

ARQUITETURA DE COMPUTADORES
PROJETO DO HARDWARE MÍNIMO DE UM COMPUTADOR
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO, FC-UNESP
TRABALHO DE CONCLUSÃO DA DISCIPLINA (TCD)

Aluno: Davi Augusto Neves Leite

RA: 191027383

E-mail: davi.neves@unesp.br

1. DESCRIÇÃO DO PROJETO

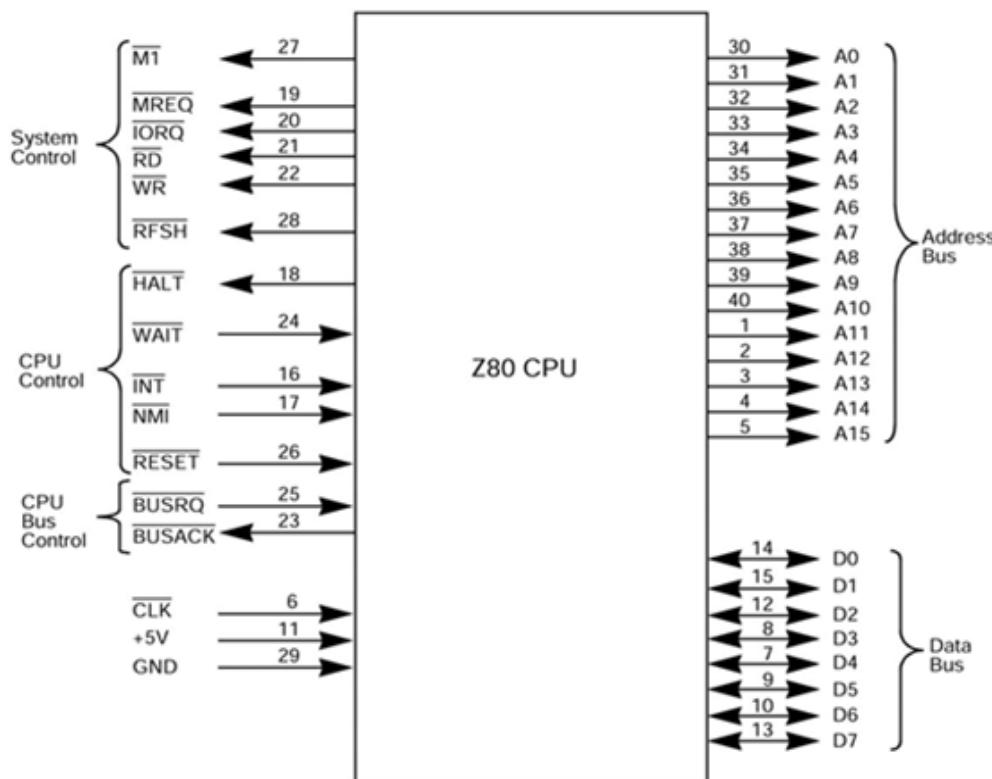
O projeto consiste na construção do chamado hardware mínimo com os seguintes componentes: um Z80 (microprocessador), uma EEPROM AT28C64B (memória EEPROM de programa), duas CMOS W24129A (memória RAM de dados), dois LATCH 74373 (Latch para I/O), sessenta e quatro botões e um LCD 16x2. Além disso, foi utilizado um decodificador 3x8 (de modelo 74138) para acesso aos dispositivos de I/O por memória isolada. Dessa forma, o circuito construído será capaz de realizar algumas operações básicas, a partir de um teclado matricial 8x8, tendo as saídas mostradas no LCD.

O hardware mínimo consiste num circuito em que seja possível as operações básicas de um computador, utilizando a arquitetura mais básica possível: uma CPU, memórias e dispositivos de entrada/saída. Consequentemente, o custo mais baixo permite que esse tipo de hardware seja utilizado para fins didáticos ou de testes.

2. O PROCESSADOR

O microprocessador Zilog Z80 possui 8 bits de dados e 16 bits de endereços (em torno de 64kbytes posições de memória), sendo capaz de transferir até 1 byte de cada vez. Foi amplamente utilizado na década de 80 em sistemas embarcados, centrais telefônicas e em microcomputadores. Um exemplo de aparelho que utilizou esse chip foi o console Sega Mega Drive.

Segue abaixo o desenho esquemático do Zilog Z80.



Os pinos utilizados no projeto foram:

- Controle do Sistema: \overline{IORQ} (20), \overline{RD} (21), \overline{WR} (22);
- Controle da CPU: \overline{INT} (16), \overline{RESET} (26);
- Barramento de Endereços: $A0$ à $A15$ (em ordem: 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 1, 2, 3, 4, 5);
- Barramento de Dados: $D0$ à $D7$ (em ordem: 14, 15, 12, 8, 7, 9, 10, 13);
- Básicos: \overline{CLK} (6), VCC (11), GND (29).

O \overline{IORQ} , uma abreviatura de IO REQUEST, consiste numa saída em nível baixo que indica se o barramento de endereços é válido para leitura ou escrita de dispositivos de entrada e saída. Em outras palavras, a partir da programação (com o comando PORT) é possível diferenciar um mesmo endereço de acesso para memória e I/O.

O \overline{RD} e \overline{WR} ativam, em nível lógico baixo, as operações de leitura e escrita, respectivamente. Semelhante ao anterior, é possível controlar as operações de escrita e leitura a partir de software. Além disso, quando ambos estão em nível lógico alto, é indicado o estado de alta impedância dos dispositivos, ou seja, ocorre uma “desativação temporária” que permite a economia de energia.

O \overline{INT} , também chamado de INTERRUPT, é uma entrada ativa em nível lógico baixo que permite a interrupção do processador, a partir de algum dispositivo, para a execução de uma sub-rotina. Nesse projeto, essa interrupção está atrelada com o teclado 8x8 (dispositivo de I/O) em que se exige a execução da sub-rotina de verificação da tecla digitada.

O \overline{RESET} consiste numa entrada ativa em nível lógico baixo que permite a reinicialização do processador e trazendo-o ao seu estado inicial. Geralmente, é utilizado a partir de um erro de processamento em que o sistema não pode prosseguir. Além disso, esse pino deve ser ligado a uma chave externa que permite a sua conexão com o fio terra por um determinado tempo (para realizar a reinicialização).

O barramento de endereços (A_0 à A_{15}) simboliza uma série de conexões utilizadas na especificação de um endereço utilizado para leitura ou escrita dos dados. Já o barramento de dados (D_0 à D_7) simboliza o máximo de bits (informação) possível de se transferir para cada endereço. Em outras palavras, o Z80 possibilita a leitura ou escrita de 8 bits (ou 1byte) para cada um dos seus 64kbytes de endereço.

O \overline{CLK} consiste num pino básico de entrada de qualquer processador: o controle de operações do sistema. Em outras palavras, esse pino deve ser ligado a um circuito eletrônico, cujo principal componente é um cristal, que gere ondas quadradas (variando em nível alto e baixo, constantemente) em uma frequência entre 2.5MHz e 8MHz para o Z80.

O VCC e GND simbolizam, respectivamente, as conexões à fonte de energia (de +5V) e ao fio terra.

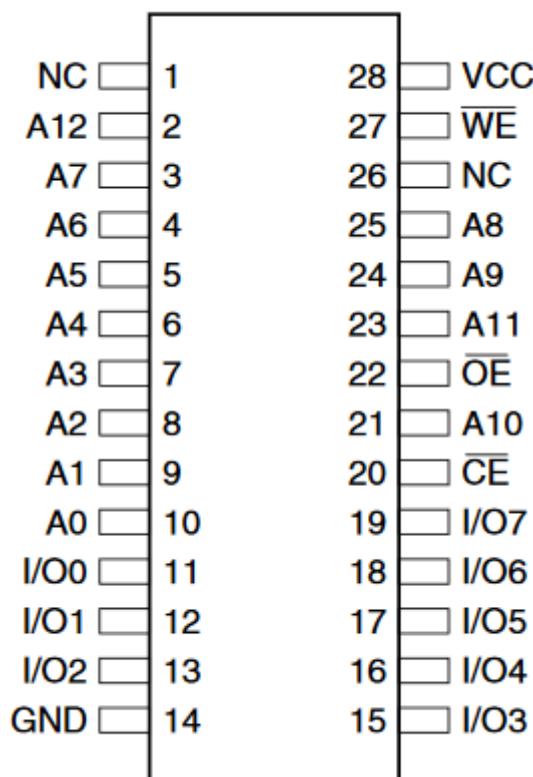
3. AS MEMÓRIAS

O projeto utilizou-se de três memórias, separadas em dois tipos: uma de programas (EEPROM AT28C64B) e duas de dados (CMOS W24129A).

- **EEPROM AT28C64B**

Consiste numa memória básica de **leitura eletricamente apagável e programável** com capacidade máxima de 64K, dividida em 13 linhas de endereço (8K) com 8 bits para troca de dados.

O seu desenho esquemático pode ser visto abaixo:



Os pinos utilizados no projeto foram:

- Barramento de Endereços: A_0 à A_{12} (em ordem: 10, 9, 8, 7, 6, 5, 4, 3, 25, 24, 21, 23, 2);
- Barramento de Dados: I/O_0 à I/O_7 (em ordem: 11, 12, 13, 15, 16, 17, 18, 19);
- Controle da Memória: \overline{CE} , \overline{OE} ;
- Básico: VCC , GND .

O barramento de endereços (A0 à A12) simboliza uma série de conexões utilizadas na especificação de um endereço da memória para a leitura dos dados pelo processador. Já o barramento de dados (I/O0 à I/O7) simboliza o máximo de bits (informação) possível de se ler para cada endereço dessa memória.

O \overline{CE} consiste numa entrada ativa em nível lógico baixo que permite a ativação e uso da memória pelo processador. Em outras palavras: é possível desabilitar temporariamente a memória a partir desta entrada (quando é enviado um sinal em nível lógico alto), possibilitando a economia de energia.

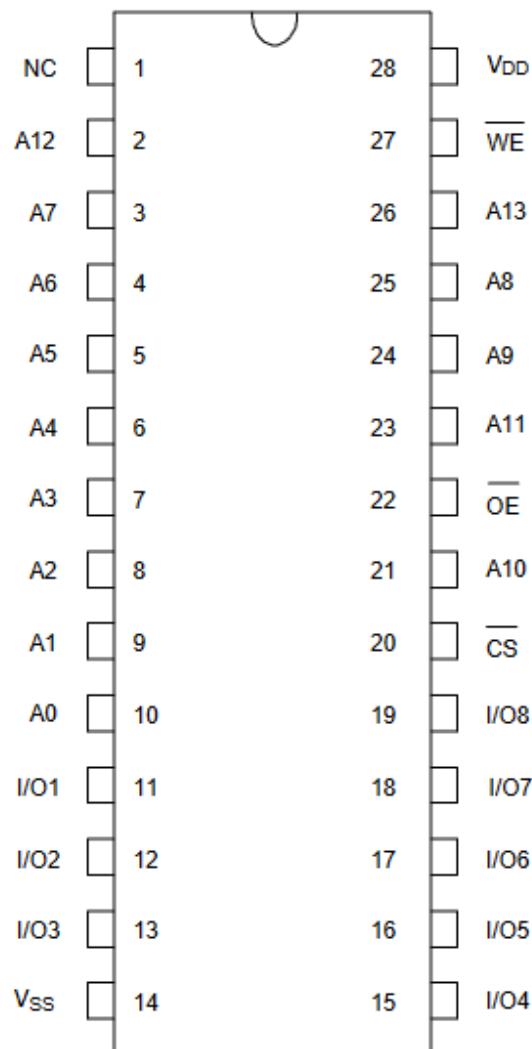
O \overline{OE} consiste numa entrada ativa em nível lógico baixo que permite a ativação para leitura de dados da memória pelo processador. Ou seja, é por meio deste pino em que o processador indica se estará lendo os dados da memória.

O VCC e GNC simbolizam, respectivamente, as conexões à fonte de energia (de +5V) e ao fio terra.

- **CMOS W24129A**

Consiste numa memória de **escrita e leitura e acesso aleatório** (RAM) do tipo CMOS estática, com alta velocidade, baixo consumo de energia e capacidade máxima de 128K, dividida em 14 linhas de endereço (16K) com 8 bits para troca de dados.

O seu desenho esquemático pode ser visto abaixo:



Os pinos utilizados no projeto foram:

- Barramento de Endereços: A_0 à A_{13} (em ordem: 10, 9, 8, 7, 6, 5, 4, 3, 25, 24, 21, 23, 2, 26);
- Barramento de Dados: I/O_1 à I/O_8 (em ordem: 11, 12, 13, 15, 16, 17, 18, 19);
- Controle da Memória: \overline{CS} , \overline{OE} , \overline{WE} ;
- Básico: VDD (ou VCC), VSS (ou GND).

O barramento de endereços (A0 à A13) simboliza uma série de conexões utilizadas na especificação de um endereço da memória para a leitura ou escrita dos dados pelo processador. Já o barramento de dados (I/O1 à I/O8) simboliza o máximo de bits (informação) possível de se ler ou escrever para cada endereço dessa memória.

O \overline{CS} consiste numa entrada ativa em nível lógico baixo que permite a ativação e uso da memória pelo processador. Em outras palavras: é possível desabilitar temporariamente a memória a partir desta entrada (quando é enviado um sinal em nível lógico alto), possibilitando a economia de energia.

O \overline{OE} consiste numa entrada ativa em nível lógico baixo que permite a ativação para leitura de dados da memória pelo processador. Ou seja, é por meio deste pino em que o processador indica se estará lendo os dados da memória.

O \overline{WE} consiste numa entrada ativa em nível lógico baixo que permite a ativação para escrita de dados na memória pelo processador. Ou seja, é por meio deste pino em que o processador indica se estará escrevendo os dados na memória.

O VDD e VSS simbolizam, respectivamente, as conexões da fonte de energia (de +5V) e ao fio terra.

4. O MAPA DE MEMÓRIAS

O mapa de memórias, conceito utilizado para separar os endereços máximos de acesso a cada memória pelo processador, foi dividido da seguinte forma:

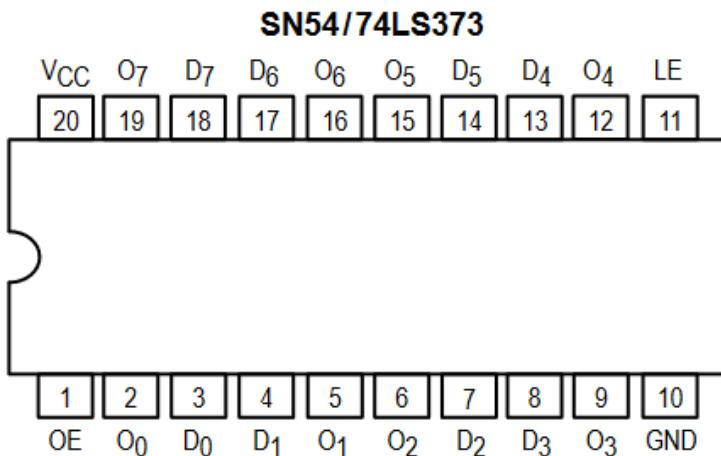
MEMÓRIA DE PROGRAMAS EEPROM AT28C64B (8KB)	0000 0000 0000 0000b = 0000h 0001 1111 1111 1111b = 1FFFh
ESPELHO (24KB)	0010 0000 0000 0000b = 2000h 0111 1111 1111 1111b = 7FFFh
MEMÓRIA DE DADOS 1 CMOS W24129A – 1 (16KB)	1000 0000 0000 0000b = 8000h 1011 1111 1111 1111b = BFFFh
MEMÓRIA DE DADOS 2 CMOS W24129A – 2 (16KB)	1100 0000 0000 0000b = C000h 1111 1111 1111 1111b = FFFFh

5. OS DISPOSITIVOS DE ENTRADA E SAÍDA (I/O)

O projeto utilizou-se de quatro componentes de entrada e saída, separando em dois grupos: o teclado matricial 8x8, utilizando dois Latch's 74373 (um de saída OUT e um de entrada IN) e sessenta e quatro botões; e o LCD 16x2.

- **Teclado Matricial 8x8 com Latch's 74373**

O teclado matricial 8x8 está baseado no conceito de **input pull-up** que, resumidamente, consiste na passagem de corrente integralmente (nível lógico alto) entre as 64 teclas e permite, desta forma, identificar a linha e coluna da tecla quando pressionada (alterando para nível lógico baixo neste momento). Para isso, são utilizados dois Latch's 74373: um de saída (OUT) e um de entrada (IN). O desenho esquemático do Latch 74373 pode ser visto abaixo:



Um fator importante desse componente consiste em: cada entrada (D_n) possui sua saída respectiva (O_n). Além disso, tem-se os pinos LE (11) e OE (1) sendo utilizados como controle para diferenciar um Latch OUT de um Latch IN. Isso pode ser visto na tabela verdade abaixo.

LS373	D_n	LE	OE	O_n
H	H	L	H	
L	H	L	L	
X	L	L	Q_0	
X	X	H	Z*	

H = HIGH Voltage Level

L = LOW Voltage Level

X = Immaterial

Z = High Impedance

O Latch OUT tem as suas entradas (D_n) ligadas ao barramento de dados e as saídas (O_n) passando em cada linha de botões. Além disso, a entrada de controle LE possui uma lógica de combinação de controle para ser ativa somente quando o processador manipula o endereço 0000h (por meio do decodificador, detalhado posteriormente neste tópico). Desta forma, é possível **inserir** dados nesse Latch e, juntamente ao Latch IN, encontrar a linha e coluna respectivas da tecla pressionada.

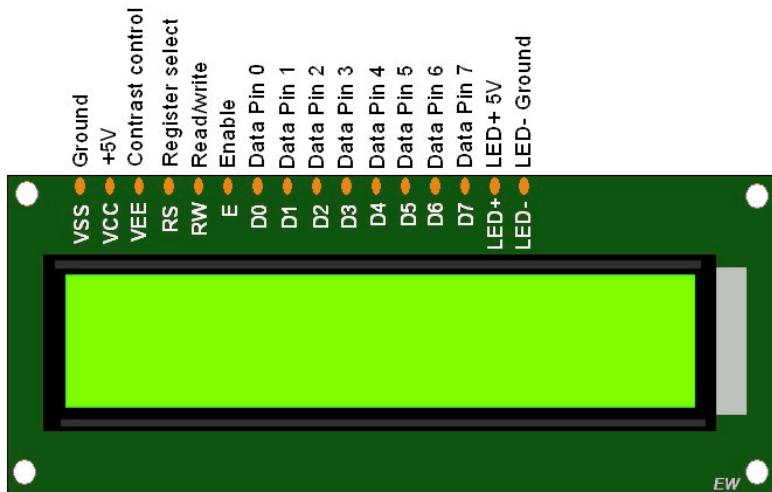
Já o Latch IN tem as suas entradas (D_n) ligadas a cada coluna de botões e as saídas (O_n) ligadas ao barramento de dados. Além disso, a entrada de controle OE possui uma lógica de combinação de controle para ser ativa somente quando o processador manipula o endereço 0001h (por meio do

decodificador, detalhado posteriormente neste tópico). Desta forma, é possível **ler** dados desse Latch e, juntamente ao Latch OUT, encontrar a linha e coluna respectivas da tecla pressionada.

- **LCD 16x2**

Consiste num componente de saída que é utilizado para **exibir** status ou parâmetros em diversos sistemas. Neste projeto, o LCD 16x2 será utilizado em combinação com o teclado para a realização de comandos pré-definidos do LCD e para mostrar alguns caracteres.

O desenho esquemático do componente segue abaixo:

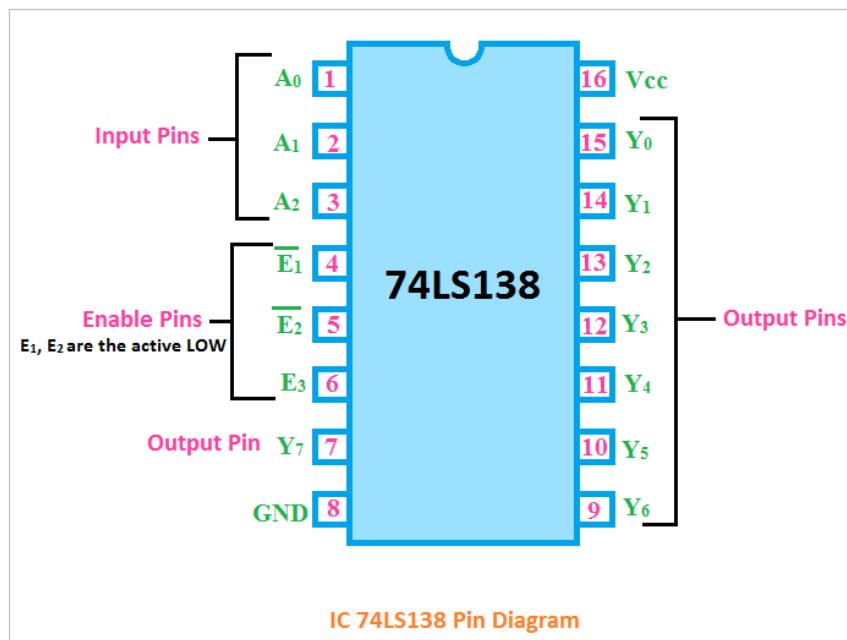


Com relação as funcionalidades de cada pino:

1. VSS (1): conexão ao fio terra;
2. VCC (2): conexão à fonte de energia (+ 5V);
3. VEE (3): controle de contraste, realizado por um potenciômetro no projeto;
4. RS (4): controle do **tipo de escrita**, sendo o nível lógico baixo para escrita de **comandos** e o nível lógico alto para escrita de **dados**.
5. RW (5): controle entre **leitura** e **escrita** do LCD;
6. E (6): ativador do chip, quando está em nível lógico alto;
7. D0/D7 (em ordem: 7, 8, 9, 10, 11, 12, 13, 14): pino de dados, tendo o D7 como uma auxiliar de sinal para quando o dispositivo está ocupado;
8. LED+ (15): conexão à fonte de energia (+ 5V);
9. LED- (16): conexão ao fio terra.

No projeto, houve uma lógica de combinação de controle para a ativação do chip por meio do \overline{IORQ} do processador e das saídas Y2 e Y3 do decodificador (detalhado abaixo e no tópico 6). Além disso, as saídas do decodificador também vão diferenciar o **tipo de escrita**. Por último, o controle de **leitura e escrita** é relacionado diretamente ao \overline{WR} do processador, sendo diferenciado via software.

Para a separação dos endereços de acesso aos I/O (Latch's e LCD), utilizou-se um decodificador 3x8 de modelo 74138, tendo as entradas de seleção A0/A1/A2 (em ordem: 1, 2, 3) ligadas diretamente e respectivamente nos pinos A0/A1/A2 do processador. O seu desenho esquemático pode ser visto logo abaixo:



Além disso, tem-se a seguinte tabela verdade dos pinos de saída (indicados por Y0/Y7):

INPUTS						OUTPUTS							
<u>E₁</u>	<u>E₂</u>	<u>E₃</u>	A ₀	A ₁	A ₂	Y ₀	Y ₁	Y ₂	Y ₃	Y ₄	Y ₅	Y ₆	Y ₇
H	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	L	X	X	X	H	H	H	H	H	H	H	H
L	L	H	L	L	L	L	H	H	H	H	H	H	H
L	L	H	H	L	L	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	L	H	H	H	H	H
L	L	H	H	H	L	H	H	H	L	H	H	H	H
L	L	H	L	L	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	L	H	H
L	L	H	L	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	L

H - High, L - Low, X - Don't Care

IC 74138 Truth Table

Desta forma, ocorreu a seguinte separação de acesso aos dispositivos de I/O com base nos endereços do A2/A1/A0 do processador (também vista no mapa de endereços de I/O do tópico 6): 000b para o Latch OUT (Y0), 001b para o Latch IN (Y1), 002b (Y2) e 003b (Y3) para o controle do **tipo de escrita e ativação** do LCD 16x2.

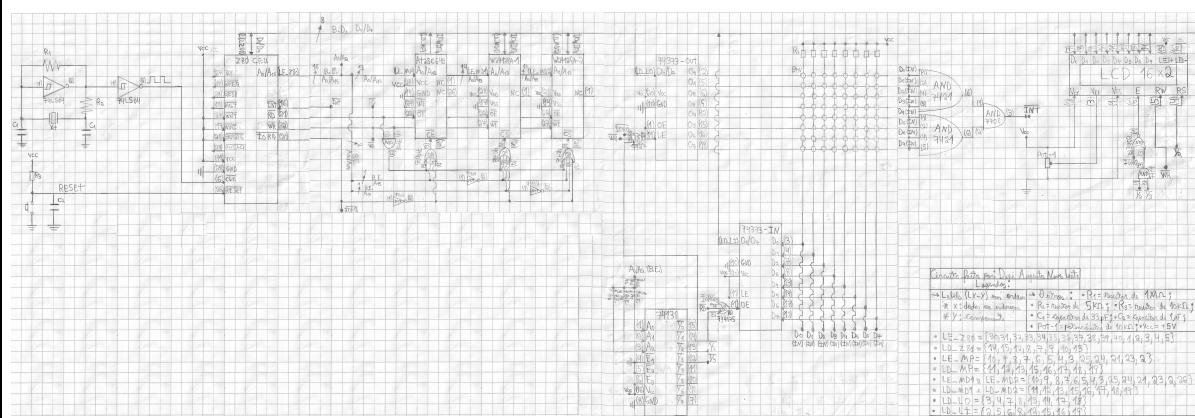
6. O MAPA DE ENDEREÇOS DE ENTRADA E SAÍDA (I/O)

Para a seleção dos dispositivos de I/O, como abordado ao final do tópico anterior, foi utilizado as saídas Y0/Y3 do decodificador 74138, de tal forma que se segue o mapa de endereços de entrada e saída abaixo.

Componentes de I/O	Componentes Auxiliares	Relação entre: A2-A1-A0 do Z80 e a saída ativa do Decodificador
Teclado Matricial 8x8	Latch OUT	000b = Y0
	Latch IN	001b = Y1
LCD 16x2	-	002b = Y2 003b = Y3

Desta forma, confere-se ao acesso dos I/O por memória isolada, ou seja, é possível utilizar um mesmo endereço do processador para a memória e para o I/O.

7. O ESQUEMA DO HARDWARE DO COMPUTADOR



8. A ROTINA DE CONTROLE DO TECLADO

A rotina consiste em se obter o valor da linha e coluna da respectiva tecla pressionada pelo usuário, por meio de inserções de dados no Latch OUT e verificando as mudanças pelo Latch IN.

INICIO_DO_ALGORITMO

/*Essa sub-rotina será executada quando o \overline{INT} do Z80 for ativado (estiver em nível lógico baixo)*/

INTERRUPCAO

SUBROTINA LINHA_COLUNA_TECLADO

//Iniciando o Latch OUT em nível baixo (Latch IN estará em nível alto)

POR[T][0000] = 00h;

//Recuperando o valor do Latch de Entrada

VALOR_IN = PORT[0001];

//Definindo os contadores de linha e coluna

LINHA = 0;

COLUNA = 0;

//Auxiliares para comparação

AUX_COMPARA = 01h;

//Resultado das Comparações

RESUL_COLUNA, RESUL_LINHA;

//COLUNA

//Realizando a comparação OR para obter a coluna

RESUL_COLUNA = VALOR_IN OR AUX_COMPARA;

//Repetição para obter a coluna

ENQUANTO (RESUL_COLUNA != 255)

COLUNA = COLUNA + 1;

AUX_COMPARA = AUX_COMPARA SHL 1;

RESUL_COLUNA = VALOR_IN OR AUX_COMPARA;

```

FIMENQUANTO

//Após obter a coluna, é necessário a manipulação para obter a linha

//Recolocar o valor 01h na AUX_COMPARA

AUX_COMPARA = 01h;

/*Como as teclas são ativadas internamente por "chaves" (que se interligam no circuito
quando pressionada), basta analisar qual a mudança ocorrida no Latch IN com
"inserções" de alto nível no Latch OUT*/

```



```

//LINHA

//Inserindo em sucessivas posições do Latch OUT

PORT[0000] = AUX_COMPARA;

RESUL_LINHA = PORT[0001];

ENQUANTO (RESUL_LINHA != 255)

    LINHA = LINHA + 1;

    AUX_COMPARA = AUX_COMPARA SHL 1;

    PORT[0000] = AUX_COMPARA;

    RESUL_LINHA = PORT[0001];

FIMENQUANTO

FIM_SUBROTINA

//Com isso, LINHA e COLUNA marcaram a respectiva tecla pressionada

FIM_DO_ALGORITMO

```

9. AS ROTINAS DE CONTROLE DO LCD

As rotinas consistem em: realizar a função pré-definida de limpar o display do LCD e na visualização de uma mensagem também pré-definida. Sendo assim, juntando com a rotina do teclado, as seguintes sub-rotinas serão realizadas: limpar o display ao apertar o botão 1x1 e escrever uma mensagem pré-definida ao apertar o botão 1x2.

INICIO_DO_ALGORITMO

//Declaração de variáveis dos endereços de controle do LCD

Y2 = 02h; //Escrita de Comandos

Y3 = 03h; //Escrita de Dados

//SUB-ROTINA DE LIMPAR O LCD: realizada com 0001h em dados (pré-definido)

SUBROTINA LIMPA_LCD

//Verifica se o botão apertado está na primeira linha e primeira coluna

SE (LINHA == 0 AND COLUNA == 0)

/*Ativando o dispositivo para escrita de comandos e inserindo os dados para limpar o LCD*/

PORT[Y2] = 01h;

FIMSE

FIMSUBROTINA

//SUB-ROTINA DE ESCREVER "PEREA": em tabela ASCII

SUBROTINA ESCREVE_PEREA

//Verifica se o botão apertado está na primeira linha e segunda coluna

SE (LINHA == 0 AND COLUNA == 1)

/*Ativando o dispositivo para escrita de dados e inserindo a letra 'P'*/

PORT[Y3] = 'P'; //ASCII = 50h

/*Ativando o dispositivo para escrita de dados e inserindo a letra 'E'*/

PORT[Y3] = 'E'; //ASCII = 45h

/*Ativando o dispositivo para escrita de dados e inserindo a letra 'R'*/

PORT[Y3] = 'R'; //ASCII = 52h

/*Ativando o dispositivo para escrita de dados e inserindo a letra 'E'*/

PORT[Y3] = 'E'; //ASCII = 45h

/*Ativando o dispositivo para escrita de dados e inserindo a letra 'A'*/

PORT[Y3] = 'A'; //ASCII = 41h

FIMSE

FIMSUBROTINA

FIM_DO_ALGORITMO