

Arquitetura de Computadores

PROF. ISAAC

Display

Tipos de Display

LCD Alfanumérico



LCD Gráfico



OLED



TFT





Display LCD

Tipos de Display LCD Alfanumérico

8x2



8 colunas e 2 linhas

16x2



16 colunas e 2 linhas

16x1



16 colunas e 1 linha

16x4



16 colunas e 4 linhas

20x4

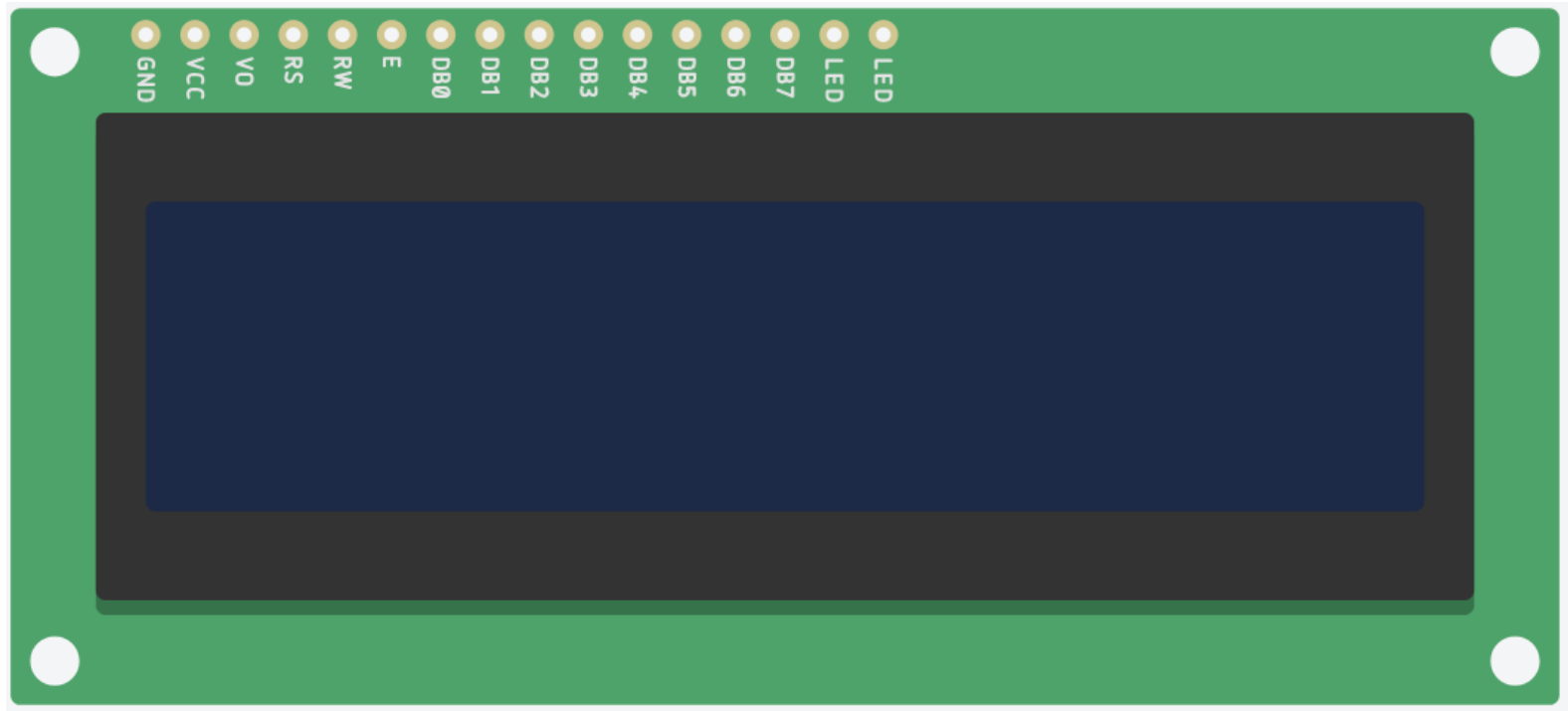


20 colunas e 4 linhas

CISC - Complex Instruction Set Compute

O Display LCD possui um processador para que os caracteres sejam escritos facilmente apenas enviando o número ASC do caractere que deva aparecer no Display.

Pinagem Display

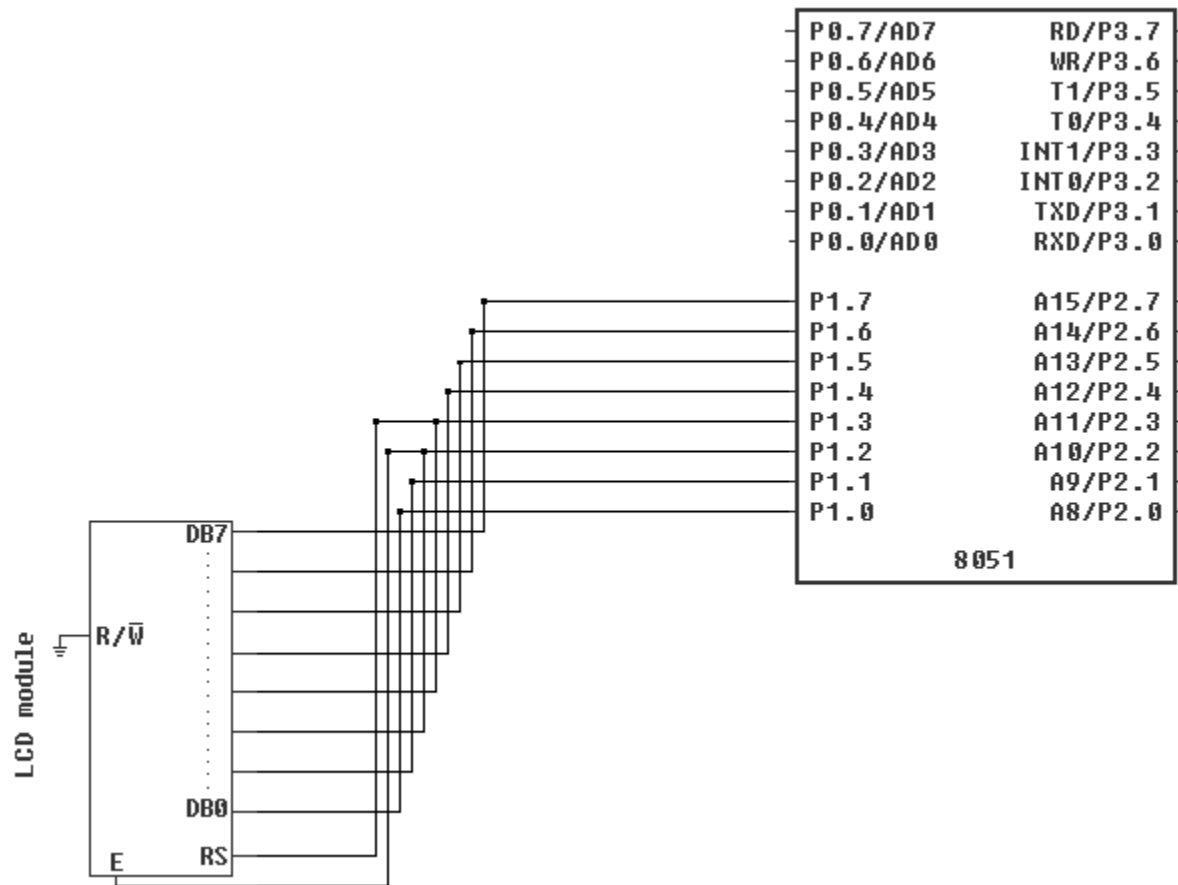


Padrão dos pinos do Display LCD

Pino	Função	Descrição
1	Alimentação	GND
2	Alimentação	VCC ou +5V
3	V0	Tensão para ajuste de contraste
4	RS	1 – Dado, 0 - Instrução
5	R/W	1 – Leitura, 0 - Escrita
6	E	1 ou (1 → 0) – Habilita, 0 - Desabilitado
7	D0 (LSB)	Barramento de Dados
8	D1	
9	D2	
10	D3	
11	D4	
12	D5	
13	D6	
14	D7 (MSB)	
15	A (qdo existir)	Anodo para LED backlight
16	K (qdo existir)	Catodo para LED backlight

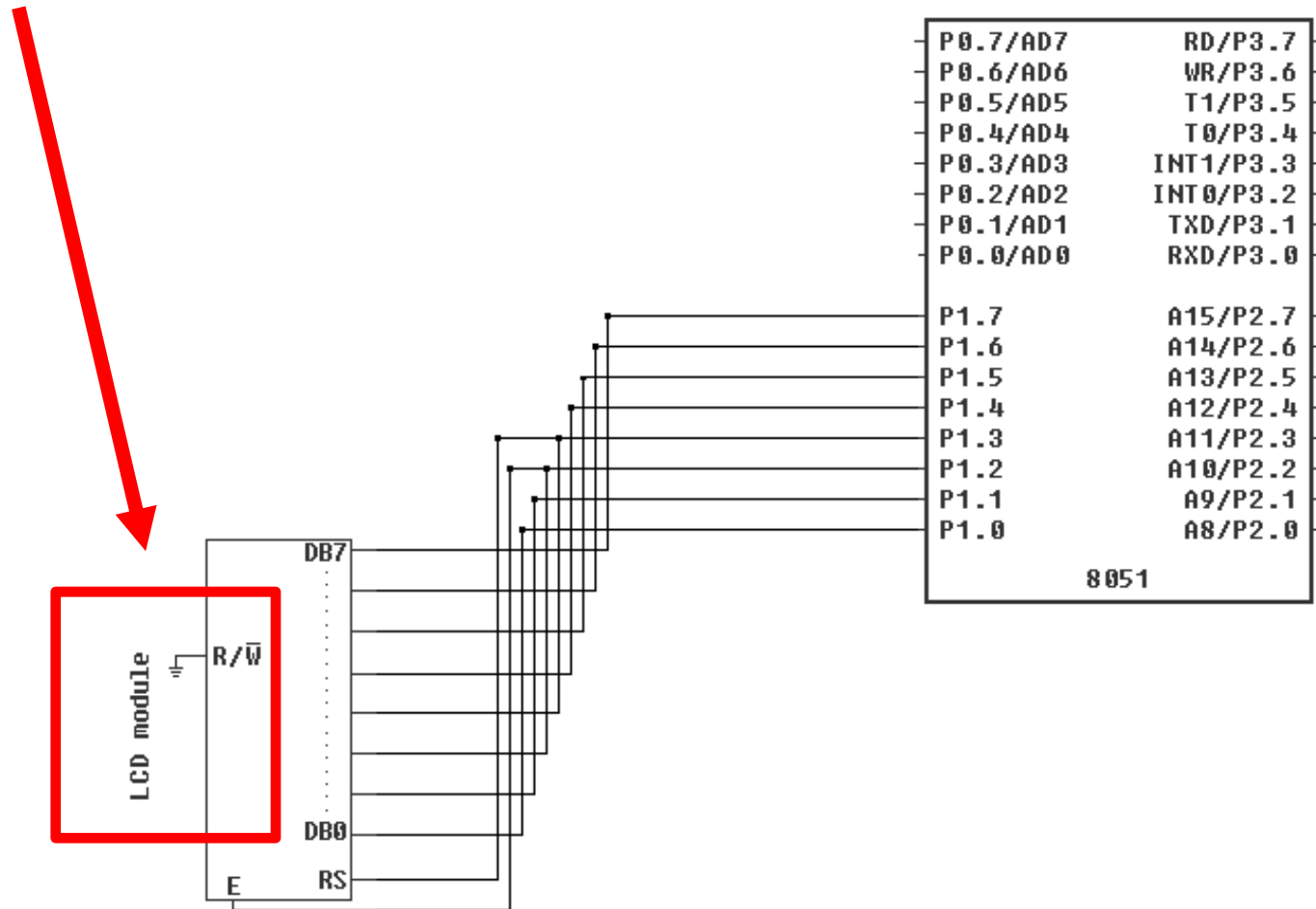
Ligação do Display LCD no 8051 – edisim51

No edisim51 o display possui alguns dos pinos apresentados na Tabela anterior. Esses pinos estão ligados no Port P1.



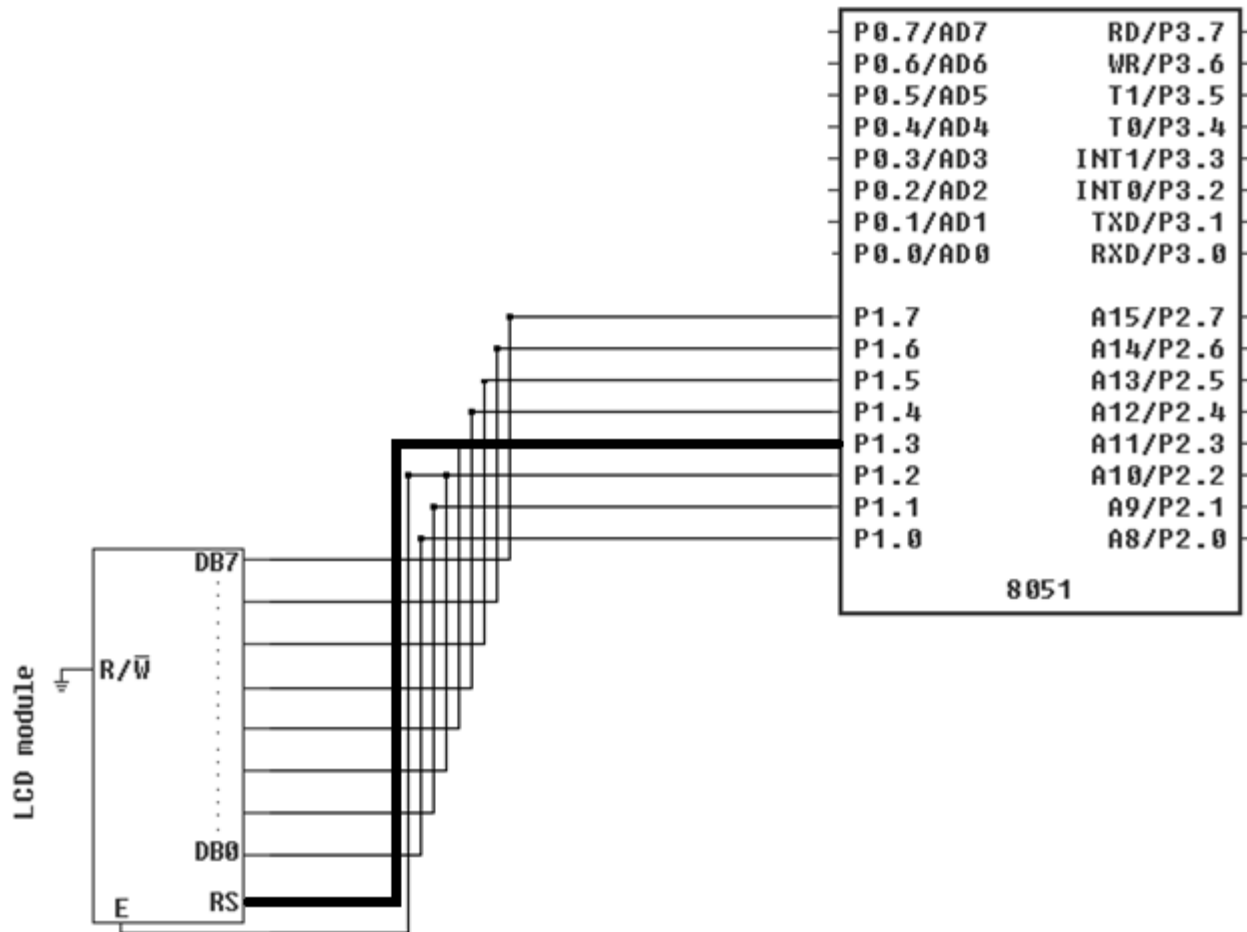
Ligação do Display LCD no 8051 – edisim51

Observe que o pino R/W do display está ligado no GND, ou seja, **R/W = 0**, portanto o LCD só pode ser utilizado no modo escrita.



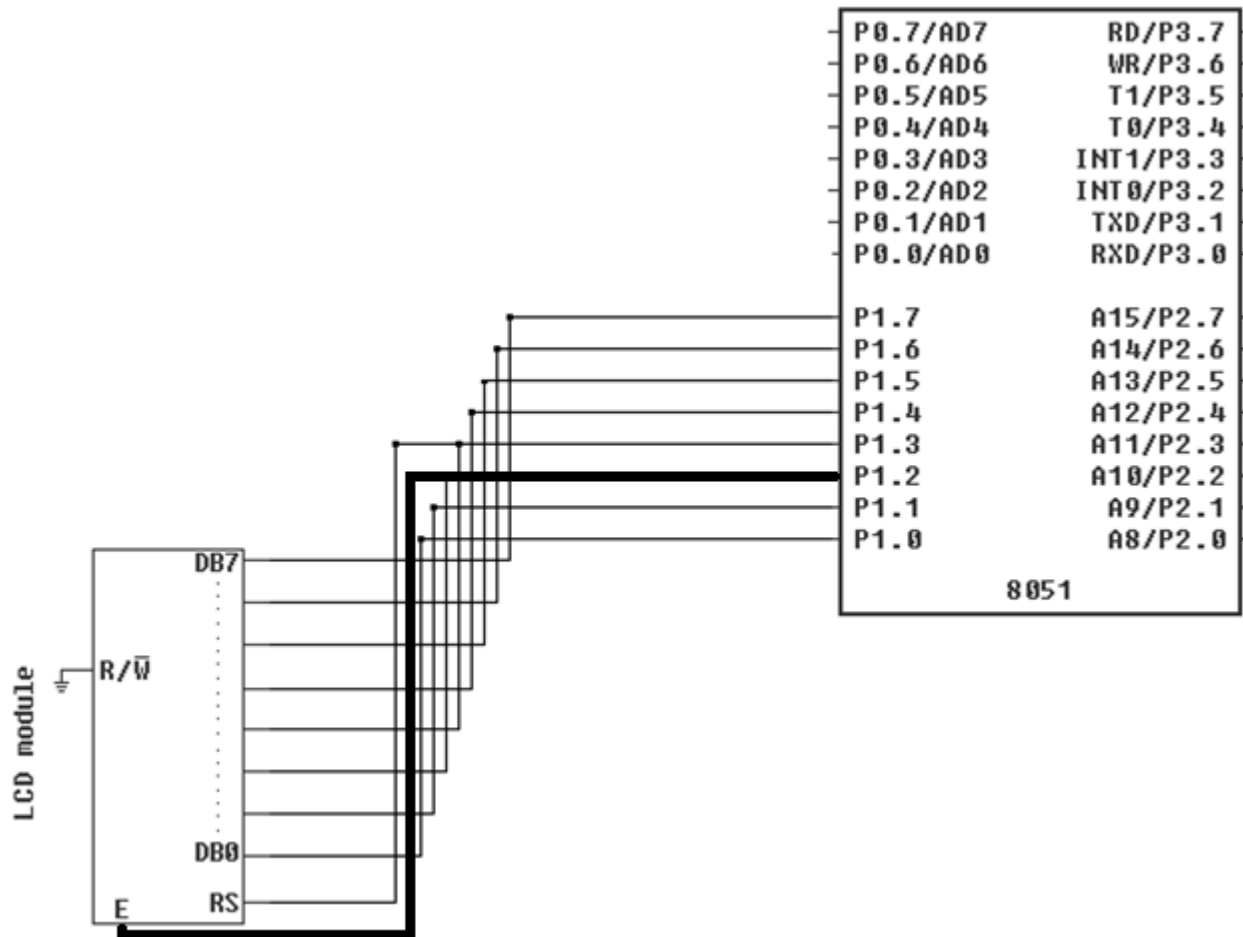
Ligação do Display LCD no 8051 – edisim51

O pino **RS** do display está ligado no port **P1.3**. O pino RS é utilizado para determinar se será enviado dado ou instrução para o display.



Ligação do Display LCD no 8051 – edisim51

O pino **EN** do display está ligado no port **P1.2**. O pino EN é utilizado para habilitar o envio de uma instrução ou dado. No edisim51 usaremos a EN como borda de descida ($1 \rightarrow 0$).



Instruções do Display LCD

Instrução	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	Operação Executada	Tempo
NOP	0	0	0	0	0	0	0	0	0	0	Sem Operação	0
Limpa Display	0	0	0	0	0	0	0	0	0	1	Limpa LCD e retorna cursor para 1a posição	1,65ms
Retorn. Cursor	0	0	0	0	0	0	0	0	1	x	Retorna posição do cursor para a origem (1a. posição da 1a. linha). Mensagens no display não são alteradas.	40µs
Exibição LCD	0	0	0	0	0	0	0	1	I/D	S	Define direção de movimento do cursor(I/D) e deslocamento automático no display (S).	40µs
Controle LCD	0	0	0	0	0	0	1	D	C	B	Ativa display (D), liga/desliga cursor (C) e habilita cursor piscante (B).	40µs
Deslocam. Cursor / LCD	0	0	0	0	0	1	S/C	R/L	x	x	Desloca display ou move cursor (S/C), especificando a direção (R/L).	40µs
Modo LCD	0	0	0	0	1	DL	N	F	x	x	Define largura dos dados enviados (DL), número de linhas (N) e fonte de caracter (F).	40µs
End. CGRAM	0	0	0	1	Endereço CGRAM						Define endereço da RAM gráfica (CGRAM). Dado deve ser enviado na sequência.	40µs
Posic. Cursor	0	0	1	Posição do cursor (0-15;)							Define posição do cursor no display. Dado deve ser enviado na sequência.	40µs
Estado LCD	0	1	BF	Posição em uso							Indicador de LCD ocupado (BF) e posição	0
Escrita Dado	1	0	Dado								Escreve dado no display	40µs
Leitura Dado	1	1	Dado								Lê dado do display	40µs
x : Tanto faz	I/D	1 0	Incrementa Decrementa						R/L	1 0	Deslocamento para a direita Deslocamento para a esquerda	
	S	1 0	Deslocamento automático de mensagem						DL	1 0	Interface de 8 bits Interface de 4 bits	
	D	1 0	Display ativo (exibe) Display inativo (apagado)						N	1 0	2 linhas 1 linha	
	C	1 0	Cursor ativo (exibe) Cursor inativo (apagado)						F	1 0	5x10 pixels 5x7 pixels	
	B	1 0	Cursor em modo piscante						CGRAM : Character Generator RAM			
	S/C	1 0	Desloca mensagem Move cursor									

Endereços de posição do cursor

Os endereços da DDRAM servem para deslocarmos o curso para a posição (linha e coluna) onde será escrito no Display:

Character located	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DDRAM address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
DDRAM address	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F



Programação

Programação de escrita no Display

Primeiramente, antes de começar a escrever no display, deveremos configura-lo, conforme informações do Datasheet do fabricante.

O edsim51 emula o Display LCD 16x2 da HITACHI.

O Datasheet esta disponível no link:

<https://www.edsim51.com/8051simulator/HD44780.pdf>

Subrotina de inicialização do display no edsim51

```
; initialise the display
; see instruction set for details
lcd_init:

    CLR RS      ; clear RS - indicates that instructions are being sent to the module

; function set
    CLR P1.7    ; |
    CLR P1.6    ; |
    SETB P1.5   ; |
    CLR P1.4    ; | high nibble set

    SETB EN     ; |
    CLR EN      ; | negative edge on E

    CALL delay  ; wait for BF to clear
                ; function set sent for first time - tells module to go into 4-bit mode
; Why is function set high nibble sent twice? See 4-bit operation on pages 39 and 42 of HD44780.pdf.

    SETB EN     ; |
    CLR EN      ; | negative edge on E
                ; same function set high nibble sent a second time

    SETB P1.7   ; low nibble set (only P1.7 needed to be changed)

    SETB EN     ; |
    CLR EN      ; | negative edge on E
                ; function set low nibble sent
    CALL delay  ; wait for BF to clear
```

subrotina de inicialização do display no edsim51

```
; entry mode set
; set to increment with no shift
CLR P1.7      ; |
CLR P1.6      ; |
CLR P1.5      ; |
CLR P1.4      ; | high nibble set

SETB EN       ; |
CLR EN        ; | negative edge on E

SETB P1.6     ; |
SETB P1.5     ; | low nibble set

SETB EN       ; |
CLR EN        ; | negative edge on E

CALL delay    ; wait for BF to clear

; display on/off control
; the display is turned on, the cursor is turned on and blinking is turned on
CLR P1.7      ; |
CLR P1.6      ; |
CLR P1.5      ; |
CLR P1.4      ; | high nibble set

SETB EN       ; |
CLR EN        ; | negative edge on E

SETB P1.7     ; |
SETB P1.6     ; |
SETB P1.5     ; |
SETB P1.4     ; | low nibble set

SETB EN       ; |
CLR EN        ; | negative edge on E

CALL delay    ; wait for BF to clear
RET
```

Programação de escrita no Display

Depois de configurado conforme datasheet do fabricante já podemos escrever no display.

O código a seguir apresenta uma subrotina para escrever um caractere no display.

Subrotina que escreve um caractere no Display

Instrução	RS	D7	D6	D5	D4	D3	D2	D1	D0
Escrita no Display	1	Dado							

sendCharacter :

```
SETB RS                ; set RS - indicates that data is being sent to module
MOV C, ACC.7            ; |
MOV P1.7, C            ; |
MOV C, ACC.6            ; |
MOV P1.6, C            ; |
MOV C, ACC.5            ; |
MOV P1.5, C            ; |
MOV C, ACC.4            ; |
MOV P1.4, C            ; | high nibble set

SETB EN                ; |
CLR EN                 ; | negative edge on E

MOV C, ACC.3            ; |
MOV P1.7, C            ; |
MOV C, ACC.2            ; |
MOV P1.6, C            ; |
MOV C, ACC.1            ; |
MOV P1.5, C            ; |
MOV C, ACC.0            ; |
MOV P1.4, C            ; | low nibble set

SETB EN                ; |
CLR EN                 ; | negative edge on E

CALL delay              ; wait for BF to clear
RET
```

Programação de escrita no Display

Agora no programa principal devemos chamar a subrotina de inicialização e após a inicialização escrever no display.

```
org 0000h
```

```
LJMP START
```

```
org 0030h
```

```
START:
```

```
acall lcd_init
```

```
MOV A, #'F'
```

```
CALL sendCharacter ; send data in A to LCD module
```

```
MOV A, #'E'
```

```
CALL sendCharacter ; send data in A to LCD module
```

```
MOV A, #'I'
```

```
CALL sendCharacter ; send data in A to LCD module
```

```
JMP $
```

Programação de escrita no Display

Observe que o texto foi escrito no display a partir da primeira linha na primeira coluna, isso devido a posição de inicialização do cursor.



Programação de escrita no Display

Agora escreveremos uma subrotina que posiciona o cursor em qualquer linha e coluna do display.

Subrotina que posiciona o cursor no Display

Instrução	RS	D7	D6	D5	D4	D3	D2	D1	D0
Posiciona o Cursor	0	1	Posição do Cursor						

posicionaCursor :

CLR RS ; clear RS - indicates that instruction is being sent to module

SETB P1.7 ;|

MOV C, ACC.6 ;|

MOV P1.6, C ;|

MOV C, ACC.5 ;|

MOV P1.5, C ;|

MOV C, ACC.4 ;|

MOV P1.4, C ;| high nibble set

SETB EN ;|

CLR EN ;| negative edge on E

MOV C, ACC.3 ;|

MOV P1.7, C ;|

MOV C, ACC.2 ;|

MOV P1.6, C ;|

MOV C, ACC.1 ;|

MOV P1.5, C ;|

MOV C, ACC.0 ;|

MOV P1.4, C ;| low nibble set

SETB EN ;|

CLR EN ;| negative edge on E

CALL delay ; wait for BF to clear

RET

Character located	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DDRAM address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
DDRAM address	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F

Programação de escrita no Display

Escrevendo no meio do display

```
org 0000h
```

```
LJMP START
```

```
org 0030h
```

```
START:
```

```
acall lcd_init
```

```
mov A, #06h
```

```
ACALL posicionaCursor ; Posiciona o cursor na coluna 06 da primeira linha
```

```
MOV A, #'F'
```

```
CALL sendCharacter ; send data in A to LCD module
```

```
MOV A, #'E'
```

```
CALL sendCharacter ; send data in A to LCD module
```

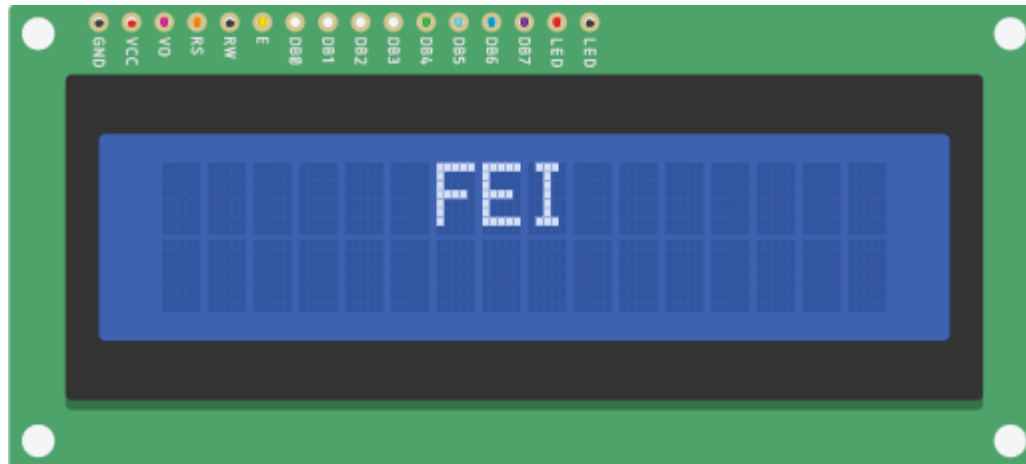
```
MOV A, #'I'
```

```
CALL sendCharacter ; send data in A to LCD module
```

```
JMP $
```

Programação de escrita no Display

Observe que o texto agora foi escrito no display a partir da 7ª coluna da primeira linha, isso devido ao posicionamento do cursor.



Subrotina que limpa o display

Instrução	RS	D7	D6	D5	D4	D3	D2	D1	D0
Limpa Display	0	0	0	0	0	0	0	0	1

;Limpa todo o display e retorna o cursor para primeira posição

clearDisplay :

CLR RS ; clear RS - indicates that instruction is being sent to module

CLR P1.7 ; |

CLR P1.6 ; |

CLR P1.5 ; |

CLR P1.4 ; | high nibble set

SETB EN ; |

CLR EN ; | negative edge on E

CLR P1.7 ; |

CLR P1.6 ; |

CLR P1.5 ; |

SETB P1.4 ; | low nibble set

SETB EN ; |

CLR EN ; | negative edge on E

CALL delay ; wait for BF to clear

RET

Subrotina que retorna o cursor

Instrução	RS	D7	D6	D5	D4	D3	D2	D1	D0
Retorna Cursor	0	0	0	0	0	0	0	1	1

;Retorna o cursor para primeira posição sem limpar o display

retornaCursor :

CLR RS ; clear RS - indicates that instruction is being sent to module

CLR P1.7 ; |

CLR P1.6 ; |

CLR P1.5 ; |

CLR P1.4 ; | high nibble set

SETB EN ; |

CLR EN ; | negative edge on E

CLR P1.7 ; |

CLR P1.6 ; |

SETB P1.5 ; |

SETB P1.4 ; | low nibble set

SETB EN ; |

CLR EN ; | negative edge on E

CALL delay ; wait for BF to clear

RET

Programação de escrita no Display

Escrevendo no meio do display e retornando o cursor

```
org 0000h  
LJMP START
```

```
org 0030h
```

```
START:
```

```
acall lcd_init
```

```
mov A, #06h
```

```
ACALL posicionaCursor ; Posiciona o cursor na coluna 06 da primeira linha
```

```
MOV A, #'F'
```

```
ACALL sendCharacter ; send data in A to LCD module
```

```
MOV A, #'E'
```

```
ACALL sendCharacter ; send data in A to LCD module
```

```
MOV A, #'I'
```

```
ACALL sendCharacter ; send data in A to LCD module
```

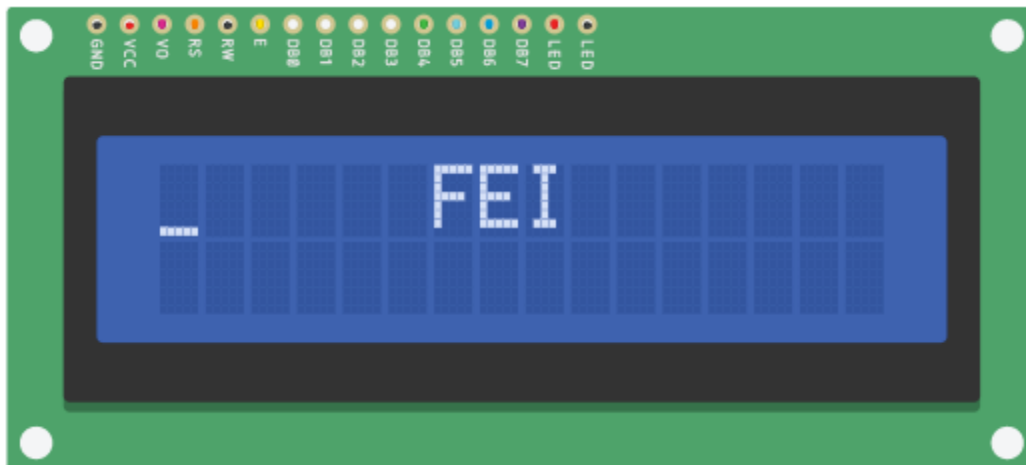
```
ACALL retornaCursor
```

```
JMP $
```

Programação de escrita no Display

Observe que o texto agora foi escrito no display a partir da 7ª coluna da primeira linha, e o cursor está na primeira posição.

Isso significa que a escrita começará na posição que o cursor está.



Caracteres de escrita no Display

No display LCD cada caractere é representado por um número, onde a maioria dos caracteres são a mesma numeração da tabela ASCII.



Padrão de Caracteres do Display

Lever 4 Bits	Upper 4 bits	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111	
		CG RAM (1)																
xxxx0000					0	a	P	`	P				-	タ	ミ	α	ρ	
xxxx0001	(2)			!	1	A	Q	a	q				。	ア	チ	厶	ä	q
xxxx0010	(3)			"	2	B	R	b	r				「	イ	ツ	×	ρ	θ
xxxx0011	(4)			#	3	C	S	c	s				」	ウ	テ	モ	ε	∞
xxxx0100	(5)			\$	4	D	T	d	t				、	エ	ト	ヲ	μ	Ω
xxxx0101	(6)			%	5	E	U	e	u				・	オ	ナ	ユ	σ	ü
xxxx0110	(7)			&	6	F	V	f	v				ヲ	カ	ニ	ヨ	ρ	Σ
xxxx0111	(8)			'	7	G	W	g	w				ア	キ	ヌ	ラ	g	π
xxxx1000	(1)			(8	H	X	h	x				イ	ク	ネ	リ	、	Σ
xxxx1001	(2))	9	I	Y	i	y				ウ	ケ	ル	ル	'	Y
xxxx1010	(3)			*	:	J	Z	j	z				エ	コ	ハ	レ	j	≠
xxxx1011	(4)			+	;	K	[k	[オ	サ	ヒ	ロ	×	π
xxxx1100	(5)			,	<	L	¥	l	l				カ	シ	フ	ワ	Φ	円
xxxx1101	(6)			-	=	M]	m	}				ユ	ス	ハ	ン	も	÷
xxxx1110	(7)			.	>	N	^	n	+				ヨ	セ	ホ	ゝ	ん	
xxxx1111	(8)			/	?	O	_	o	←				ッ	ソ	マ	°	ö	■

Programação de escrita no Display

Escrevendo no meio do display e retornando o cursor

```
org 0000h  
LJMP START
```

```
org 0030h
```

```
START:
```

```
acall lcd_init
```

```
mov A, #06h
```

```
ACALL posicionaCursor ; Posiciona o cursor na coluna 06 da primeira linha
```

```
MOV A, #46h
```

```
ACALL sendCharacter ; send data in A to LCD module
```

```
MOV A, #45h
```

```
ACALL sendCharacter ; send data in A to LCD module
```

```
MOV A, #49h
```

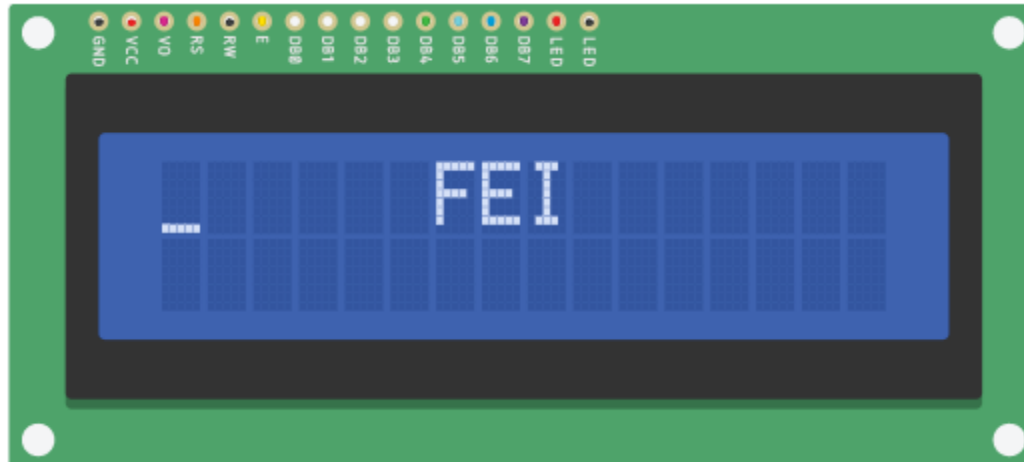
```
ACALL sendCharacter ; send data in A to LCD module
```

```
ACALL retornaCursor
```

```
JMP $
```

Programação de escrita no Display

Observe que o texto que foi escrito no display.



Bibliografia

ZELENOVSKY, R.; MENDONÇA, A. Microcontroladores Programação e Projeto com a Família 8051. MZ Editora, RJ, 2005.

Gimenez, Salvador P. Microcontroladores 8051 - Teoria e Prática, Editora Érica, 2010.