

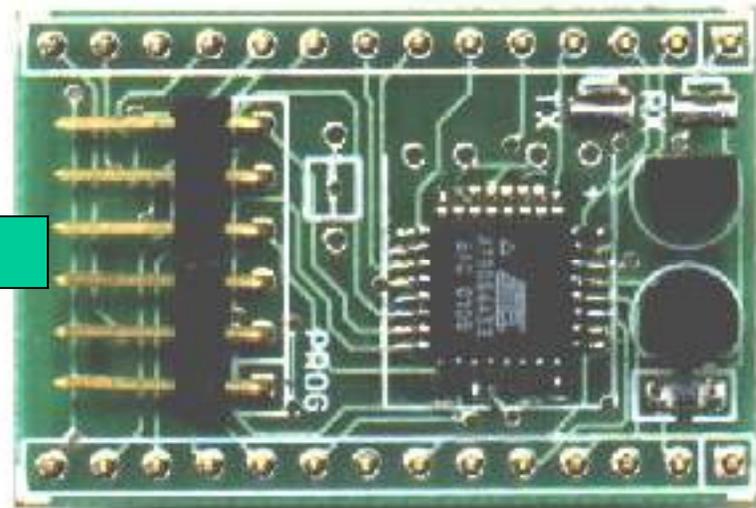
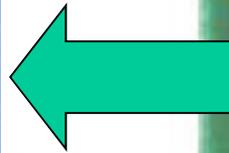
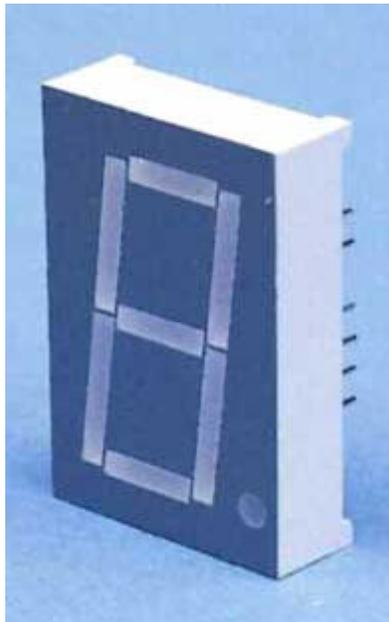
8051

AULA 9

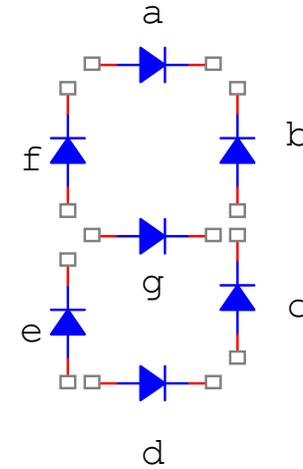
**Interface com Displays**

Prof. Adilson Gonzaga

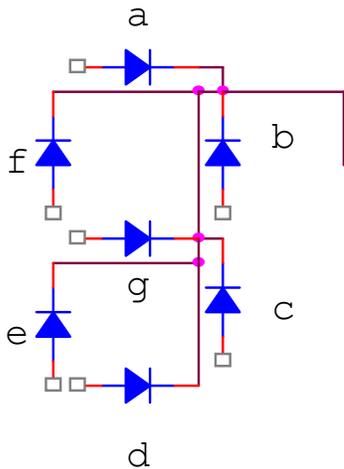
# Interface com Displays de 7 Segmentos



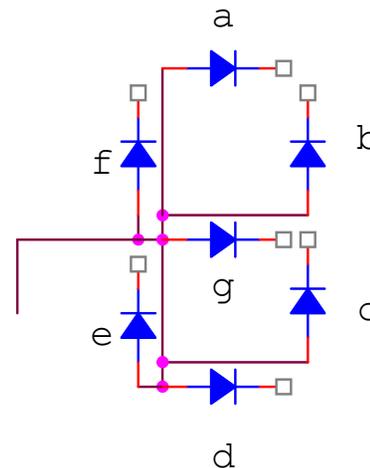
Um Display de 7 segmentos é formado por 7 LED's (a,b,c,d,e,f,g) que são previamente encapsulados e conectados de duas maneiras:



Catodo Comum:



Anodo Comum:



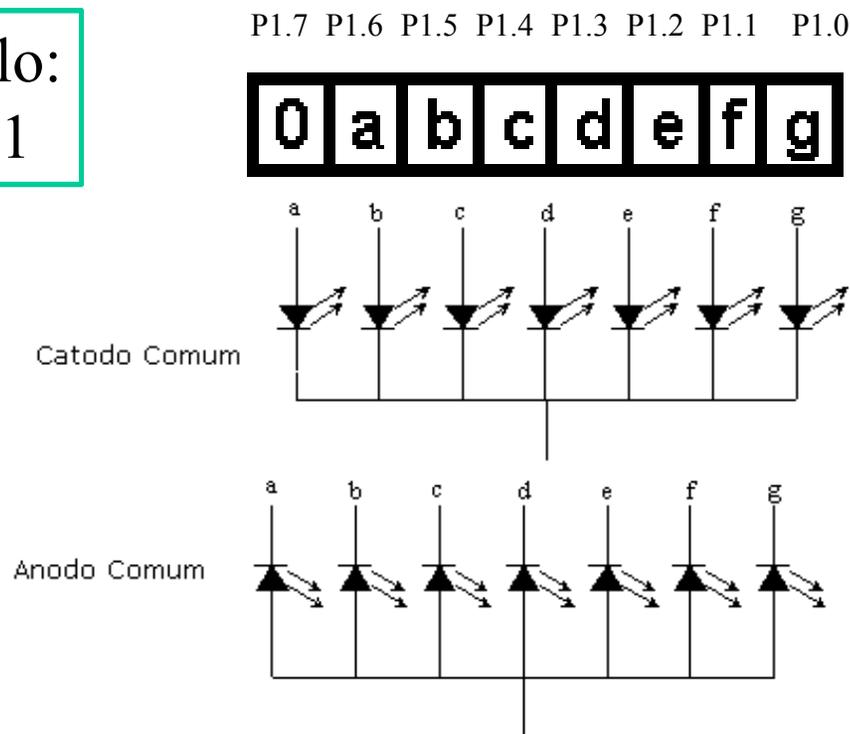
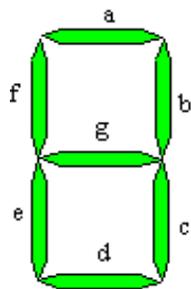
No Catodo Comum acende-se cada LED conectando-se o Comum ao GND e aplicando-se valor lógico 1 em cada segmento que se quer acender.

No Anodo Comum acende-se cada LED conectando-se o Comum ao VCC e aplicando-se valor lógico 0 em cada segmento que se quer acender.

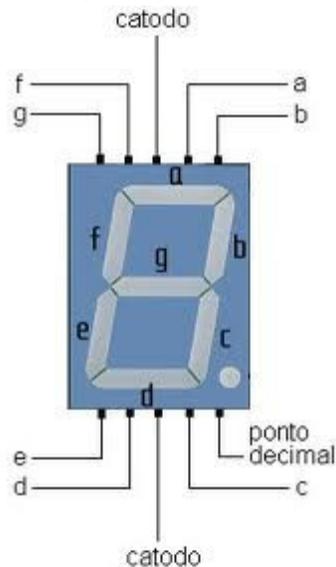
# Usando Bits de Porta

Para se interfacear um Display de 7 Segmentos com um Microcontrolador, deve-se determinar quais bits serão usados para acionar os LED's dos segmentos.

Exemplo:  
Porta P1



## Tabela de codificação de 7 segmentos para um Display Catodo Comum



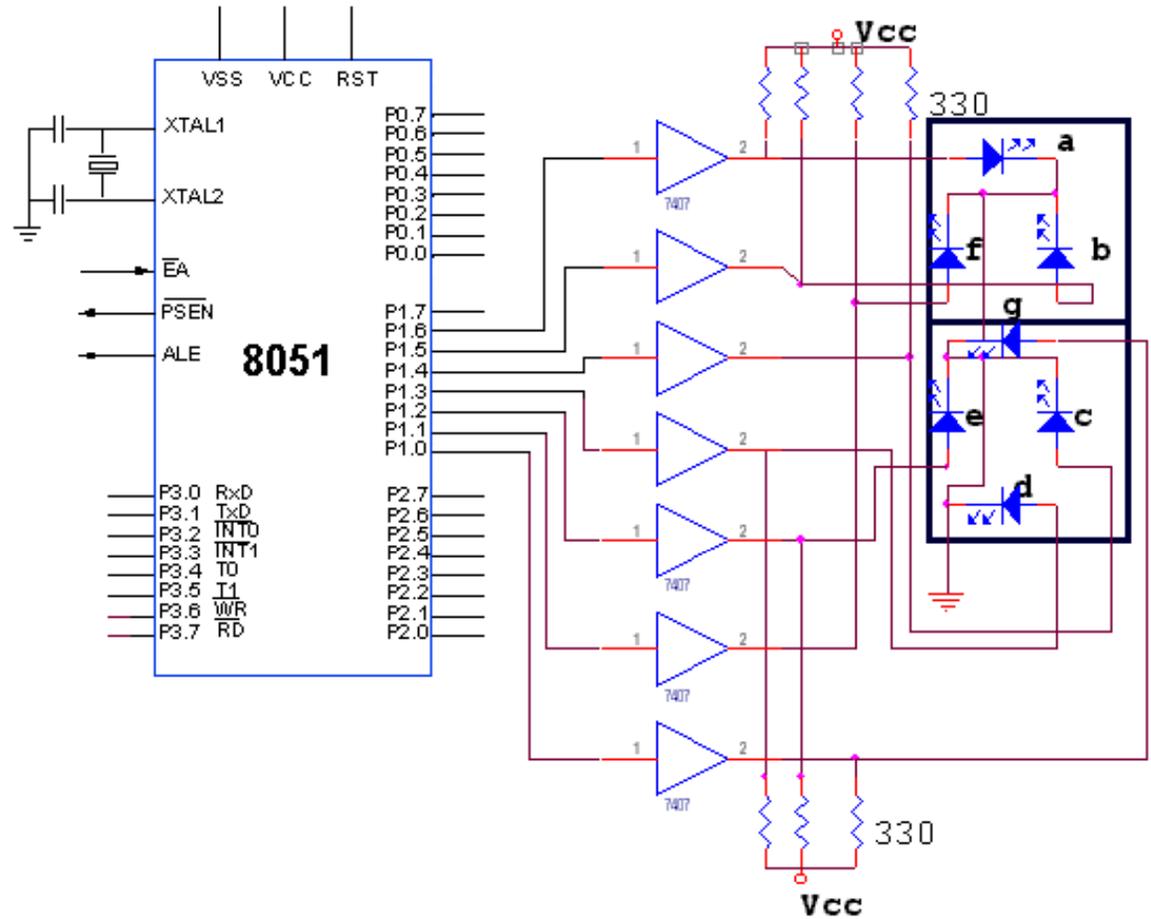
Byte a ser enviado para a porta do Microcontrolador para geração dos dígitos:



O Bit Mais Significativo é zero para Displays sem Ponto Decimal, caso contrário este bit será usado para ativar o ponto decimal.

a	b	c	d	e	f	g	Digito	Display	Hexa
1	1	1	1	1	1	0	0	0	7E
0	1	1	0	0	0	0	1	1	30
1	1	0	1	1	0	1	2	2	6D
1	1	1	1	0	0	1	3	3	79
0	1	1	0	0	1	1	4	4	33
1	0	1	1	0	1	1	5	5	5B
1	0	1	1	1	1	1	6	6	5F
1	1	1	0	0	0	0	7	7	70
1	1	1	1	1	1	1	8	8	7F
1	1	1	1	0	1	1	9	9	7B
1	1	1	0	1	1	1	A	A	77
0	0	1	1	1	1	1	b	b	1F
1	0	0	1	1	1	0	C	C	4E
0	1	1	1	1	0	1	d	d	3D
1	0	0	1	1	1	1	E	E	4F
1	0	0	0	1	1	1	F	F	47

Exemplo de Interface direta de 1 Display de 7 segmentos, catodo comum, com o 8051



Pode-se também utilizar um Decodificador integrado para 7 segmentos e interfacear o decodificador com o Microcontrolador

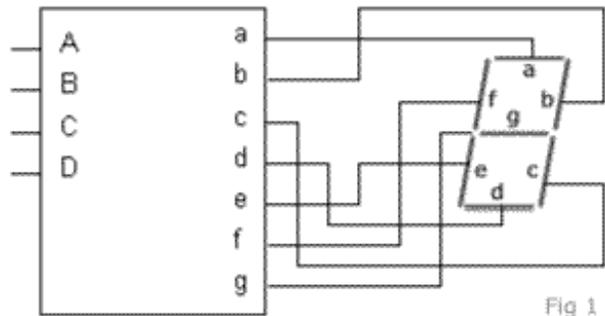


Fig 1



Uma boa aplicação para se utilizar Acesso a TABELA !!!

Subrotina de acionamento do Display de 7 segmentos (O dígito a ser mostrado no Display entra pelo Acumulador – de 00 a 0F)

### Utilizando como ponteiro o DPTR

```
DISPLAY:  MOV     P1,#00      ;Apaga todos os segmentos
          MOV     DPTR,#TAB  ;
          MOVC    A,@A+DPTR
          MOV     P1,A
          RET
; Tabela com os códigos Hexadecimais correspondentes aos dígitos de 7 segmentos
; em ordem crescente (0,1,2,3,4,5,6,7,8,9,A,b,C,d,E,F)
TAB:      DB     7EH,30H,6DH,79H,33H,5BH,5FH,70H,7FH,7BH,77H,1FH,4EH,3DH,4FH,47H
```

Uma boa aplicação para se utilizar Acesso a TABELA !!!

Subrotina de acionamento do Display de 7 segmentos (O dígito a ser mostrado no Display entra pelo Acumulador – de 00 a 0F)

Utilizando como ponteiro o PC

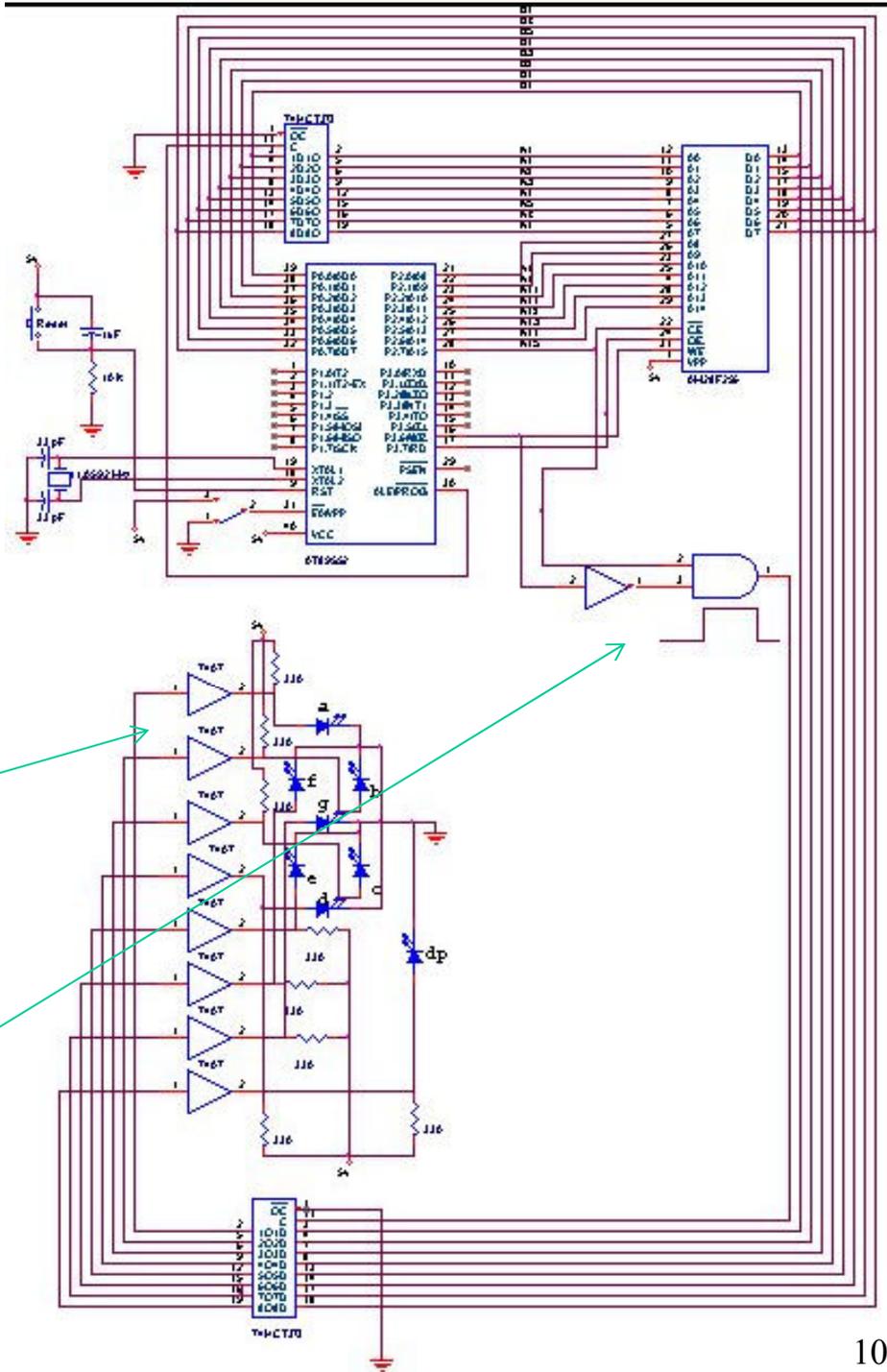
```
DISPLAY:  ADD     A,#02
          MOV     P1,#00
          MOVC   A,@A+PC
          MOV     P1,A
          RET
TAB:      DB 7EH,30H,6DH,79H,33H,5BH,5FH,70H,7FH,7BH,77H,1FH,4EH,3DH,4FH,47H
```

# Usando Mapeamento de Memória

## Display Catodo Comum

dp	g	f	e	d	c	b	a
D7	D6	D5	D4	D3	D2	D1	D0

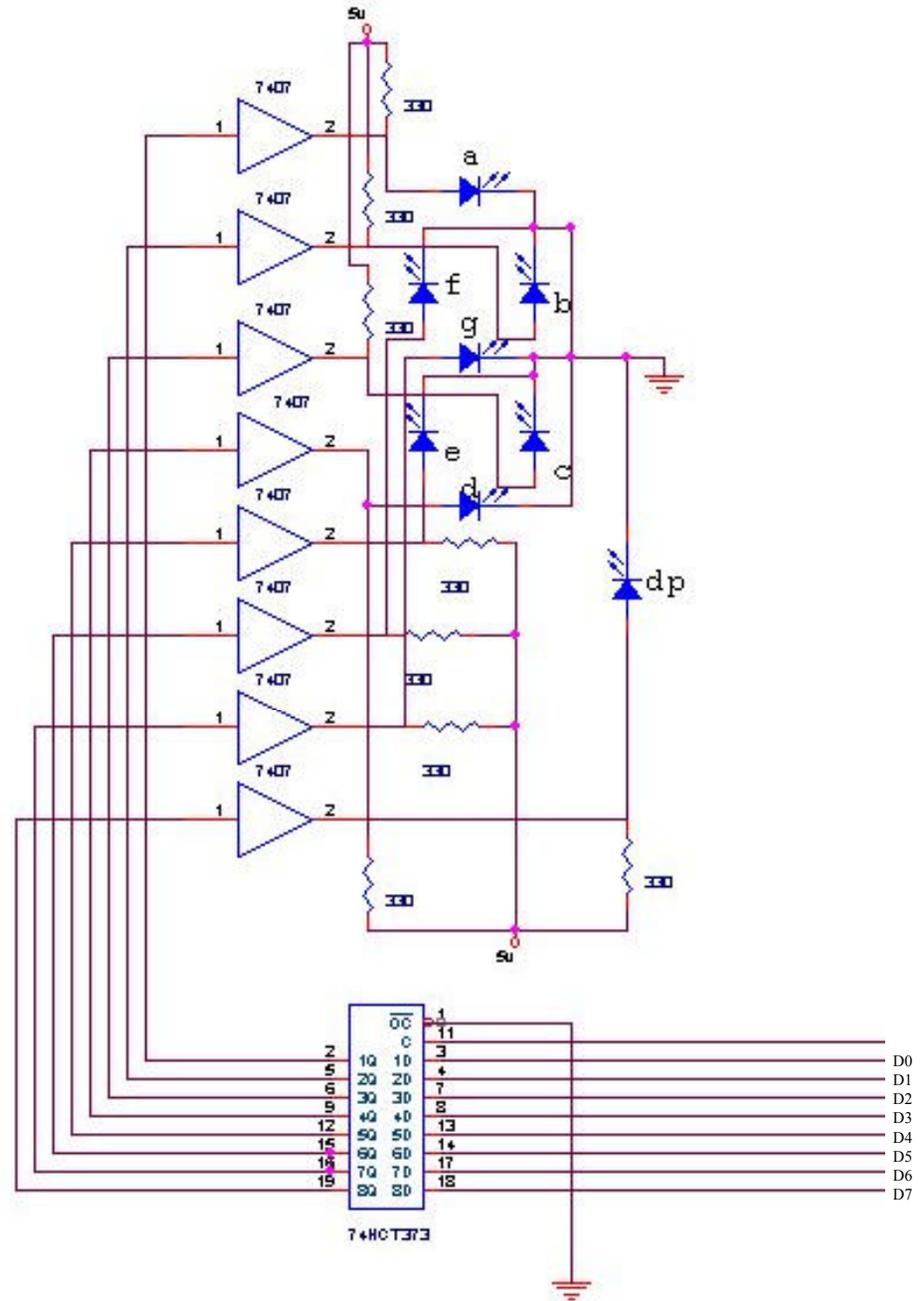
Mapeamento:  
De 8000h a FFFFh  
(escrita em qualquer endereço que tenha A15 = 1)



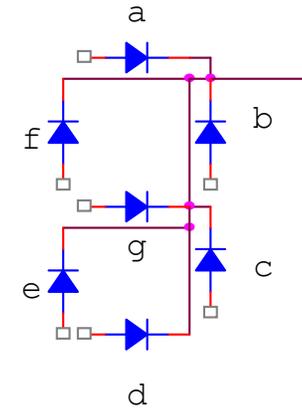
dp	g	f	e	d	c	b	a
D7	D6	D5	D4	D3	D2	D1	D0

Escrever no Display de 7 segmentos:

```
MOV    A, #Byte
MOV    DPTR,#8000h
MOVX   @DPTR,A
```



dp	g	f	e	d	c	b	a
D7	D6	D5	D4	D3	D2	D1	D0



Exemplos:

Apagar todos os segmentos

```
MOV A,#00
MOV DPTR,#8000H
MOVX @DPTR,A
```

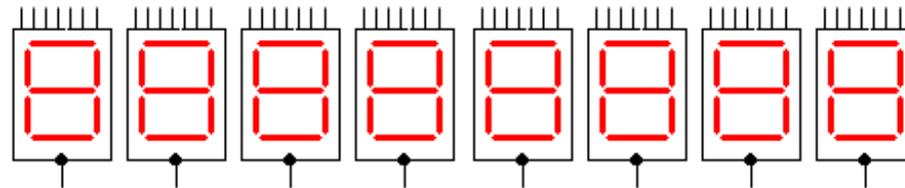
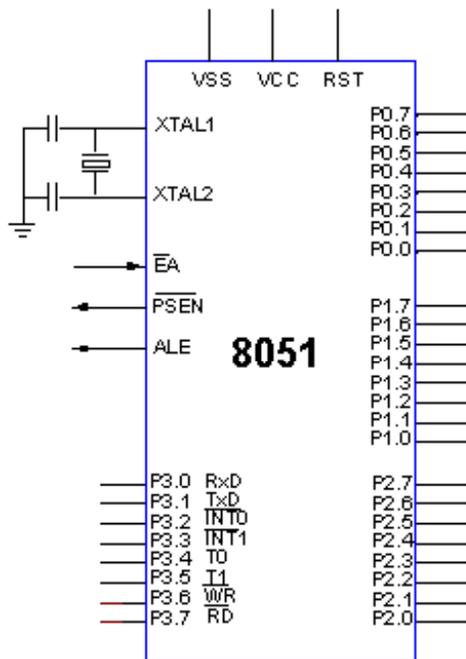
0	0	0	0	0	0	0	0
dp	g	f	e	d	c	b	a
D7	D6	D5	D4	D3	D2	D1	D0

Acender o número 0

```
MOV A,#3Fh
MOV DPTR,#8000H
MOVX @DPTR,A
```

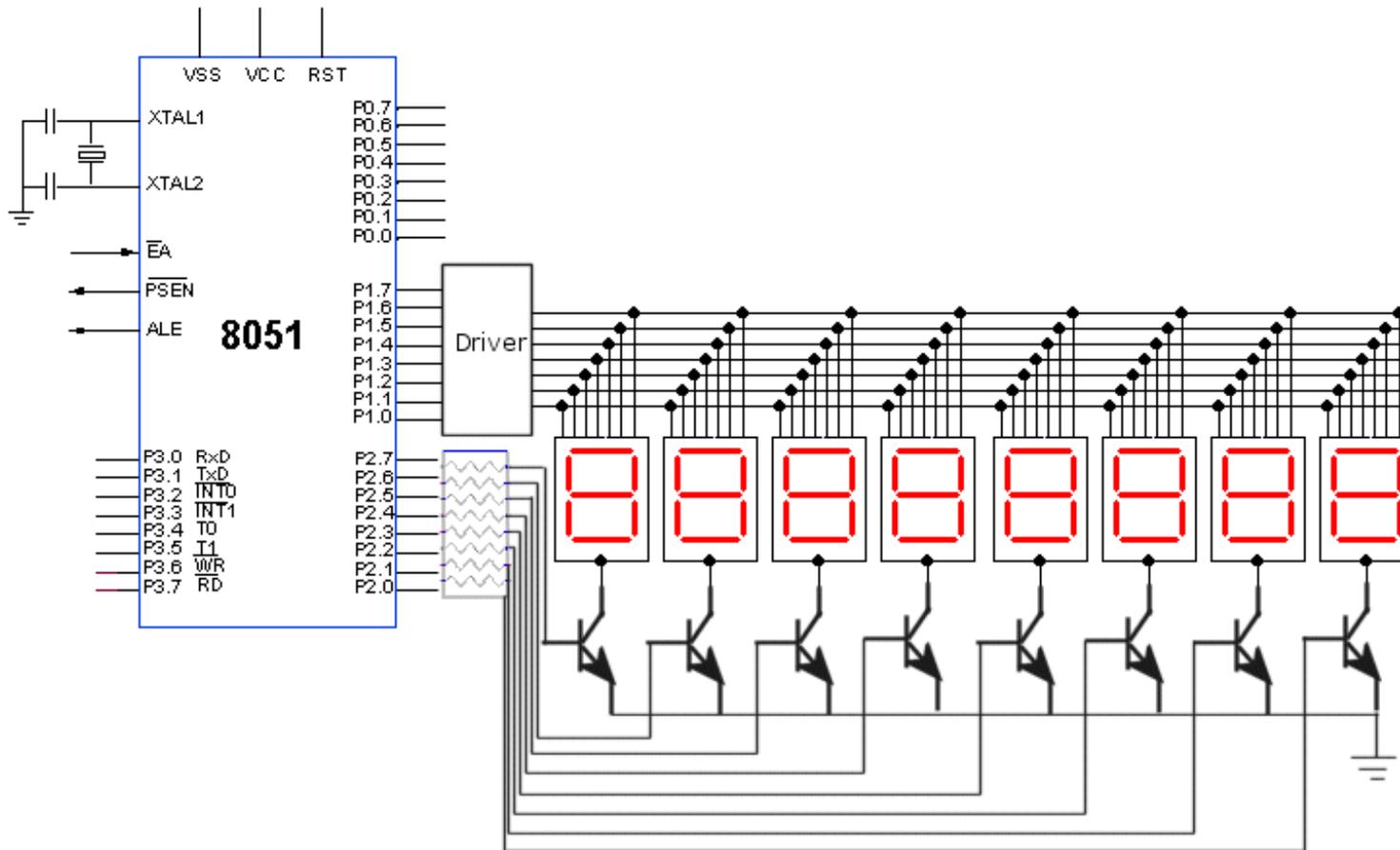
0	0	1	1	1	1	1	1
dp	g	f	e	d	c	b	a
D7	D6	D5	D4	D3	D2	D1	D0

# Como interfacear ao 8051 um conjunto de 8 Displays de 7 segmentos utilizando interface direta com os Bits de Porta?



Seriam necessárias 8 Portas de I/O ?

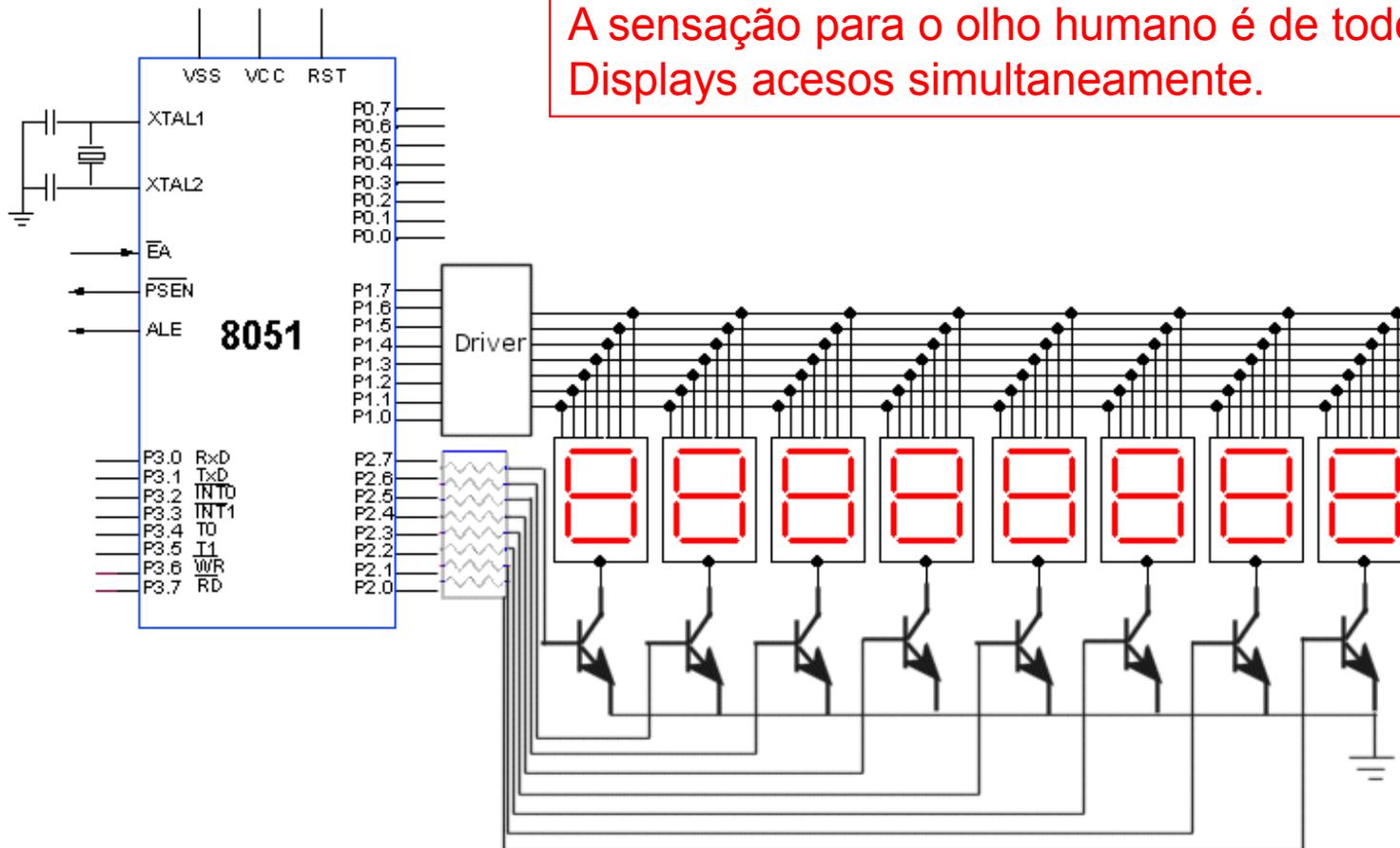
## Multiplexação de Displays de 7 Segmentos



Com apenas duas Portas do 8051 (P1 e P2) é possível Multiplexar no tempo o comando de todos os Displays de 7 Segmentos.

A Porta P1 mantém o valor do código Hexadecimal correspondente ao dígito a ser aceso e a Porta P2 indica em qual dos 8 Displays será aceso o dígito equivalente.

- Portanto, deve ser realizada uma varredura do dígito menos significativo para o dígito mais significativo, controlada pela Porta P2, alterando-se o valor de cada dígito no tempo, através da Porta P1.



## Sub-rotina de Multiplexação de 8 Displays de 7 segmentos.

Os códigos hexadecimais correspondentes a cada dígito a ser aceso devem ser armazenados nas posições 30h a 37h (BUFFER).

```
DMUX:      MOV      R0,#30H ; Início do Buffer do Display LSB
           MOV      A,#01  ;
CONT:      ACALL   APAGA
           MOV      P1,@R0 ; Código hexa-7 do dígito enviado ao Display
           MOV      P2,A   ; na posição dada pelo Bit rotacionado em A
           RL       A
           INC      R0
           CJNE    R0,#38H,DIGIT
DIGIT:     JC      CONT
           ACALL   APAGA
           RET

;Sub-rotina que mantém todos os Displays apagados a cada ciclo
;
APAGA:     MOV      P1,#00
           MOV      P2,#00
           RET
```

**Exemplo:** Se o número a aparecer nos Displays for:

**87509246**

As posições de memória (BUFFER) deverão conter:

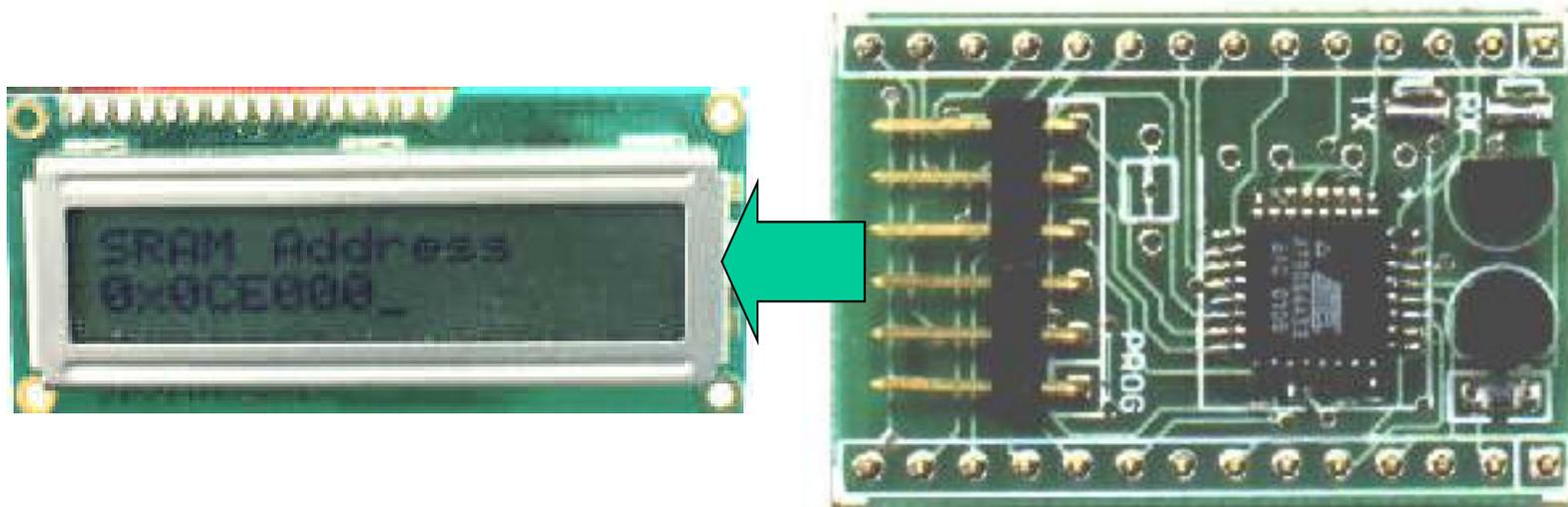
30h	5Fh	<b>6</b>
31H	33h	<b>4</b>
32h	6Dh	<b>2</b>
33h	7Bh	<b>9</b>
34h	7Eh	<b>0</b>
35h	5Bh	<b>5</b>
36h	70h	<b>7</b>
37h	7Fh	<b>8</b>

## Exercício (Entregar pelo site até a próxima aula)

- 1) Mapear em memória 8 Displays Multiplexados de 7 segmentos Anodo Comum (escrita em endereços superiores a 8000h) e uma RAM Externa de 0000 a 7FFFh.
- 2) Escrever um programa em Assembly do 8051 que envie para os Displays o valor numérico em BCD do conteúdo das posições de memória externa:
  - (1000)h → 2 displays Menos Significativos
  - (2000)h → 2 displays seguintes
  - (3000)h → próximos 2 displays
  - (4000)h → 2 displays Mais Significativos

Os conteúdos das posições de Memória são valores em hexadecimal de 00 a 63h

## Interface com LCD – Liquid Crystal Display



- Alguns dos LCDs mais utilizados são os displays de 16x2 e 20x2.
  - Isto significa 16 e 20 caracteres em cada uma das duas linhas do display respectivamente.



- O **HD44780** é o controlador padrão mais popular utilizado pelos fabricantes de LCD.
- Permite fazer uma comunicação de forma simples com a maioria dos LCDs.

- O padrão da indústria para módulos de LCDs baseados no controlador **HD44780** permite utilizar Displays com até 80 caracteres.

- Para isso, o circuito do controlador possui um conector de 14 pinos com as funções mostradas na tabela:

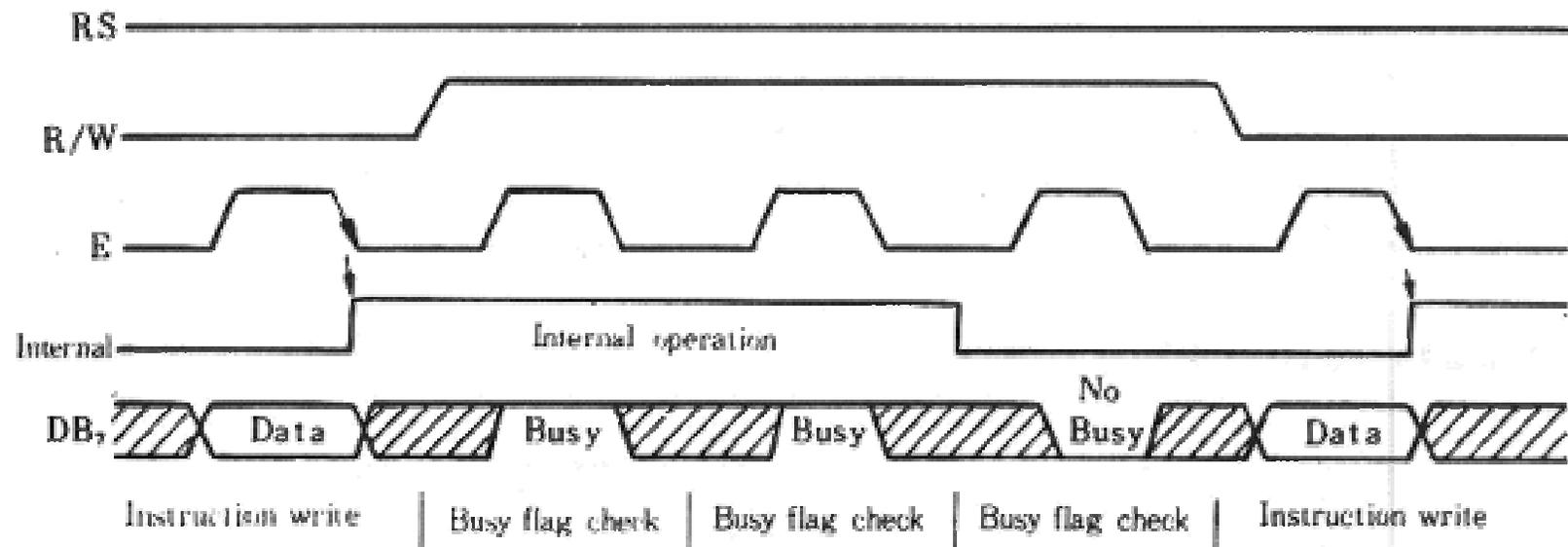
- O HD44780 requer 3 linhas de controle e também 4 ou 8 linhas de I/O para o bus de dados. O usuário deve selecionar operação com 4 ou 8 bits no bus de dados.

- Se um Módulo de LCD tiver mais que 80 caracteres, o circuito do controlador terá um conector de 16 pinos e a tabela com a nomenclatura dos pinos difere da mostrada ao lado.

Pin number	Symbol	Level	I/O	Function
1	Vss	-	-	Power supply (GND)
2	Vcc	-	-	Power supply (+5V)
3	Vee	-	-	Contrast adjust
4	RS	0/1	I	0 = Instruction input 1 = Data input
5	R/W	0/1	I	0 = Write to LCD module 1 = Read from LCD module
6	E	1, 1->0	I	Enable signal
7	DB0	0/1	I/O	Data bus line 0 (LSB)
8	DB1	0/1	I/O	Data bus line 1
9	DB2	0/1	I/O	Data bus line 2
10	DB3	0/1	I/O	Data bus line 3
11	DB4	0/1	I/O	Data bus line 4
12	DB5	0/1	I/O	Data bus line 5
13	DB6	0/1	I/O	Data bus line 6
14	DB7	0/1	I/O	Data bus line 7 (MSB)

## Operação do controlador de LCD HD44780 no modo 8 Bits:

### Ciclo de escrita de Instrução



RS = 0 → Instrução

RS = 1 → Dado

RW = 0 → Escrita

E = 0-1-0 → Habilita a escrita

## Conjunto de Instruções para programação do controlador de LCD HD44780

DDRAM → RAM de Dados do Display

CGRAM → RAM do Gerador de Caracteres do Display

Instrução	Palavra de Instrução								Atividade executada pelo controlador		
	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		Tempo	
Clear display	0	0	0	0	0	0	0	1	Apaga o display e retorna o cursor para a posição de home (Endereço 0).	1.64ms	
Cursor home	0	0	0	0	0	0	1	*	Retorna o cursor para a posição de home (Endereço 0). O conteúdo da DDRAM não muda.	1.64ms	
Entry mode set	0	0	0	0	0	1	ID	S	Direção do Cursor → VD=0 → decrementa, VD=1 → Incrementa. S = 1 → Desloca o Display S=0 → Não desloca o Display	40uS	
Display On/Off control	0	0	0	0	1	D	C	B	D = 1 → Liga Display C = 1 → Liga Cursor B = 1 → Pisca Cursor	40uS	
Cursor/display shift	0	0	0	1	S/C	R/L	*	*	S/C = 0 → Move o cursor S/C = 1 → Desloca o Display, R/L = 0 → Desloca à esquerda R/L = 1 → Desloca à direita O conteúdo da DDRAM não muda.	40uS	
Function set	0	0	1	DL	N	F	*	*	DL=0 → Modo 4 Bit DL=1 → Modo 8 Bit N=0 → 1 linha N=1 → 2 linhas F=0 → Matriz de 5/7 pontos F=1 → Matriz de 5/10 pontos	40uS	
Set CGRAM address	0	1	CGRAM address							Estabelece o endereço da CGRAM. Os dados da CGRAM são enviados ou recebidos após este comando.	40uS
Set DDRAM address	1	DDRAM address							Estabelece o endereço da DDRAM. Os dados da DDRAM são enviados ou recebidos após este comando..	40uS	
Read busy-flag and address counter	BF	CGRAM / DDRAM address							BF=0 → o controlador aceita instruções BF=1 → controlador ocupado Leitura do conteúdo do contador de endereço da CGRAM ou DDRAM.	0uS	
Write to CGRAM or DDRAM	write data								Escreve dados na CGRAM ou DDRAM.	40uS	
Read from CGRAM or DDRAM	read data								Lê dados da CGRAM ou DDRAM.	40uS	

## DDRAM → RAM de Dados do Display

Com N=0 → Display de 1 linha, os endereços dos caracteres são:

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27

Display size	Visible	
	Character positions	DDRAM addresses
1*8	00..07	0x00..0x07
1*16	00..15	0x00..0x0F
1*20	00..19	0x00..0x13
1*24	00..23	0x00..0x17
1*32	00..31	0x00..0x1F
1*40	00..39	0x00..0x27



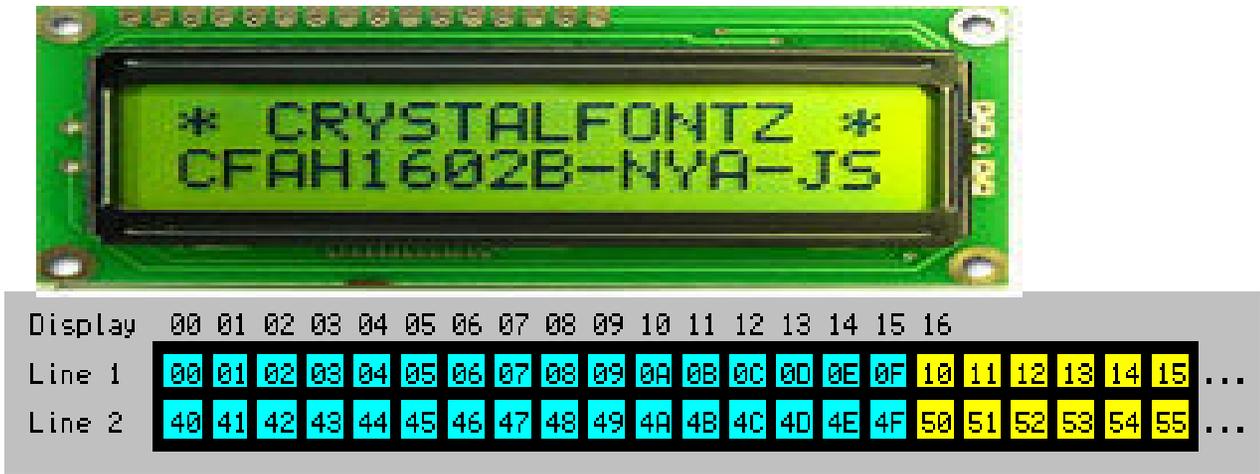
Com N=1 → Display de 2 linhas, os endereços dos caracteres são:

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66	67

Display size	Visible	
	Character positions	DDRAM addresses
2*16	00..15	0x00..0x0F + 0x40..0x4F
2*20	00..19	0x00..0x13 + 0x40..0x53
2*24	00..23	0x00..0x17 + 0x40..0x57
2*32	00..31	0x00..0x1F + 0x40..0x5F
2*40	00..39	0x00..0x27 + 0x40..0x67



**Exemplo:** Para um LCD de 2 linhas no formato 2x16, os endereços da DDRAM, que são visíveis no Display, são os anotados em azul na figura:



Logo, para se escrever um caractere na primeira posição da linha 2 deve-se escrever a Instrução  $80h + 40h = C0h$ , onde:

80h → Estabelece o endereço da DDRAM e,

40h → Estabelece o endereço da primeira posição na linha 2

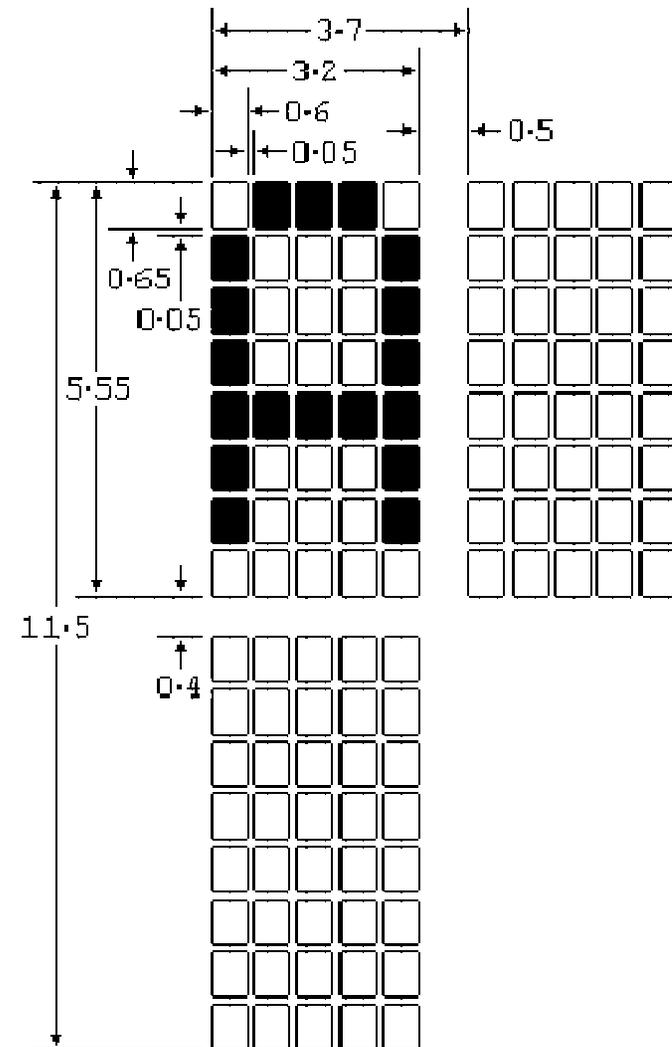
Set DDRAM address	1	DDRAM address	Estabelece o endereço da DDRAM. Os dados da DDRAM são enviados ou recebidos após este comando..	40uS
-------------------	---	---------------	---	------

1 1 0 0 0 0 0 0

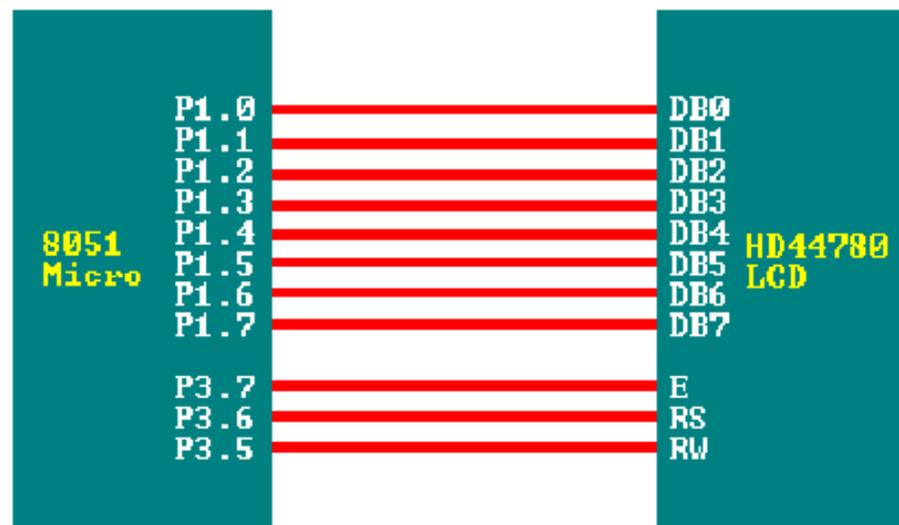
# Conjunto de Carcteres ASCII aceitos e gerados pelo controlador de LCD HD44780

Fonte de 5x7 pontos

	0	0	0	0	0	0	1	1	1	1	1	1	
	0	0	0	1	1	1	1	0	1	1	1	1	
	0	1	1	0	0	1	1	1	1	0	0	1	
	0	0	1	0	1	0	1	0	1	0	1	0	
xxxx0000		0	@	P	`	P		-	9	ε	α	ρ	
xxxx0001		!	1	A	Q	a	q	。	ア	チ	4	ä	q
xxxx0010		"	2	B	R	b	r	「	イ	ツ	×	β	θ
xxxx0011		#	3	C	S	c	s	」	ウ	〒	ε	ε	ω
xxxx0100		\$	4	D	T	d	t	、	エ	ト	μ	μ	Ω
xxxx0101		%	5	E	U	e	u	・	オ	ナ	1	σ	Ü
xxxx0110		&	6	F	V	f	v	ヲ	カ	ニ	ヨ	ρ	Σ
xxxx0111		'	7	G	W	g	w	フ	キ	ヲ	ラ	q	π
xxxx1000		(	8	H	X	h	x	イ	ク	ネ	リ	フ	̄
xxxx1001		)	9	I	Y	i	y	ウ	ケ	ル	リ	ウ	
xxxx1010		*	:	J	Z	j	z	エ	コ	ハ	レ	i	〒
xxxx1011		+	;	K	[	k	[	オ	サ	ヒ	ロ	×	斤
xxxx1100		,	<	L	¥	l	l	ハ	シ	フ	フ	φ	円
xxxx1101		-	=	M	]	m	]	ユ	ズ	ハ	ン	も	÷
xxxx1110		.	>	N	^	n	^	ヨ	セ	ホ	ハ	ん	
xxxx1111		/	?	O	_	o	_	ツ	マ	°	ö	■	



## Exemplo de Interface de um LCD (2x16) baseado no controlador HD44780 com o Microcontrolador 8051



```

    ORG      0
; Nome das linhas de dados e controle

    DB0     EQU      P1.0
    DB1     EQU      P1.1
    DB2     EQU      P1.2
    DB3     EQU      P1.3
    DB4     EQU      P1.4
    DB5     EQU      P1.5
    DB6     EQU      P1.6
    DB7     EQU      P1.7
    ENAB    EQU      P3.7
    RS      EQU      P3.6
    RW      EQU      P3.5
    DATA   EQU      P1
;*****
; Programa Principal *
; Exemplo: Escrever HELLO WORLD no LCD, sendo a *
; primeira palavra no início da primeira linha e a *
; segunda palavra a partir da posição 10 da segunda*
; linha do LCD. *
;*****
    LCALL   INIT_LCD
    LCALL   CLEAR_LCD
    MOV     A,#'H'
    LCALL   WRITE_TEXT
    MOV     A,#'E'
    LCALL   WRITE_TEXT
    MOV     A,#'L'
    LCALL   WRITE_TEXT
    MOV     A,#'L'
    LCALL   WRITE_TEXT
    MOV     A,#'O'
    LCALL   WRITE_TEXT
    MOV     A,#4AH      ; Posicionando o cursor
    LCALL   POS_LCD
    MOV     A,#'W'
    LCALL   WRITE_TEXT
    MOV     A,#'O'
    LCALL   WRITE_TEXT
    MOV     A,#'R'
    LCALL   WRITE_TEXT
    MOV     A,#'L'
    LCALL   WRITE_TEXT
    MOV     A,#'D'
    LCALL   WRITE_TEXT
    SJMP   $
;*****

```

Exemplo de um Programa para escrever HELLO WORLD no LCD

A palavra HELLO deverá ser escrita na primeira posição da primeira linha do Display e a palavra WORLD deverá ser escrita na posição 10 da segunda linha do Display



```

;*****
; Sub-rotina de Inicialização do controlador do LCD*
;*****

```

```
INIT_LCD:
```

```

CLR    RW      ; Inicia ciclo de escrita no LCD
SETB   ENAB    ; Habilita o LCD
CLR    RS      ; Modo Instrução
MOV    DATA,#38h ; Display de 2 linhas, Modo 8 Bits
CLR    ENAB    ; Desabilita o LCD
LCALL  WAIT_LCD ; Aguarda o LCD executar o comando

```

```
;*****
```

```

SETB   ENAB
CLR    RS
MOV    DATA,#0Eh ; Liga o LCD e o Cursor
CLR    ENAB
LCALL  WAIT_LCD

```

```
;*****
```

```

SETB   ENAB
CLR    RS
MOV    DATA,#06h ; LCD em recepção e cursor move para
                  ; a direita a cada caracter
CLR    ENAB
LCALL  WAIT_LCD

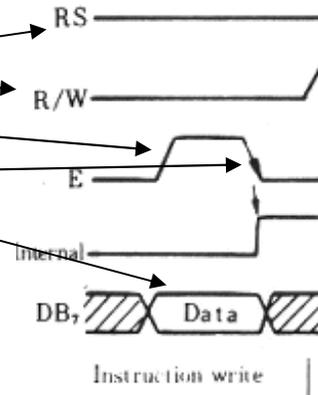
```

```

RET

```

```
;*****
```



0 0 1 1 1 0 0 0

38h

Function set	0	0	1	DL	N	F	*	*	DL=0 → Modo 4 Bit DL=1 → Modo 8 Bit N=0 → 1 linha N=1 → 2 linhas F=0 → Matriz de 5*7 pontos F=1 → Matriz de 5*10 pontos
--------------	---	---	---	----	---	---	---	---	--

0 0 0 0 1 1 1 0

0Eh

Display On/Off control	0	0	0	0	1	D	C	B	D = 1 → Liga Display C = 1 → Liga Cursor B = 1 → Pisca Cursor
------------------------	---	---	---	---	---	---	---	---	---

0 0 0 0 0 1 1 0

06h

Entry mode set	0	0	0	0	0	1	ID	S	Direção do Cursor → ID=0 → decrementa, ID=1 → Incrementa. S = 1 → Desloca o Display S=0 → Não desloca o Display
----------------	---	---	---	---	---	---	----	---	--

```

;*****
; Sub-rotina para limpar a tela do LCD      *
;*****

```

CLEAR\_LCD:

```

CLR     RW
SETB   ENAB
CLR     RS
MOV     DATA,#01h
CLR     ENAB
LCALL  WAIT_LCD
RET

```

0 0 0 0 0 0 0 1

01h

Clear display	0	0	0	0	0	0	0	1	Apaga o display e retorna o cursor para a posição de home (Endereço 0)
---------------	---	---	---	---	---	---	---	---	--

```

;*****

```

Display	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	...
Line 1	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	...
Line 2	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	54	55	...

A = 4Ah

Décima posição da segunda linha

```

;*****
; Sub-rotina para posicionar o cursor do LCD      *
; A posição deve estar armazenada no Acumulador  *
;*****

```

POS\_LCD:

```

CLR     RW
SETB   ENAB
CLR     RS
ADD     A,#80H
MOV     DATA,A
CLR     ENAB
LCALL  WAIT_LCD
RET

```

1 0 0 0 0 0 0 0 80h

0 1 0 0 1 0 1 0 4Ah

1 1 0 0 1 0 1 0 CAh

Set DDRAM address	1	DDRAM address	Estabelece o endereço da DDRAM. Os dados da DDRAM são enviados ou recebidos após este comando..
-------------------	---	---------------	---

```

;*****
; Sub-rotina para escrever um caractere no LCD      *
; O Carater em ASCII deve estar no Acumulador.   *
;*****

```

```

WRITE_TEXT:
    CLR     RW
    SETB   ENAB
    SETB   RS           ; Modo Dados
    MOV    DATA,A
    CLR    ENAB
    LCALL  WAIT_LCD
    RET

```

```

;*****

```



```

;*****
; Sub-rotina que espera o LCD executar um comando *
;*****

```

```

WAIT_LCD:
    SETB RW           ; Inicia o ciclo de leitura no LCD
    SETB ENAB        ; Habilita o LCD
    CLR RS           ; Modo Instrução
    MOV DATA,#0FFh ; Garante P1 como entrada
    JB DB7,$        ; Se o bit DB7 estiver em alto,
                   ; o LCD está ocupado
    CLR ENAB        ; Desabilita o LCD
    CLR RW           ; Finaliza o ciclo de leitura no LCD
    RET

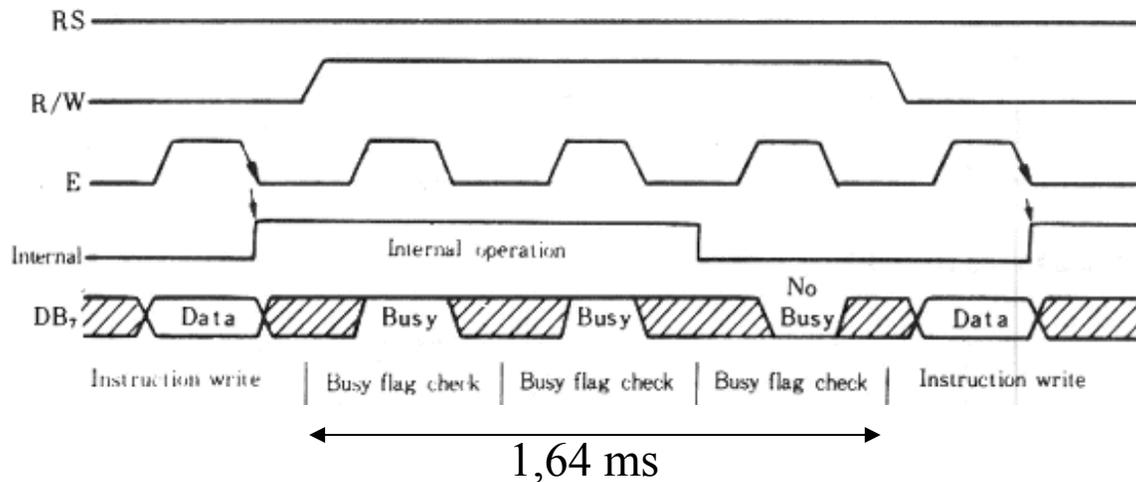
```

```

;*****

```

Como os tempos de respostas do LCD não ultrapassam 1,64 ms, uma outra forma de fazer a Sub-rotina de espera (WAIT\_LCD) é utilizar uma rotina de Atraso de aproximadamente 2 ms.



Cursor home	0	0	0	0	0	0	1	*	Retorna o cursor para a posição de home (Endereço 0). O conteúdo da DDRAM não muda.	1.64ms
Entry mode set	0	0	0	0	0	1	ID	S	Direção do Cursor → ID=0 → decrementa, ID=1 → Incrementa. S = 1 → Desloca o Display S=0 → Não desloca o Display	40uS

```

WAIT_LCD:  MOV     R1,#122    ; Atraso de 2 ms
LOOP:     MOV     R0,#06
          DJNZ    R0,$
          DJNZ    R1,LOOP
          RET
  
```