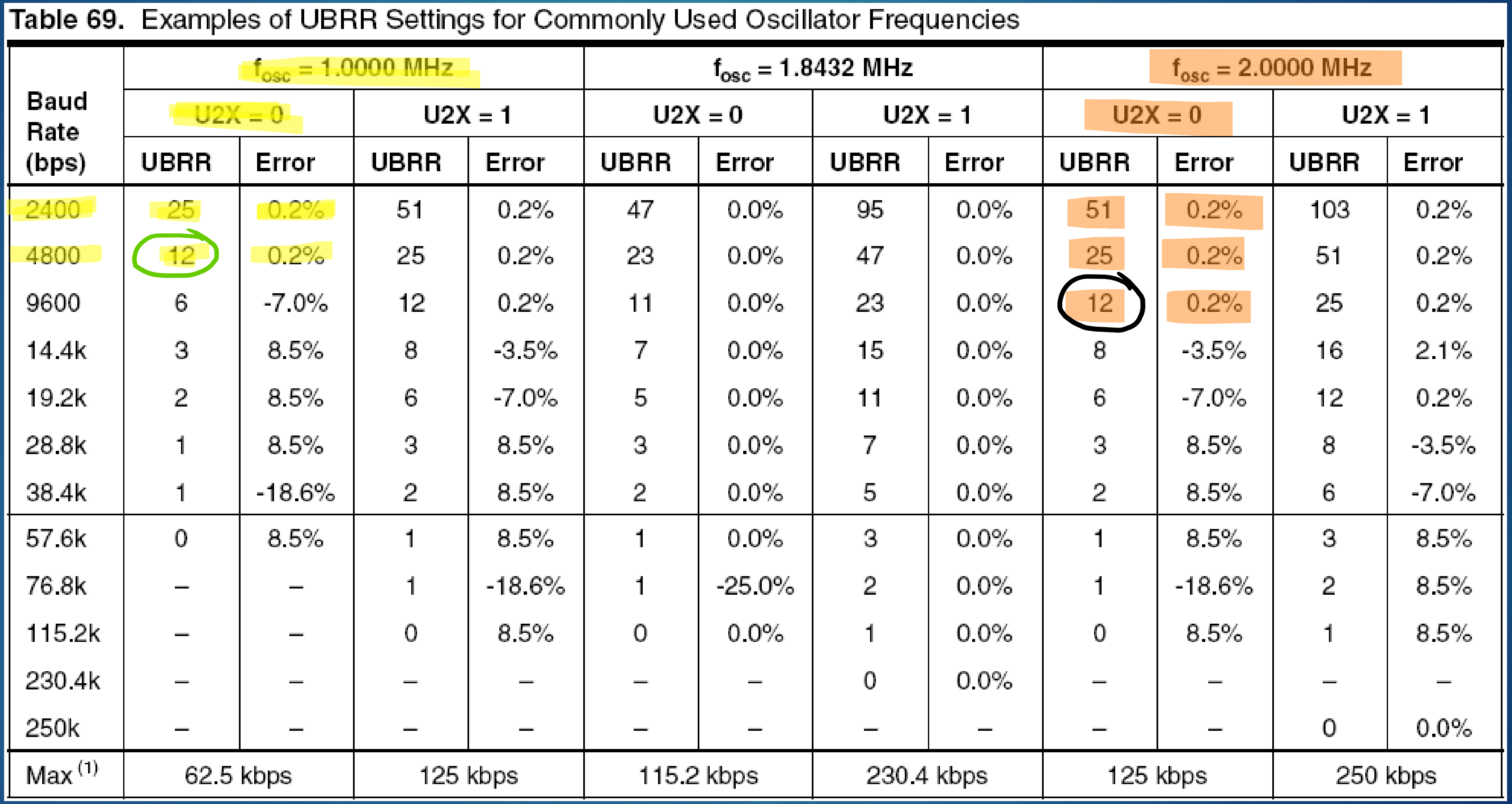


Table 69. Examples of UBRR Settings for Commonly Used Oscillator Frequencies

Baud Rate (bps)	f_{osc} = 1.0000 MHz				f_{osc} = 1.8432 MHz				f_{osc} = 2.0000 MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	25	0.2%	51	0.2%	47	0.0%	95	0.0%	51	0.2%	103	0.2%
4800	12	0.2%	25	0.2%	23	0.0%	47	0.0%	25	0.2%	51	0.2%
9600	6	-7.0%	12	0.2%	11	0.0%	23	0.0%	12	0.2%	25	0.2%
14.4k	3	8.5%	8	-3.5%	7	0.0%	15	0.0%	8	-3.5%	16	2.1%
19.2k	2	8.5%	6	-7.0%	5	0.0%	11	0.0%	6	-7.0%	12	0.2%
28.8k	1	8.5%	3	8.5%	3	0.0%	7	0.0%	3	8.5%	8	-3.5%
38.4k	1	-18.6%	2	8.5%	2	0.0%	5	0.0%	2	8.5%	6	-7.0%
57.6k	0	8.5%	1	8.5%	1	0.0%	3	0.0%	1	8.5%	3	8.5%
76.8k	–	–	1	-18.6%	1	-25.0%	2	0.0%	1	-18.6%	2	8.5%
115.2k	–	–	0	8.5%	0	0.0%	1	0.0%	0	8.5%	1	8.5%
230.4k	–	–	–	–	–	–	0	0.0%	–	–	–	–
250k	–	–	–	–	–	–	–	–	–	–	0	0.0%
Max [(1)]	62.5 kbps		125 kbps		115.2 kbps		230.4 kbps		125 kbps		250 kbps	

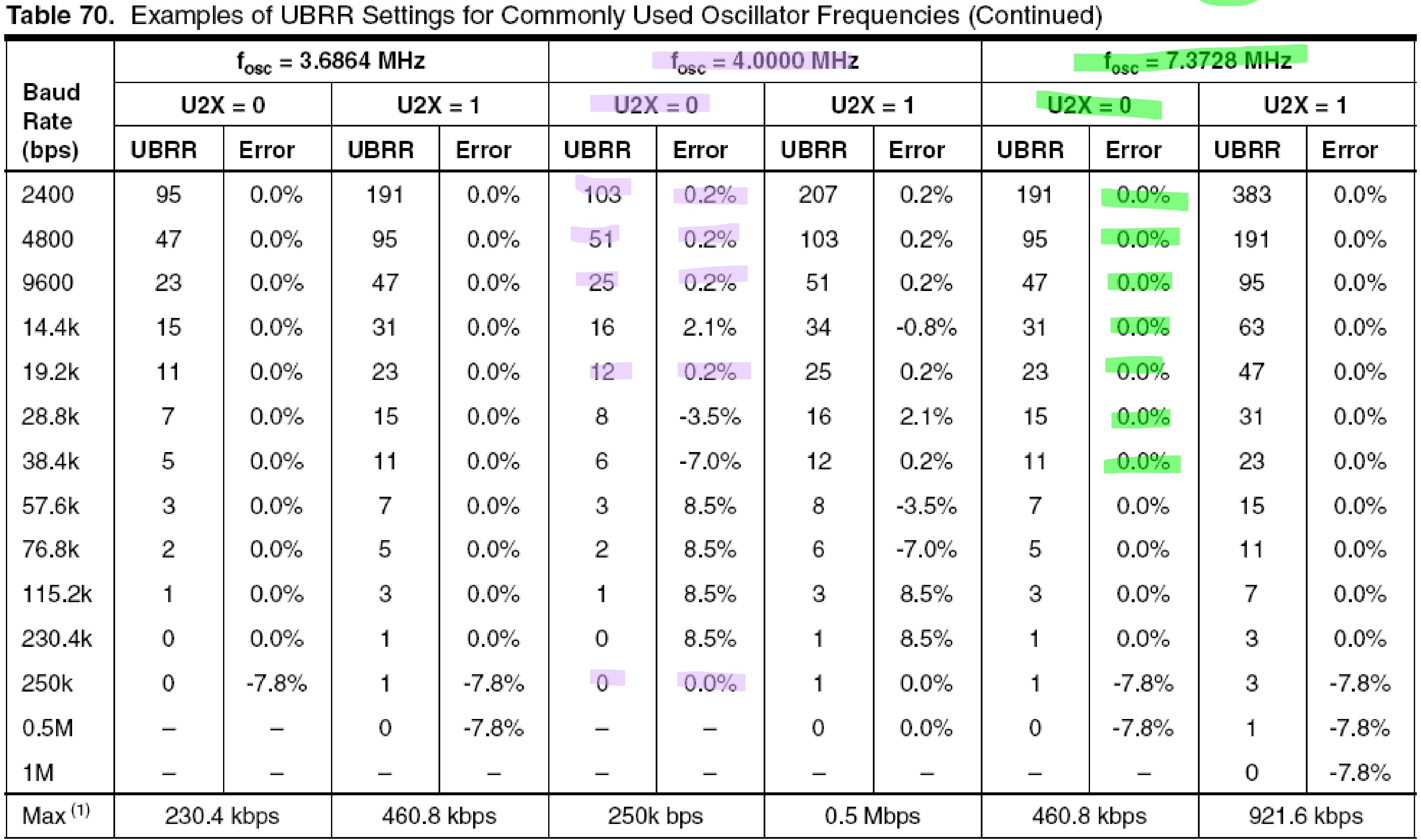


Table 70. Examples of UBRR Settings for Commonly Used Oscillator Frequencies (Continued)

Baud Rate (bps)	f_{osc} = 3.6864 MHz				f_{osc} = 4.0000 MHz				f_{osc} = 7.3728 MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error	UBRR	Error
2400	95	0.0%	191	0.0%	103	0.2%	207	0.2%	191	0.0%	383	0.0%
4800	47	0.0%	95	0.0%	51	0.2%	103	0.2%	95	0.0%	191	0.0%
9600	23	0.0%	47	0.0%	25	0.2%	51	0.2%	47	0.0%	95	0.0%
14.4k	15	0.0%	31	0.0%	16	2.1%	34	-0.8%	31	0.0%	63	0.0%
19.2k	11	0.0%	23	0.0%	12	0.2%	25	0.2%	23	0.0%	47	0.0%
28.8k	7	0.0%	15	0.0%	8	-3.5%	16	2.1%	15	0.0%	31	0.0%
38.4k	5	0.0%	11	0.0%	6	-7.0%	12	0.2%	11	0.0%	23	0.0%
57.6k	3	0.0%	7	0.0%	3	8.5%	8	-3.5%	7	0.0%	15	0.0%
76.8k	2	0.0%	5	0.0%	2	8.5%	6	-7.0%	5	0.0%	11	0.0%
115.2k	1	0.0%	3	0.0%	1	8.5%	3	8.5%	3	0.0%	7	0.0%
230.4k	0	0.0%	1	0.0%	0	8.5%	1	8.5%	1	0.0%	3	0.0%
250k	0	-7.8%	1	-7.8%	0	0.0%	1	0.0%	1	-7.8%	3	-7.8%
0.5M	–	–	0	-7.8%	–	–	0	0.0%	0	-7.8%	1	-7.8%
1M	–	–	–	–	–	–	–	–	–	–	0	-7.8%
Max [1]	230.4 kbps		460.8 kbps		250k bps		0.5 Mbps		460.8 kbps		921.6 kbps	

UDR dato

UCSRA flags / Errores recep /

} no config

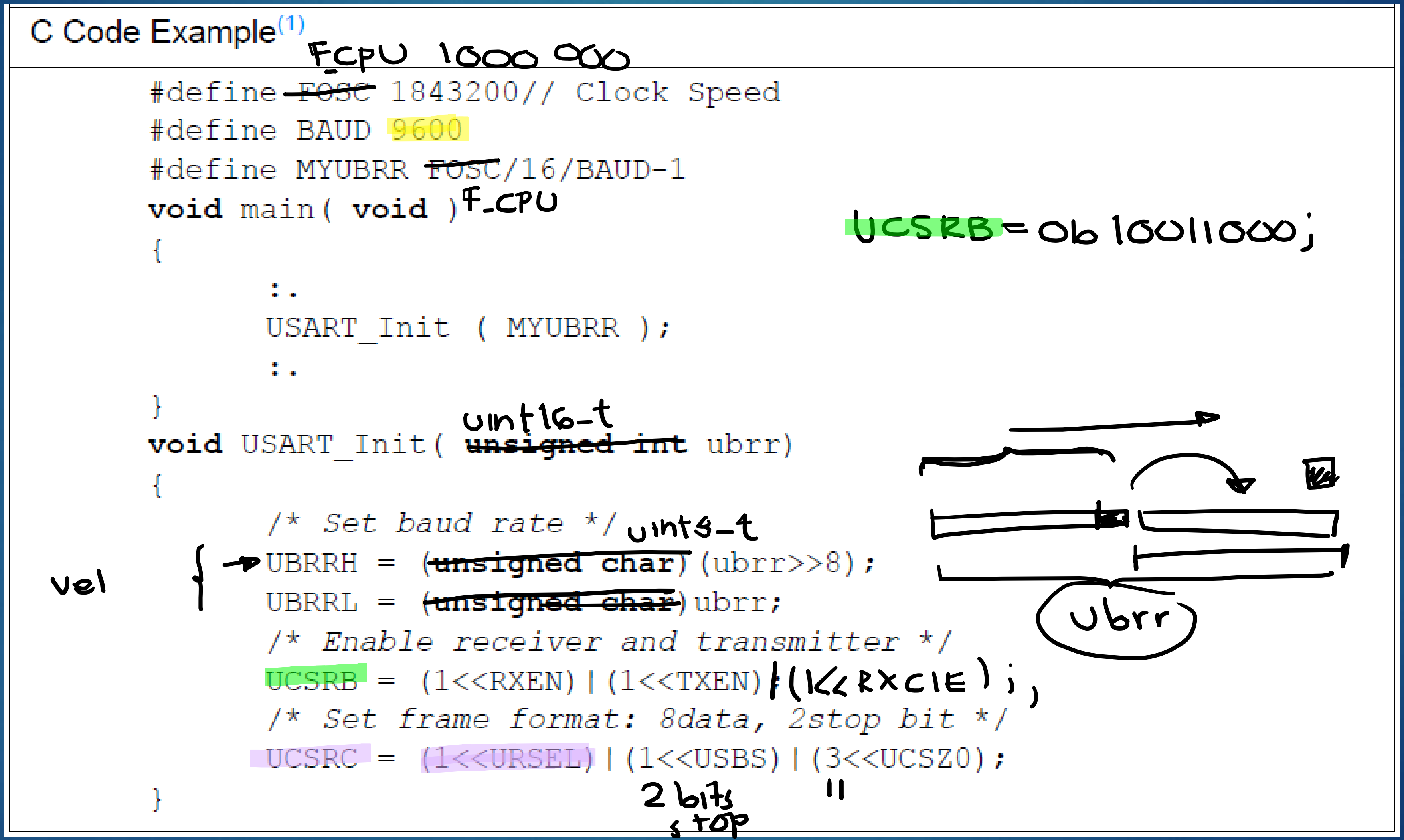
- - - - -

UCSRB Interrup / transm y/o recep / #bits / 9° bit ←

UCSRC paridad / bits parada / #bits ←

✓ UBRR (UBRRH:UBRRL) baud rate ←

INICIALIZACIÓN / CONFIGURACION



C Code Example(1)

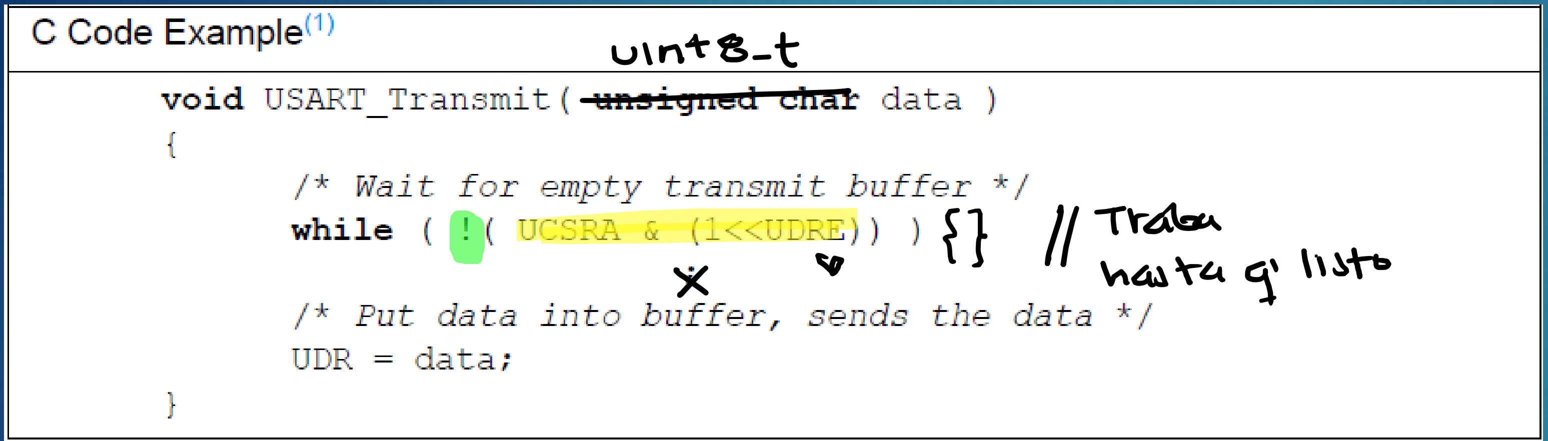
```
                FCPU 1000 000
#define ~~FOSC~~ 1843200// Clock Speed
#define BAUD 9600
#define MYUBRR ~~FOSC~~/16/BAUD-1
void main( void )F_CPU
{
      :.
      USART_Init ( MYUBRR );
      :.
}
                 uint16_t
void USART_Init( ~~unsigned int~~ ubrr)
{
      /* Set baud rate */ uint8_t
vel {→UBRRH = (~~unsigned char~~)(ubrr>>8);
    { UBRRL = (~~unsigned char~~)ubrr;
      /* Enable receiver and transmitter */
      UCSRB = (1<<RXEN)|(1<<TXEN)|(1<<RXCIE);,
      /* Set frame format: 8data, 2stop bit */
      UCSRC = (1<<URSEL)|(1<<USBS)|(3<<UCSZ0);
                          2 bits      ||
                          stop
}
```

UCSRB=0b10011000;

```
void main (void) {
      ≡
      USART_Transmit (5);
      ≡
}
```

falsø = Ø

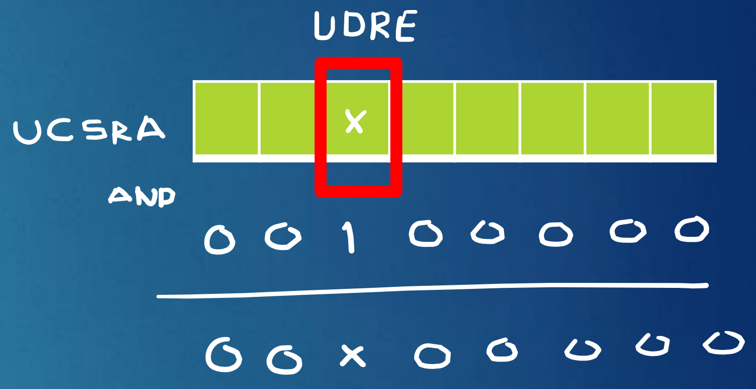
verdad = no cero



C Code Example[1]

uint8_t

```
void USART_Transmit( ~~unsigned char~~ data )
{
      /* Wait for empty transmit buffer */
      while ( !( UCSRA & (1<<UDRE)) ) {}   // Traba hasta q' listo
                    X          ♡

      /* Put data into buffer, sends the data */
      UDR = data;
}
```



si UDRE Ø ∴ ♡ = 0
NO LISTO
!(falso)
verdad

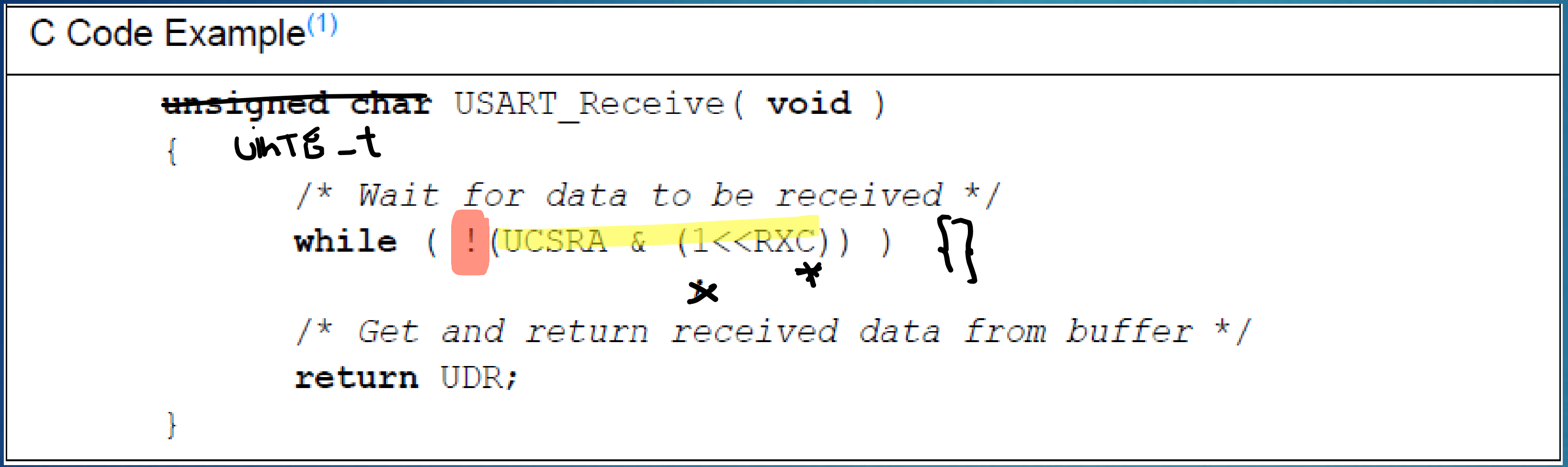
si UDRE=1 ∴ ♡ = 1
LISTO
!(verdadero)
falso

RECiBiR

. . . || traba

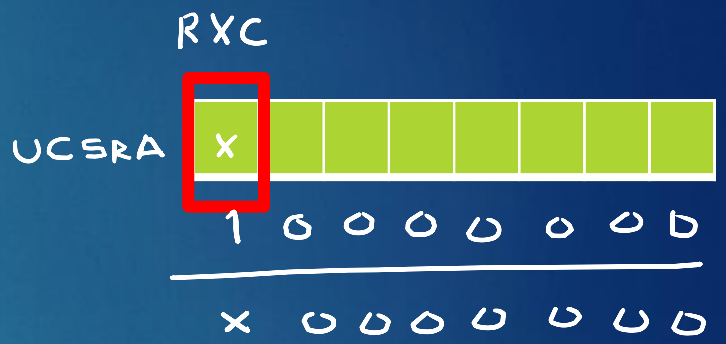
Vs utilizar interrupción RXCIE

∅ - falso
no ∅ - verdad



C Code Example[(1)]

```
~~unsigned char~~ USART_Receive( void )
{   uint8_t
        /* Wait for data to be received */
        while ( !(UCSRA & (1<<RXC)) )  {}
                               *      *

        /* Get and return received data from buffer */
        return UDR;
}
```



si RXC = 0 ∴ * = 0

↗ NO DATOS

!(falso)

verdad

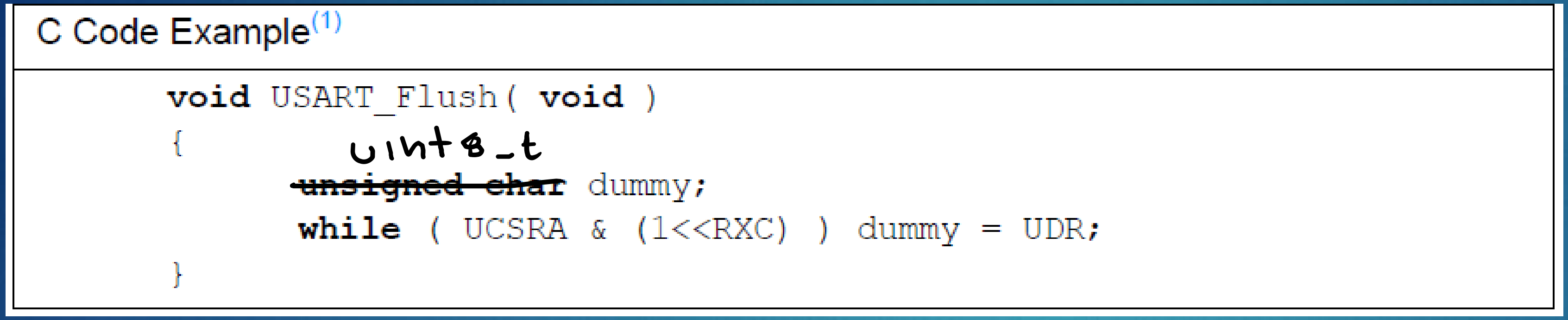
si RXC = 1 ∴ * = 1

si DATOS

!(verdad)

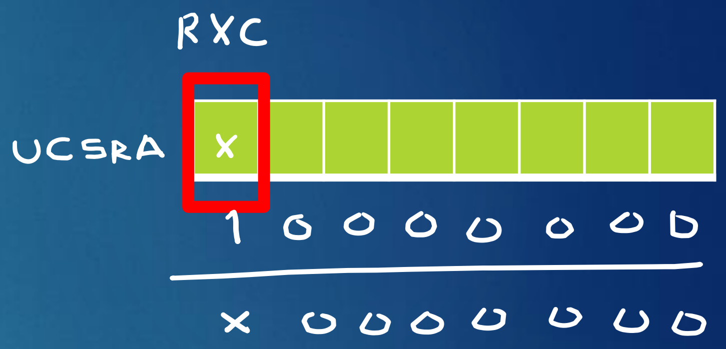
falso

USART_Flush();



C Code Example[1]

```
void USART_Flush( void )
{
            uint8_t
      ~~unsigned char~~ dummy;
      while ( UCSRA & (1<<RXC) ) dummy = UDR;
}
```



si RXC=0 ∴ *=0 (falso)

↗ NO DATOS

si RXC = 1 ∴ *=1 (verdad)

SI DATOS