Laboratório sobre o Modelo de Computador de Programa Armazenado – Arquitetura Cleópatra: Bloco de Dados, Bloco de Controle

Prática: Finalização e Simulação do Processador Cleópatra

Recursos: Projeto Parcial da Arquitetura Cleópatra (esquemático) e CAD Foundation

Parte I – Introdução e Objetivos

O **Laboratório sobre o Modelo de Computador de Programa Armazenado e Programação em Linguagem de Montagem** envolveu a prática com a linguagem de montagem (em inglês, *assembly language*) do processador Cleópatra, mediante emprego do sistema de desenvolvimento de software do mesmo. Neste Laboratório, a ênfase muda para a implementação em esquemáticos do processador Cleópatra, passando de preocupações arquiteturais para problemas organizacionais.

Neste Laboratório, partir-se-á de uma versão parcialmente implementada da Cleópatra. O Bloco de Dados (BD) e o Bloco de Controle (BC) estão quase prontos. O trabalho prático incluirá completar o projeto do Processador e simulá-lo corretamente, para um pequeno programa em linguagem de montagem. As partes a completar são os decodificadores de escrita e leitura dos registradores internos e a ROM de tradução de conteúdos do IR para microendereços. Após terminado o diagrama de esquemáticos da Cleópatra, deve-se realizar a simulação do mesmo, usando um script de simulação a ser implementado pelo grupo de alunos. Um exemplo de script (arquivo Completa.cmd) e de suas formas de onda de saída (arquivo Completa.des) estão disponíveis no projeto.

O objetivo desta aula é compreender, concluir o projeto do processador Cleópatra, e simular o projeto completo. A simulação total de um trecho de programa dado deve ser realizada.

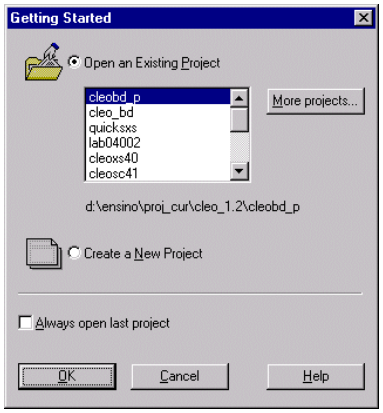
Obs: Use o documento de especificação da Cleópatra disponível a partir homepage da disciplina, o_cleo2.01.doc

Parte II – Bloco de dados - Leitura e estudo (atividade prévia ao laboratório)

Tarefa 1: Ler atentamente os capítulos 2 e 3 do texto de referência da arquitetura Cleópatra. Quatro tópicos devem ser bem compreendidos pelo aluno:

- Interface entre o processador e o mundo externo (no nosso caso, com a memória), e os sinais de reset e de relógio (clock).
- Interface entre o bloco de controle (BC) e bloco de dados (BD).
- Estrutura interna do BD.
- Estrutura interna do BC.

Parte III – Bloco de Dados - projeto parcial



Tarefa 2: Executar passos abaixo.

1. Buscar o arquivo Cleobd_p.zip da homepage da disciplina. (não descompacte este arquivo manualmente, use a opção de menu File → Restore Project do Gerenciador do Foundation)
2. Abra o Foundation. Deverá aparecer a seguinte tela:

Selecione a opção Cancel. Se for aberto o projeto anterior, ignore-o.

3. No menu "File", escolha a opção "*Restore Project*". Escolha o arquivo *Cleobd_p.zip* no seu diretório de trabalho. Esta opção irá descompactar o arquivo "*Cleobd_p.zip*". Cuidar para indicar no "*wizard*" o local correto onde instalar o projeto, evitando trabalhar no disco H: (desempenho!).

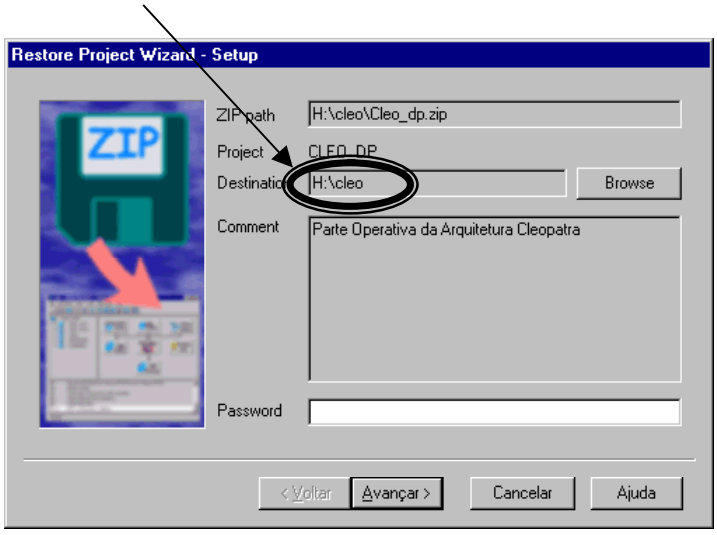


Figura 1 - Janela de descompactação de projeto no sistema Foundation.

4. Caso o projeto não seja aberto após recuperação, vá ao menu File → Open Project. Abrir o projeto cleobd_p.

Parte IV - Implementação dos codificadores de escrita/leitura

Tarefa 3: Executar passos abaixo.

1. Abra o esquemático, entre no Bloco de Dados e observe as entradas/saídas (folha de esquemático *cleo_top*), que deve aparecer como na janela abaixo.

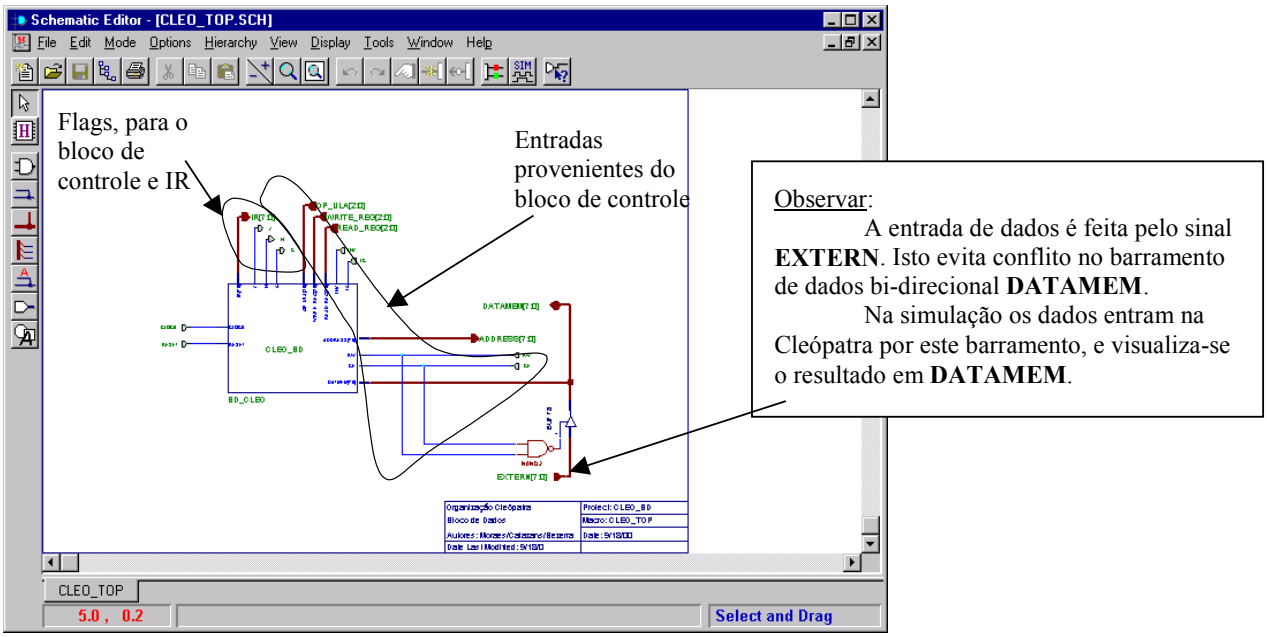


Figura 2 – Esquemático de nível mais alto do Bloco de Dados Cleópatra.

2. Percorra a hierarquia, verificando a estrutura do Bloco de Dados.

Observar, que além dos registradores, há três importantes blocos: ULA, DECW e DECR. O bloco ULA está implementado. Os blocos DECW e DECR devem ser construídos conforme a especificação da arquitetura, veja um resumo nas tabelas abaixo. Os códigos para leitura e escrita dos registradores devem ser gerados pelo bloco de controle, cada um como um vetor de três bits, denominados READ_REG[2:0] e WRITE_REG[2:0], respectivamente. A explicação do comportamento associado a estas tabelas verdade encontra-se no texto o_cleo2.01.doc.

Tabela 1 - Especificação do bloco DECW.

WRITE_REG	LD_MAR	LD_MDR	LD_IR	LD_PC	LD_AC	LD_RS
000	1	0	0	0	0	0
001	0	1	0	0	0	0
010	0	0	1	0	0	0
011	0	0	0	1	0	0
100	0	0	0	0	1	0
101	0	0	0	0	0	1
110	0	1	0	1	0	0
111	0	0	0	0	0	0

Tabela 2 - Especificação do bloco DECR. (Obs.: Colocar um inversor na saída de cada sinal, pois os tristates tem ativação negada).

READ_REG	RMDR	RIR	RPC	RAC	RRS
000	0	0	0	0	0
001	1	0	0	0	0
010	0	1	0	0	0
011	0	0	1	0	0
100	0	0	0	1	0
101	0	0	0	0	1
110	1	0	0	1	0
111	1	0	1	0	0

3. Implementar agora estes dois blocos.
4. Finalmente, note que o biestável que armazena o qualificador de Overflow não aparece, a ULA não gera esta informação. Tarefa 4: Altere o Bloco de Dados para inserir estas estruturas. Considere que a carga do biestável de Overflow é comandada pelo sinal LCV (antes LC, mude).

Parte V – Microssimulação do bloco de dados

Tarefa 5: Executar passos abaixo.

1. Após completar o esquemático, chame o simulador, e no menu "*file → load waveform*" escolha "cleo_bd.tve". Trata-se do resultado de uma simulação realizada corretamente. Deverá ser exibida a janela da Figura 3.

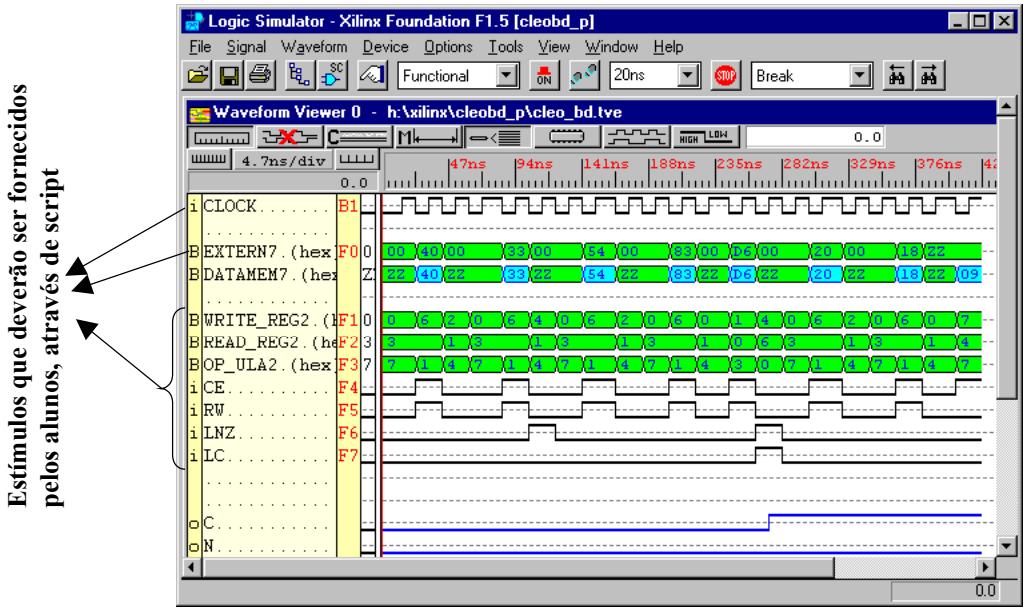


Figura 3 –Uma simulação correta do Bloco de Dados Cleópatra.

Tarefa 6: **Estude, Pense e Responda**: qual o trecho de código objeto cuja simulação é mostrada na Figura acima? Quantas instruções este trecho possui e quais são elas? Não se esqueça de mencionar o modo de endereçamento de cada instrução, quando isto for aplicável.

- Na janela de simulação estão indicados os estímulos que devem ser fornecidos à simulação pelo aluno, que são: clock, reset (ausente na Figura), conteúdo da memória (use para isto o vetor EXTERN) e os comandos (WRITE_REG, READ_REG, OP_ULA, CE, RW, LNZ, LCV).

NÃO HÁ SCRIPT EXEMPLO. O MESMO DEVE SER DESENVOLVIDO PELO GRUPO COMO PARTE DESTE LABORATÓRIO. NÃO ESQUECER QUE OS TEMPOS NO SCRIPT SÃO CUMULATIVOS, E NÃO ABSOLUTOS!

- Devem ser visualizados os seguintes sinais: saídas de todos os registradores internos (não mostrado na Figura, use o comando *watch* no seu próprio script), estado dos qualificadores e estado da memória (DATAMEM).

2. Cada grupo deve realizar a simulação completa de um trecho de programa com 3 instruções.

Lista de grupos de instrução a serem trabalhadas (1 por grupo de alunos):

Grupo 1	Grupo 2	Grupo 3	Grupo 4	Grupo 5	Grupo 6	Grupo 7	Grupo 8	Grupo 9	Grupo 10
LDA 0F1h	LDA 12h,I	LDA 0A5h,I	LDA 0C5h	LDA 56h,I	LDA 77h	LDA 48h,I	LDA 0F9h	LDA 0D0h,I	LDA 0CCh
ADD 0E0h,I	AND 25h	STA 91h	AND 56h,I	ADD 69h	STA 91h,I	OR 34h	ADD 0F8h	STA 0F8h	ADD 18h,I
JC 0FAh,I	JN 0FAh,R	JZ 0FAh	JN 0FAh,R	JV 0FAh,I	JN 0FAh	JV 0FAh,I	JV 0FAh,R	JV 0FAh	JC 0FAh,R

Obs.1: Para os modos direto e indireto o aluno deverá escolher os conteúdos de memória.

Obs.2: Escolher os dados de forma a forçar que sempre ocorra o desvio na última instrução.

Exemplo: supondo que o seu seja o "Grupo 1", vocês teriam o seguinte mapa de memória:

Endereço	0	1	2	3	ABh	E0h	F1h
Conteúdo	LDA	F1h	ADD	E0h	12h	ABh	34h

Ou seja, vocês atribuiriam o conteúdo ABh à posição 0E0h, o conteúdo 34h à posição 0F1h e o conteúdo 12h à posição 0ABh. No final da micro-simulação, vocês deverão ter no acumulador a soma 34h+12h, ou seja, 46h.

Obs.: É necessário saber exatamente o número de ciclos que cada instrução leva para ser executada, bem como todos os sinais de controle gerados pelo BC a cada ciclo de relógio. A documentação completa para tanto é a descrição da µROM da Cleópatra (disponível na área de download da disciplina).

Parte VI - Preparando o Trabalho no Bloco de Controle

Siga os mesmos passos da **Parte III** para recuperar o projeto semi-pronto do processador completo, a partir do arquivo "**Cleo_p.zip**" na homepage da disciplina. Após copiar e restaurar o arquivo, abra o projeto e o editor de esquemáticos. O Bloco de Dados deste projeto está completo, mas o Bloco de Controle não. Entrando três níveis na hierarquia, chega-se até o interior do BC. A Figura 4 à esquerda mostra como deve aparecer a tela neste nível, que apresenta as duas partes principais do BC: a MicroROM e o Seqüenciador. A MicroROM está completa, e falta apenas descrever uma das partes do Seqüenciador, qual seja, o A tradução de IR e qualificadores para Microendereços. Entrando dentro do Seqüenciador, é possível visualizar esta parte do projeto a nível de interface. A Figura 4 à direita mostra como deve aparecer a tela do Editor de Esquemáticos, após descer na hierarquia como indicado. O retângulo abaixo e ao centro do esquemático da Figura mencionada é o que deve ser implementado nesta aula.

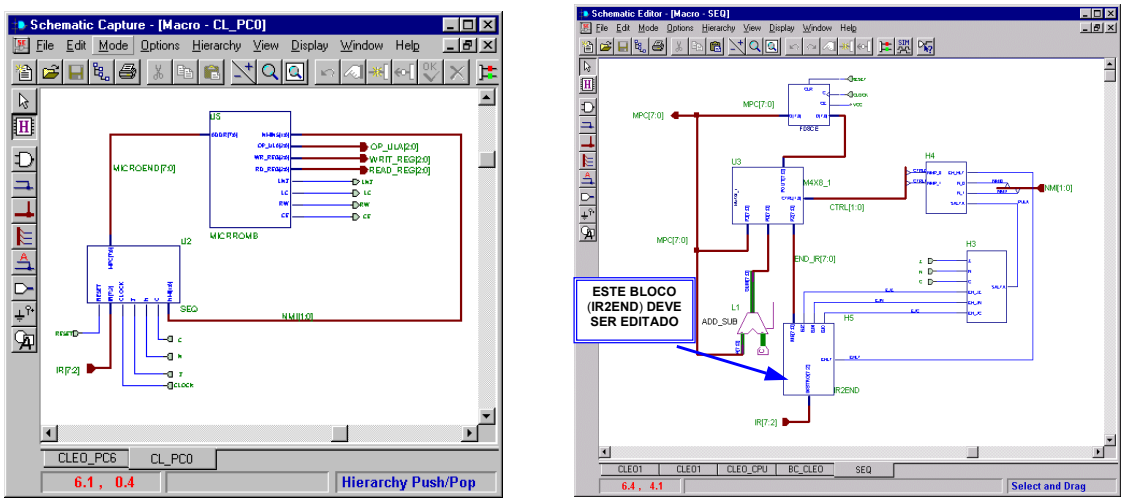


Figura 4 – Esquemáticos intermediários do Bloco de Controle Cleópatra.

Parte VII - Implementação do Bloco IR2END

O bloco IR2END teve apenas sua interface e lógica especial definidas no projeto parcial. Para implementar seus conteúdos, será necessário implementar uma ROM. Para tanto, existe um utilitário do sistema de CAD Foundation que automatiza a criação de blocos regulares complexos, e que deverá ser empregado. Trata-se da ferramenta denominada **LogiBlox Module Generator**. Os conteúdos da citada ROM devem ser inferidos a partir da Tabela da última página da especificação Cleópatra (Versão 2.0 ou mais atual).

Para criar a ROM em questão, inicie pela escolha, no editor de esquemáticos, do menu: *Tools ➔ LogiBLOX Module Generator*. Esta escolha abre uma janela de diálogo para criar diversos blocos de forma automática. Clique no campo *Module Types* e escolha *Memories*. A partir daí a janela de diálogo deve apresentar-se como na Figura 5.

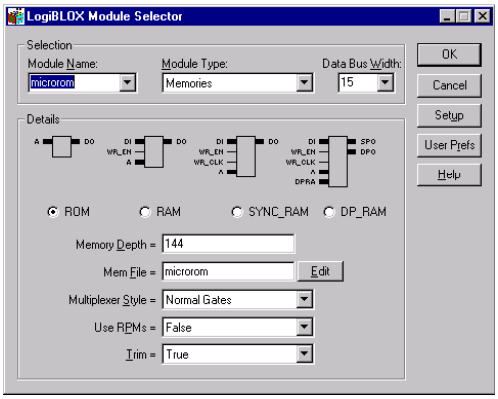


Figura 5 – Janela de parametrização de módulos da ferramenta LogiBlox Module Generator.

Para prosseguir, **configure** os campos desta janela (os números da janela estão errados), (1)especificando um nome para a ROM, (2)garantindo que a opção ROM é a escolhida, (3)especificando o número de saídas (campo Data Bus Width), (4)o número de linhas da ROM (campo Memory Depth) e (5)um nome do arquivo que conterá o texto de especificação da ROM (pode ser o mesmo nome do módulo). No Apêndice deste laboratório há detalhes de como preencher a ROM. Caso haja dúvidas sobre como preencher os campos, clique em *Help*, clique em *Index*, e digite ROM para chegar na opção de ajuda para módulos do tipo ROM. Após configuradas todas as opções, e não antes, deve-se editar o arquivo, clicando-se na opção *Edit*. As instruções de como preencher o arquivo com os dados da ROM encontram-se nas mesmas páginas de ajuda mencionadas antes. Terminando a edição do arquivo, basta sair do editor e clicar em *Ok* para que a geração da ROM tenha lugar. Caso haja erros, o arquivo foi mal preenchido. Deve-se então corrigir os erros e re-executar o processo de geração.

Erros comuns:

1) O número de linhas da ROM deve ser múltiplo de 16;
2) O número de dados especificados no arquivo deve ser exatamente igual ao número de linhas da ROM. Se a geração da ROM tiver sucesso, uma nova célula existe na biblioteca do projeto, com o nome do módulo escolhido. Agora basta inserir este módulo e conectá-lo aos pinos de entrada e saída do bloco IR2END, não esquecendo verificar se tudo está certo com as portas lógicas detectoras de HLT/JN/JZ/JC/JV.

Parte VIII – Simulação

Tarefa 7: Executar passos abaixo.

1. Simule a ROM recém criada, antes de mais nada. O procedimento é muito simples: coloque uma seqüência de valores na entrada da ROM (digamos todos os valores de 0/00H a 63/3FH) e observe a saída da ROM implementada, verificando se seu comportamento confere com a especificação no documento Cleópatra.
2. Simule o bloco IR2END e simule o Seqüenciador;
3. Finalmente, realize a execução de um programa pela Cleópatra como um todo. Escolha um programa com pelo menos seis (6) instruções em linguagem de montagem, simule-o no ambiente de desenvolvimento de software Cleópatra e simule exatamente o mesmo comportamento em hardware, no simulador do Foundation. A qualidade e abrangência do programa escolhido será importante na avaliação (máximo de instruções distintas, máximo de modos de endereçamento usado, etc.).

ATENÇÃO1: Note que existe uma defasagem de meio período de relógio entre ações no BD e no BC, para que os comandos do BC e os qualificadores do BD estabilizem após uma mudança, antes de serem usados pelo outro bloco (ver inversor entre o sinal de clock de entrada e o BD).

ATENÇÃO2: Existem alguns problemas na simulação, devido ao pessimismo do modelo usado pelo simulador lógico do Foundation para barramentos tristate. Caso sua simulação acuse erro de conflito nos barramentos BUSA e BUSB, escolha ignorar o erro até o fim da simulação. Após, analise se existe algum problema na simulação (linhas em X ou Z). Se estas não existirem, não há problemas com a simulação.

Parte IX - A fazer e a entregar

A fazer: Executar todas as tarefas propostas acima.

A Entregar (**Todo o material deve ser entregue em meio magnético**):

- Projeto completo, arquivado via Foundation (opção de menu File → Archive Project... do Gerenciador de Projeto), com o scripts de simulação e uma execução para cada um destes (após gerar a forma de onda, use a opção de menu File → Save Simulation State...).
- O relatório do Laboratório;
- resposta ao(s) item(ns) com menção **Estude, Pense e Responda** deste documento (incluindo o Apêndice).

Percentuais de nota atribuídos às Tarefas:

Item Avaliado	Percentual
Projeto Completo	**30%**
Simulação Correta do Trecho de Programa no BD	**20%**
Simulação Correta do Trecho de Programa no processador completo	**30%**
Respostas às questões	**20%**

Apêndice - O que deve ser colocado na ROM IR2END

- O endereço da ROM é formado pelos 6 bits mais significativos do IR.
  Supor:

STA # ,D: 24H ou 0010 0100
O endereço na ROM serão os 6 bits mais significativos, ou 001001 = 9 (decimal) ou 09H (hexadecimal)

- Na posição 9 da ROM deverá ser escrito o valor 04. Atenção: a posição 9 corresponde à linha 10, pois o endereço começa em 0.

Instrução	Valor em Hexa da Instrução	Endereço correspondente p/ a ROM (decimal)	Conteúdo da ROM (µendereço de Início)
NOT	00	0	03H
STA # ,D	24	9	04H
STA ,I	28	10	08H
STA ,R	2C	11	0EH
LDA #	40	16	12H
LDA ,D	44	17	15H
LDA ,I	48	18	1AH
LDA ,R	4C	19	21H

Continuar ...

- No arquivo texto deverá ser inserido o conteúdo da ROM após o campo DATA:

```
; Data Section
; Specifies data to be stored in different addresses
; e.g., DATA 0:A, 1:0
RADIX 16
DATA
03              ; posição 0
03
03
03
03
03
03
03
04              ; endereço correspondente ao STA # ,D
08              ; endereço correspondente ao STA ,I
……
```

Conteúdo que deve ser preenchido a mão pelo aluno.

Tarefa 8: **Estude, Pense e Responda:** Porque nas 8 primeiras posições da ROM se armazena o mesmo valor (03H)? Isto ocorre em outras partes desta ROM? Onde?