CAPÍTULO IX

PORTA SERIAL

9.1. INTRODUÇÃO

A porta serial existente na família MCS-51 é "full duplex", quer dizer, pode transmitir e receber dados simultaneamente. Tem um buffer que permite receber um segundo byte antes que o byte previamente recebido tenha sido retirado (lido) do registro de recepção. Mas, se o primeiro byte não tiver sido lido no tempo em que o segundo byte se completa, um dos dois será perdido.

9.2. REGISTROS ENVOLVIDOS

A porta serial possui um registrador chamado SBUF, o mesmo que se usa para enviar ou receber dados pela porta serial. Na realidade, o nome SBUF se refere a dois registros, um somente para leitura por onde se recebem os dados que chegam pela porta serial e outro somente para escrita por onde se transmitem dados pela porta serial.

O modo de operação da porta serial é controlado pelo registro **SCON,** que é ilustrado na figura 9.1. A figura 9.2 apresenta um resumo dos modos de operação.

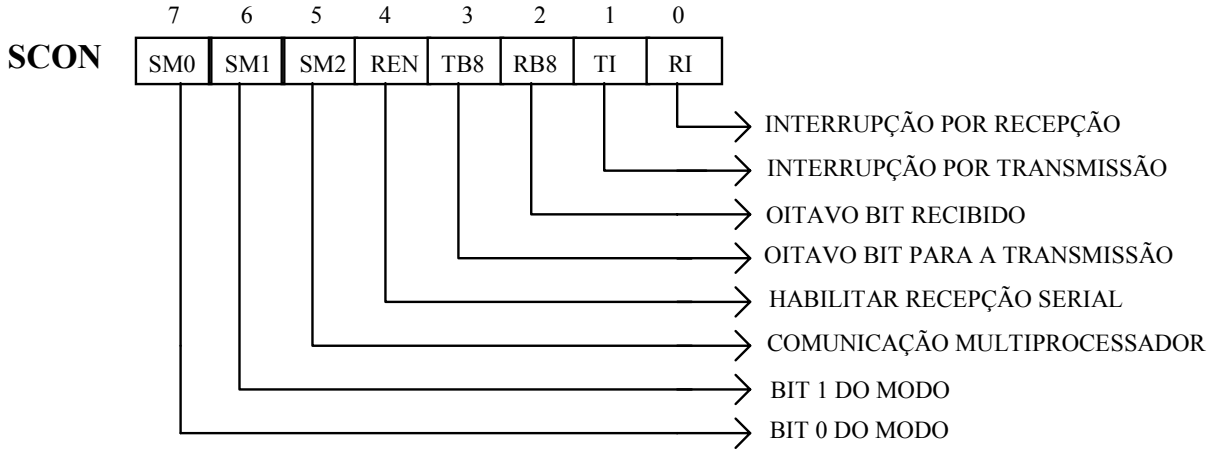


Figura 9.1. Descrição do registro SCON.

SM0	SM1	MODO	DESCRIÇÃO	FREQÜÊNCIA
0	0	0	registro de deslocamento	clock/12
0	1	1	UART de 8 bits	variável
1	0	2	UART de 9 bits	clock/12 ou clock/64
1	1	3	UART de 9 bits	variável

Figura 9.2. Modos de operação da porta serial.

SM2→ comunicação multiprocessador (habilitada com SM2=1):

SM2=1 e em modo 1 ==> interrupção (RI=1) com o bit de parada (stop bit) igual a 1

SM2=1 e em modo 2 ou 3 ==> interrupção (RI=1) se for recebido RB8=1.

REN→ habilita a recepção serial (Reception Enable)

TB8 → oitavo bit a ser transmitido nos modos 2 e 3.

RB8 → oitavo bit recebido nos modos 2 e 3.

TI → flag de interrupção por término de transmissão pela porta serial:

MODO 0 → ativado no final do oitavo bit

DEMAIS → ativado no começo do bit de parada

RI → flag de interrupção por recepção pela porta serial

MODO 0 → ativado no final do oitavo bit

DEMAIS → ativado na metade do bit de parada

Observação: os flags TI e RI não são apagados por hardware e por isso a rotina de interrupção deve fazê-lo, ou seja, estes flags devem ser zerados por software.

O registro PCON também toma parte na geração do baud rate da porta serial. Este registro é ilustrado na figura 9.2.

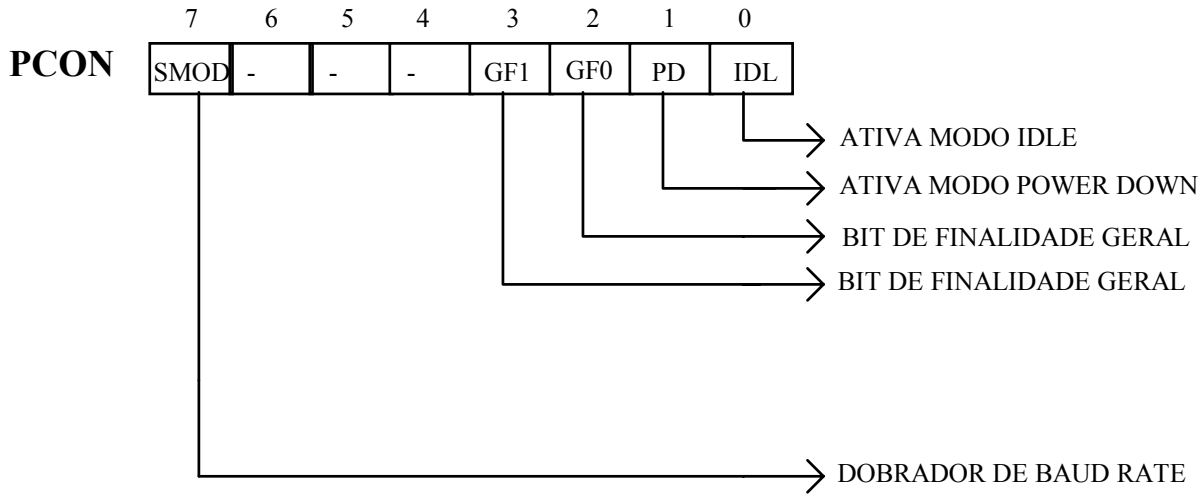


Figura 9.3. Descrição do registro PCON.

9.3. MODOS DE OPERAÇÃO

Existem quatro modos de operação da porta serial, cada um com uma finalidade. Será apresentado um resumo de cada modo:

9.3.1. Modo 0 (síncrono, 8 bits)

Neste modo os dados entram e saem pelo pino RXD. O pino TXD fornece o clock para o deslocamento (shift). São transmitidos e recebidos dados de 8 bits (LSB primeiro). O baud rate é de 1/12 da freqüência do clock (igual a 1 ciclo de máquina).

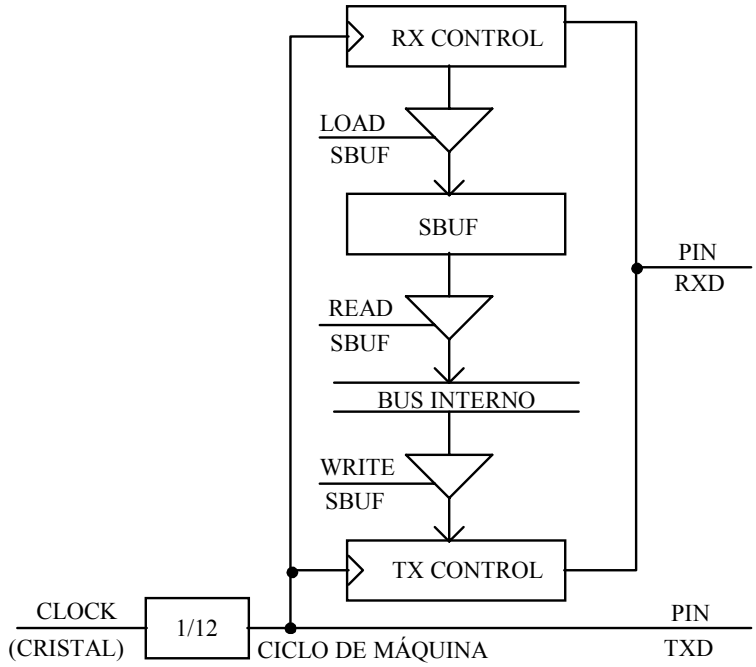
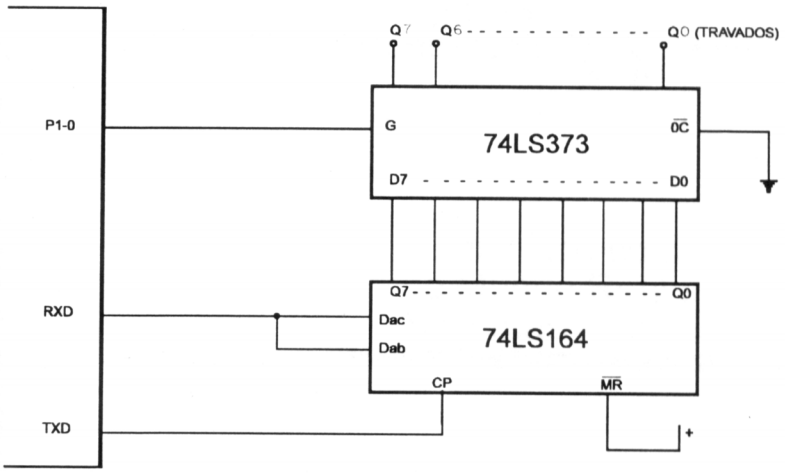


Figura 9.4. Esquema da porta serial em modo 0.

Este modo pode ser usado para expandir as portas de I/O do microcontrolador usando um esquema como o da figura a seguir:



9.3.2. Modo 1 (assíncrono, 8 bits)

No Modo 1 são operados 10 bits com um baud rate programável. Recebe-se pelo pino RXD e transmite-se pelo pino TXD. A figura 9.5 ilustra o frame de bits serial. O bit de partida (START) é sempre zero e o bit de parada (STOP) é sempre um na transmissão. Na recepção o bit de parada é colocado no bit RB8.

START	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	STOP

Figura 9.5. Frame de bits gerado pela porta serial em modo 1.

Neste modo o baud rate é gerado pelo contador/temporizador 1. A cada 16 (SMOD=1) ou 32 (SMOD=0) transbordamentos (overflows) é enviado um pulso para o circuito serial. A figura 9.6 ilustra o esquema do gerador do baud rate e também apresenta algumas fórmulas para o cálculo do valor a ser programado no contador/temporizador.

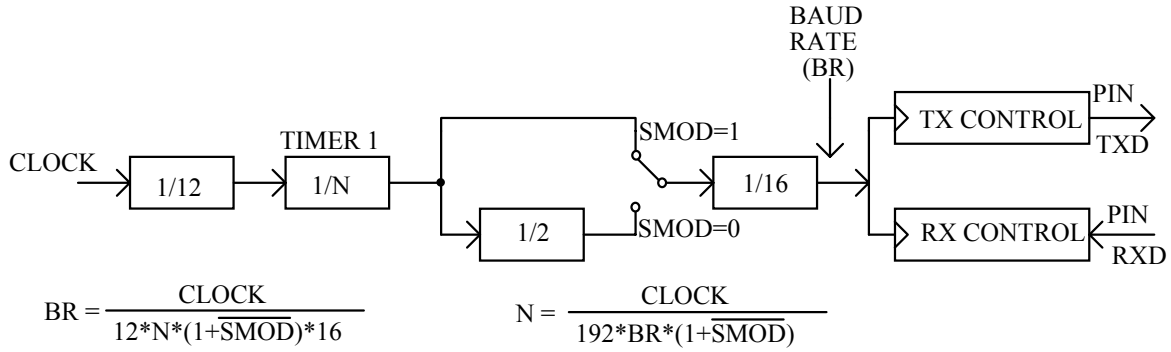


Figura 9.6. Esquema para a geração do baud rate (BR) no modo 1.

Usando o contador/temporizador 1 programa-se qualquer baud rate. Na grande maioria dos casos tem-se o baud rate e é buscado o valor N a ser programado no contador/temporizador 1 (o mais cômodo é usar o modo 2 - auto recarga). Um problema que sempre existe é o erro gerado pela aproximação do valor a ser programado no contador/temporizador. Isto provoca a geração de baud rates ligeiramente diferentes.

Exemplo: Usando um cristal de 3,575611 MHz, qual é o valor a ser programado no timer 1 para operar a 9600 bauds (usar SMOD=0) ?

N = 3575611 / (384*9600) = 0,9699 (aproximadamente 1)

Usando timer 1 no modo 2 tem-se: 256 - 1 = 255 (TH1=TL1=255).

Observação: a transferência do dado recebido para o registro SBUF (e RB8, que guarda o stop bit) e a ativação do flag RI somente acontecerá se:

RI = 0 e (SM2= 0 ou o bit de parada =1).

9.3.3. Modo 2 (assíncrono, 9 bits, baud rate fixo)

Este modo opera com 11 bits e o baud rate pode ser de 1/32 ou 1/64 do clock. O bit TB8/RB8 serve para transmitir a paridade ou gerar um segundo bit de parada (TB8=1). O frame de bits usado neste modo é ilustrado na figura 9.7.

START	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	TB8	STOP

Figura 9.7. Frame de bits gerado pela porta serial no modo 2.

O único controle que se tem sobre o baud rate é através do uso do bit SMOD. Se SMOD=0, trabalha-se com BR=clock/32 e se SMOD=1, trabalha-se com BR=clock/64. A figura 9.8 ilustra o esquema para geração do baud rate no modo 2.

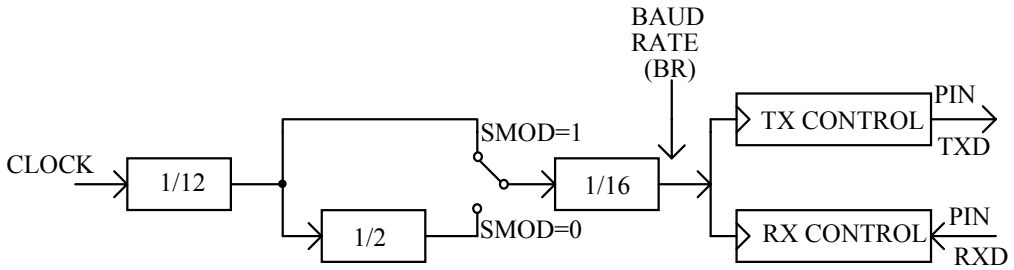


Figura 9.8. Esquema para a geração do baud rate (BR) no modo 2.

Na transmissão o bit 8 é copiado do bit TB8. Na recepção o bit 8 é copiado para o bit RB8. Os bits TB8 e RB8 estão no registro SCON.

O dado recebido somente é carregado no SBUF (e RB8) se:

RI=0 e (SM2=0 ou bit 9 (stop bit) = 1).

9.3.4. Modo 3 (assíncrono, 9bits, baud rate variável)

É idêntico ao modo 2, exceto que a geração do baud rate é idêntico ao modo1. A figura 9.9 ilustra o frame de bits e a geração do baud rate.

START	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	TB8	STOP

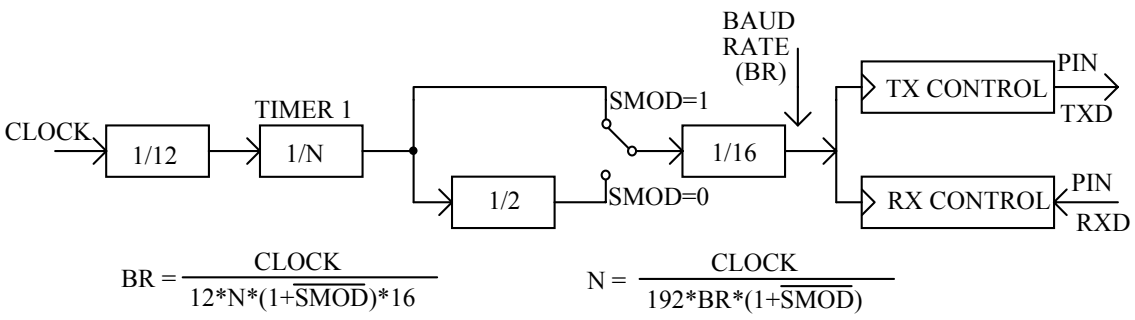


Figura 9.9. Frame de bits e geração do baud rate para a porta serial no modo 3.

9.4. CUIDADOS COM A PORTA SERIAL

Nos quatro modos a transmissão se inicia quando é escrito um byte em SBUF. A recepção é habilitada quando REN=1 (no MODO 0 exige-se também RI=0). Não se esqueça de que somente existe uma única interrupção dedicada à porta serial, portanto ela é gerada por duas condições:

- Término da transmissão de um byte → flag TI
- Término da recepção de um byte → flag RI

Deve-se testar os flags RI e TI para determinar se a interrupção foi por transmissão ou por recepção. Os flags RI e TI devem ser apagados por software.

Um outro cuidado muito importante é com a geração do baud rate. Muitas vezes não se consegue a comunicação serial devido à imprecisão do baud rate. Portanto o problema a seguir é proposto: Programar o contador/temporizador 1 para gerar 9600 bauds para a porta serial operando em modo 3, sendo que o cristal é de 4 MHz.

Solução: N = 4000000/(384*9600) = 1,085 (SMOD=0).

Na solução calculou-se que N=1,085 mas somente podem ser programados números interos; assim, caso se aproxime para 1, o baud rate gerado será de BR=4000000/384=10416,7. Será que irá funcionar bem com esse baud rate ?

Para responder esta pergunta é necessário compreender o que acontece com a transmissão serial. Esta transmissão é assíncrona, quer dizer, pode iniciar em qualquer instante; o inicio é caracterizado pela presença de um bit de partida (START). Uma vez iniciada, deve-se garantir a duração de cada bit. Para cada byte é transmitido um bit de partida, os bits de dados e um ou dois bits de parada. O que se deve buscar é garantir que não haja um erro muito grande neste frame de bits. A figura 9.10 ilustra o caso da porta serial operando em modo 3.

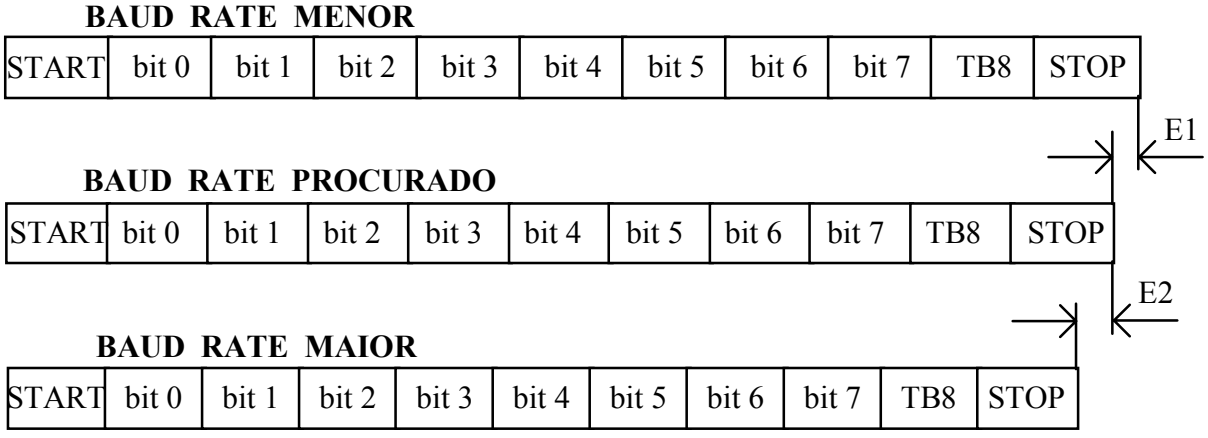


Figura 9.10. Erros provocados por diferenças no baud rate.

A fase mais critica é a recepção do último bit porque poderá apresentar uma defasagem muito grande (rotuladas de E1 e E2 na figura). Não se pode especificar um valor limite para esses

erros pois diversos fatores, particulares para cada caso, precisam ser levados em consideração. Como um valor prático e que funciona na grande maioria dos casos arbitra-se que o valor do erro, para o último bit, deve ser menor que 40% da duração de um bit (usando BR exato).

$$\left|\frac{11}{BRp} - \frac{11}{BR}\right| \leq \frac{40}{100} * \frac{1}{BR}$$

BRp $\rightarrow$ baud rate programado

BR $\rightarrow$ baud rate exato

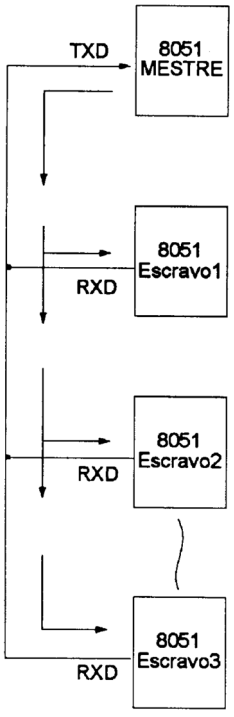
A equação pode ser simplificada para : $$\left|\frac{1}{BRp} - \frac{1}{BR}\right| \leq \frac{1}{27,5 * BR}$$

Como pode ser verificado, com um cristal de 4 MHz não se conseguirá transmitir a 9600 pois o erro do baud rate será muito grande.

$$\left|\frac{1}{10416,7} - \frac{1}{9600}\right| \leq \frac{1}{27,5 * 9600}$$

8,167 μs $\leq$ 3,788 μs $\rightarrow$ falso.

9.5. Comunicação entre vários 8051



Os modos 2 e 3 permitem interligar vários 8051, sendo um mestre e vários escravos. Nestes modos temos:

- 1 start bit;
- 8 bits de dados;
- um nono bit que vai para o bit RB8 (na recepção) ou pode ser escolhido 0 ou 1 na transmissão escrevendo-se em TB8;
- 1 stop bit

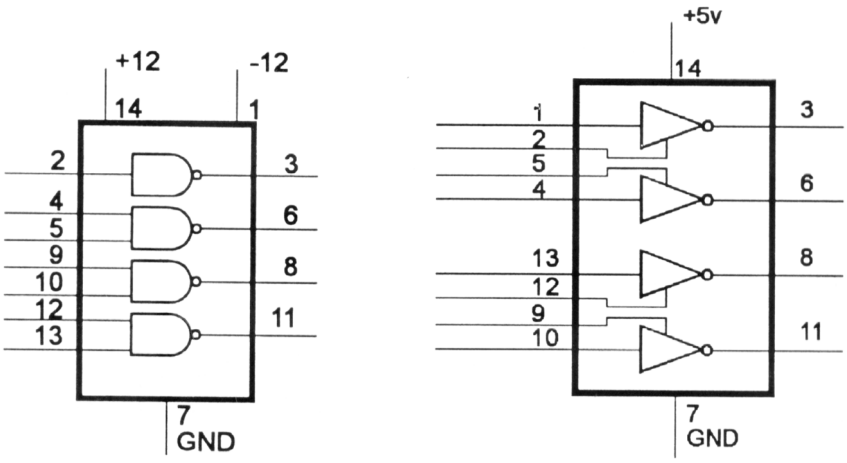
Note que se SM2 = 1 e RB8 = 1, a interrupção da serial será atendida.

O algoritmo de comunicação consiste em:

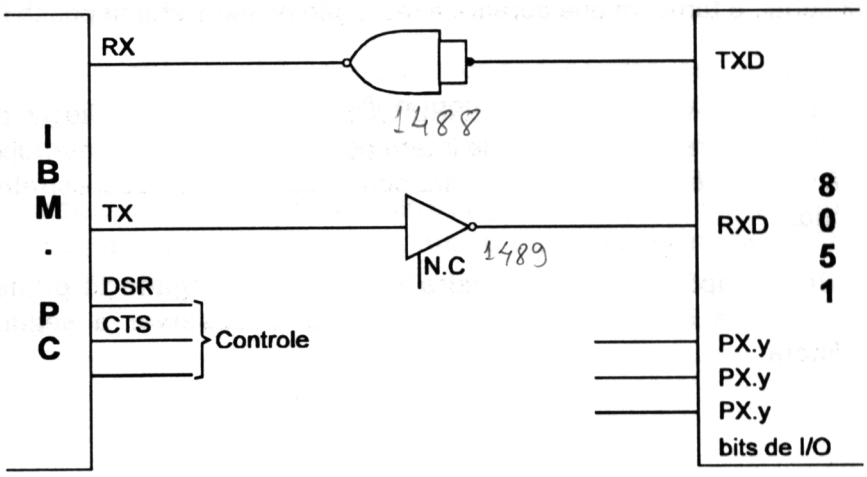
1) No início, todos os escravos estão com SM2 = 1
2) Quando o mestre quiser enviar dados para algum escravo, ele escreverá 1 em seu bit TB8 e então enviará serialmente o endereço do escravo desejado, e como teremos todos os bits RB8 em 1, **todos** escravos serão interrompidos para verificar se é seu o endereço enviado.
3) O escravo selecionado zerará o seu bit SM2 e estará preparado para receber os dados, os quais terão agora o nono bit (RB8) em 0.
4) Os demais escravos permanecerão com SM2 em 1 e, dessa forma, não serão mais interrompidos pois os dados têm RB8 = 0.
5) Em resumo, o mestre envia endereços com o nono bit em 1 e os dados com o nono bit em 0, portanto, se o mestre desejar se comunicar com outro escravo basta enviar o novo endereço com o nono bit em 1.

9.6. Comunicação serial entre o 8051 e o PC

Devido à diferença de tensão entre os níveis TTL e os usados no padrão RS-232C (PC), é necessário usar circuitos conversores para interfacear o microcontrolador com a porta serial do PC. A conversão pode ser feita através de circuitos transistorizados discretos (veja as figuras 11.10, 11.11 e 11.12 e leia a seção 11.4) ou através de circuitos integrados como o 1488 (conversor de TTL para RS-232C) e o 1489 (conversor de RS-232C para TTL). A seguir estão os esquemas dos dois integrados e o esquema básico de interligação entre o 8051 e a serial do PC.



CIs 1488 (esquerda) e 1489 (direita)



9.7. EXERCÍCIOS

Exercício 9.1. TX_SER

Usando um loop infinito, transmitir pela porta serial todos os caracteres ASCII de "0" a "Z". Usar 9600 bauds, 8 bits de dados, 1 bit de partida e 2 bits de parada.

Para este caso o modo 3 é o mais adequado e TB8=1 será usado para gerar um bit de parada. A transmissão será feita por interrupção, quer dizer, a cada byte transmitido haverá uma interrupção. O contador/temporizador 1 será programado para modo 2 (auto-recarga).

SCON

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
1	1	0	0	1	0	0	0

TMOD

GATE	C/*T	M1	M0	GATE	C/*T	M1	M0
0	0	1	0	0	0	0	0

IE

EA	-	-	ES	ET1	EX1	ET0	EX0
1	-	-	1	0	0	0	0

Figura 9.11. Inicialização de os registros para o exercício 9.1.

Cálculo do divisor formado pelo contador/temporizador 1:

N = 3575611/(384*9600) = 0,9699 → 1. (TH1 = TL1 = 256 - 1 = 255).

```
;TX_SER.ASM
                DEFSEG PROG, CLASS=CODE, START=0
                SEG             PROG
;
BR_9600         EQU             255
;
                ORG             RESET
                AJMP            INIC
;
                ORG             SINT
                AJMP            SERIAL
;
                ORG             50H
INIC            MOV             TMOD,#20H       ;TIMER 1 EM MODO 2
                MOV             TH1,#BR_9600    ;PROGRAMAR BAUD RATE
                MOV             TL1,#BR_9600
                SETB            TR1             ;INICIAR TIMER 1
                MOV             SCON,#0C8H      ;MODO 3 COM TB8=1
                MOV             IE,#90H         ;HAB INTERRUP SERIAL
                MOV             A,#"0"          ;PRIMERO ASCII
                SETB            TI              ;TX PRIMEIRO ASCII
                SJMP            $               ;LOOP INFINITO
;
                ORG             100H
SERIAL          CLR             TI              ;APAGAR FLAG
                MOV             SBUF,A          ;TRANSMITIR
                INC             A
                CJNE            A,#"Z"+1,SER1   ;FOI O ULTIMO ?
                MOV             A,#"0"          ;REINICIALIZAR
SER1            ACALL           RETARDO         ;ATRASAR TRANSMISSAO
                RETI
;
RETARDO         MOV             R7,#0
AQUI            DJNZ            R7,AQUI
                RET
                END
```

Exercício 9.2. RX_SER

O circuito deverá responder aos seguintes comandos que chegam pela porta serial:

1 → acender todos os leds	0 → apagar todos os leds
R → acender led vermelho	r → apagar led vermelho
A → acender led amarelo	a → apagar led amarelo
V → acender led verde	v → apagar led verde

Estes comandos vão chegar pela porta serial usando o formato: 1 bit de partida, 8 bits de dados, 2 bits de parada, com um baud rate de 9600.

```
;RX_SER.ASM
               DEFSEG PROG, CLASS=CODE, START=0
               SEG            PROG
;
BR_9600        EQU            255
LED_VERMELHO   EQU            P1.0
LED_AMAR       EQU            P1.1
LED_VERDE      EQU            P1.2
CHEGOU         EQU            32.0
;
               ORG            RESET
               AJMP           INIC
;
               ORG            SINT
               AJMP           SERIAL
;
               ORG            50H
INIC           MOV            TMOD,#20H       ;TIMER 1 EM MODO 2
               MOV            TH1,#BR_9600    ;PROGRAMAR BAUD RATE
               MOV            TL1,#BR_9600
               SETB           TR1             ;INICIAR TIMER 1
               MOV            SCON,#0D0H      ;MODO 3 COM REN=1
               MOV            IE,#90H         ;HAB INTERRUP SERIAL
               CLR            CHEGOU          ;APAGAR FLAG
ESPERA         JNB            CHEGOU,ESPERA   ;AGUARDAR UM COMANDO
               CLR            CHEGOU
;
               CJNE           A,#"0",LB1
               CLR            LED_VERMELHO    ;CHEGOU 0
               CLR            LED_AMAR
               CLR            LED_VERDE
               SJMP           ESPERA
;
LB1            CJNE           A,#"1",LB2
               SETB           LED_VERMELHO    ;CHEGOU 1
               SETB           LED_AMAR
               SETB           LED_VERDE
               SJMP           ESPERA
;
LB2            CJNE           A,#"R",LB3
               SETB           LED_VERMELHO    ;CHEGOU R
               SJMP           ESPERA
LB3            CJNE           A,#"r",LB4
               CLR            LED_VERMELHO    ;CHEGOU r
               SJMP           ESPERA
;
LB4            CJNE           A,#"A",LB5
               SETB           LED_AMAR        ;CHEGOU A
               SJMP           ESPERA
LB5            CJNE           A,#"a",LB6
               CLR            LED_AMAR        ;CHEGOU a
               SJMP           ESPERA
;
LB6            CJNE           A,#"V",LB7
               SETB           LED_VERDE       ;CHEGOU V
               SJMP           ESPERA
LB7            CLR            LED_AMAR        ;CHEGOU v
               SJMP           ESPERA
;
SERIAL         CLR            RI              ;APAGAR FLAG DE INTERRUP
               MOV            A,SBUF          ;COLOCAR DADO NO Acc
               SETB           CHEGOU
               RETI
               END
```