Programando displays de cristal líqüido a partir da porta paralela do PC

Programming liquid crystal display from PC parallel port

Carlos Sica[1]

Resumo

Como toda máquina, computadores utilizam uma forma específica de processar dados, utilizam a energia elétrica para representar seus dados. Naturalmente para tornar estes dados compreensíveis ao ser humano, são necessárias interfaces entre a máquina e o homem.

Em computadores pessoais, a interface mais utilizada é o monitor de vídeo, porém, em máquinas que executam tarefas específicas como o forno de microondas, relógios, telefones entre outros, não é necessário utilizar um produto tão complexo quanto a tela de um computador. Neste caso é muito comum se utilizar os display's de 7 segmentos ou os de cristal líqüido.

Este artigo busca elucidar o funcionamento dos displays de cristal líqüido, principalmente pelo fato de serem operados por modernos dispositivos chamados controladores, que se assemelham ao microcontroladores ou microprocessadores. O controlador mais citado na literatura eletrônica talvez seja o HD44780 da Hitachi, porém, vários outros seguem o mesmo padrão como o S6A0069 ou KS0066U da Sansung que controlam o módulo PC1601-A da Powertip, que vamos destacar neste artigo. Além disso, será explorada sua potencialidade através de exemplos implementados em laboratório desenvolvidos em linguagem assembly 8088, compatível com os computadores PC atuais.

Palavras Chave

Controlador, Display

Abstract

As such as al machine, computers use a particular kind to data process. To transform this data to comprehensive form, are necessary interface between the machine and human.

In personal computer, the most used interface is the video monitor, but, in machine that run specific jobs, as such as, microwave stove, watches, phones and so on, it's not necessary to use expensive or complex devices, it's frequent to use a 7 segment or liquid crystal displays.

This paper looking for to explain the working of liquid crystal display, principally because it are operated by up to date device named controller, that are similar to micro-controllers and microprocessors.

The most referenced controller on literature maybe is the HD44780 from Hitachi, but, a lot of follow its default, as such as, S6A0069 and KS0066U from Samsung that to control the PC1601-A Powertip's module that going to be showed in this paper. Additionally, will be explored its potentiality from examples implemented into laboratory and developed in 8088's assembly language, PC compatible.

Key Words

Controller, Display

---

[1] Carlos Benedito Sica de Sica – Professor do Departamento de Informática da Universidade Estadual de Maringá, Avenida Colombo, 5790 – Bloco 19/DIN

1. Introdução

Os Displays de Cristal líqüido são conhecidos pelas letras LCD, que significam "**L**iquid **C**ristal **D**isplay". São interfaces que utilizam uma tecnologia moderna para representar letras, números e símbolos advindos de microprocessadores ou de microcontroladores. Basicamente a técnica utilizada é a de polarizar o cristal líqüido que muda de cor quando energizado. Para 'desenhar' as letras, números e outros símbolos, cada dígito é composto por uma matriz, fabricado tipicamente com 5 colunas e 8 linhas, das quais, em geral, a última é utilizada para o cursor, por este motivo alguns usuários o atribuem o tamanho de 5x7 ao invés de 5x8.

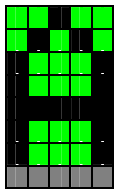

Figura 1) matriz 5x8 dos caracteres

Existem diversos fabricantes deste tipo de interface, mas todos os módulos de LCD estudados possuem um circuito integrado encarregado de controlar a matriz de cristal líquido, presente em sua arquitetura, onde são representados os caracteres e outros símbolos. Este circuito integrado, chamado controlador é o coração do display, pois, se encarrega de gerar os sinais adequados para energizar os pontos certos de cada matriz para 'acender' os símbolos desejados, além de efetuar operações, tais como, limpar o display, piscar o cursor e rolar a linha, provocando efeitos de movimento. O tamanho dessa matriz pode variar de modelo para modelo.

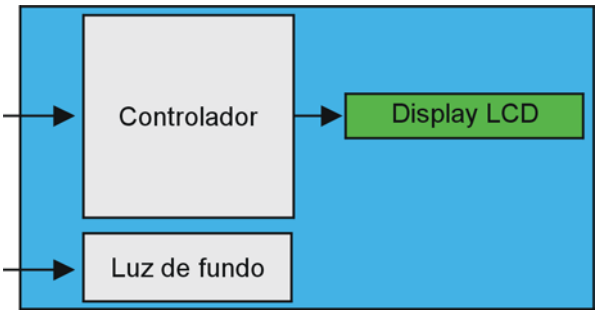

Figura 2) Módulo LCD

Pelo fato do display propriamente dito, não trabalhar sem o controlador, deve-se considerar sempre o conjunto, o qual é chamado de módulo LCD, relação qual, é mostrada na Figura 2.

2. A arquitetura do LCD

Para detalhar o módulo LCD, é necessário estudar sua arquitetura. A parte externa compõe-se pela pinagem de acesso a parte interna.

A pinagem dos displays de cristal líqüido obedecem à terminologia abaixo:

Pino	***Símbolo***	***Função***	***Descritivo***
1	Vss	Alimentação	0 Volts (terra)
2	Vdd	Alimentação	+ 5 Volts
3	Vo	Ajuste de contraste	Deve receber um sinal que varia de 0 a 5 Volts para ajustar o contraste. Se for ligado diretamente ao terra terá o contraste no máximo.
4	RS	Bit de seleção de registrador	Quando colocado em 0 volts, seleciona o registrador de instruções (IR). Quando colocado em 5 volts, seleciona o registrador de dados (DR)
5	R/-W	Bit de seleção de operação	Quando colocado em 0 volts, seleciona a operação de escrita. Quando colocado em 5 volts, seleciona a operação de leitura.
6	E	Bit de 'enable'	Habilita a escrita na memória, após o registrador IR ou DR ter recebido a informação.
7	DB0	Linha 0 do barramento de dados	Bit 2^0 (menos significativo) a ser escrito no registrador IR ou DR
8	DB1	Linha 1 do barramento de dados	Bit 2^1 a ser escrito no registrador IR ou DR
9	DB2	Linha 2 do barramento de dados	Bit 2^2 a ser escrito no registrador IR ou DR
10	DB3	Linha 3 do barramento de dados	Bit 2^3 a ser escrito no registrador IR ou DR
11	DB4	Linha 4 do barramento de dados	Bit 2^4 a ser escrito no registrador IR ou DR
12	DB5	Linha 5 do barramento de dados	Bit 2^5 a ser escrito no registrador IR ou DR
13	DB6	Linha 6 do barramento de dados	Bit 2^6 a ser escrito no registrador IR ou DR
14	DB7	Linha 7 do barramento de dados	Bit 2^7 a ser escrito no registrador IR ou DR
15	A	Alimentação	Anôdo (+) da luz de fundo
16	K	Alimentação	Katôdo (-) da luz de fundo

Tabela 1) pinagem externa do módulo LCD

2.1. Arquitetura interna

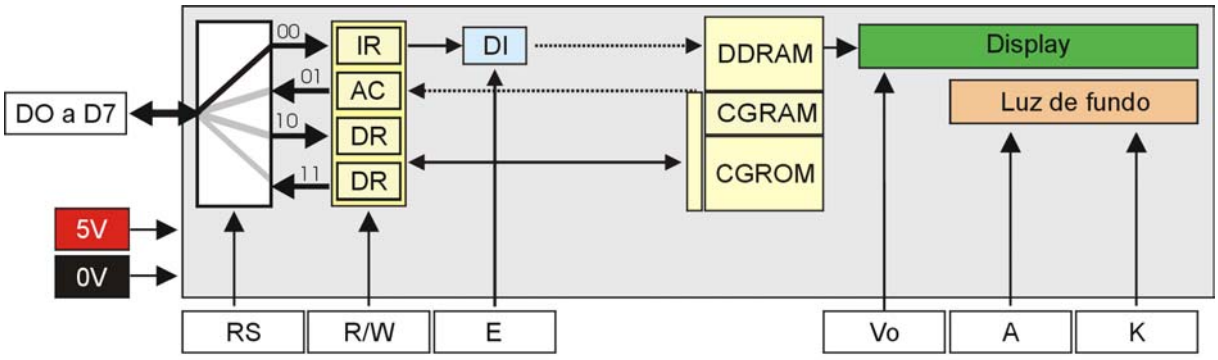

Figura 3) arquitetura interna hipotética (proposta pelo autor)

Existem displays com e sem luz de iluminação do fundo, no segundo caso, os pinos A (15) e K (16) não são válidos ou não existem.

A unidade DI é chamada decodificador de instruções, responsável por atuar endereçando as memórias e o display.

2.2. Registradores

Os LCD's possuem três registradores de 8 bits aos quais o programador tem acesso.

- O registrador de instruções, chamado IR (**I**nstruction **R**egister) comanda o display de acordo com as operações de configuração e instruções detalhadas na Tabela 4, inclusive a de apontamento para a memória com os endereços descritos na Tabela 2 e Tabela 3, bem como, os endereços que posicionam o caractere escrito na DDRAM (item 2.3.2).
- O registrador de dados, DR (**D**ata **R**egister), serve para armazenar os dados que serão escritos no display, para tanto, RS deve conter 1 lógico e R/W 0 lógico. Estes dados seguem o padrão ASCII (**A**merican **S**tandard **C**ode **I**nternational **I**nterchange) e também podem ser programados pelo usuário através da memória CGRAM (item 2.3.3). Além disso, numa operação de leitura, ele fornecerá o dado nele armazenado. Na operação de leitura o bit R/W conterá 1 lógico.
- O registrador contador de endereço é chamado de AC (**A**ddress **C**ounter), ele é ajustado pelo IR com o endereço de memória onde será escrito o próximo caractere. Ele pode ser incrementado automaticamente quando um caractere é escrito ou decrementado quando uma informação é lida da DDRAM, para tanto, o módulo deve estar programado para tal, de acordo com as instruções descritas na Tabela 4. Esse endereço pode poder ser lido nos bits D0 a D6 do barramento quando RS=0 e R/W=1. O último bit do barramento, D7 conterá o BF (**B**usy **F**lag).

2.3. Memórias

Os controladores de display de cristal líqüido possuem um bloco de memória que totaliza 384 bytes efetivos. Essa memória se divide em três áreas, uma somente leitura (CGROM) e duas de escrita/leitura (DDRAM e CGRAM).

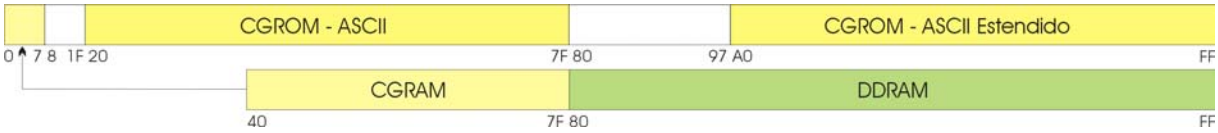

Figura 4) conjunto de memória

2.3.1. CGROM

A CGROM (**C**haracter **G**enerator **R**ead **O**nly **M**emory) é a parte do controlador que não pode ser modificada pelo programador, ela possui 192 caracteres pré-programados, endereçados de 20h a 7Fh e de A0h a FFh.

Quando se envia um código ASCII para o LCD, através do registrador DR, o controlador utiliza este código como endereço e verifica na memória CGROM o mapa de bits correspondente aquele caractere, para depois enviá-lo ao display propriamente dito através da DDRAM, por este motivo, é chamada de geradora de caracteres.

Cada um desses endereços corresponde a uma letra do alfabeto, número ou símbolo que, em sua maioria, estão presentes na tabela ASCII. Por exemplo, o endereço 41h corresponde à letra A, o 42h à letra B e assim sucessivamente, porém, alguns displays não seguem esse padrão, pelo fabricante ter preferido reservar uma parte da memória para escrever caracteres, por exemplo, de línguas orientais. A Figura 5 mostra o mapeamento da memória CGROM.

Como os primeiros códigos da tabela ASCII não guardam caracteres visíveis, os fabricantes optaram por criar uma área de memória passível de programação, para que o programador pudesse criar caracteres especiais, esta seqüência de endereços é mapeada na CGRAM, conforme será descrito no item 2.3.3.

Figura 5) mapeamento da CGROM (Hitachi, manual HD 44780U)

2.3.2. DDRAM

O controlador possui uma memória RAM de dados de 128 bytes (80h), endereçados de 80h a FFh, onde são escritos os caracteres que vão aparecer no display automaticamente (comandados pelo controlador). Estes caracteres podem ser advindos da CGROM ou da CGRAM.

Ela é dividida em blocos, que se referem às linhas do display. A primeira linha inicia em 80h e vai até BFh, a segunda linha inicia em C0h e vai até FF compondo um total de 64 caracteres cada linha.

Descrição	***Modo***	***RS***	***RW***	***IR***
Endereços DDRAM*	Primeira posição da primeira linha	0	0	80
	Primeira posição da segunda linha	0	0	C0

Tabela 2) endereços da DDRAM

A maioria dos módulos disponíveis no mercado, portanto, obedecem ao padrão de no máximo 2 linhas de 64 caracteres. O LCD utilizado como exemplo, o PC1601-A se apresenta no formato 1 linha de 16 caracteres, conforme ilustra a Figura 6.

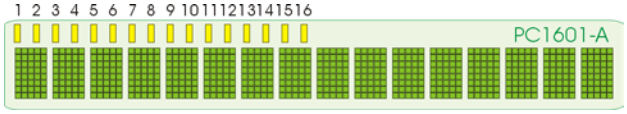

Figura 6) exemplo de módulo

Na prática, porém, esse display trabalha segundo os padrões industriais de 2 duas linhas, obrigando o programador a trabalhar como se ele tivesse de 8 caracteres em cada linha (veja Figura 7), endereçados de 80h a 87h e de C0h a C7h. Portanto, para escrever uma frase corretamente nele deve-se imaginar uma janela que mostra apenas o início das duas linhas.

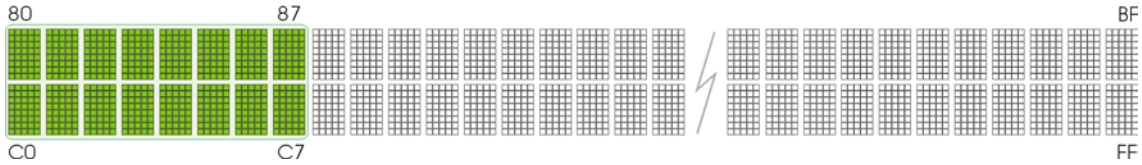

Figura 7) o real funcionamento do PC1601-A

Para escrever um caractere através da DDRAM coloca-se o LCD em modo de escrita (R/W=0) de dados (RS=1) e o código ASCII do caractere no barramento (DR), depois se executa a transição do pino E de nível alto para nível baixo.

Já, para escrever uma instrução coloca-se o LCD em modo de escrita (R/W=0) de instruções (RS=0) e o código da instrução no barramento (DR), depois se executa a transição do pino E de nível alto para nível baixo.

Quando se deseja escrever um caractere em um endereço desejado, basta enviar este endereço como se fosse uma instrução (através do IR) precedendo o envio do caractere (através do DR).

2.3.3. CGRAM

Os LCDs como os estudados, se baseiam em matrizes de pontos para representar cada caractere. Os pontos que 'desenham' os símbolos baseados no código ASCII, estão pré-definidos na CGROM, como discutido no item 2.3.1. Existe, porém, uma área de memória chamada CGRAM (**Caractere Ge**nerator RAM) na qual o programador pode definir caracteres especiais que não constam na tabela ASCII.

A CGRAM possui tipicamente 8 matrizes para compor os caracteres customizáveis. Cada matriz utiliza 8 bytes totalizando uma memória de 64 bytes acessados através de um conjunto de endereços mostrados na Tabela 3.

ASCII	Endereço CGRAM (Hexadecimal)
0	40 a 47
1	48 a 4F
2	50 a 58
3	58 a 5F
4	60 a 67
5	68 a 6F
6	70 a 77
7	78 a 7F

Tabela 3) endereços da CGRAM

Cada uma dessas matrizes, normalmente é formada por 5 bits que representam as colunas e 8 bits que, por sua vez, representam as linhas. Apesar do primeiro argumento, que representa as colunas, utilizar apenas 5 bits, permanece o padrão de hardware de 1 byte. Assim, cada matriz ocupa 64 bits como ilustrado na Figura 8.

endereço	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	binário	hexa	decimal
40									00000100	04	4
41									00001010	0A	10
42									00000100	04	4
43									00011111	1F	31
44									00000100	04	4
45									00001010	0A	10
46									00010001	11	17
47									00010001	11	17

Figura 8) matriz de pontos associada com os bits e bytes da memória CGRAM

Para programar os caracteres na CGRAM, previamente compostos de acordo com a Figura 8, é necessário ajustar RS e RW em 0 lógico, para escrever no registrador IR o endereço inicial do caractere a ser programado de acordo com a Tabela 3 e, após, atribuir 1 lógico ao registrador RS, visando enviar a seqüência de 8 bytes. A CGRAM incrementará os endereços automaticamente.

Para escrever estes caracteres no display, envia-se cada caractere da CGRAM para a DDRAM especificando os códigos de 00 a 07, que correspondem aos "endereços" das matrizes seqüencialmente programadas. Desta forma, os bytes programados nos endereços de 40 a 47 compõem o caractere especial codificado como 00 e, por seqüência, o caractere composto no intervalo de 78 a 7F corresponderá ao código 07.

3. Programação

As instruções abaixo são utilizadas para programar o LCD e devem ser enviadas para o registrador IR, após isto, a linha de 'enable' deve receber a transição de 1 para 0 lógico, somente nesse momento a instrução é executada no display. Quando um endereço é enviado, como os descritos nas duas últimas linhas descritas na Tabela 4, o registrador AC é atualizado com tal valor.

Modo (Rs=0 e R/W=0)									***IR***
Limpa o display (DDRAM) e posiciona o cursor no inicio	0	0	0	0	0	0	0	1	01
Posiciona cursor no início (sem apagar DDRAM)	0	0	0	0	0	0	1	x	02-03
Desloca cursor para esquerda após escrever caractere	0	0	0	0	0	1	0	0	04
Desloca mensagem para esquerda após escrever caractere	0	0	0	0	0	1	0	1	05
Desloca cursor para direita após escrever caractere	0	0	0	0	0	1	1	0	06
Desloca mensagem para esquerda após escrever caractere	0	0	0	0	0	1	1	1	07
Desliga display e cursor, mantendo os dados na DDRAM	0	0	0	0	1	0	0	0	08
Desliga display, dados pemanecem na DDRAM, cursor pisca	0	0	0	0	1	0	0	1	09
Desliga display e liga cursor fixo	0	0	0	0	1	0	1	0	0A
Desliga display, mantendo os dados na DDRAM, liga cursor	0	0	0	0	1	0	1	1	0B
Liga o display e esconde o cursor piscante	0	0	0	0	1	1	0	0	0C
Liga o display e o cursor fica piscando	0	0	0	0	1	1	0	1	0D
Liga o display e o cursor fica fixo	0	0	0	0	1	1	1	0	0E
Liga o display e o cursor fica alternante	0	0	0	0	1	1	1	1	0F
Desloca cursor para esquerda e decrementa AC	0	0	0	1	0	0	x	x	10-13
Desloca cursor para direita e incrementa AC	0	0	0	1	0	1	x	x	14-17
Desloca mensagem para esquerda e cursor acompanha	0	0	0	1	1	0	x	x	18-1B
Desloca mensagem para direita e cursor acompanha	0	0	0	1	1	1	x	x	1C-1F
Interface 4 bits, display de 1 linha e matriz 5x8	0	0	1	0	0	0	x	x	20-23
Interface 4 bits, display de 1 linha e matriz 5x11	0	0	1	0	0	1	x	x	24-27
Interface 4 bits, display de 2 linha e matriz 5x8	0	0	1	0	1	0	x	x	28-2B
Interface 4 bits, display de 2 linha e matriz 5x11	0	0	1	0	1	1	x	x	2C-2F
Interface 8 bits, display de 1 linha e matriz 5x8	0	0	1	1	0	0	x	x	30-33
Interface 8 bits, display de 1 linha e matriz 5x11	0	0	1	1	0	1	x	x	34-37
Interface 8 bits, display de 2 linha e matriz 5x8	0	0	1	1	1	0	x	x	38-3B
Interface 8 bits, display de 2 linha e matriz 5x11	0	0	1	1	1	1	x	x	3C-3F
Endereços para escrever na CGRAM	0	1	AC5	AC4	AC3	AC2	AC1	AC0	40-7F
Endereços para escrever/ler na DDRAM	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	80-FF

Tabela 4) código das instruções que comandam o LCD

3.1. Ligação entre o PC e o DISPLAY

Os exemplos a seguir visam comandar o LCD através da porta paralela do PC, para tanto, utiliza-se a ligação descrita na Figura 9.

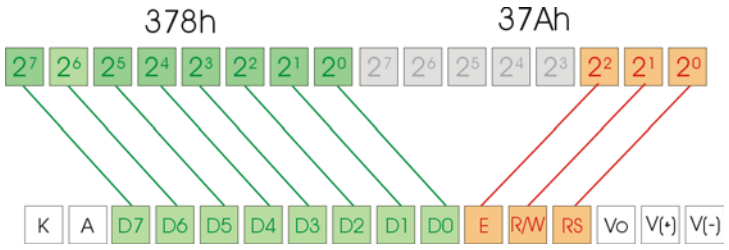

Figura 9) ligação entre o PC e LCD

3.2. Rotinas básicas para programar um display

Ao longo deste artigo, foi debatida a seqüência para executar uma instrução ou definir o cursor em uma posição da memória DDRAM.

A primeira rotina assembly apresentada é a que envia e executa uma instrução:

```
envia_i proc
   ;este procedimento espera que ah contenha a instrução a ser executada
   mov   dx,37Ah  ;endereço da porta do PC ligada aos pinos de controle do LCD
   mov   al,00000011b   ; E=0, RW=0, RS =0 (o pc inverte os dois primeiros bits)
   out   dx,al    ;envia palavra de controle para o barramento

   mov   dx,378h  ;endereço da porta do PC ligada ao registrador de dados do LCD
   mov   al,ah    ;ah contém a palavra de controle a ser enviada
   out   dx,al    ;envia a palavra de controle para o barramento

   mov   dx,37Ah  ;endereço da porta do PC ligada aos pinos de controle do LCD
   mov   al,00000111b   ; E=1, RW=0, RS=0 (o pc inverte os dois primeiros bits)
   out   dx,al    ;envia palavra de controle para o barramento
   call  atraso   ;para garantir E=1 durante um tempo

   mov   al,00000011b   ; E=0, RW=0, RS=0 (o pc inverte os dois primeiros bits)
   out   dx,al    ;envia palavra de controle para o barramento
   call  atraso   ;para garantir E=0 durante um tempo

   ret
envia_i endp
```

Para enviar um dado para a DDRAM e, conseqüentemente, escrevê-lo no display, utilizou-se a rotina abaixo:

```
envia_d proc
   mov   dx,37Ah  ;endereço da porta do PC ligada aos pinos de controle do LCD
   mov   al,00000010b   ; E=0, RW=0, RS=1 (o pc inverte os dois primeiros bits)
   out   dx,al    ;envia palavra de controle para o barramento

   mov   dx,378h  ;endereco da porta do PC ligada ao registrador de dados do LCD
   mov   al,ah    ;ah contém a palavra de controle a ser enviada
   out   dx,al    ;envia o código do caractere para o barramento

   mov   dx,37Ah  ;endereço da porta do PC ligada aos pinos de controle do LCD
   mov   al,00000110b   ; E=1, RW=0, RS=1 (o pc inverte os dois primeiros bits)
   out   dx,al    ;envia palavra de controle para o barramento
   call  atraso   ;para garantir E=1 durante um tempo

   mov   al,00000010b   ; E=0, RW=0, RS=1 (o pc inverte os dois primeiros bits)
   out   dx,al    ;envia palavra de controle para o barramento
   call  atraso   ;para garantir E=0 durante um tempo

   ret
envia_d endp
```

3.3. A transição do Enable

Para que uma instrução seja executada ou que um dado seja realmente escrito no display, a linha de enable deve sofrer uma transição de 1 para 0 num intervalo de tempo suficientemente longo para que o decodificador de instruções perceba ou que o display tenha tido tempo para acender toda a matriz correspondente ao código recebido. O tempo necessário para gerar um transição de 1 para 0 foi testado no PC e abaixo é apresentada a rotina que provoca o atraso exigido.

```
atraso proc
       mov   cx,20000
volta: loop volta
       ret
atraso endp
```

3.4. Inicialização do LCD

Ao ligar o módulo LCD pode ser necessário executar algumas instruções que farão a preparação dele. Em primeiro lugar, deve-se especificar que tipo de LCD está sendo programado (veja Tabela 4) e, depois, o modo de operação desejada, tais como, definir o comportamento do cursor e da frase a ser escrita. Vale destacar que alguns módulos trazem uma rotina de inicialização embutida, que é executada automaticamente cada vez que recebe alimentação.

```
; tipo de display
   mov   ah,38H  ;duas linhas e matriz de 5x8 (exemplo: PC1601A POWERTIP)
   call  envia_i

; limpa display e move cursor para o inicio
   mov   ah,01H
   call  envia_i

; sentido do descolamento do cursor
   mov   ah,06H  ;configura cursor com auto-incremento para a direita
   call  envia_i

; controle do comportamento cursor
   mov   ah,0cH  ;configura cursor piscante
   call  envia_i
```

3.5. Exemplo de programação da CGRAM

Considerando o item 2.3.3 que descreve a forma de programar a CGRAM, é apresentada agora a rotina que programa um caractere especial na CGRAM. Após esta programação, pode-se enviar o símbolo criado, através do código 00 (veja exemplo no item 3.6)

```
   mov ah, 040H
   call envia_i  ;envia endereço inicial da CGRAM
   mov ah, 04H
   call envia_d  ;envia programação 1a linha da matriz

   mov ah, 0aH
   call envia_d  ;envia programação 2a linha da matriz

   mov ah, 0aH
   call envia_d  ;envia programação 3a linha da matriz

   mov ah, 04H
   call envia_d  ;envia programação 4a linha da matriz

   mov ah, 01fH
   call envia_d  ;envia programação 5a linha da matriz

   mov ah, 04H
   call envia_d  ;envia programação 6a linha da matriz

   mov ah, 0aH
   call envia_d  ;envia programação 7a linha da matriz

   mov ah, 011H
   call envia_d  ;envia programação 8a linha da matriz
```

3.6. Envia um caractere

```
   mov   ah,080h   ;posiciona o cursor na primeira posição da primeira linha
   call  envia_i   ;envia o endereço
   mov   ah,'x'    ;define o código ascii do caractere
   call  envia_d   ;envia a letra x para a primeira posição da primeira linha
   mov   ah,0C7h   ;posiciona o cursor na primeira posição da segunda linha
   call  envia_ i  ;envia o endereço
   mov   ah,0      ;define o código do caractere especial programado na CGRAM
   call  envia_d   ;envia o símbolo para a primeira posição da segunda linha
```

3.7. Enviando uma seqüência de 16 caracteres

Para escrever uma frase, é necessário enviar repetidamente os caracteres contidos nela, porém, o programador precisa ajustar somente o endereço da primeira posição e o LCD fará o incremento do endereço automaticamente.

Lembre-se que o LCD utilizado como exemplo tem uma característica muito especial: apesar de possuir apenas uma linha com 16 caracteres (16x1), ele se comporta como se tivesse duas linhas de 8 caracteres (8x2).

Por este motivo, deve-se enviar os 8 primeiros caracteres para o endereço 80h, que é a posição inicial da memória de dados DDRAM, isto também pode ser feito automaticamente quando envia-se a instrução '01', que limpa o display e fixa o cursor na posição inicial da primeira linha, porém, quando chegar ao nono caractere, deve-se redirecionar o cursor para a posição C0h que é o endereço inicial da segunda linha.

```
envia_16 proc
;SI deve estar apontando para o início do conjunto de 16 caracteres

       mov   ah,01H          ;limpa display e posiciona o cursor em 80h
       call  envia_i
       mov bx,0              ;BX será o contador de caracteres

L1:    cmp byte ptr [si],0   ;verifica se é fim da frase
       je fim
       mov ah,[si]           ;pega caractere apontado por SI (SI->'x')
       call envia_d          ;envia caractere
       inc si                ;aponta para o próximo caractere da frase
       cmp bx, 7             ;verifica se já escreveu 8 caracteres
       je L2ini              ;se sim, mudará o endereço para C0h
       inc bx                ;se não, conta caractere
       jmp L1                ;e volta para enviar mais um

L2ini: mov   ah,0c0H         ;define endereço da segunda linha
       call  envia_i         ;e posiciona o cursor nele

L2:    cmp byte ptr [si],0
       je fim
       mov ah, [si]
       call envia_d
       inc si
       cmp bx, 15
       je fim
       inc bx
       jmp L2

fim:  ret
envia_16 endp
```

3.8. Enviando uma frase completa

Se o procedimento apresentado no item 3.7 for utilizado repetidamente, pode-se visualizar uma frase inteira se deslocando no display.

```
envia_f proc
;SI deve apontar para o início da frase

; deve-se provocar um atraso grande para ver a frase passando devagar no display
        mov cx, 1000
mais:   push cx
        call atraso
        pop cx
        loop mais

linha1: push si             ;preserva SI, pois, envia_16 irá alterá-lo
        call envia_16       ;imprime 16 caracteres a partir de SI
        pop si
        inc si              ;SI aponta para o próximo caractere
```

```
        jmp linha1          ;envia mais 16

fim:    ret
envia_f endp
```

4. Conclusão

Programar displays de cristal líqüido tem sido objeto de estudo dos cursos de graduação de muitas disciplinas de universidades com cursos de base tecnológica, por este motivo, a abordagem deste tema técnico foi feita de forma didática, visando elucidar o funcionamento de um dispositivo muito aplicado na indústria, o controlador de displays de cristal líqüido. O teor didático foi obtido através da realização de testes em sala de aula, onde os alunos puderam experimentar os conceitos documentados pelo professor e as técnicas desenvolvidas em forma de algorítimo.

As experiências que geraram os trechos de códigos fonte apresentados, podem se tornar objeto de trabalhos futuros se transportados para outras máquinas, como é o caso de microcontroladores. Por outro lado, esse conhecimento deve abrir horizontes para grupos de trabalho e de pesquisa que se dedicam ao desenvolvimento de sistemas em linguagem de alto nível.

5. Bibliografia

Celso Renato G. Providelo, Thiago Pinheiro Felix da Silva e Lima, Desenvolvimento da interface com o módulo LCD baseado no HD44780 utilizando o MC68HC705J1A, http://gdm.sel.eesc.sc.usp.br/newpage/documents/lcd/

Fred Cox, http://paginas.terra.com.br/lazer/fredcox/lcd/gerador.htm

Ilton Barbacena, Cláudio Afonso Fleury, Display LCD, http://www.eletronica.etc.br/igor/apostilaselet/

Julyan Ilett, How to use Intelligent Liquid Crystal Displays, http://www.epemag.wimborne.co.uk/resources.htm

Hitachi, Manual técnico do controlador HD44780

Sansung, Manual técnico d controlador S6A0069 e KS0066U

Powertip, Manual técnico do módulo PC1601-A da