

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL

FACULDADE DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Professor Juliano D'Ornelas Benfica
Laboratório de Processadores I

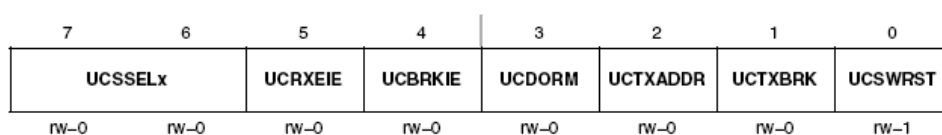
Tutorial de uso das portas seriais do MSP430F2xx:

2010

1 Configuração da Serial

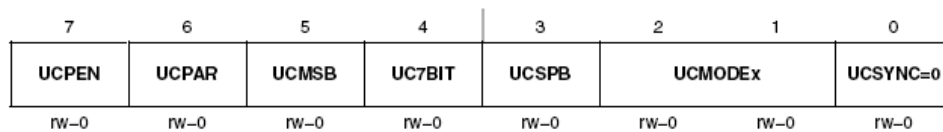
Para a utilização da serial é necessário a configuração dos seguintes registradores:

- **UCAxCTL1, USCLAx Control Register 1:** Este registrador possui 8 bits para configuração conforme mostrado na figura abaixo:



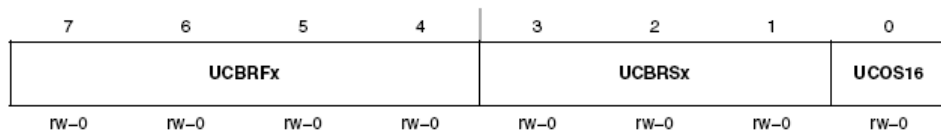
- **BIT 7-6 - UCSSELx** - Seletor de fonte de clock - Seguem a seguinte configuração:
 - * 00 - UCLK
 - * 01 - ACLK
 - * 10 - SMCLK
 - * 11 - SMCLK
- **BIT 5 - UCRXEIE** - Bit de indicação de erro de recepção, caractere errado.
 - * 0 - Recepção normal.
 - * 1 - Recepção com erro.
- **BIT 4 - UCBRKIE** - Bit de indicação de erro de recepção, BREAK.
 - * 0 - Recepção normal.

- * 1 - Recepção com erro.
 - **BIT 3 - UCDORM** - Configuração do modo sleep.
 - * 0 - Modo normal.
 - * 1 - Modo sleep.
 - **BIT 2 - UCTXADDR**- Endereço de transmissão:
 - * 0 - Próximo frame transmite como dado.
 - * 1 - Próximo frame transmite como endereço.
 - **BIT 1 - UCTXBRK**- Interrompe a transmissão:
 - * 0 - Transmissão normal.
 - * 1 - Interrompe a transmissão.
 - **BIT 0 - UCSWRST**- Habilita o reset por software:
 - * 0 - Inicializa a máquina de estados da serial. **OBS: Deve ser habilitado somente após todas as outras configurações**
 - * 1 - Serial em estado de reset.
- **UCAxCTL0, USCLAx Control Register 0:** Este registrador possui 8 bits para configuração conforme mostrado na figura abaixo:



- **BIT 7 - UCPEN** - Habilitação da paridade.
 - * 0 - Paridade desabilitada.
 - * 1 - Paridade habilitada.
- **BIT 6 - UCPAR** - Tipo de paridade.
 - * 0 - Paridade PAR.
 - * 1 - Paridade ÍMPAR.
- **BIT 5 - UCMSB** - Envio do MSB ou LSB.
 - * 0 - Envia primeiro o LSB.
 - * 1 - Envia primeiro o MSB.
- **BIT 4 - UC7BIT** - Tamanho do dado.
 - * 0 - 8 bits de dados.
 - * 1 - 7 bits de dados.
- **BIT 3 - UCSPB** - Stop bits.
 - * 0 - Um stop bit.
 - * 1 - Dois stop bits.
- **BIT 2-1 - UCMODEx** - Seletor de modo de operação - Seguem a seguinte configuração:

- * 00 - Modo UART.
- * 01 - Modo idle-line para multiprocessador.
- * 10 - Modo address-bit para multiprocessador.
- * 11 - Modo UART com detecção automática de baud rate.
- **BIT 0 - UCSYNC**- Modo síncrono ou assíncrono:
 - * 0 - Modo assíncrono.
 - * 1 - Modo síncrono.
- **UCAxMCTL, USCLAx Modulation Control Register**: Este registrador possui 8 bits para configuração conforme mostrado na figura abaixo:



- **BIT 7 - BIT 4 - UCBRFx** - Seleção do primeiro estágio de modulação. Este valor deve ser usado de acordo com as tabelas 1 e 2.
- **BIT 3 - BIT 1 - UCBRSx** - Seleção do segundo estágio de modulação. Este valor deve ser usado de acordo com as tabelas 1 e 2.
- **BIT 0 - UCOS16**- Habilitação para o oversampling:
 - * 0 - Sem oversampling. Este modo deve ser utilizado caso a fonte de clock para a UART seja de baixa frequência.
 - * 1 - Com oversampling. Este modo deve ser utilizado caso a fonte de clock para a UART seja de alta frequência.
- **UCAxBR0, USCLAx Baud Rate Control Register 0 e UCAxBR1, USCLAx Baud Rate Control Register 1** - Estes registradores servem para configuração do baud rate conforme tabelas nas figuras 1, 2 e 3.

BRCLK frequency [Hz]	Baud Rate [Baud]	UCBRx	UCBRsX	UCBRFx	Max. TX Error [%]		Max. RX Error [%]	
32,768	1200	27	2	0	-2.8	1.4	-5.9	2.0
32,768	2400	13	6	0	-4.8	6.0	-9.7	8.3
32,768	4800	6	7	0	-12.1	5.7	-13.4	19.0
32,768	9600	3	3	0	-21.1	15.2	-44.3	21.3
1,048,576	9600	109	2	0	-0.2	0.7	-1.0	0.8
1,048,576	19200	54	5	0	-1.1	1.0	-1.5	2.5
1,048,576	38400	27	2	0	-2.8	1.4	-5.9	2.0
1,048,576	56000	18	6	0	-3.9	1.1	-4.6	5.7
1,048,576	115200	9	1	0	-1.1	10.7	-11.5	11.3
1,048,576	128000	8	1	0	-8.9	7.5	-13.8	14.8
1,048,576	256000	4	1	0	-2.3	25.4	-13.4	38.8
1,000,000	9600	104	1	0	-0.5	0.6	-0.9	1.2
1,000,000	19200	52	0	0	-1.8	0	-2.6	0.9
1,000,000	38400	26	0	0	-1.8	0	-3.6	1.8
1,000,000	56000	17	7	0	-4.8	0.8	-8.0	3.2
1,000,000	115200	8	6	0	-7.8	6.4	-9.7	16.1
1,000,000	128000	7	7	0	-10.4	6.4	-18.0	11.6
1,000,000	256000	3	7	0	-29.6	0	-43.6	5.2
4,000,000	9600	416	6	0	-0.2	0.2	-0.2	0.4
4,000,000	19200	208	3	0	-0.2	0.5	-0.3	0.8
4,000,000	38400	104	1	0	-0.5	0.6	-0.9	1.2
4,000,000	56000	71	4	0	-0.6	1.0	-1.7	1.3
4,000,000	115200	34	6	0	-2.1	0.6	-2.5	3.1
4,000,000	128000	31	2	0	-0.8	1.6	-3.6	2.0
4,000,000	256000	15	5	0	-4.0	3.2	-8.4	5.2
8,000,000	9600	833	2	0	-0.1	0	-0.2	0.1
8,000,000	19200	416	6	0	-0.2	0.2	-0.2	0.4
8,000,000	38400	208	3	0	-0.2	0.5	-0.3	0.8
8,000,000	56000	142	7	0	-0.6	0.1	-0.7	0.8
8,000,000	115200	69	4	0	-0.6	0.8	-1.8	1.1
8,000,000	128000	62	4	0	-0.8	0	-1.2	1.2
8,000,000	256000	31	2	0	-0.8	1.6	-3.6	2.0

Figura 1: Usando o USCOS16=0

12,000,000	9600	1250	0	0	0	0	-0.05	0.05
12,000,000	19200	625	0	0	0	0	-0.2	0
12,000,000	38400	312	4	0	-0.2	0	-0.2	0.2
12,000,000	56000	214	2	0	-0.3	0.2	-0.4	0.5
12,000,000	115200	104	1	0	-0.5	0.6	-0.9	1.2
12,000,000	128000	93	6	0	-0.8	0	-1.5	0.4
12,000,000	256000	46	7	0	-1.9	0	-2.0	2.0
16,000,000	9600	1666	6	0	-0.05	0.05	-0.05	0.1
16,000,000	19200	833	2	0	-0.1	0.05	-0.2	0.1
16,000,000	38400	416	6	0	-0.2	0.2	-0.2	0.4
16,000,000	56000	285	6	0	-0.3	0.1	-0.5	0.2
16,000,000	115200	138	7	0	-0.7	0	-0.8	0.6
16,000,000	128000	125	0	0	0	0	-0.8	0
16,000,000	256000	62	4	0	-0.8	0	-1.2	1.2

Figura 2: Usando o USCOS16=0

BRCLK frequency [Hz]	Baud Rate [Baud]	UCBRx	UCBRsX	UCBRFx	Max. TX Error [%]		Max. RX Error [%]	
1,048,576	9600	6	0	13	-2.3	0	-2.2	0.8
1,048,576	19200	3	1	6	-4.6	3.2	-5.0	4.7
1,000,000	9600	6	0	8	-1.8	0	-2.2	0.4
1,000,000	19200	3	0	4	-1.8	0	-2.6	0.9
1,000,000	57600	1	7	0	-34.4	0	-33.4	0
4,000,000	9600	26	0	1	0	0.9	0	1.1
4,000,000	19200	13	0	0	-1.8	0	-1.9	0.2
4,000,000	38400	6	0	8	-1.8	0	-2.2	0.4
4,000,000	57600	4	5	3	-3.5	3.2	-1.8	6.4
4,000,000	115200	2	3	2	-2.1	4.8	-2.5	7.3
4,000,000	230400	1	7	0	-34.4	0	-33.4	0
8,000,000	9600	52	0	1	-0.4	0	-0.4	0.1
8,000,000	19200	26	0	1	0	0.9	0	1.1
8,000,000	38400	13	0	0	-1.8	0	-1.9	0.2
8,000,000	57600	8	0	11	0	0.88	0	1.6
8,000,000	115200	4	5	3	-3.5	3.2	-1.8	6.4
8,000,000	230400	2	3	2	-2.1	4.8	-2.5	7.3
8,000,000	460800	1	7	0	-34.4	0	-33.4	0
12,000,000	9600	78	0	2	0	0	-0.05	0.05
12,000,000	19200	39	0	1	0	0	0	0.2
12,000,000	38400	19	0	8	-1.8	0	-1.8	0.1
12,000,000	57600	13	0	0	-1.8	0	-1.9	0.2
12,000,000	115200	6	0	8	-1.8	0	-2.2	0.4
12,000,000	230400	3	0	4	-1.8	0	-2.6	0.9
16,000,000	9600	104	0	3	0	0.2	0	0.3
16,000,000	19200	52	0	1	-0.4	0	-0.4	0.1
16,000,000	38400	26	0	1	0	0.9	0	1.1
16,000,000	57600	17	0	6	0	0.9	-0.1	1.0
16,000,000	115200	8	0	11	0	0.9	0	1.6
16,000,000	230400	4	5	3	-3.5	3.2	-1.8	6.4
16,000,000	460800	2	3	2	-2.1	4.8	-2.5	7.3

Figura 3: Usando o USCOS16=1

- **UCAxRXBUF, USCLAx Receive Buffer Register** - Buffer de Recepção - Este registrador é onde fica o dado recebido pela serial.
- **UCAxTXBUF, USCLAx Transmit Buffer Register** - Buffer de Transmissão - Este registrador é onde fica o dado recebido pela serial.

2 Habilitação da Interrupção

Para habilitação da interrupção da porta serial, temos as seguintes configurações:

- **IE2, Interrupt Enable Register 2** - Registrador de habilitação da interrupção pela recepção e/ou pela transmissão da porta serial, conforme figura abaixo:

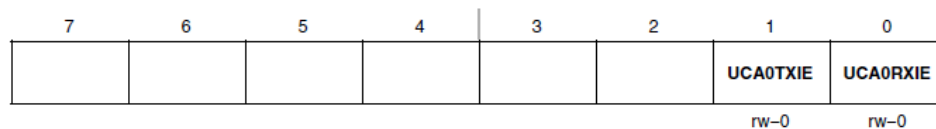


Figura 4: Habilita interrupção

UCA0TXIE - Interrupção de Transmissão:

- 0 : Interrupção Desabilitada.
- 1 : Interrupção Habilitada.

UCA0RXIE - Interrupção de Recepção:

- 0 : Interrupção Desabilitada.
- 1 : Interrupção Habilitada.

- **IFG2, Interrupt Flag Register 2** - Registrador de contém as flags de sinalização da interrupção pela recepção e/ou pela transmissão da porta serial, conforme figura abaixo:

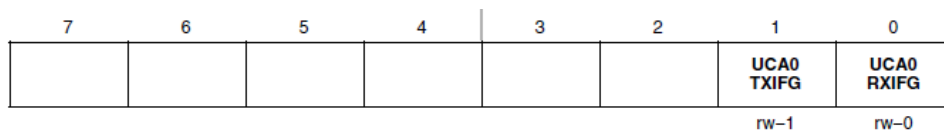


Figura 5: Flags de sinalização de interrupção

UCA0TXIFG - Flag de sinalização de interrupção pela transmissão:

- 0 : Sem interrupção pendente.
- 1 : Interrupção pendente.

UCA0RXIFG - Flag de sinalização de interrupção pela recepção:

- 0 : Sem interrupção pendente.
- 1 : Interrupção pendente.

3 Exemplos de Utilização

3.1 Exemplos de utilização sem interrupção

1. Exemplo de função em C de inicialização da porta serial:

```
void init_serial(void)
{
    P3DIR |= 0x10;    //configura pino de TX como saída
    P3DIR &= ~0x20;   //configura pino de RX como entrada
    P3SEL = 0x30;     //seleciona a funcionalidade uart dos pinos P3.4 e P3.5,5
    UCA0CTL1 = UCSSEL_SMCLK | UCSWRST; //0x81, configura SMCLK e serial parada
    UCA0CTL0 = 0x0;   //8bits, sem paridade, 1 stop bit
    UCA0BR0 = 104;    //ver tabela com UCOS16=1 para clock de 16MHz e baud 9600
    UCA0BR1 = 0;      //ver tabela com UCOS16=1 para clock de 16MHz e baud 9600
    UCA0MCTL = UCBRF_3 | UCBRS_0 | UCOS16; // ver tabela UCBRF=3,UCBRS=0,UCOS16=1
    UCA0CTL1 &= ~UCSWRST; // Inicializa a máquina de estados da serial
}
```

2. Exemplo de função em C de transmissão da porta serial:

```
void envia_serial(int dado)
{
    while(!(IFG2 & 2)); //enquanto não terminou de enviar espera
    UCA0TXBUF = dado;   //envia o dado
    IFG2 &= (~2);       //limpa flag de transmissão
}
```

3. Exemplo de função em C de recepção da porta serial:

```
int U0getchar(void)
{
    while(!(IFG2 & 1)); //enquanto não recebe espera
    IFG2 &= (~1);       //limpa flag de recepção
    return UCA0RXBUF;   //retorna o dado recebido
}
```

3.2 Exemplo de utilização com interrupção

Exemplo de função em C de inicialização da porta serial com interrupção pelo RX:

```
void init_serial(void)
{
    P3DIR |= 0x10;    //configura pino de TX como saída
    P3DIR &= ~0x20;   //configura pino de RX como entrada
    P3SEL = 0x30;     //seleciona a funcionalidade uart dos pinos P3.4 e P3.5,5
    UCA0CTL1 = UCSSEL_SMCLK | UCSWRST; //0x81, configura SMCLK e serial parada
    UCA0CTL0 = 0x0;    //8bits, sem paridade, 1 stop bit
    UCA0BR0 = 104;     //ver tabela com UCOS16=1 para clock de 16MHz e baud 9600
    UCA0BR1 = 0;       //ver tabela com UCOS16=1 para clock de 16MHz e baud 9600
    UCA0MCTL = UCBRF_3 | UCBRS_0 | UCOS16; // ver tabela UCBRF=3,UCBRS=0,UCOS16=1
    UCA0CTL1 &= ~UCSWRST; // Inicializa a máquina de estados da serial
    IE2 |= UCA0RXIE;   //habilita interrupção pela recepção
    _EINT();            //habilita interrupções do processador
}
```

Exemplo de função em C de atendimento da interrupção pelo RX:

```
interrupt (USCIABORX_VECTOR) Serial_0(void)
{
    while (!(IFG2&UCA0TXIFG)); //Enquanto não terminou de transmitir espera
    UCA0TXBUF = UCA0RXBUF;     //envia o dado que chega pela serial
}
```