IFPB - Instituto Federal de Educação, Ciência e Tecnologia da Paraíba

Disciplina: Sistemas Embarcados

Professor: Alexandre Sales Vasconcelos

Aluno: Joab da Silva Maia

03 - Atividades - USART e SPI (ATmega328)

Resolução de questões

Questão 1)

Observando a função de cada bit e seu respectivo modo de operação. Podemos ver qual valor será carregado no registrador UCSR0C. Configure UART0 com 7N2 como solicitado na pergunta. Inicialmente verificamos a tabela a função dos bits numerados de 7 à 0 do UART0.

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	

Bits: 7 a 0

Função: Seleciona o modo de operação: síncrono, assíncrono, reservado e master SPI.

Bits: 5 a 4

Função: Selecionam o tipo de verificação de paridade: desabilitado, reservado, ativado odd e even.

Bits: 3

Função: Seleciona o número de bits de parada a serem inseridos pelo transmissor.

Bits: 2 a 1

Função: Define o tamanho do caractere (número de bits) recebido e/ou transmitido.

Bits: 0

Função: Define a relação entre a mudança de saída e entrada de dados e o clock.

Por convenção, configuraremos USART0 em modo assíncrono e sem paridade. Como você pode na pergunta 7N2, teremos 2 bits de parada e caracteres de 7 bits. Portanto Podemos configurar o valor do bit a ser carregado no registro de acordo com os valores da tabela abaixo:

conf	assíncrono		sem paridade		2bits	caracteres com 7 bits		assíncrono
bits	7	6	5	4	3	2	1	0
valor	0	0	0	0	1	1	0	0

O valor a ser carregado será: 00001100.

Questão 2)

```
#define USART_BAUDRATE 300
#define BAUD_PRESCALE (((fosc/(16*BAUDRATE)))-1)

int main void(){
        UCSRB = (1 << RXEN) | (1 << TXEN);
        URSRC = (1 << URSEL) | (1 << UCSZ0) | (1 << UCSZ1);
        UBRRH = (BAUD_PRESCALE >> 8);
        UBRRL = BAUD_PRESCALE;
        for(;;)
        {
        while((UCSRA & (1 << RXC)) == 0) ();
        ReceivedByte = UDR;
        while((UCSRA & (1 << UDRE)) == 0) ();
        UDR = ReceivedByte;
        }
}
```

Questão 3)

```
DDRB |= (1 << PINB2) | (1 << PINB3) | (1 << PINB5) | (0 <<
PINB4);
SPCR &= ~(1<<MSTR);
SPCR |= (1<<SPR0)|(1<<SPR1);
SPCR |= (1<<SPIE);
SPCR |= (1<<SPE);
SPDR = data;
while(!(SPSR & (1 << SPIF)));
SPDR = 0xFF;
while(!(SPSR & (1 << SPIF)));
data = SPDR;
```