EEL7030 - Microprocessadores

LCS

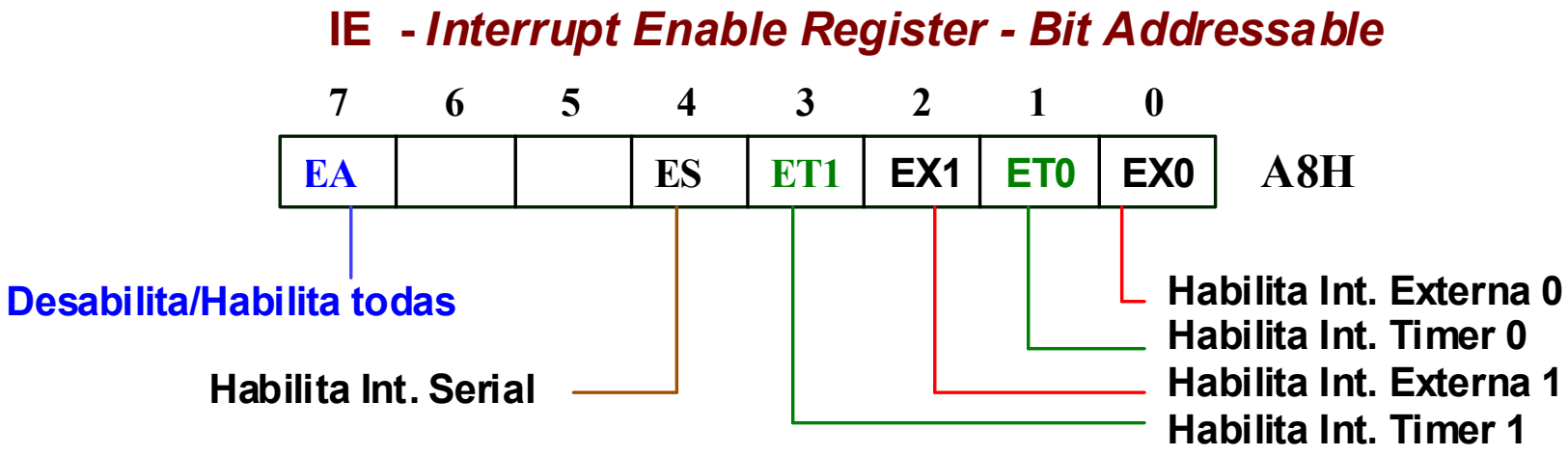
Laboratório de
Comunicações
e Sistemas
Embarcados

Prof. Raimes Moraes

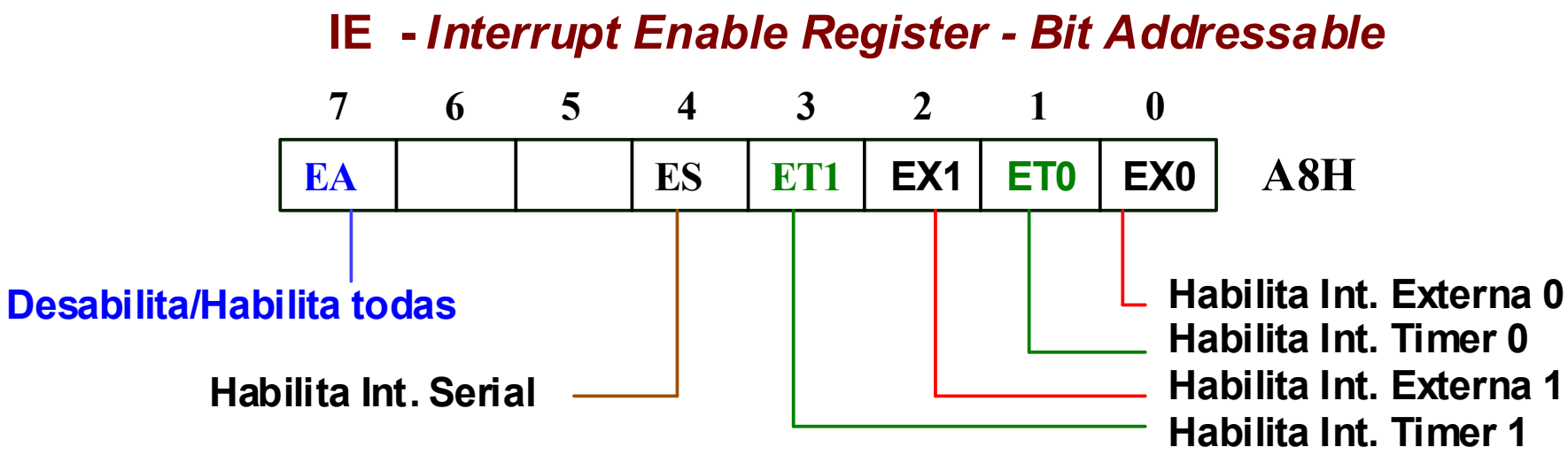
EEL - UFSC

Fontes de interrupção e endereços dos tratadores de interrupção do 8051

Fontes de Interrupção	Endereços dos Tratadores (Hexadecimal)
Externa 0	0003
Timer 0	000B
Externa 1	0013
Timer 1	001B
Serial	0023



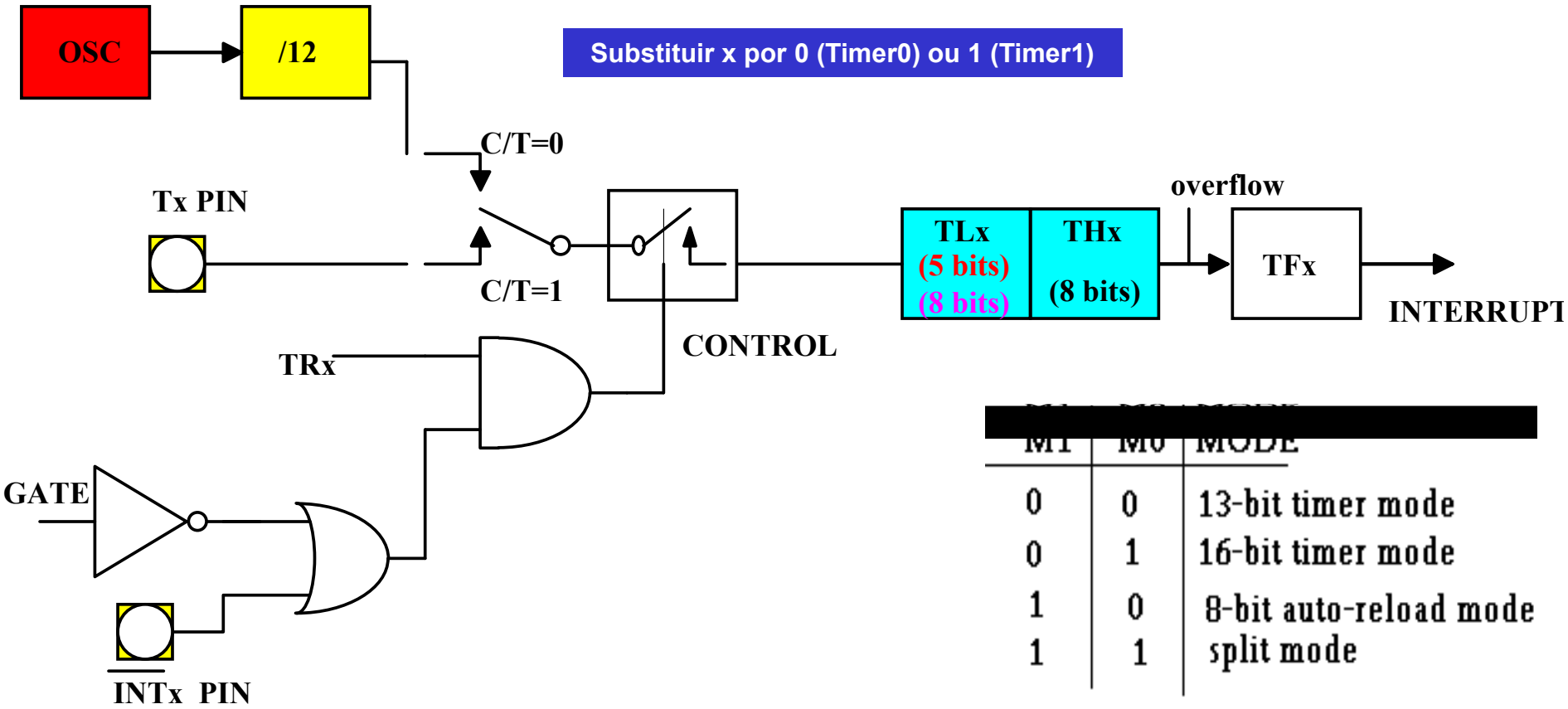
Habilitação das Interrupções dos Timers



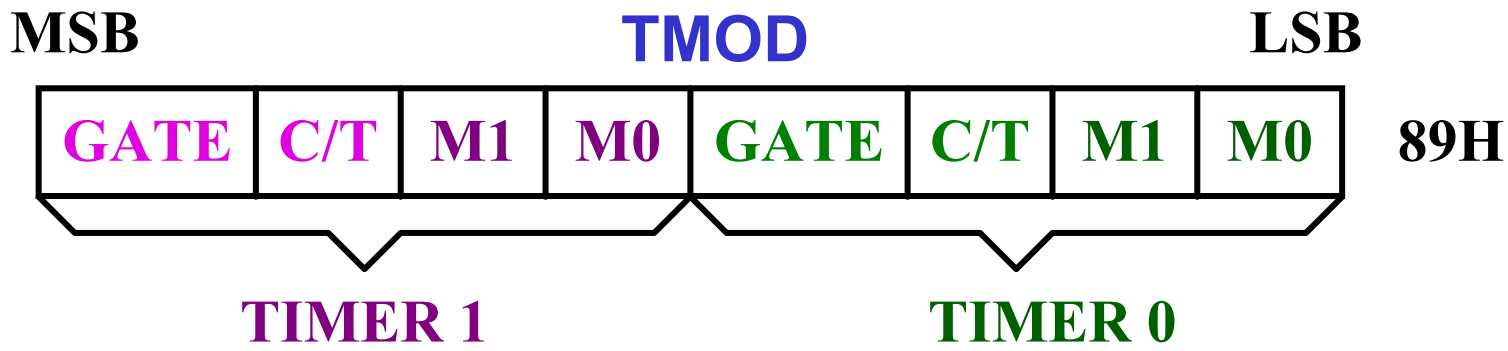

```
MOV  IE,#10001010B;      habilita TMR0 E TMR1
ou
SETB  ET0
SETB  ET1
SETB   EA
```

Temporizadores / Contadores

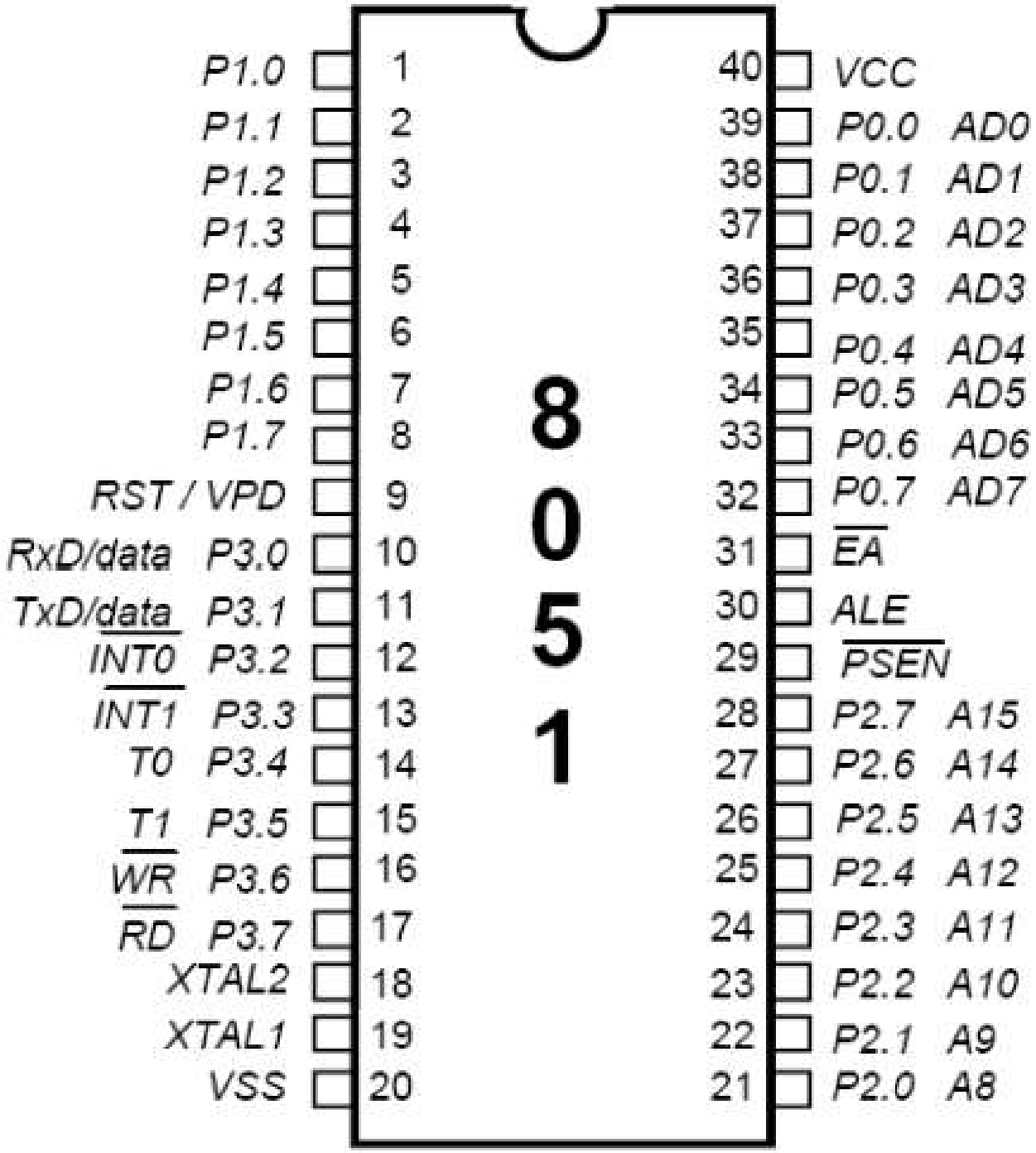
Modos 0 (13 bits) e 1 (16 bits)



M1	M0	MODE
0	0	13-bit timer mode
0	1	16-bit timer mode
1	0	8-bit auto-reload mode
1	1	split mode



Pinos do 8051



Registrador Timer Control (TCON)

MSB LSB

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

88H

ITx - *Interrupt control bit*. 1 => borda de descida
0 => nível lógico baixo

IEx - *External Interrupt flag*.
Setado pelo hardware quando interrupção é detectada.
Zerado pelo hardware ao desviar o fluxo de execução do código para o tratador de interrupção.

TRx – *Timer Run*
Deve ser setado pelo programador (Ex: SETB TR0) para iniciar a contagem.

TFx – *Timer flag*
Setado pelo hardware quando há *overflow* do contador (No modo 1, FFFFH => 0000H).
Zerado pelo hardware ao desviar o fluxo de execução do código para o tratador de interrupção.

Programação dos Timers

1. **Alocar tratador para o contador/temporizador no seu devido endereço;**

2. **Habilitar interrupção do contador/temporizador no registrador IE (EA, ETx);**

3. **Especificar o modo de funcionamento do contador/temporizador (TMOD);**

4. **Especificar intervalo de contagem (THx e TLx);**

5. **Iniciar contagem (setb TRx);**

6. **Ao término da contagem, realizar recarga para os modos: 0, 1 e 3.**

OBS1: Caso se deseje medir a largura de pulso de sinal aplicado em /INTx, fazer GATE= '1'. A contagem ocorrerá durante intervalo de tempo no qual o pino /INTx estiver em nível lógico alto.

OBS2: É possível testar, frequentemente, se houve encerramento da contagem sem habilitar interrupções (*polling* dos *flags* TFx) em programas mais dedicados.

Exemplo: Faça um programa que escreva na porta P1, a cadeia de 16 caracteres: 'Microcontrolador' à taxa de 10 kbits/s ((1/10000) = 100 us). Usar Timer 0 no Modo 1. Cristal: 12 MHz

Contagem com Timer 0: Usando Interrupção

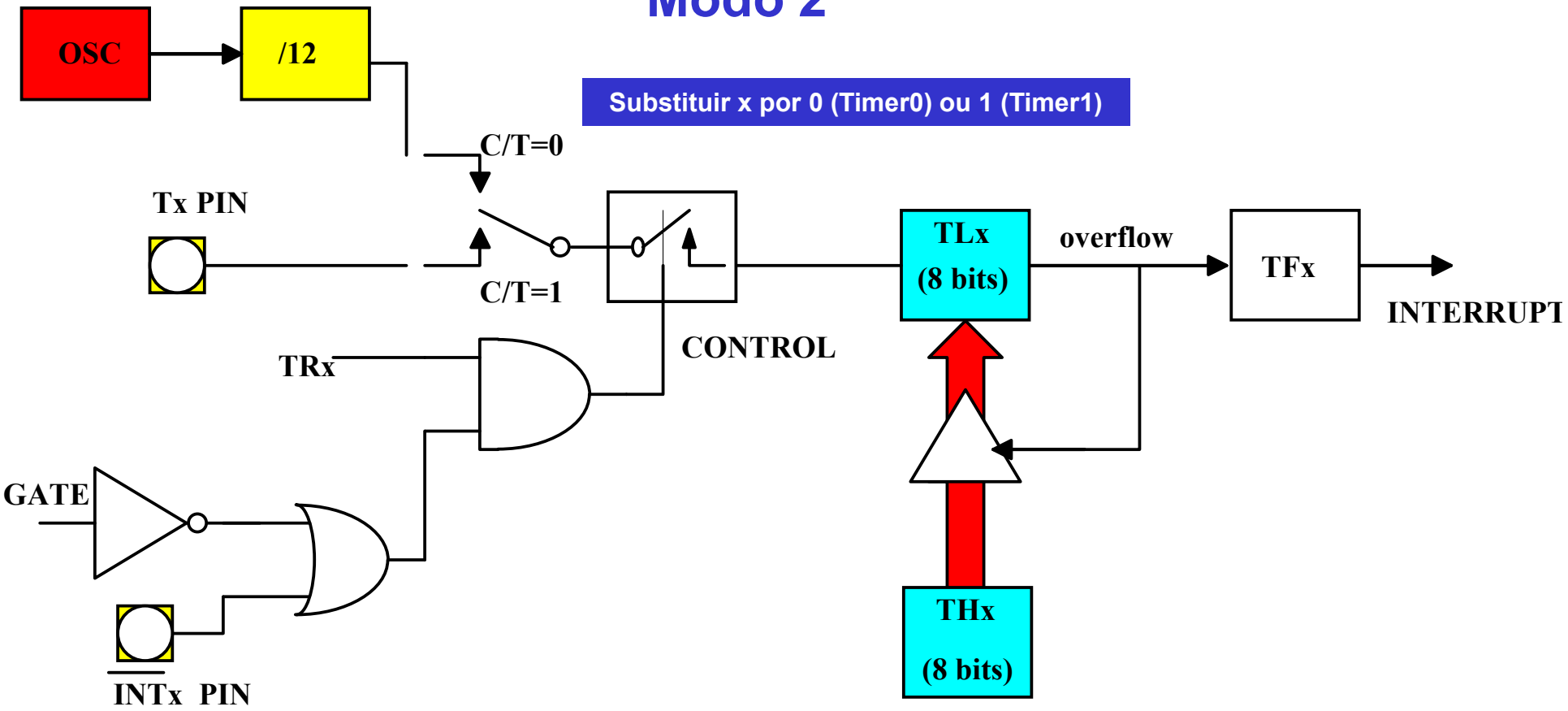
```
reset     equ   0h
ltmr0     equ   0bh ; local do tratador
state     equ   20h


    org reset          ;PC=0 depois de reset
    jmp inicio


    org         ltmr0
    mov         th0,#0ffh
    mov         tl0,#09ch
    mov         state,#1h
    reti


inicio:
    mov    ie,#10000010b     ; habilita tmr0
    mov    tmod,#01h         ; modo 1

    mov         th0,#0ffh
    mov         tl0,#09ch
```

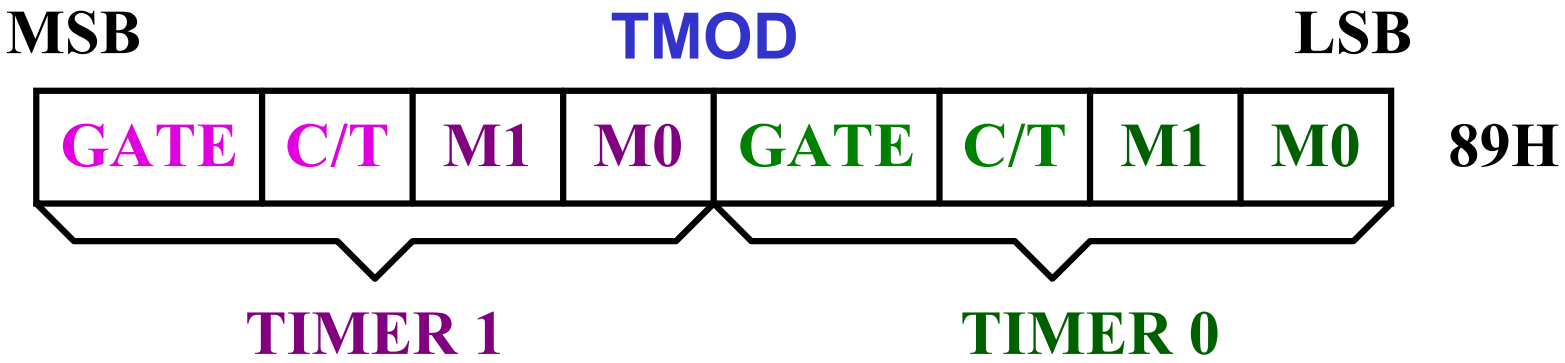
```
    mov    state,#0h ;inicialização
    mov    r0,# state
    mov    dptr,#tabela
    mov    r1,#0

    setb   tr0


volta:    cjne       @r0,#1,volta


              mov      state,#0h
              mov      a,r1
              movc     a,@a+dptr
              mov      p1,a
              inc      r1
              cjne     r1,#16,volta
              clr      tr0
              jmp      $

tabela: db 'Microcontrolador'
              end
```

Contagem com Timer 0: Usando *Polling*

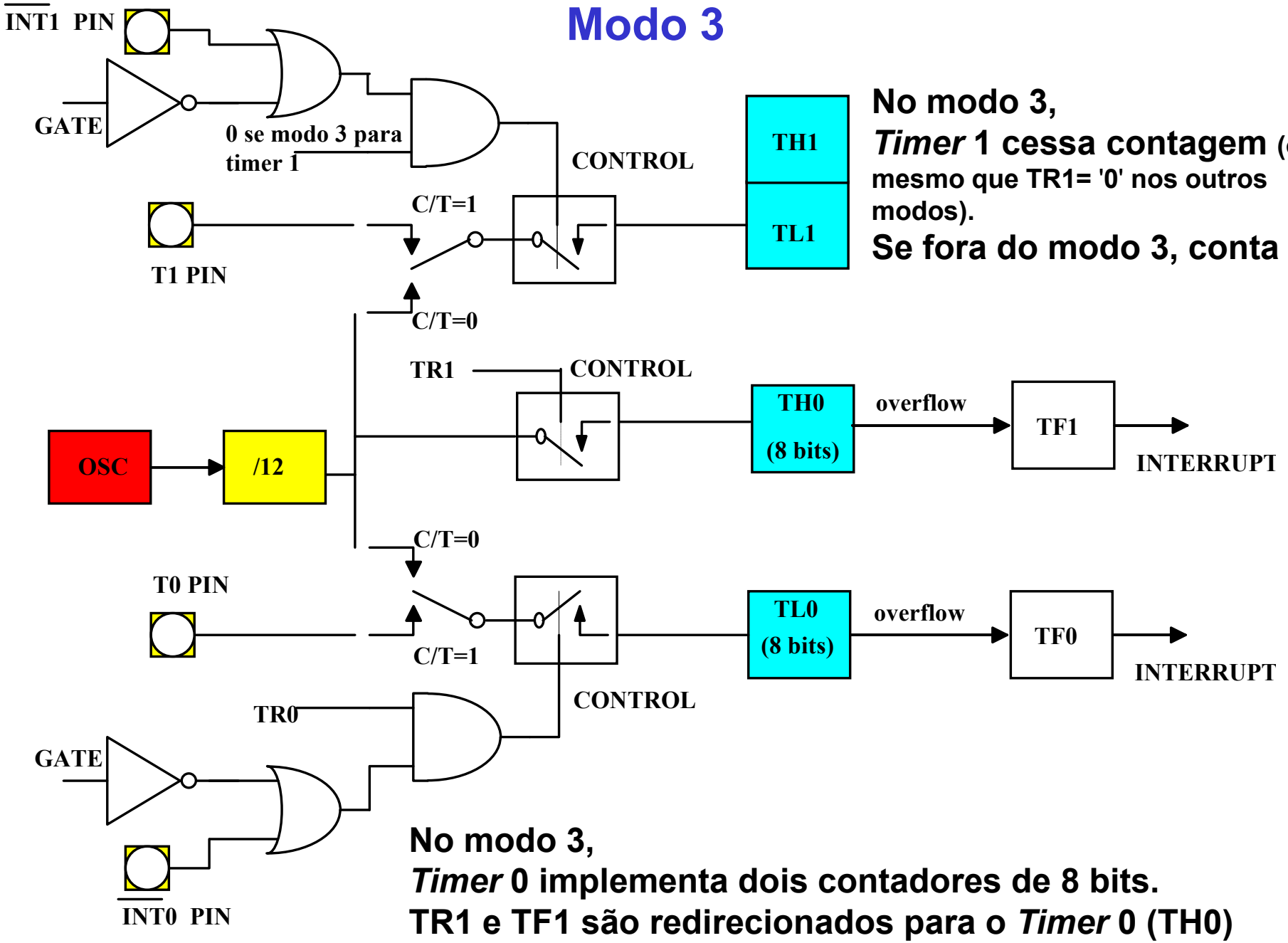
```
        org     0h              ; PC=0 depois de reset
        mov     tmod,#01h       ; timer0 no modo 1
        mov     th0,#0ffh
        mov     tl0,#09ch

        mov     dptr,#tabela
        mov     r1,#0
        setb    tr0

volta:  jnb     tf0,volta
        mov     tl0,#09ch
        mov     th0,#0ffh
        clr     tf0
        mov     a,r1
        movc    a,@a+dptr
        mov     p1,a
        inc     r1
        cjne    r1,#16,volta
        clr     tr0
        jmp     $

tabela: db 'Microcontrolador'
        end
```

Temporizadores / Contadores
Modo 2





Temporizadores / Contadores
Modo 3

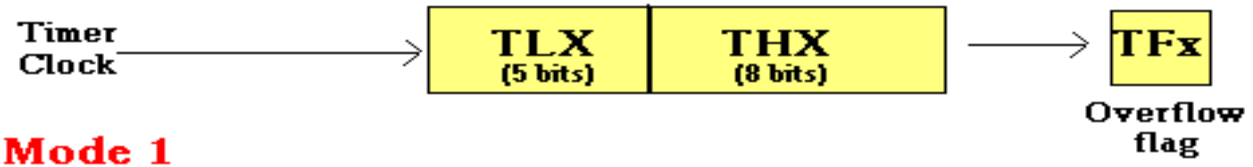


No modo 3,
***Timer* 1 cessa contagem** (o mesmo que TR1= '0' nos outros modos).
Se fora do modo 3, conta

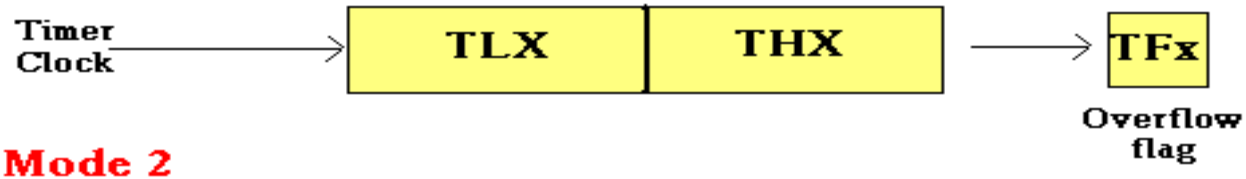
No modo 3,
***Timer* 0 implementa dois contadores de 8 bits.**
TR1 e TF1 são redirecionados para o *Timer* 0 (TH0)

Resumo

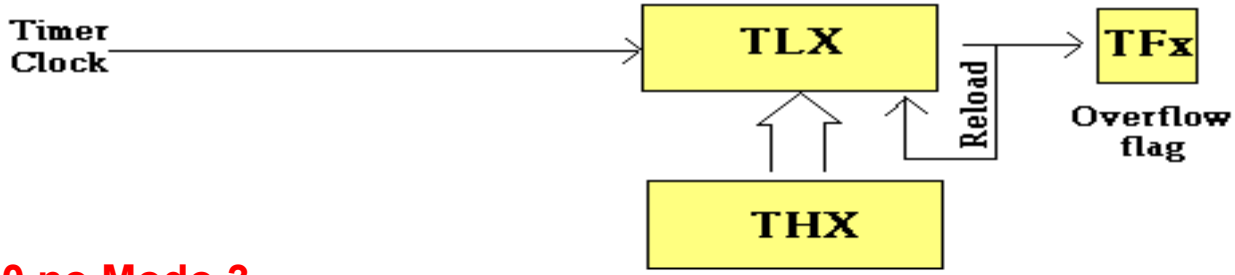
Mode 0



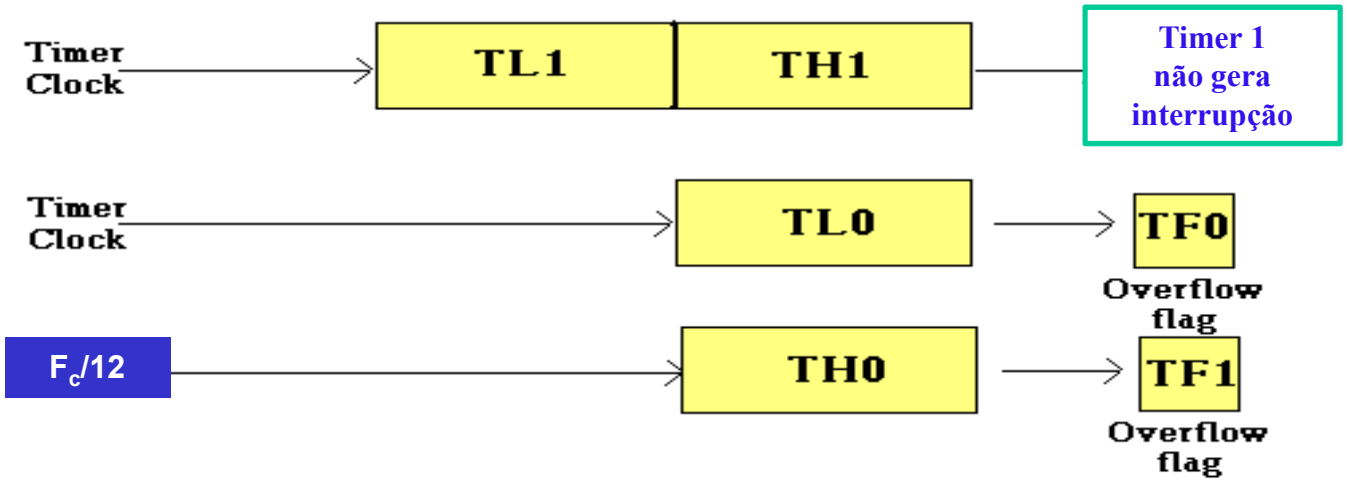
Mode 1



Mode 2

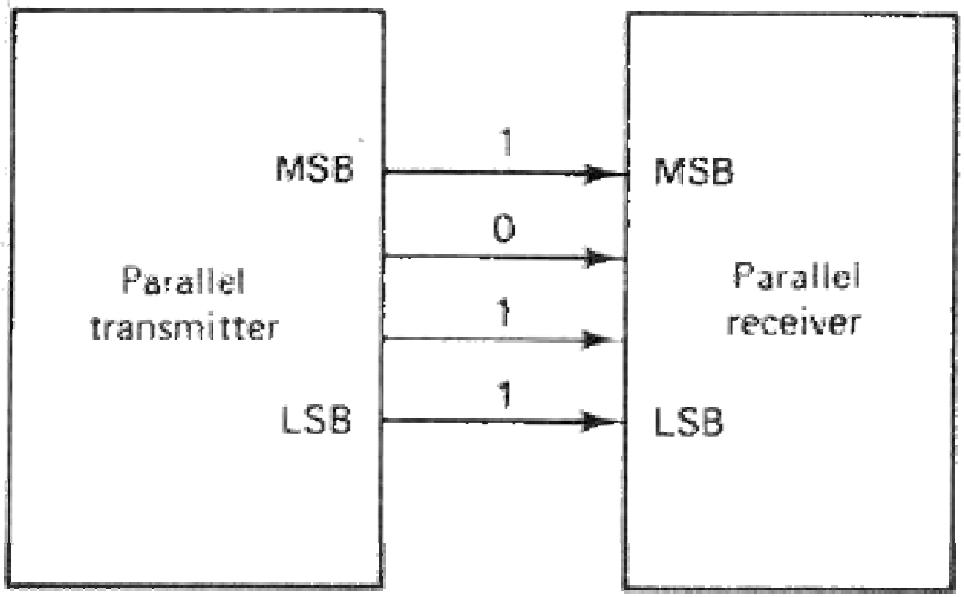


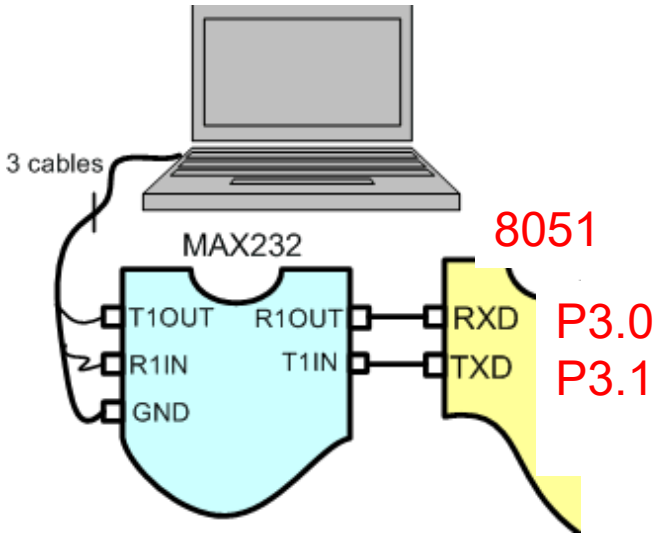
Timer 0 no Modo 3

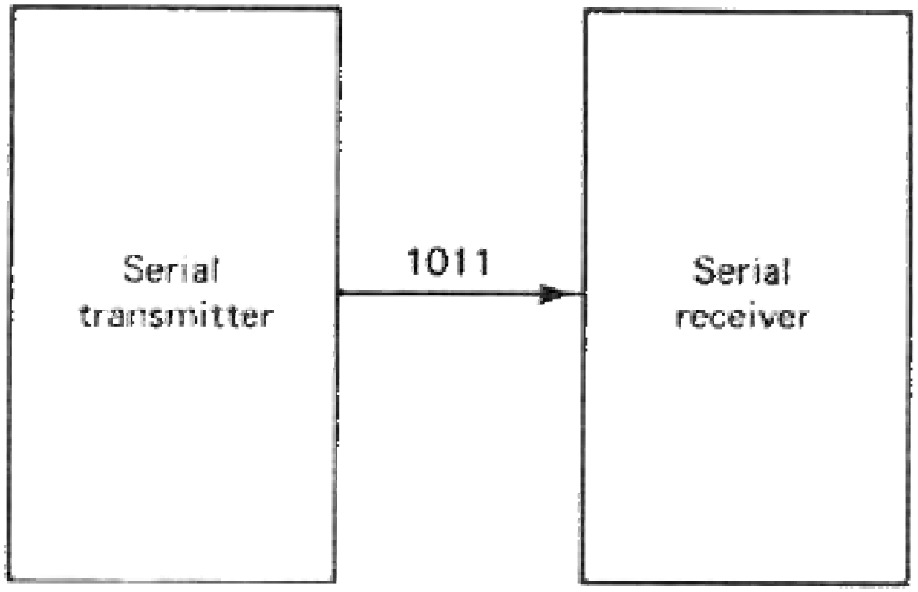


Substituir x por 0 (Timer0) ou 1 (Timer1)

Transmissão Serial



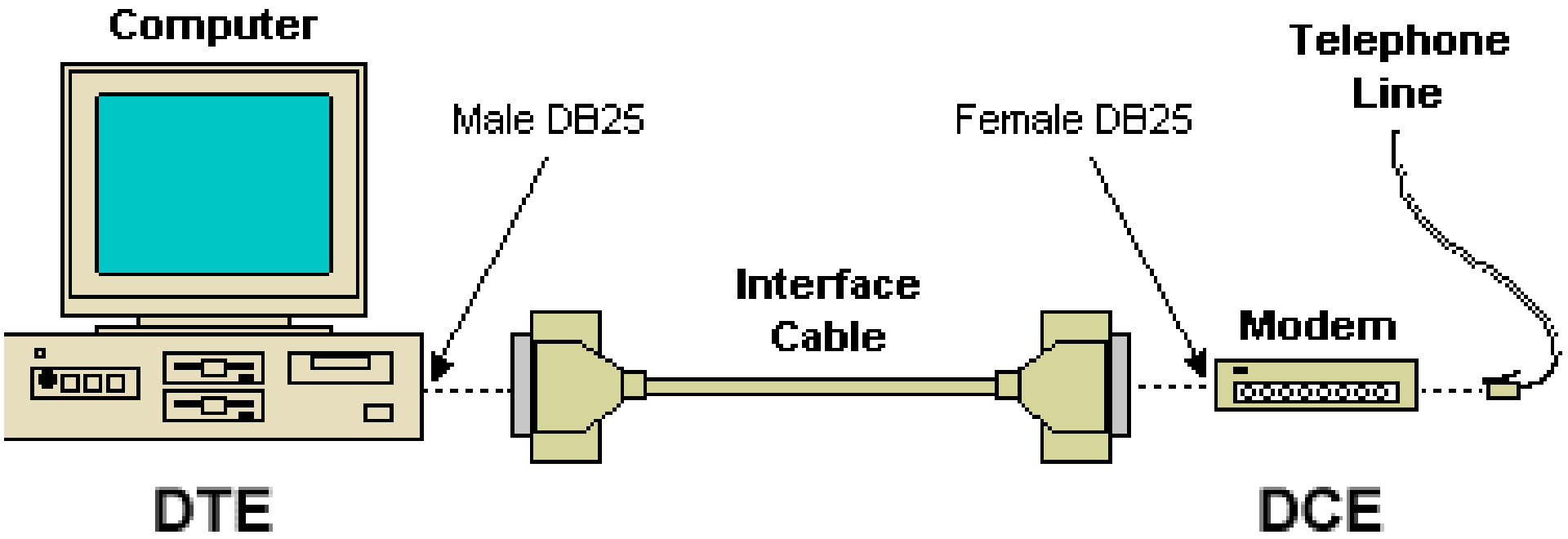




Transmissão Serial

USART: *Universal Synchronous Asynchronous Receiver Transmitter*

EIA RS-232C: padrão industrial para a comunicação serial de dados binários entre um **DTE** (terminal de dados) e um **DCE** (comunicador de dados).



Data Terminal Equipment

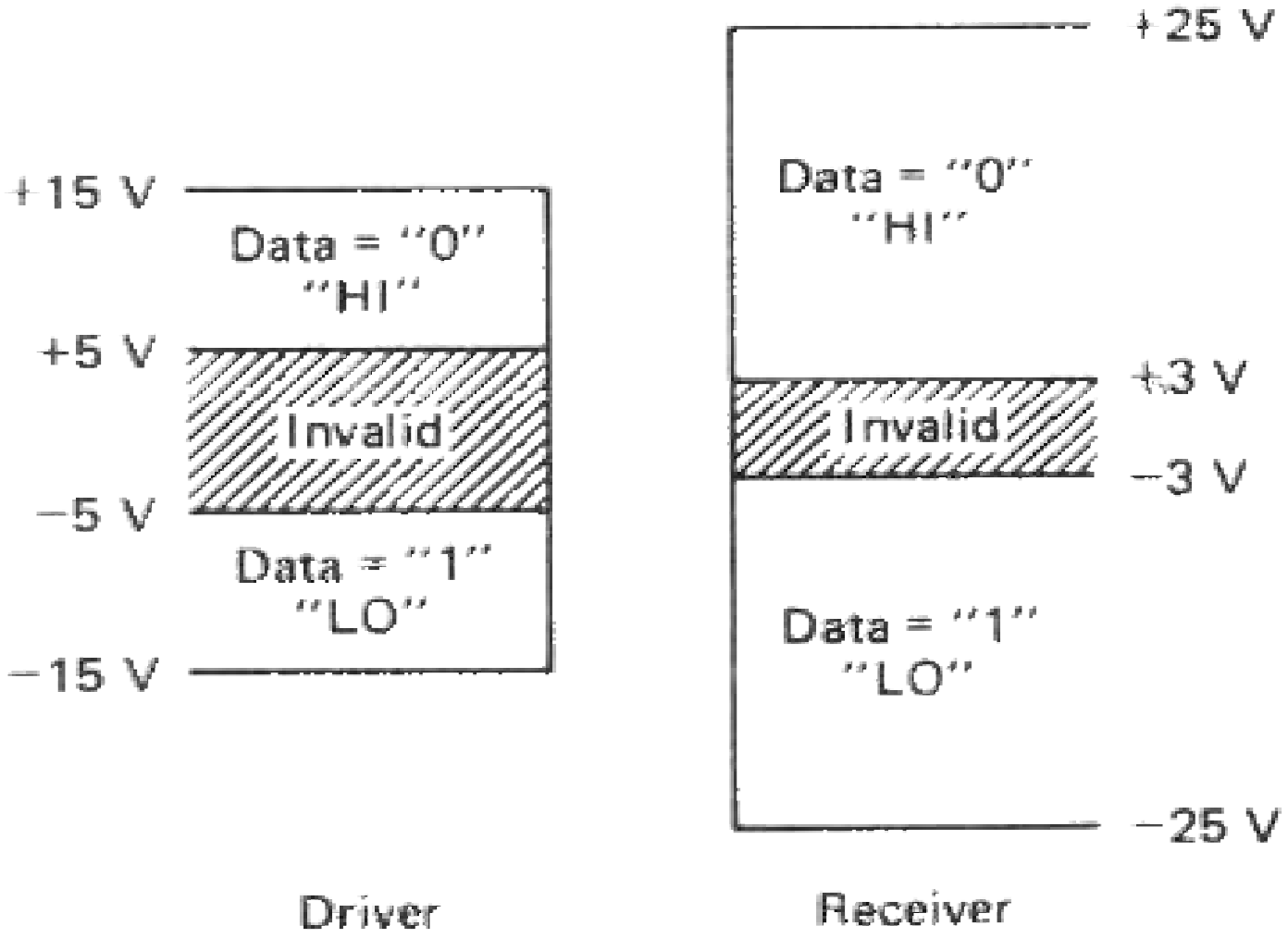
Data Communication Equipment

Foi largamente utilizado em PCs, estando ainda presente em muitos equipamentos.

Níveis Lógicos EIA RS-232C

(*Electronic Industries Association*)

- Tamanho máximo do cabo: 15 metros
- Velocidade de transmissão: 1Mbit/s



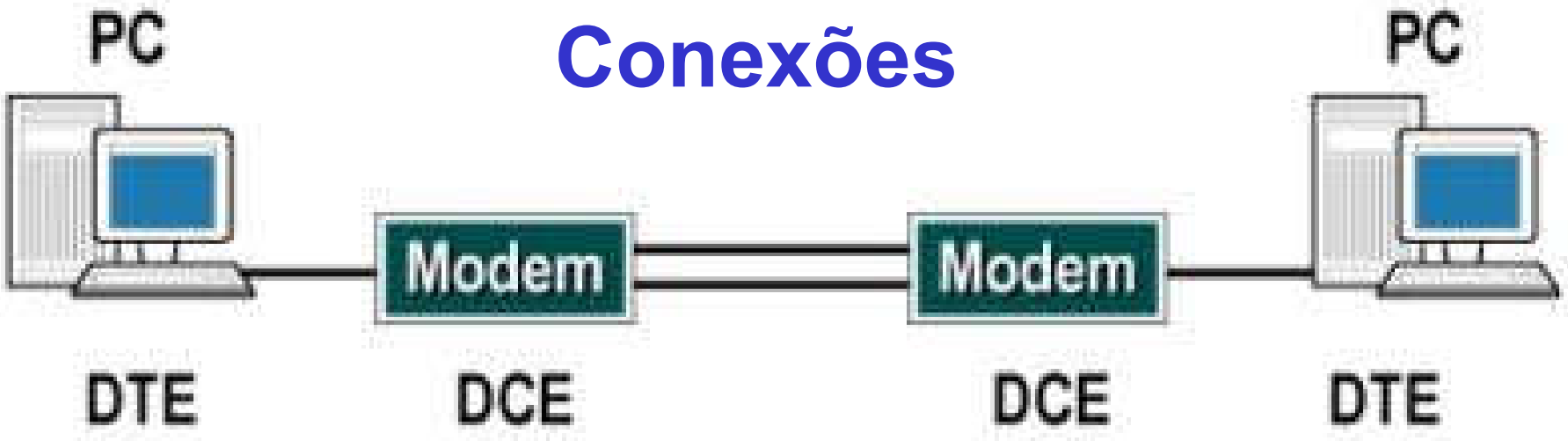
Conexões

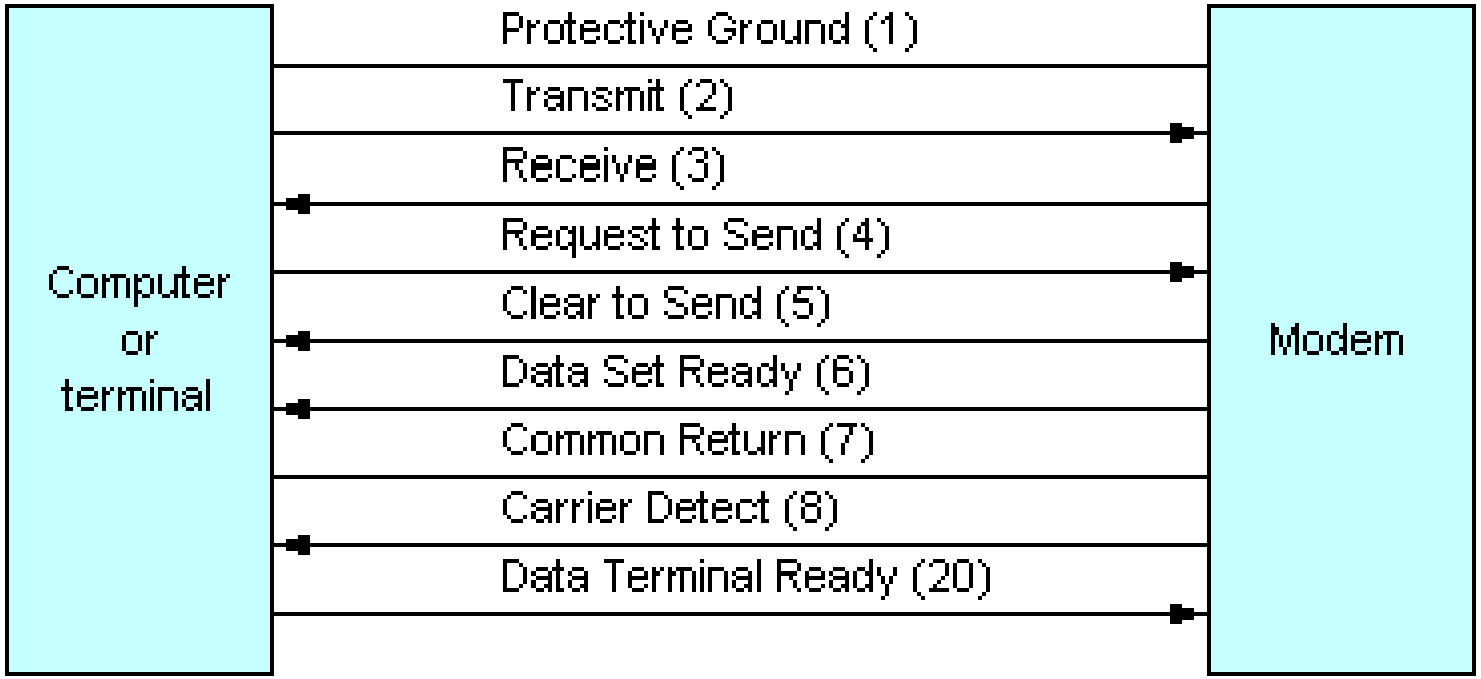
Pin				
DB25	DB9	Signal Name		Direction
1		CD	Chassis Ground	-
2	2	TD	Transmit Data	DTE → DCE
3	3	RD	Receive Data	DTE ← DCE
4	7	RTS	Request To Send	DTE → DCE
5	8	CTS	Clear To Send	DTE ← DCE
6	6	DSR	Data Set Ready	DTE ← DCE
7	5	SG	Signal Ground	-
8	1	DCD	Data Carrier Detect	DTE ← DCE
20	4	DTR	Data Terminal Ready	DTE → DCE
22	9	RI	Ring Indicator	DTE ← DCE

DTE Conector Macho

DCE Conector Fêmea

Conexões

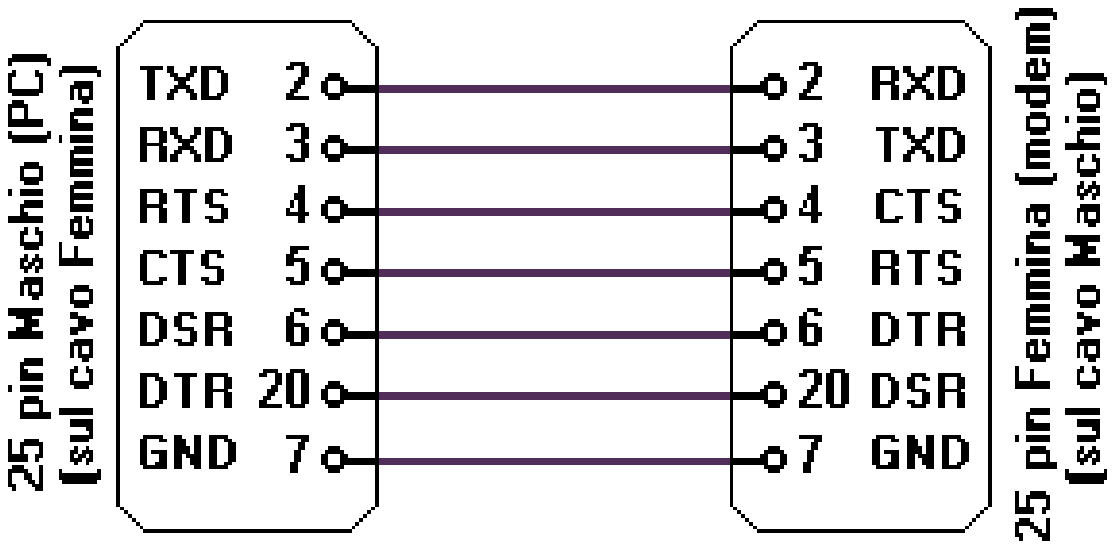




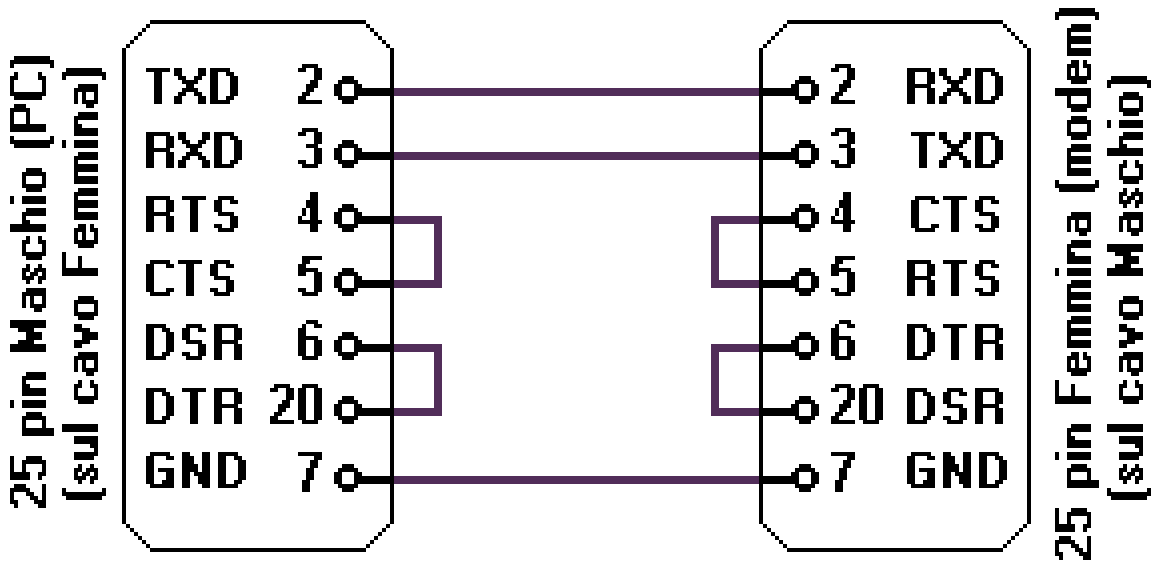
DCD (pino 8) indica quando um modem está conectado a outro modem remoto via linha telefônica

Conexões

Ideal



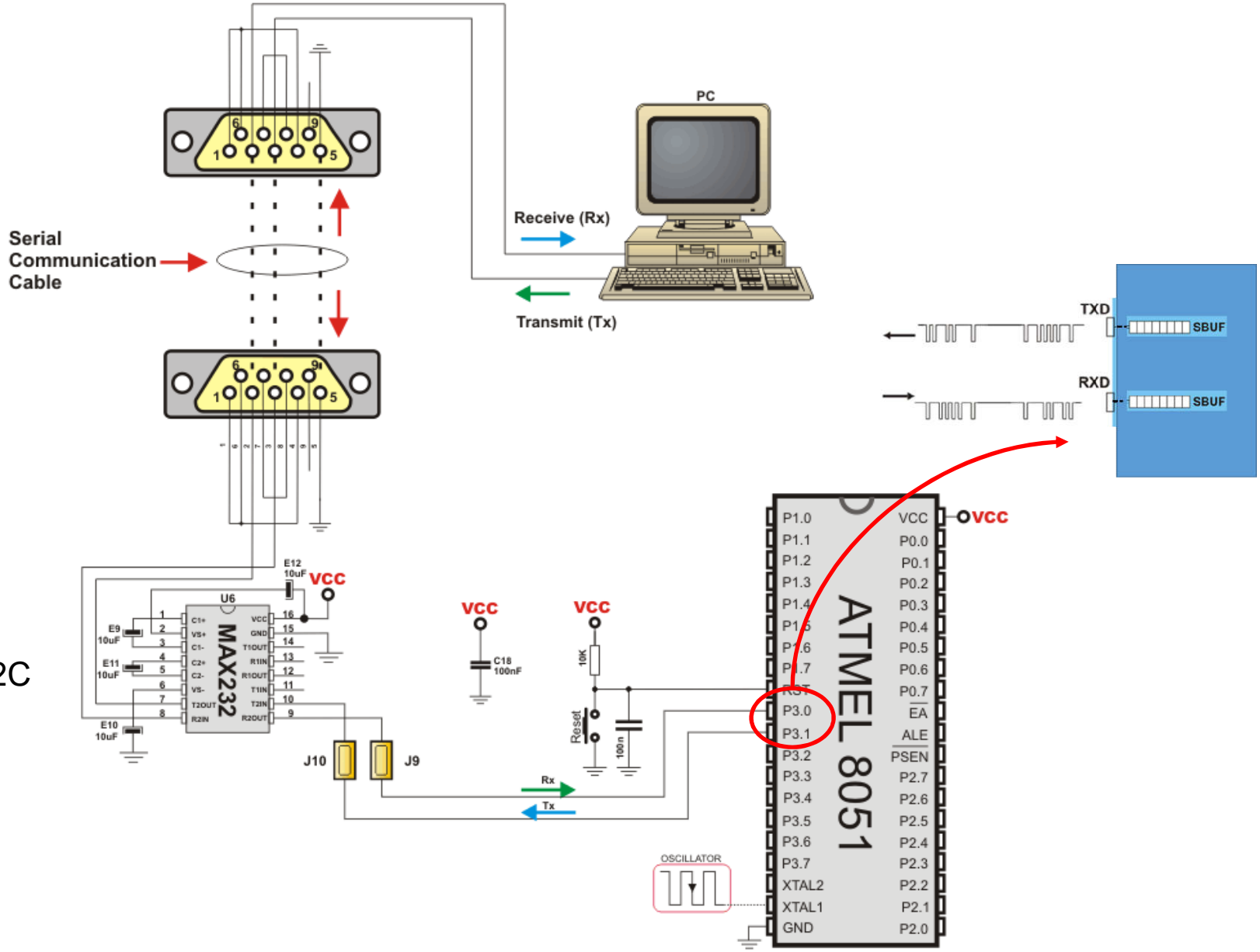
Mínimo



Handshaking

- **Troca de sinais para estabelecer comunicação condicional**
- **Os sinais DTR e DSR permitem que os dispositivos identifiquem que estão prontos para comunicação.**
- **Processo**
  - **Transmissor ativa RTS**
  - **Receptor detecta CTS por interrupção ou *polling***
  - **Receptor ativa RTS**
  - **Transmissor detecta CTS**
  - **Transmissor envia dados**

Comunicação serial entre PC e 8051



Conversor
TTL/RS232C

http://www.mikroe.com/chapters/view/70/chapter-7-development-systems/

SCON - Serial Port Control Register - Bit Addressable

7	6	5	4	3	2	1	0	
SM0	SM1	SM2	REN	TB8	RB8	TI	RI	98H

Modos
00 modo 0
01 modo 1
10 modo 2
11 modo 3

Habilita Comunicação Multiprocessador nos modos 2 e 3
Modo 1 => SM 2= 1 => RI = 1 qdo stop bit válido é recebido
Modo 0 => SM2 não tem função

Habilita Recepção

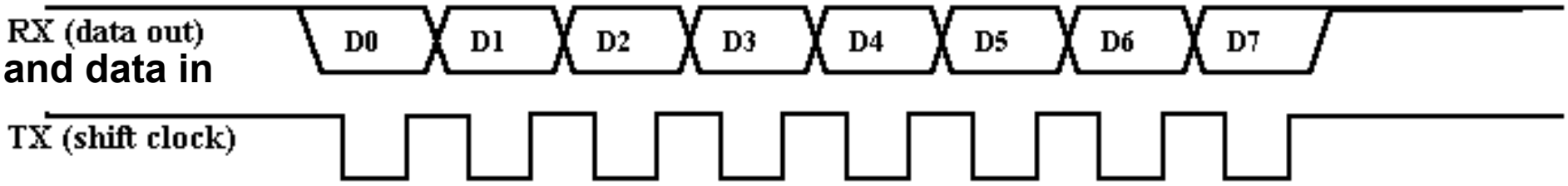
Received Interrupt
RI=1 qdo dado é recebido (exceto se SM2=1)
Resetado por software

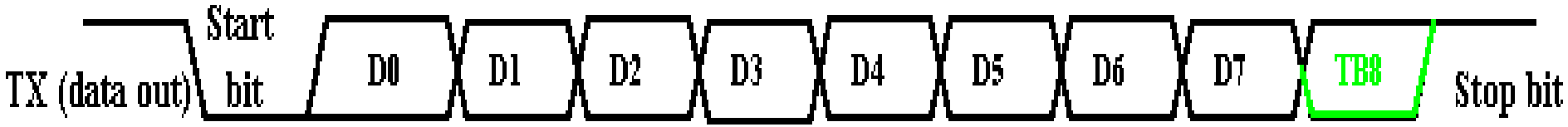
Transmited Interrupt
Resetado por software

Recebe nono bit (Modos 2 e 3)
Stop bit no Modo 1 (SM2=0)

Não utilizado no Modo 0

Nono bit transmitido (Modos 2 e 3)



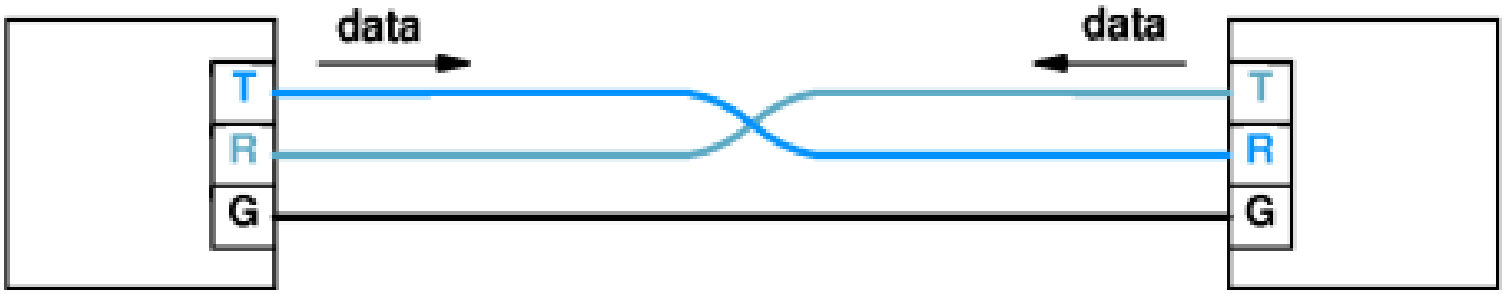


Modos de Programação da Serial

SM0	SM1	MODO	PROTOCOLO	BITS	TAXA
0	0	0	SÍNCRONO - HD	8	Fclock/12
0	1	1	ASSÍNCRONO - FD	10	Variável
1	0	2	ASSÍNCRONO - FD	11	Fclock/32 ou /64
1	1	3	ASSÍNCRONO - FD	11	Variável

Terminologia

Full Duplex (FD): dados são (ou podem ser) transmitidos e recebidos simultaneamente por ambos os dispositivos conectados, requerendo para tal, conexões adequadas.



Half Duplex (HD): dados são transmitidos ou recebidos por ambos os dispositivos conectados, mas não simultaneamente.

Programação *Baud Rate*

Modo 2 => BR= [(2^SMOD) / 64] x f osc

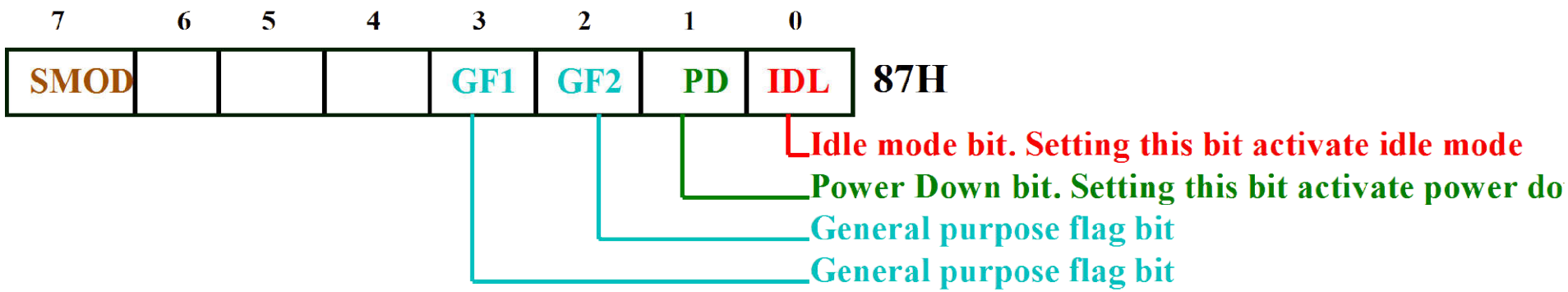
Modo 1 e 3 => Timer 1 é utilizado para estabelecer a *baud rate, de acordo com:*

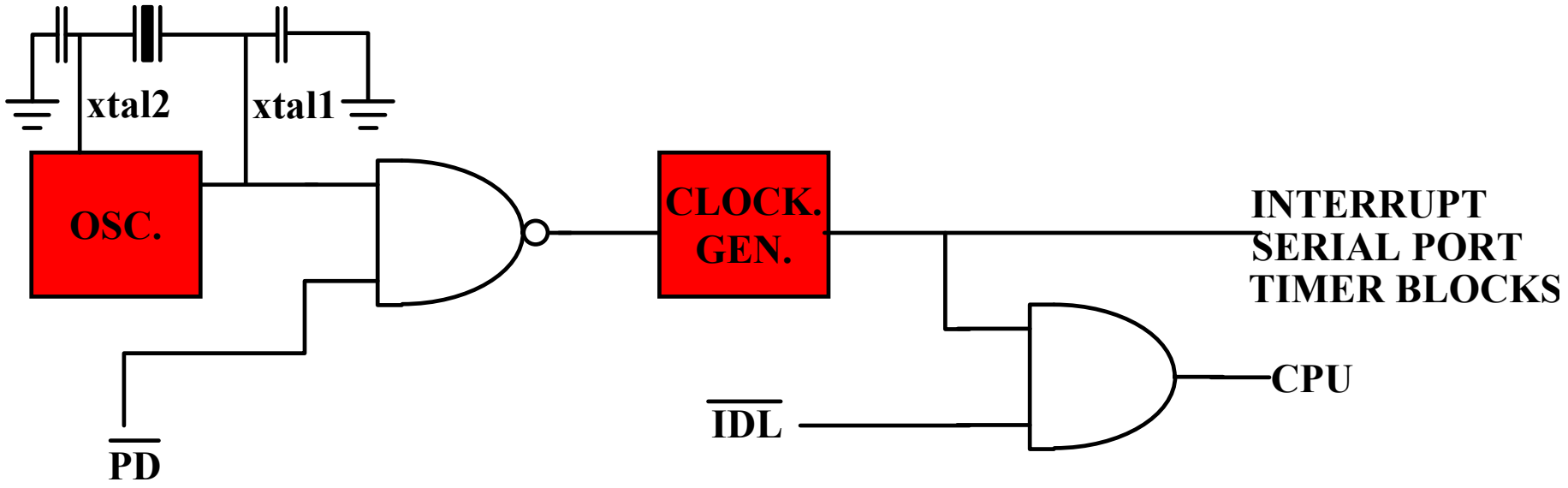
BR= [(2^SMOD) / 32] x { f osc / [12 x (256 - TH1)] }

OBS: Gate=0; TR1=1

			Timer 1		
Baud Rate	**Fosc**	**SMOD**	**C/T**	**Mode**	**Reload Value**
Mode 0 Max: 1 MHz	**12 MHz**	**X**	**X**	**X**	**X**
Mode 2 Max: 375K	**12 MHz**	**1**	**X**	**X**	**X**
Modes 1, 3: 62.5K	**12 MHz**	**1**	**0**	**2**	**FFH**
19.2K	**11.059 MHz**	**1**	**0**	**2**	**FDH**
9.6K	**11.059 MHz**	**0**	**0**	**2**	**FDH**
4.8K	**11.059 MHz**	**0**	**0**	**2**	**FAH**
2.4K	**11.059 MHz**	**0**	**0**	**2**	**F4H**
1.2K	**11.059 MHz**	**0**	**0**	**2**	**E8H**
137.5K	**11.986 MHz**	**0**	**0**	**2**	**1DH**
110K	**6 MHz**	**0**	**0**	**2**	**72H**
110K	**12 MHz**	**0**	**0**	**1**	**FEEBH**

PCON - Power Control Register





Para sair de Idle: por uma Interrupção que esteja habilitada ou por Reset.
Por Interrupção: vai para o tratador e após RETI , continua de onde parou.
Por Reset: continua de onde parou.

Para sair de Power Down - Somente por Reset (não altera RAM).

Programação da Interface Serial: Usando Interrupção

1) **Alocar tratador de interrupção da serial no endereço 23H;**

1) **Habilitar interrupção serial (setb ES; setb EA ou MOV IE,#10010000H);**

1) **Especificar o modo de funcionamento da serial (SCON). Habilitar recepção quando for o caso (REN = '1');**

1) **Nos modos 1 e 3, especificar taxa de transmissão/recepção de dados no Timer 1 (TMOD, TH1 e flip-flop SMOD do registrador PCON). No modo 2, especificar SMOD.**

1) **Disparar contagem (setb TR1);**

1) **Carregar dado em SBUF para iniciar transmissão. Carregar (TI = '1') ou ler (RI = '1') dado de SBUF quando da ocorrência da interrupção.**

1) **Ao saltar para o tratador de interrupção serial, limpar flag (TI e/ou RI) que a solicitou.**

OBS: É possível testar, frequentemente, se os flags RI e TI estão setados (indicando buffer de recepção cheio ou buffer de transmissão vazio, respectivamente) sem a necessidade de habilitar as interrupções (*polling* dos *flags*) em programas mais dedicados.

Faça um programa que transmita, serialmente, a cadeia de 16 caracteres: 'Microcontrolador'. Realize a transmissão de forma assíncrona sem envio de bit de paridade à taxa de 9600 bauds (bits/s).

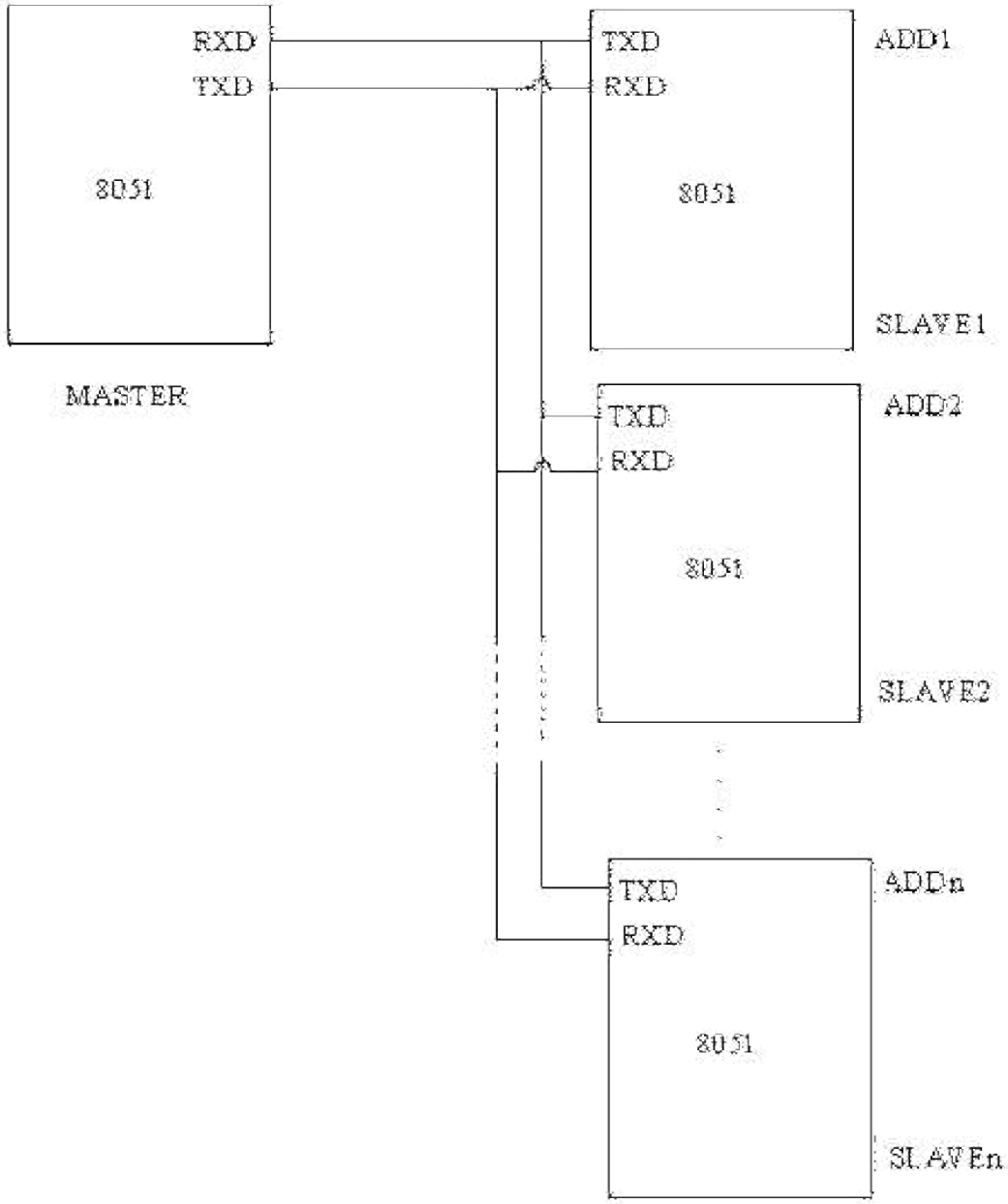
```
RESET       EQU     00H
LTSERIAL    EQU     23H ; local tratador
STATE       EQU     20H

    ORG RESET    ;PC=0 depois de reset
    JMP INICIO

    ORG     LTSERIAL
    CLR     TI
    MOV     STATE,#1H
    RETI

INICIO:  MOV       IE,#10010000B
         MOV       SCON,#01000000B
         MOV       TMOD,#00100000B
         MOV       TH1,#0FDH
         MOV       TL1,#0FDH
         MOV       PCON,#0H
         SETB      TR1
         MOV       STATE,#0H
         MOV       R0,# STATE
         MOV       DPTR,#TABELA
         MOV       R1,#1
         MOV       SBUF,# 'M'

VOLTA:   CJNE      @R0,#1,VOLTA
         MOV        STATE,#0H
         MOV       A,R1
         MOVC      A,@A+DPTR
         MOV       SBUF,A
         INC       R1
         CJNE      R1,#16,VOLTA
         CLR       TR1
         JMP       $

TABELA: DB 'Microcontrolador'

         END
```

Programação da Interface Serial: Usando *Polling*

Faça um programa que transmita, serialmente, a cadeia de 16 caracteres: 'Microcontrolador'. Realize a transmissão de forma assíncrona sem envio de bit de paridade à taxa de 9600 bauds (bits/s).

```
RESET       EQU     00H

    ORG   RESET   ;PC=0 depois de reset
    JMP   INICIO


INICIO:  MOV     SCON,#01000000B
         MOV     TMOD,#00100000B
         MOV     TH1,#0FDH
         MOV     TL1,#0FDH
         MOV     PCON,#0H
         SETB    TR1
         CLR     TI
         MOV     DPTR,#TABELA
         MOV     R1,#1
         MOV     SBUF,#'M'

VOLTA:   JNB     TI,VOLTA
         CLR      TI
         MOV     A,R1
         MOVC    A,@A+DPTR
         MOV     SBUF,A
         INC     R1
         CJNE    R1,#16,VOLTA
         CLR     TR1
         JMP     $

TABELA: DB 'Microcontrolador'
         END
```

Comunicação serial emtre múltiplos 8051 – SM2 = '1'



- **SM2: Habilita comunicação serial com múltiplos processadores nos modos 2 e 3 (modos com 9 bits de dados)**

- **Nestes modos, se SM2 for colocado em '1', RI só será setado (solicitando a execução de seu tratador de interrupção) quando o nono bit recebido (RB8) for '1'.**

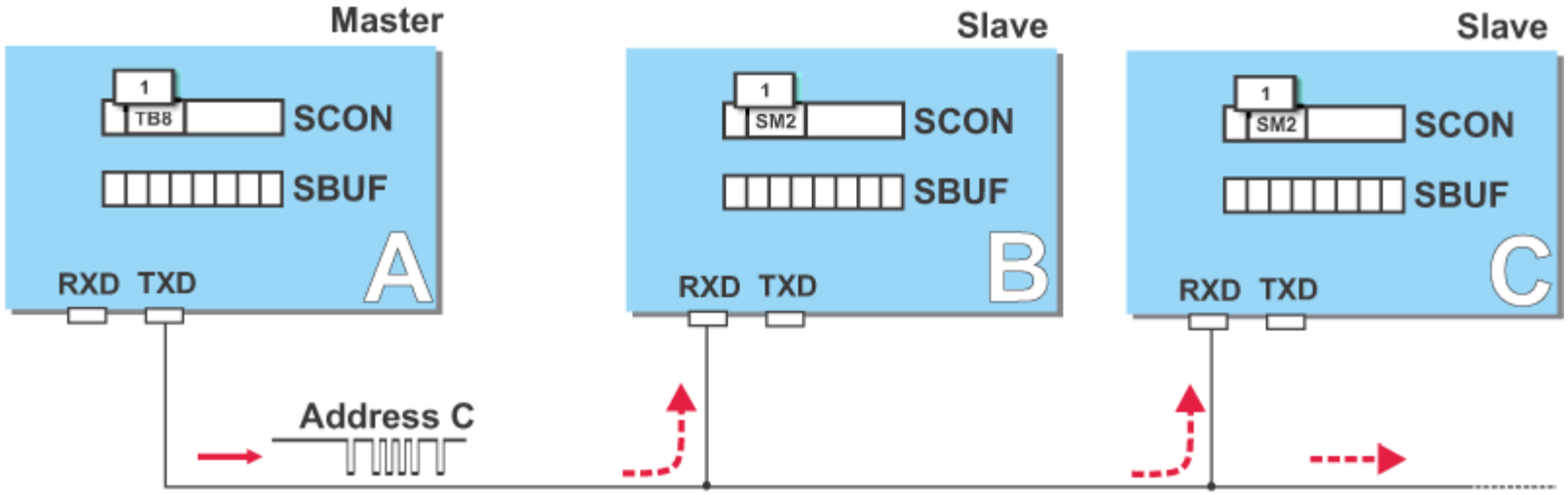
- **Desta forma, é possível que um dispositivo (microcomputador, por exemplo) comunique-se com vários 8051 em um mesmo barramento (TB8 = '1') e "peça" para um deles modificar SM2 (SM2 = '0'). A partir de então, dados enviados com o nono bit igual a '0' (TB8 = '0') serão detectados apenas pelo 8051 que modificou o SM2 (SM2 = '0'), pois este demanda a execução do seu tratador de interrupção.**

- **Ao término desta comunicação, este 8051 deve voltar a setar SM2 para que tal processo possa ser realizado com um outro 8051**

Comunicação Multi-processadores - 8051

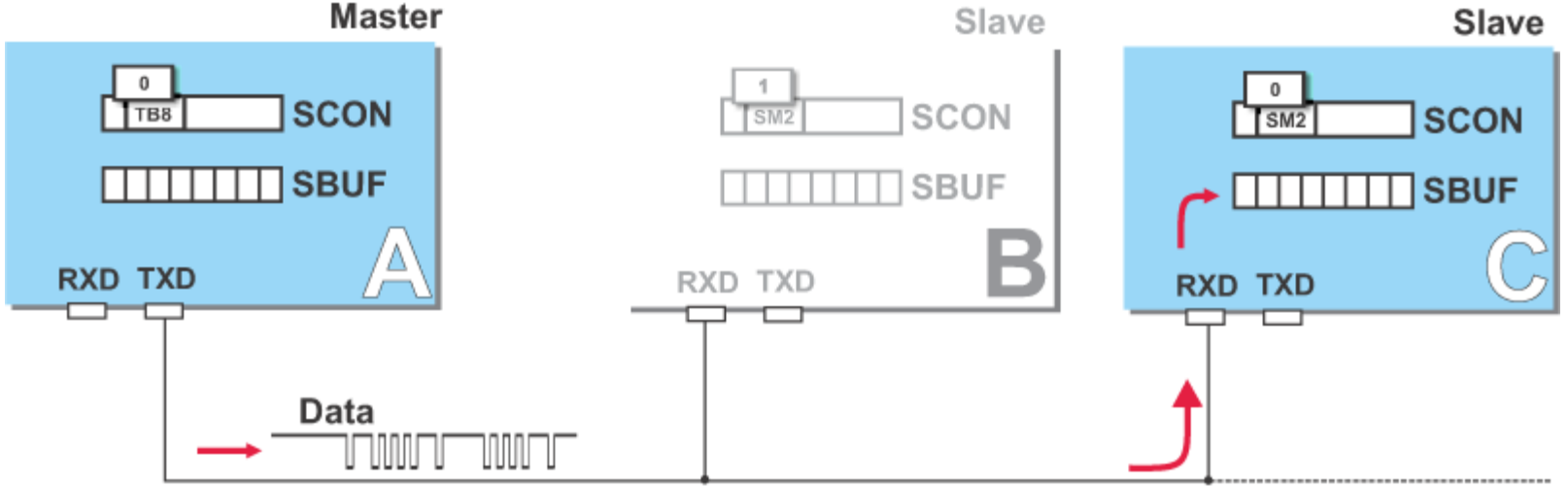
Transmissão de endereço para selecionar 8051

(TB8 do Mestre: '1' - SM2 dos slaves: '1')

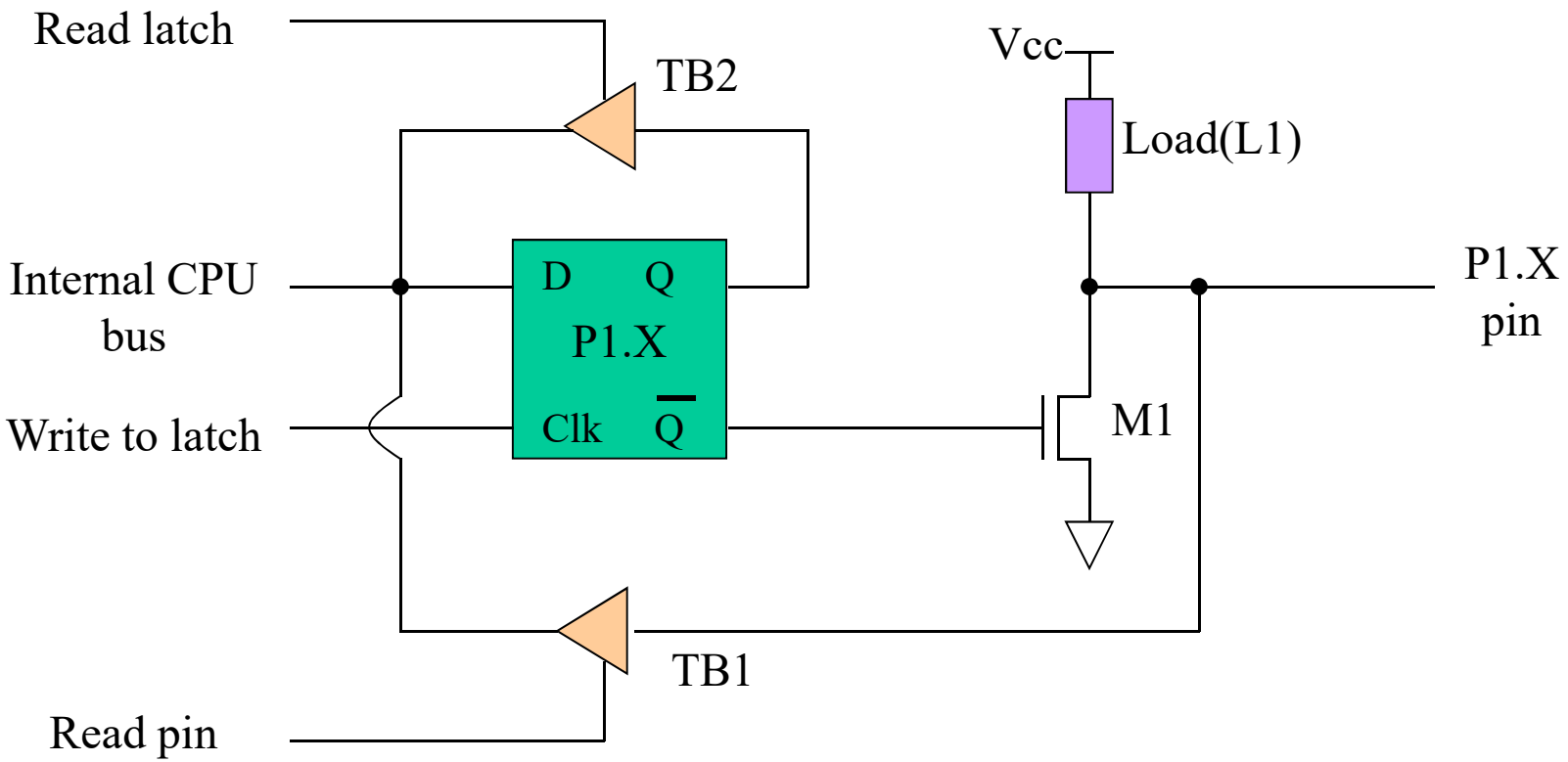


Comunicação com 8051 selecionado

(TB8 do Mestre: '0' - SM2 do slave endereçado: '0')

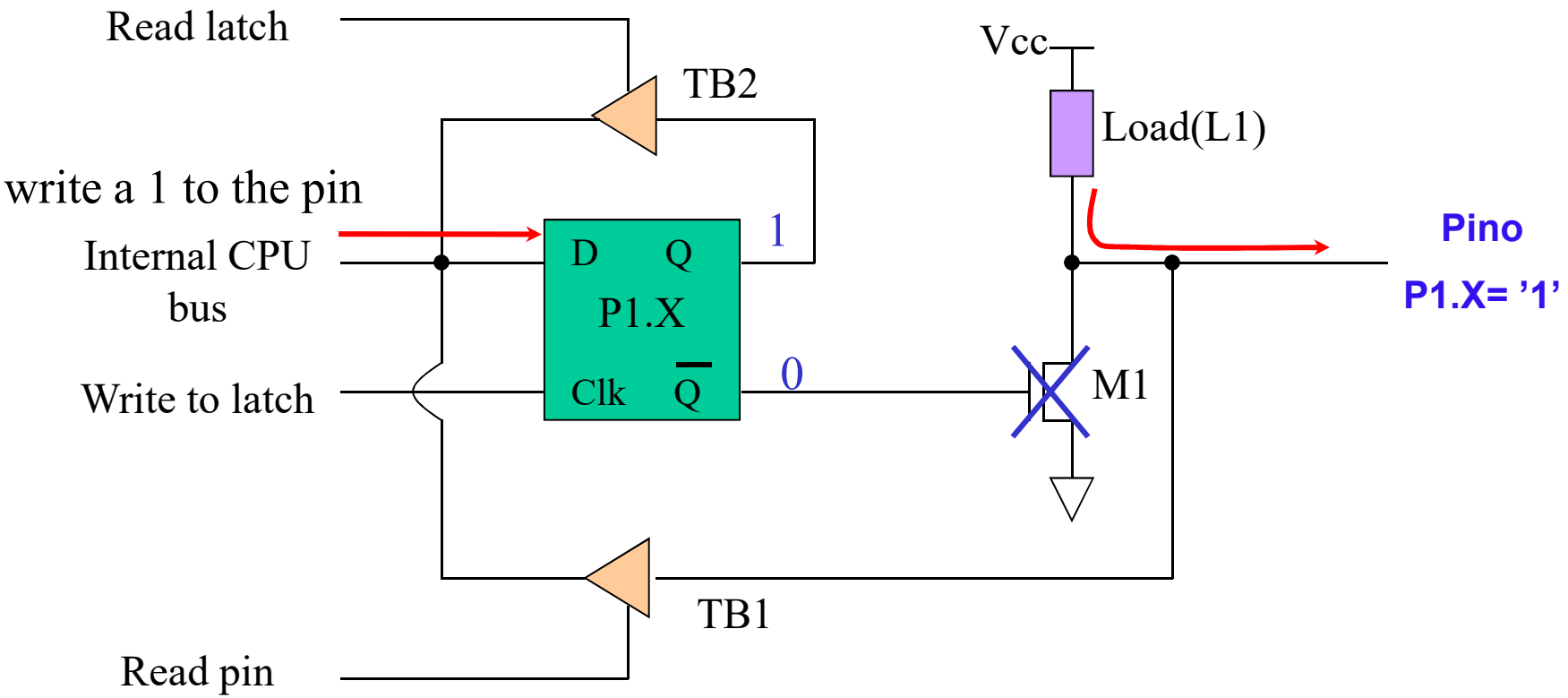


Organização Pinos da Porta P1



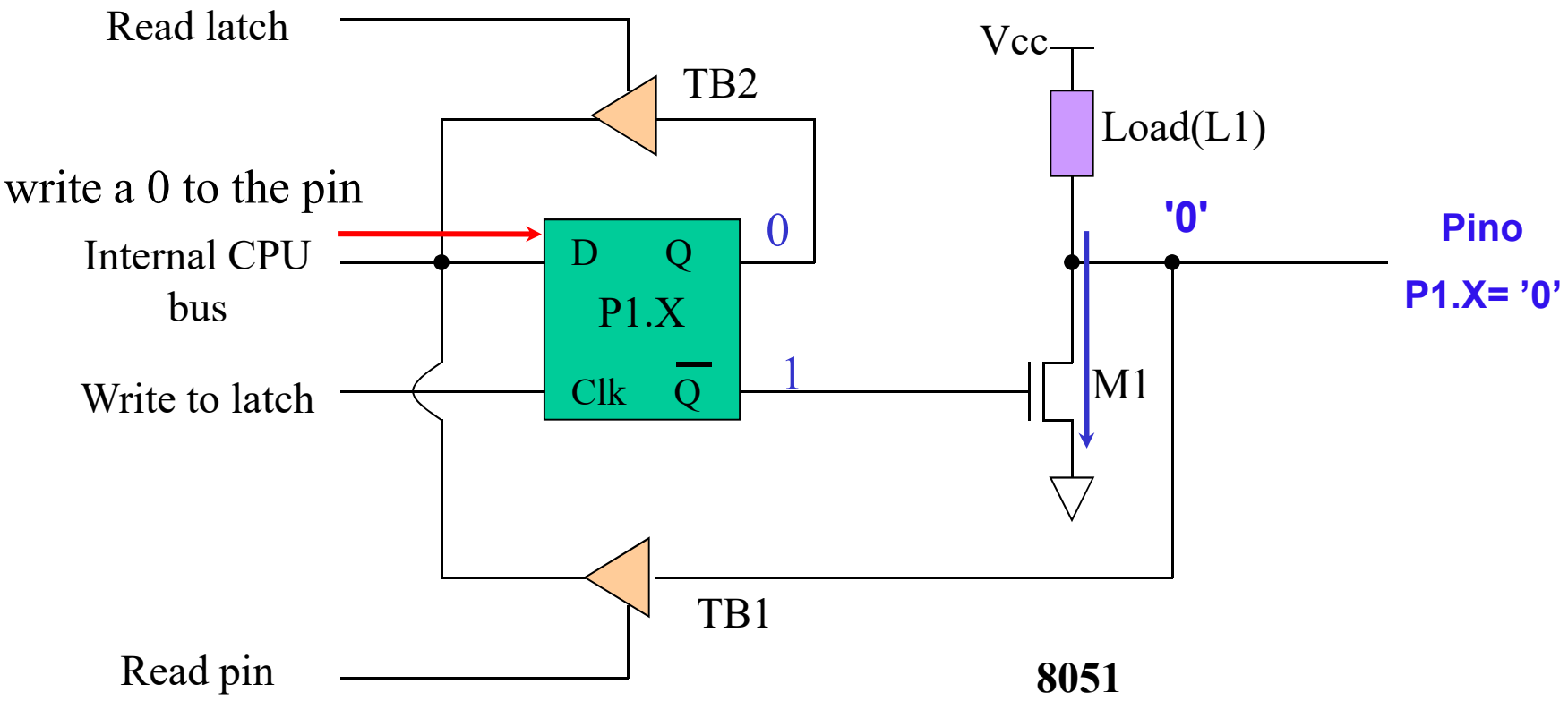
8051

Escrevendo '1' em Pino P1.X



8051

Escrevendo '0' em Pino P1.X

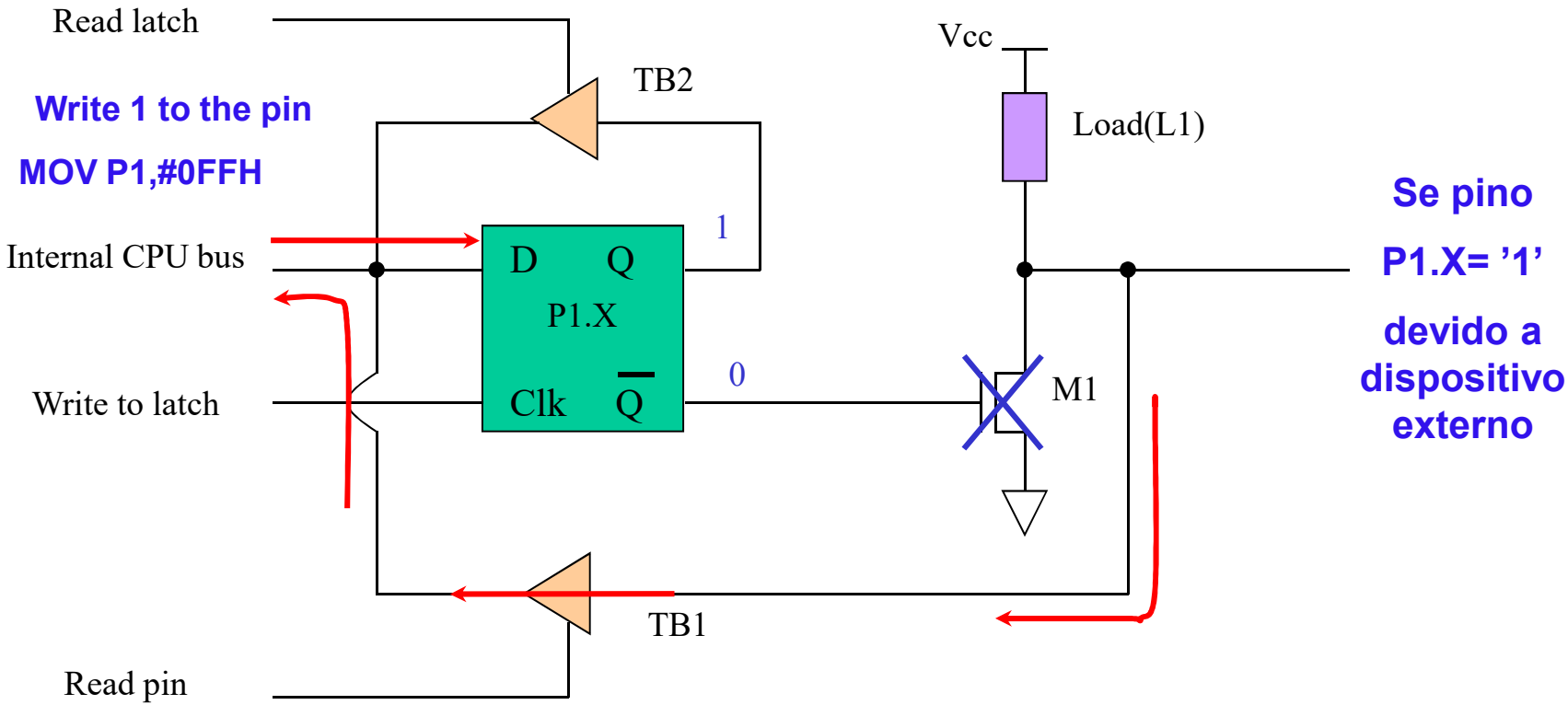


Após tal escrita , tem-se da leitura

MOV A,P1

ACC.X= '0' (independe do valor externo colocado no pino)

Lendo '1' em Pino P1.X



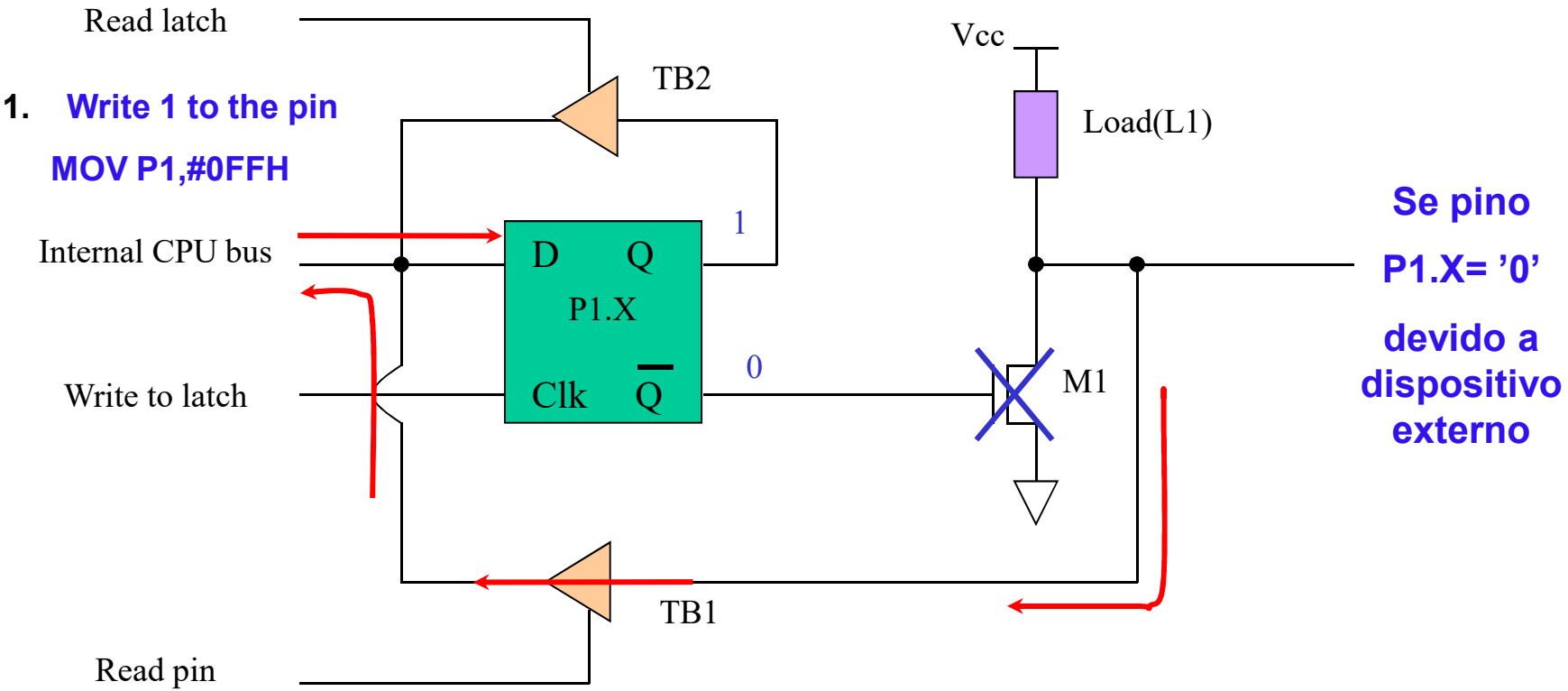
Após tal escrita , tem-se da leitura

MOV A,P1

ACC.X= '1'

8051

Lendo '0' em Pino P1.X



Após tal escrita , tem-se

MOV A,P1

ACC.X= '0'

8051

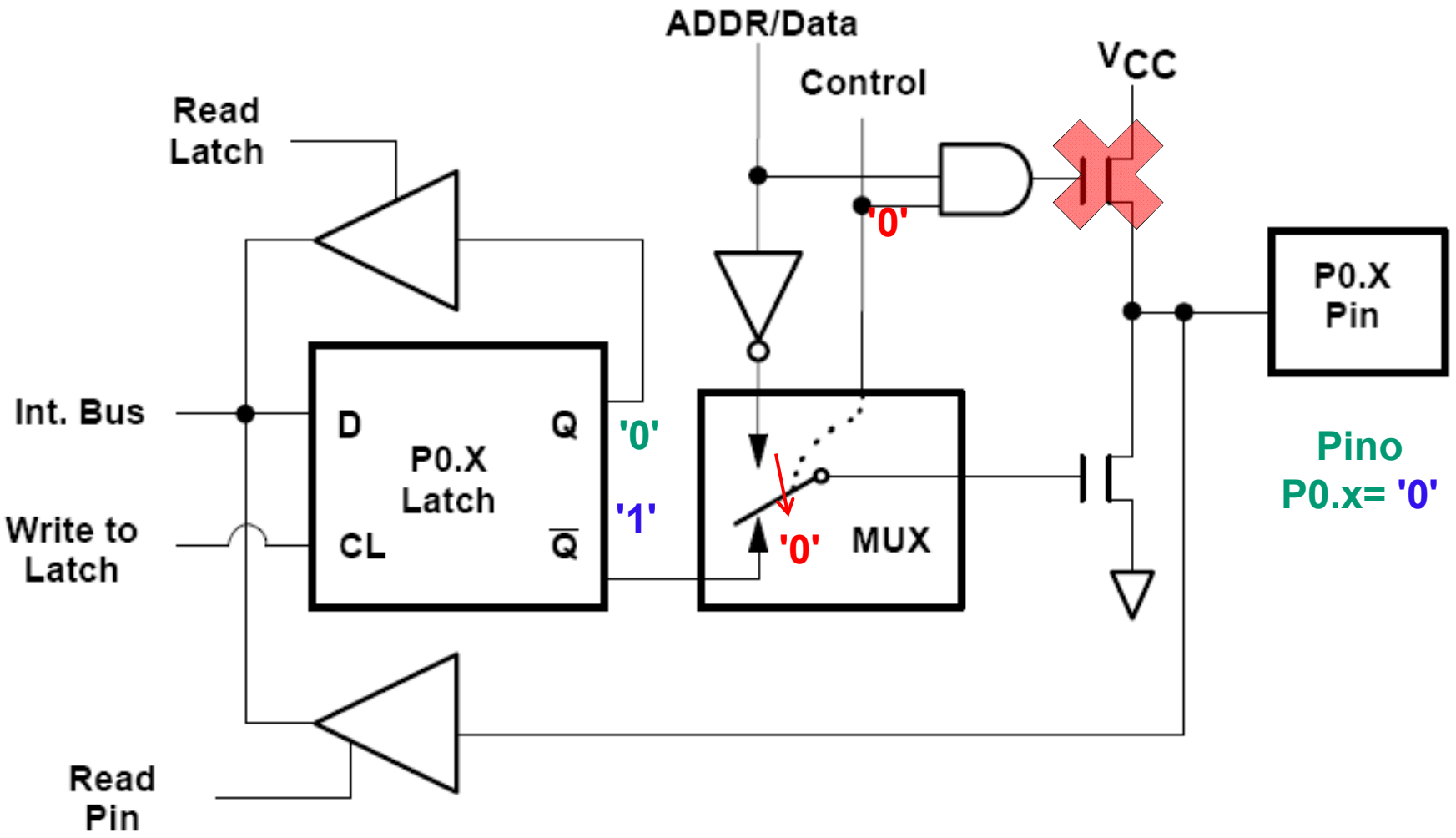
Lendo do Latch de Porta

Algumas instruções fazem a leitura do Pino (TB1) e outros da saída do Latch (TB2)

Instruções que leem do latch (“lê-modifica-escreve”). Exemplos:

ANL	**AND lógico**	**ex. ANL P1,A**
ORL	**OR lógico**	**ex. ORL P2,A**
XRL	**XOR lógico**	**ex. XRL P3,A**
CPL	**complementa bit**	**ex. CPL P3.0**
INC	**incrementa**	**ex. INC P2**
DEC	**decrementa**	**ex. DEC P2**
DJNZ	**decrementa e salta se não zero**	**ex. DJNZ P3,LABEL**
MOV PX.Y,C	**move bit de carry para bit Y da Port X**	
CLR PX.Y	**limpa bit Y da Port X**	
SETB PX.Y	**seta bit Y da Port X**	

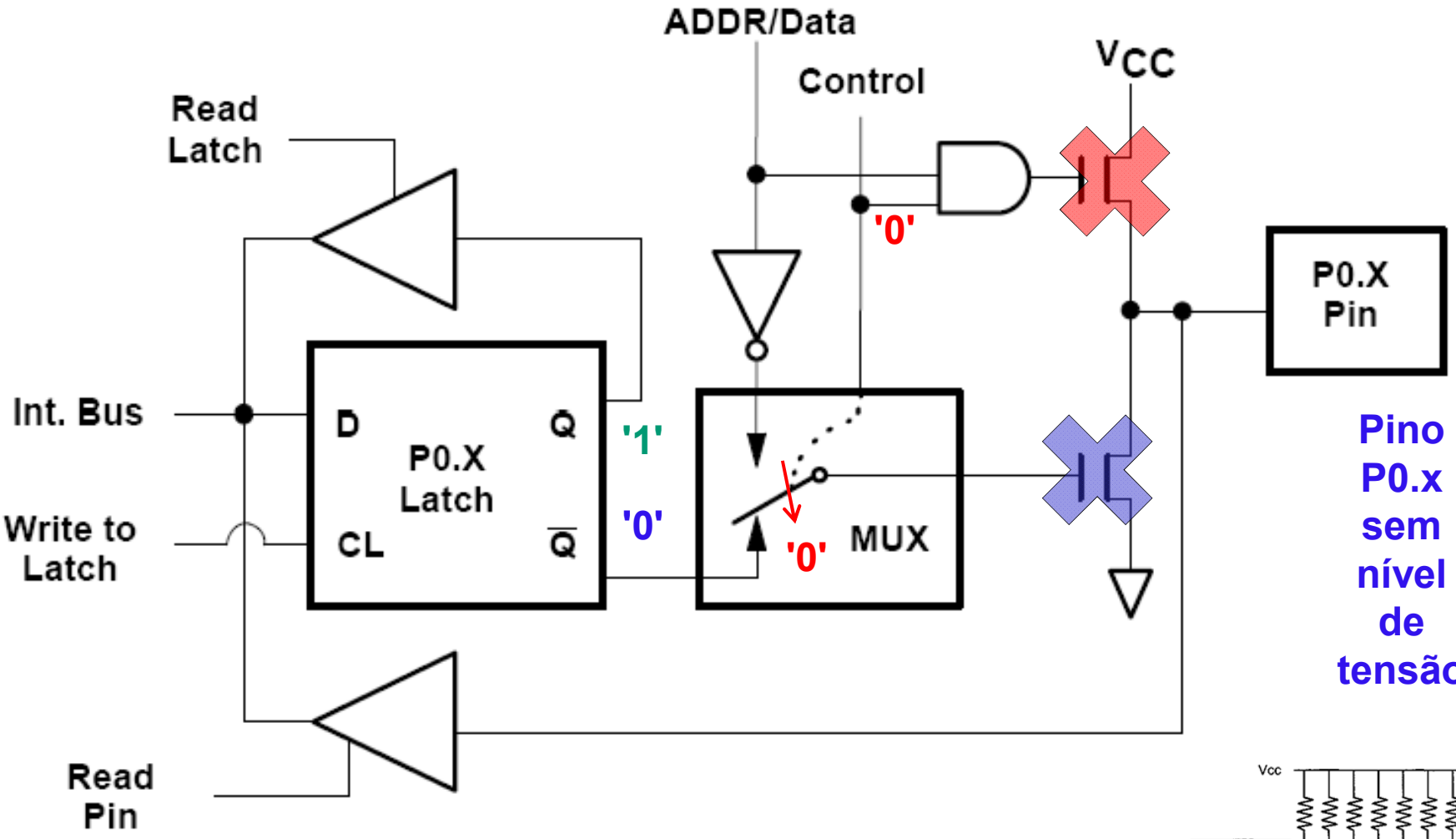
Porta 0 – MOV P0,#0 – Control '0'

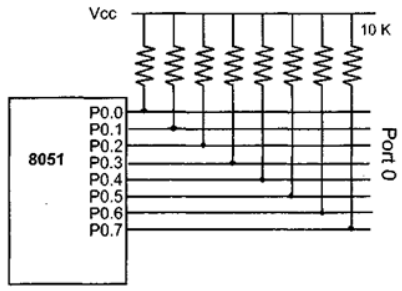


a. Port 0 Bit

Porta 0 – MOV P0,#0FFH – Control '0'

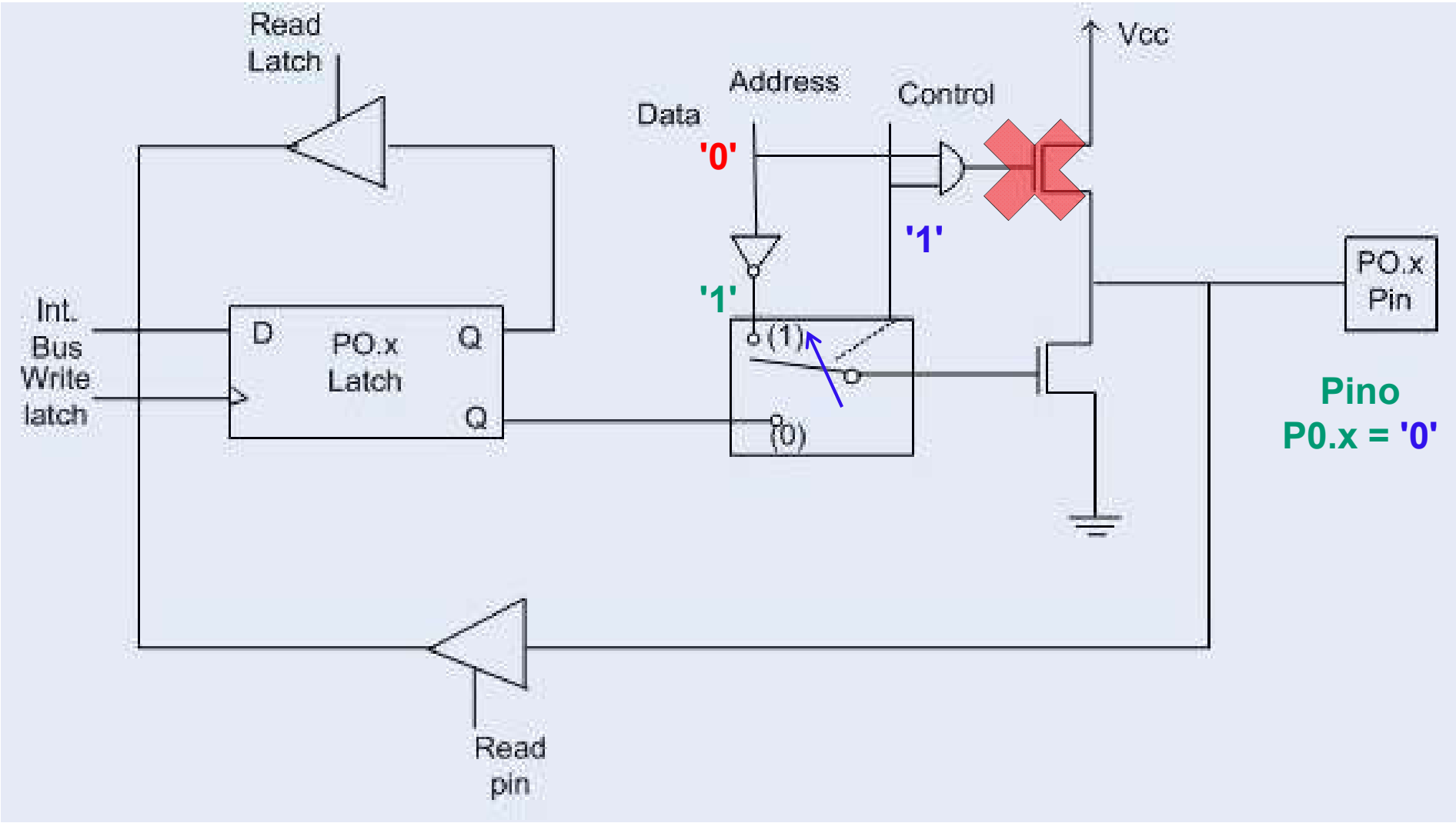
Não possui resistor de *pull up*





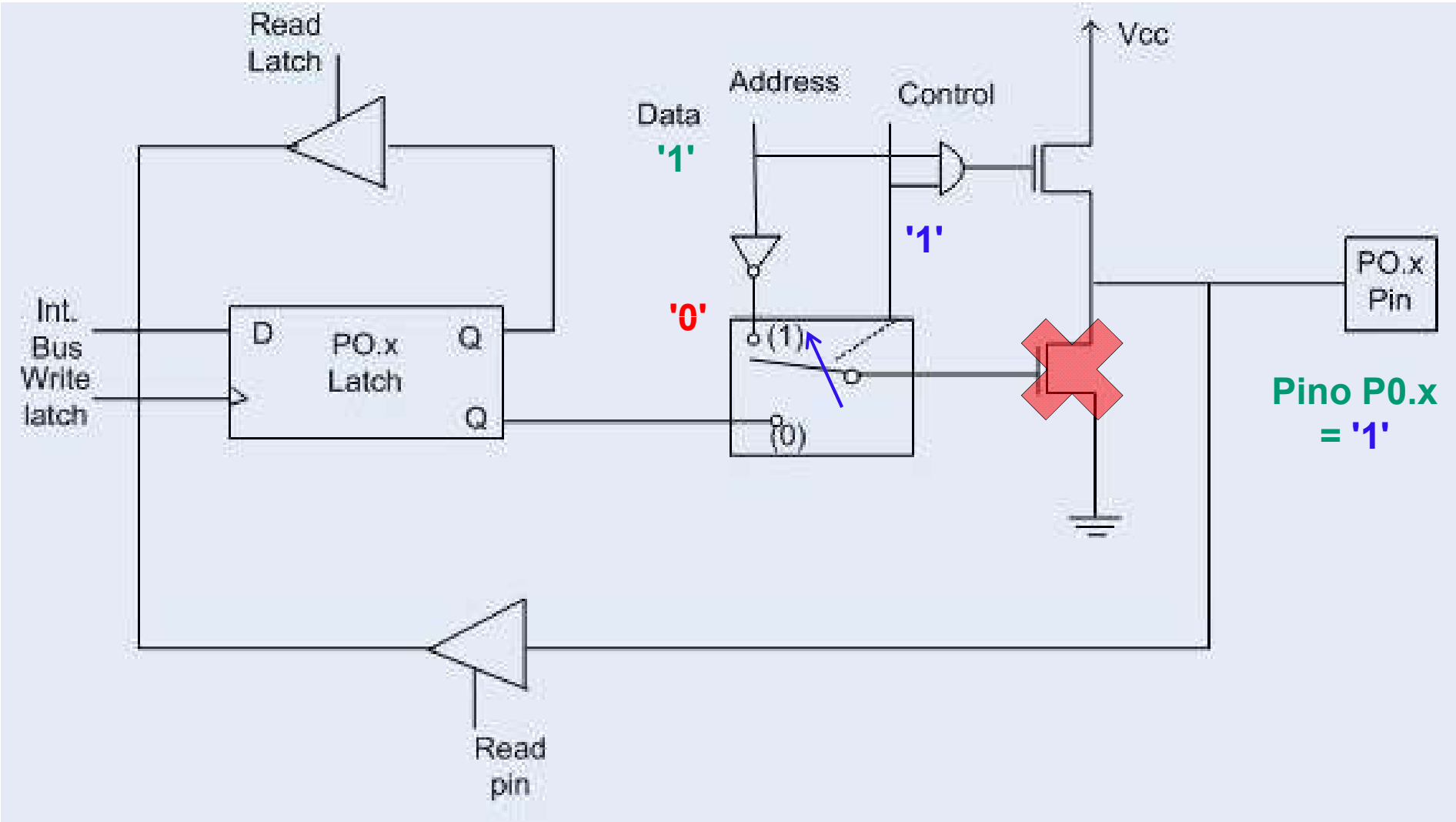
Porta 0 – Acesso à memória externa – Control '1'

Escrita de nível lógico baixo '0' em memória XRAM



Porta 0 – Acesso à memória externa – Control '1'

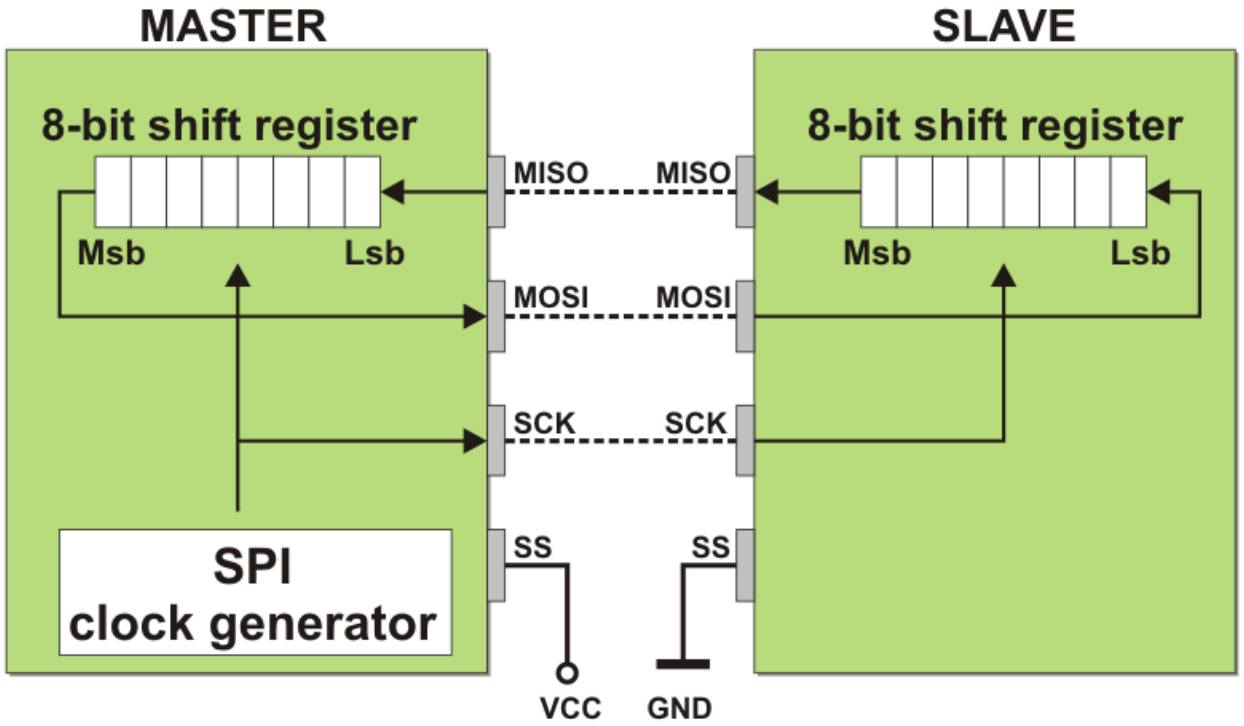
Escrita de nível lógico alto '1' em memória XRAM



Outros Modos de Transmissão Serial

SPI: Serial Peripheral Interface

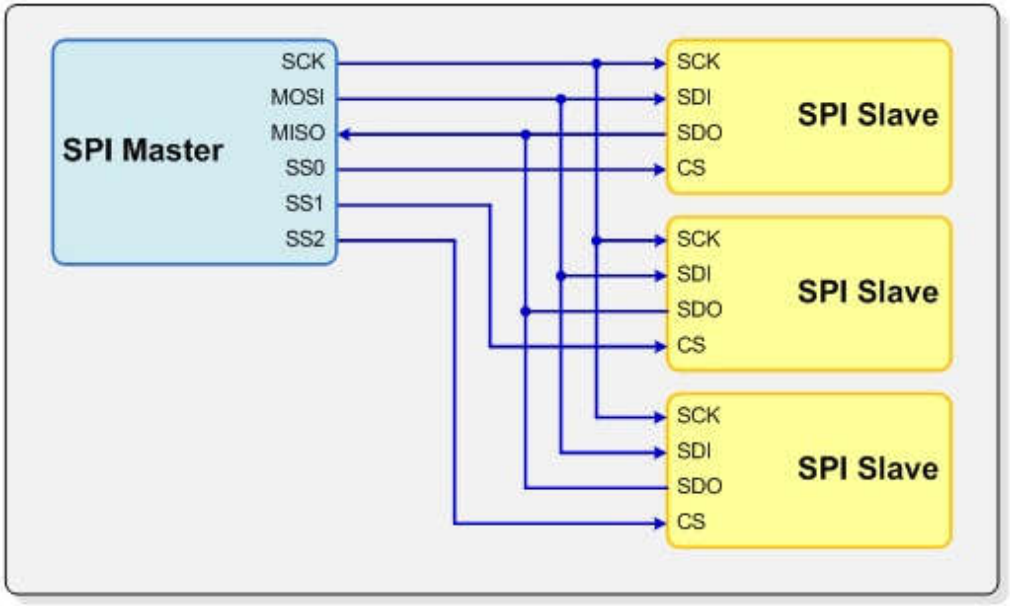
Transmissão Serial Síncrona



MOSI: Master Out Slave In
MISO: Master In Slave Out
SCK: Serial Clock (suprido pelo mestre)
SS: Slave Select

- Full Duplex, sem controle do fluxo de dados. (para receber, necessário enviar dado)
- Velocidade de transmissão de dados de dezenas de Mbi/s

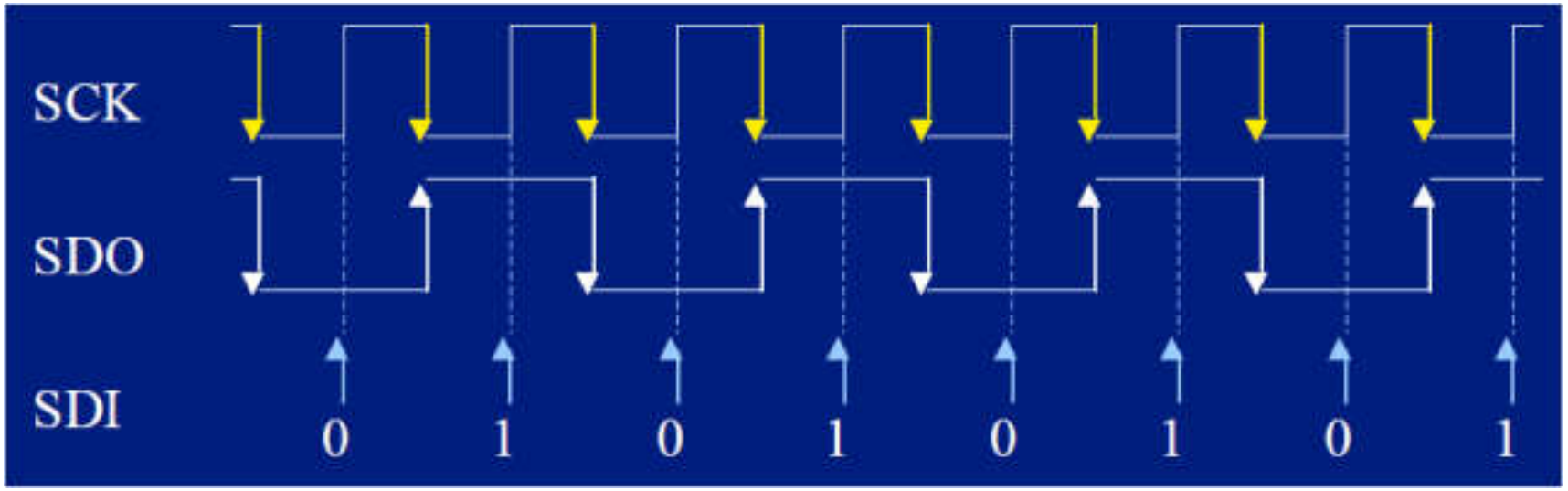
SPI - Conexão com múltiplos escravos



SCK: Serial Clock (suprido pelo mestre)
MOSI: Master Out Slave In
MISO: Master In Slave Out
SDI: Serial Data In
SDO: Serial Data Out
CS, **SSx**: Chip Select, Slave Select

SPI – Exemplo de envio/recepção de dados – CPOL=CPHA=1

dado atualizado pelo mestre na borda de descida; dado amostrado pelo escravo na borda de subida do clock



CPOL: 0 - valor base do clock = 0

CPHA: 0 - dado amostrado pelo escravo na borda de subida do clock
(dado atualizado pelo mestre na borda de descida)

CPHA: 1 - dado amostrado pelo escravo na borda de descida do clock
(dado atualizado pelo mestre na borda de subida)

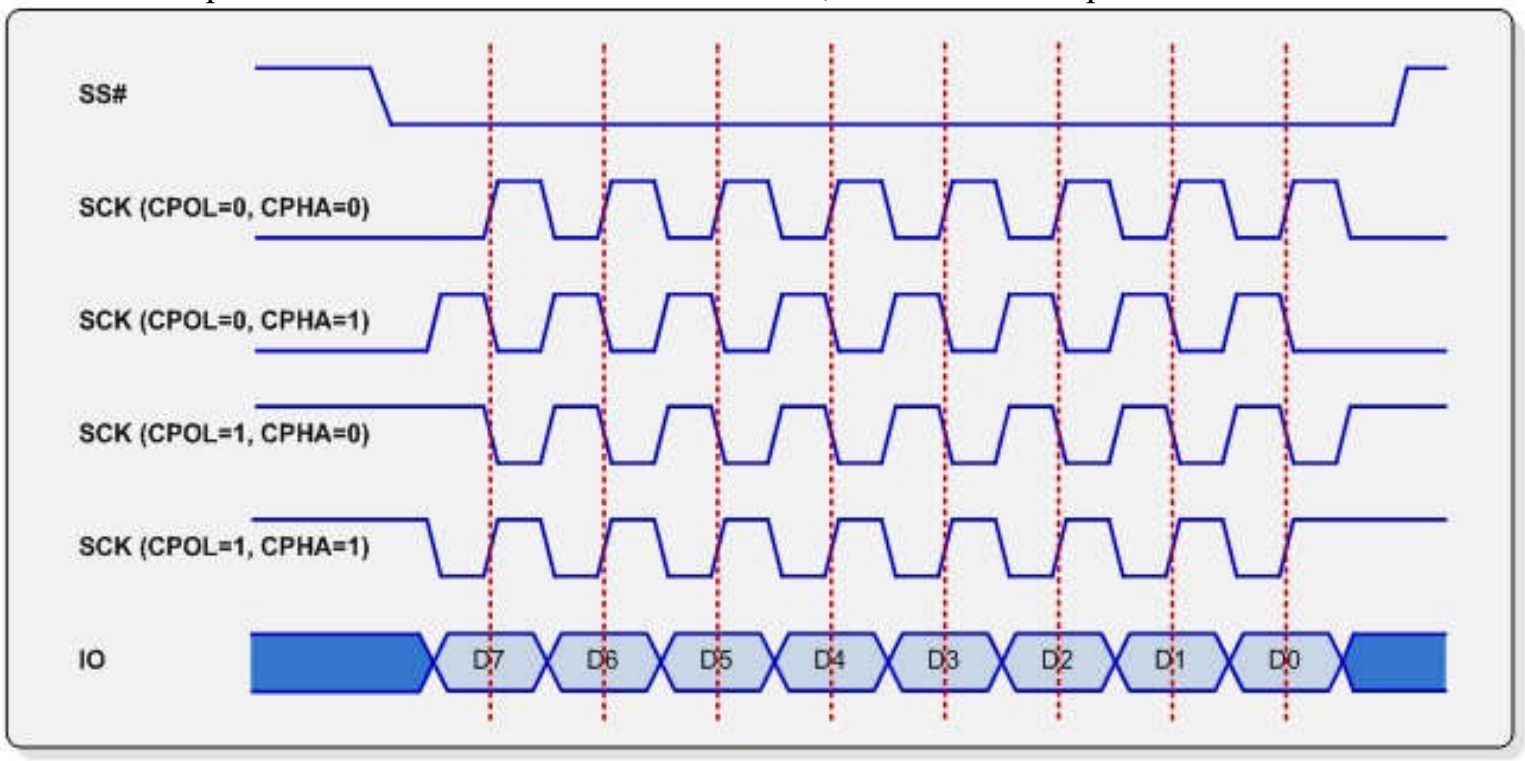
CPOL: 1 - valor base do clock = 1

CPHA: 0 - dado amostrado pelo escravo na borda de descida do clock
(dado atualizado pelo mestre na borda de subida)

CPHA: 1 – dado amostrado pelo escravo na borda de subida do clock; dado atualizado pelo mestre na borda de descida

CPOL: Polaridade do Clock
CPHA: Fase do Clock

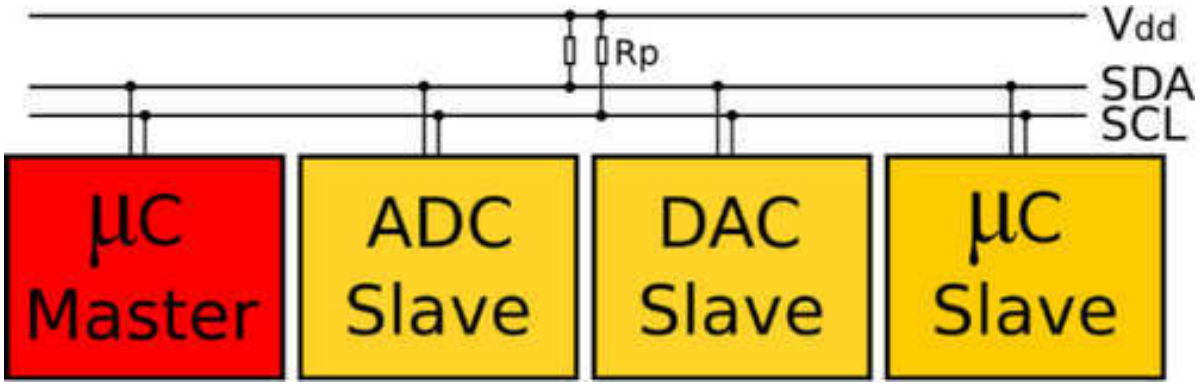
Mode	CPOL	CPHA
0	0	0
1	0	1
2	1	0
3	1	1



Transmissão Serial

I2C: *Inter-Integrated Circuit*

Mestre Único



SDA - **S**erial **DA**ta line
SCL - **S**erial **CL**ock line

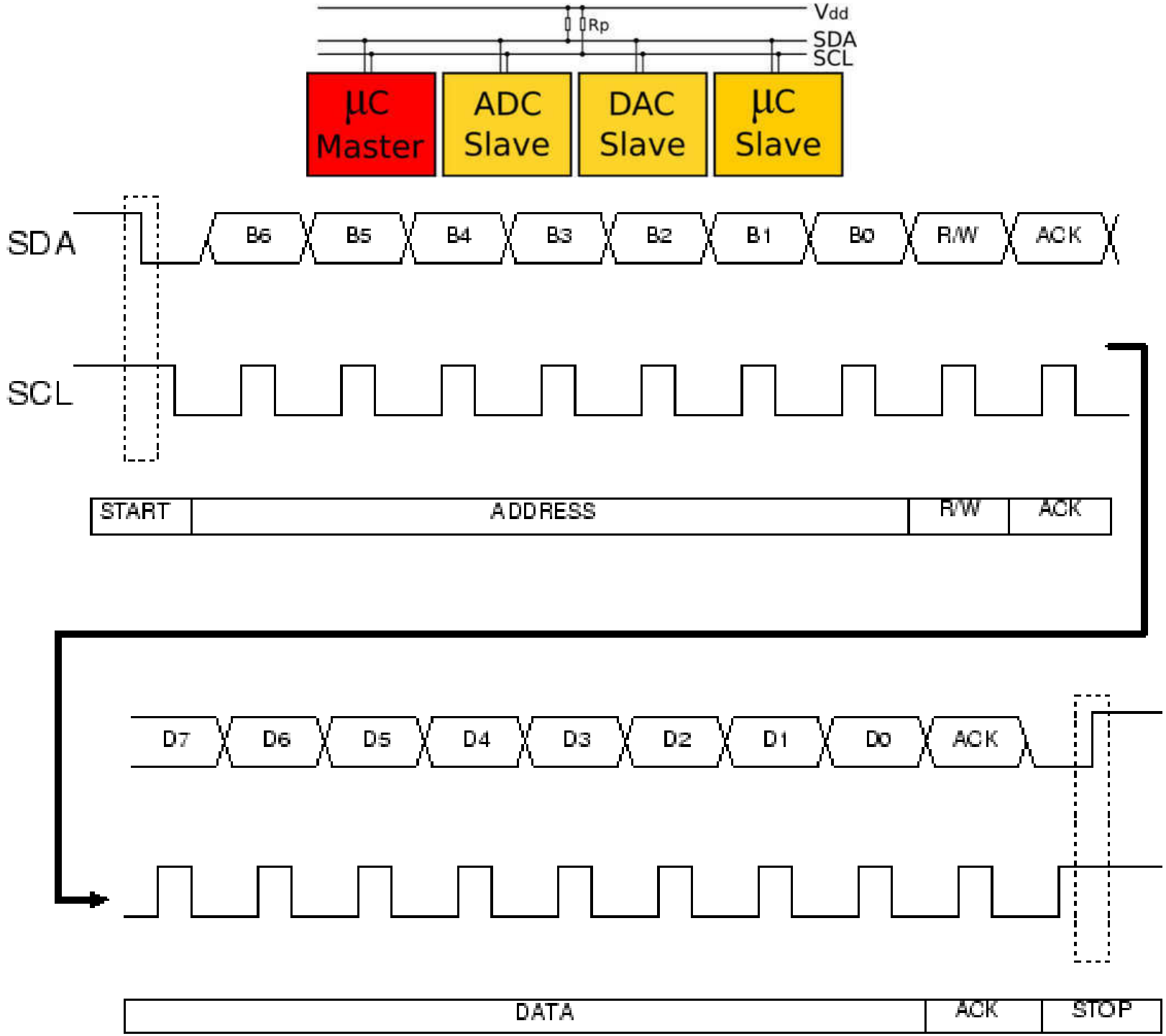
- **Mestre envia clock e endereço do escravo.**

- **7 bits são utilizados para endereçamento; há 16 endereços reservados => máximo de 112 escravos**

- **CIs tem endereço fixo ou os bits menos significativos do endereço do dispositivo são especificados por pinos**

- **Velocidade de transmissão de dados => Até 3,4 Mbi/s no modo *High Speed***

- **Mestre e escravo podem trocar de papel após stop bit .**


Vdd
Rp
SDA
SCL
µC Master
ADC Slave
DAC Slave
µC Slave
SDA
B6
B5
B4
B3
B2
B1
B0
R/W
ACK
SCL
START
ADDRESS
R/W
ACK
D7
D6
D5
D4
D3
D2
D1
D0
ACK
DATA
ACK
STOP

Comunicação de dados entre Mestre-Escravo

Etapa	Escrita	Leitura	Nro Bits	Campo
1	START bit	START bit	1	START
2	Slave address	Slave address	7	ADDRESS
3	bit 0	bit 1	1	W/R
4	Aguarda bit ‘0’ de “recebido”	Aguarda bit ‘0’ de “recebido”	1	ACK
5	Envia dados	Recebe dados	8	DATA
6	Aguarda bit ‘0’ de recebido	Envia bit ‘0’ de recebido ou vai para passo 7	1	ACK
7	STOP bit	STOP bit	1	STOP

Os passos 5 e 6 podem ser repetidos tal que múltiplos bytes sejam transmitidos ou recebidos. Ou seja, após 6, retorna-se ao passo 5 ou prossegue para o passo 7.