Atividades 03 - USART e SPI (ATmega328)

1. Qual seria o valor a ser carregado no registrador UCSR0C para configurar a USART0 com 7N2.

R: 7N2 = caracteres de 7 bits, sem paridade e 2 bits de parada.

- todos são de leitura e escrita

Bit	7	6	5	4	3	2	1	0
	UMSEL1	UMSEL0	UPM1	UPM0	USBS	UCSZ1	UCSZ0	UCPOL
Valor inicial	0	0	0	0	1	1	0	0

UMSEL = Modo assíncrono: 00
UPM = Configura a paridade
USBS = 1 - Seleciona 2 *stop bit*.,
UCSZ = Configura o tamanho de caractere : 7 bits = UCSZ1 (1), UCSZ0 (0).
UCPOL = Borda de subida na transmissão e descida na recepção: 0

2. Escreva o código para configurar a USART0 com o formato de quadro (8E2) e baud rate de 300 bps.

Obs:

Equação

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

```
#define FOSC 16000000

#define BOUD 300

#define UBRRn (FOSC/16*BOUD) - 1


void setup()  {


// set 300 as baud rate

UBRR0H = (unsigned char) (ubrr >> 8);

UBRR0L = (unsigned char)  ubrr;
```

```
// set frame format as 8E2

UCSR0C = 0b00101110;

}

void loop () {}
```

3. Escreva o código para configurar o SPI no modo de operação escravo.

MOSI = master -> slave (SDO, DO, SO)
MISO = slave -> master (SDI, DI, SI)
SCLK = clock (CLK, SCK)
SS = seleção de slave

3 registradores usados
SP**C**R = de controle
SP**S**R = de status
SP**D**R = de dados (leitura/escrita), escrita - inicia a transmissão de dados;

bits do SPCR
MSTR = seta se mestre (1) ou escravo
SPE = habilita a interface SPI (1)

bits do SPSR
SPIF = sinaliza se a transferência foi concluída (1)

No arduino
SS = pino 10 (LOW para ouvir o mestre)
MOSI = pino 11
MISO = pino 12
SCK = pino 13

R: SPI slave com base no slide do prof.

```
// --- Inicialização ---
// Seta SS (PINB2), MOSI (PINB3), SCK (PINB5), como entradas e MISO (PINB4) como saída.
DDRB |= (0 << PINB2) | (0 << PINB3) | (0 << PINB5) | (1 << PINB4);
// Habilita a SPI e configura como slave. (Não precisa configurar o clock, pq o clock dele vem do master)
SPCR = (1 << SPE) | (0 << MSTR);

// --- Recebe os dados ---
rdata = SPDR;                   // recebe o dado.
SPDR = wdata;                           // escreve um byte fictício.
SPSR = (1 << SPIF);             // sinaliza que foi enviado (escrevendo 1).
```

Ele usa o *Microchip Visual Studio*