SIMULAÇÃO E REALIZAÇÃO DE UM COMPUTADOR COM BASE NUM PROCESSADOR MONOLÍTICO

Christian Lenz Cesar

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIA (M.Sc.).

Aprovada por:

Presidente

RIO DE JANEIRO

ESTADO DA GUANABARA - BRASIL

OUTUBRO DE 1973

A G R A D E C I M E N T O S

Meus sinceros agradecimentos ao coordenador do Programa de Engenharia de Sistemas e Computação, professor Nelson Maculan, pelo apoio e ao professor Celso de Renna e Souza por ter tornado realidade a parte prática deste trabalho.

R E S U M O

O presente trabalho descreve a arquitetura de um pequeno computador realizado com base num processador monolítico e sua simulação.

Primeiramente estabelece-se os circuitos externos ao processador necessários ao seu funcionamento.

Em seguida mostra-se um simulador que executa instruções dadas em sua forma binária. O usuário pode definir dentro do simulador a organização da memória, dos periféricos e do sistema de interrupção.

Finalmente descreve-se a implementação do computador.

A B S T R A C T

This paper describes an architecture for a small computer using a microprocessor as its central processing unit.

In the first part of the work it is established the necessary external logic circuits for the microprocessor.

The second part presents a simulator that executes instructions given in binary form. The user may define the memory, peripherals and interrupt system organization inside the simulator.

Finally it is described the implementation of the computer.

ÍNDICE

Capítulos: Páginas:

A P R E S E N T A Ç Ã O

A integração de funções lógicas no início da década de 60 libertou o projetista de circuitos do trabalho de montagem dessas funções com componentes discretos. Desde então a eletrônica digital teve um desenvolvimento espantoso com a multiplicação das tecnologias de fabricação dos circuitos, cada uma trazendo novos progressos em termos de velocidade e de densidade de integração.

A integração em larga escala (LSI-Large Scale Integration), apanágio da tecnologia MOS (Metal Oxide Semiconductor), permitiu a colocação num só circuito integrado de funções cada vez mais complexas, como por exemplo, memórias a semicondutores de grande capacidade.

Foi a partir dessa tecnologia que surgiu o microprocessador, que consiste num conjunto de circuitos LSI, cada um contendo uma parte importante de um computador digital, por exemplo tôda a unidade aritmética e lógica ou mesmo a unidade central de processamento (CPU - Central Processing Unit).

O impacto desses microprocessadores se verifica especialmente em áreas antes dominadas pelo minicomputador e pelo sistema digital feito "sob medida" (special-purpose), tais como controle de processo, controle de máquina, periféricos

de computador, máquinas de calcular, instrumentação, terminais inteligentes e em comunicações, por serem de baixo custo e tamanho reduzido. Os sistemas de processamento montados com êsses novos componentes foram chamados de microcomputadores.

Lançado em 1972 o INTEL 8008 foi um dos primeiros microprocessadores vendidos no mercado. Constituido por apenas um circuito integrado de 18 pinos, é uma CPU que trabalha em paralelo sôbre palavras de oito bits, possui um repertório de 48 instruções e endereça até 16K palavras de memória.

Êste trabalho resulta do estudo aprofundado da 8008 e do microcomputador SIM8-01 da mesma companhia e tem por objetivos:

- dar ao projetista de circuitos lógicos regras e modelos para a implementação de um sistema digital baseado na 8008
- criar um suporte de programação para o projetista e usuário da 8008 a fim de facilitar a implementação e correção dos programas a serem colocados em ROM (Read Only Memory) e outras memórias

O trabalho está dividido em três capítulos. Inicialmente propõe-se uma arquitetura para um microcomputador que usa a 8008 indicando os sinais de controle necessários ao seu funcionamento bem como a maneira de gerá-los por circuito. Al-

gumas regras são estabelecidas para o projeto da memória, das entradas e saidas, e do sistema de interrupção.

No segundo capítulo descreve-se um programa em PL/1 que simula a 8008 permitindo ao usuário a definição da organização da memória, das entradas e saidas, e das interrupções, que compoem o microcomputador.

O terceiro capítulo trata da implementação real de um microcomputador usando o modelo da primeiro capítulo.

É importante acentuar que a compreensão destes capítulos depende da leitura prévia do manual da 8008, referencia básica de toda a tese. Não se pretendeu que êste trabalho fosse outro resumo da 8008. Apenas os aspectos menos claros do manual é que serão tratados nas discussões.

C A P Í T U L O I

O MICROCOMPUTADOR

I.1 - INTRODUÇÃO

A arquitetura a ser descrita para o microcomputador, resultou da experiência adquirida nos últimos anos em nossos laboratórios de circuitos digitais e da presente (1972) situação do mercado brasileiro de eletrônica.

A quase totalidade dos circuitos digitais encontrados nos revendedores das grandes cidades são funções lógicas simples. Raramente se encontram circuitos LSI. Por essas razões é que o chaveamento de informações que convergem para um mesmo ponto é tradicionalmente feito utilizando-se circuitos com saida em coletor aberto, ao contrário dos americanos que podem se dar o luxo de usar multiplexadores ou lógica a três estados.

A arquitetura repousa portanto sôbre uma técnica já bem dominada que é a de barras de coletor aberto. Esta escolha resultou em que apenas a 8008, a memória e os inversores de baixa potencia, necessários ao "interface" MOS-TTL, fossem buscados fora do Brasil.

A descrição que se segue trata os circuitos do ponto de

vista estritamente lógico. Somente nas conclusões deste capítulo é que serão discutidos os problemas práticos que poderão ocorrer na implementação.

A fim de evitar confusão usar-se-á a seguinte convenção:

microprocessador - a 8008

processador - a 8008 e o conjunto de circuitos necessários ao seu funcionamento

microcomputador - o processador com a memória, entradas e saidas, e o sistema de interrupção

I.2 - PROCESSADOR

A organização de um processador que usa a 8008 é a da figura 1.1. A barra de dados admite fluxo de informação em ambos os sentidos. As informações que saem da 8008 são colocadas na barra de memória, ou barra M, que é a continuação lógica da barra de dados. Ao contrário, a barra de entrada, ou barra E, por onde chegam as informações, está isolada por meio de porta lógica que será amostrada em T3A, quando a 8008 estiver pronta para receber o octeto de bits. Os registros RL e RH armazenam durante os ciclos PCI,PCR e PCW,as partes baixa e alta do endereço, respectivamente. Para o ciclo de entrada e saida,(PCC), RL alimenta a barra de saida, ou barra S, e RH contem o número do periférico.

Os sinais de controle da 8008 - "status","sync",CC1 e CC0 (os dois últimos obtido através de RH) - e a fase Ø2 do relógio mestre são utilizados para a geração de todos os sinais necessários ao sistema (controle de escrita na memória, sincronização da interrupção, estados, ciclos, controle de multiplexagem na barra de entrada, etc.).

As figuras 1.2 e 1.3 mostram os circuitos de geração desses sinais e a figura 1.4 um diagrama de tempo que indica o relacionamento entre os sinais mais importantes:

Ø22 - a quarta fase do relógio. É o sinal mais impor-

tante, sendo usado para a obtenção de quase todos os outros.

STROBE LO - carrega a palavra presente na barra M em RL

STROBE HI - carrega a palavra presente na barra M em RH

PCI,PCR,PCW,PCC - os ciclos do processador, obtidos pela decodificação de CC0 e CC1.

T1,T2,T3,T4,T5,T1I,WAIT,STOP - os estados do processador, obtidos pela decodificação de S0, S1 e S2.

R/W - comando de escrita em memória do dado presente na barra M. Só ocorre durante o ciclo PCW.

T3A - amostra o dado presente na barra E quando a 8008 está recebendo informações.

Esses sinais serão utilizados na obtenção dos sinais de multiplexagem das três entidades básicas da máquina ligadas à barra E (memória, periféricos de entrada e sistema de interrupção) e dos sinais de chamada a periféricos de saida ligados à barra S. Serão chamados de LIB (libere) no caso da barra de entrada e CARR (carregue) no caso da barra de saida. Os LIB são tais que apenas um poderá ocorrer (nível 1 lógico) num dado instante. Os CARR não possuem esta restrição, porém na prática, serão também mutuamente exclusivos. Isto é devido ao fato que a barra E é do tipo coletor aberto (as barras S e M não são do tipo coletor aberto).

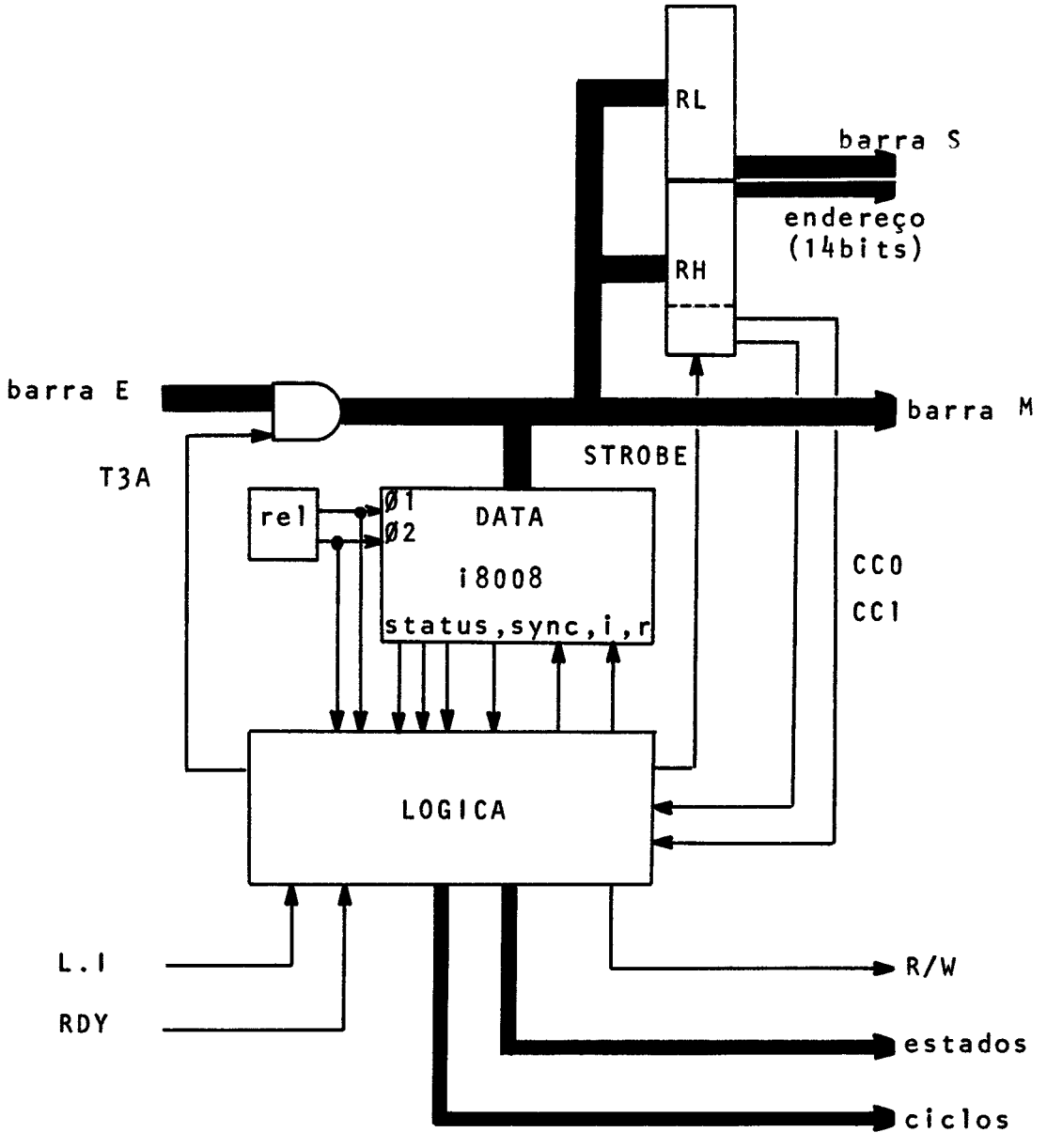
RL
barra S
endereço
(14bits)
RH
barra E
barra M
STROBE
T3A
rel
Ø1
Ø2
DATA
i8008
status,sync,i,r
CC0
CC1
LOGICA
L.I
RDY
R/W
estados
ciclos

fig. 1.1

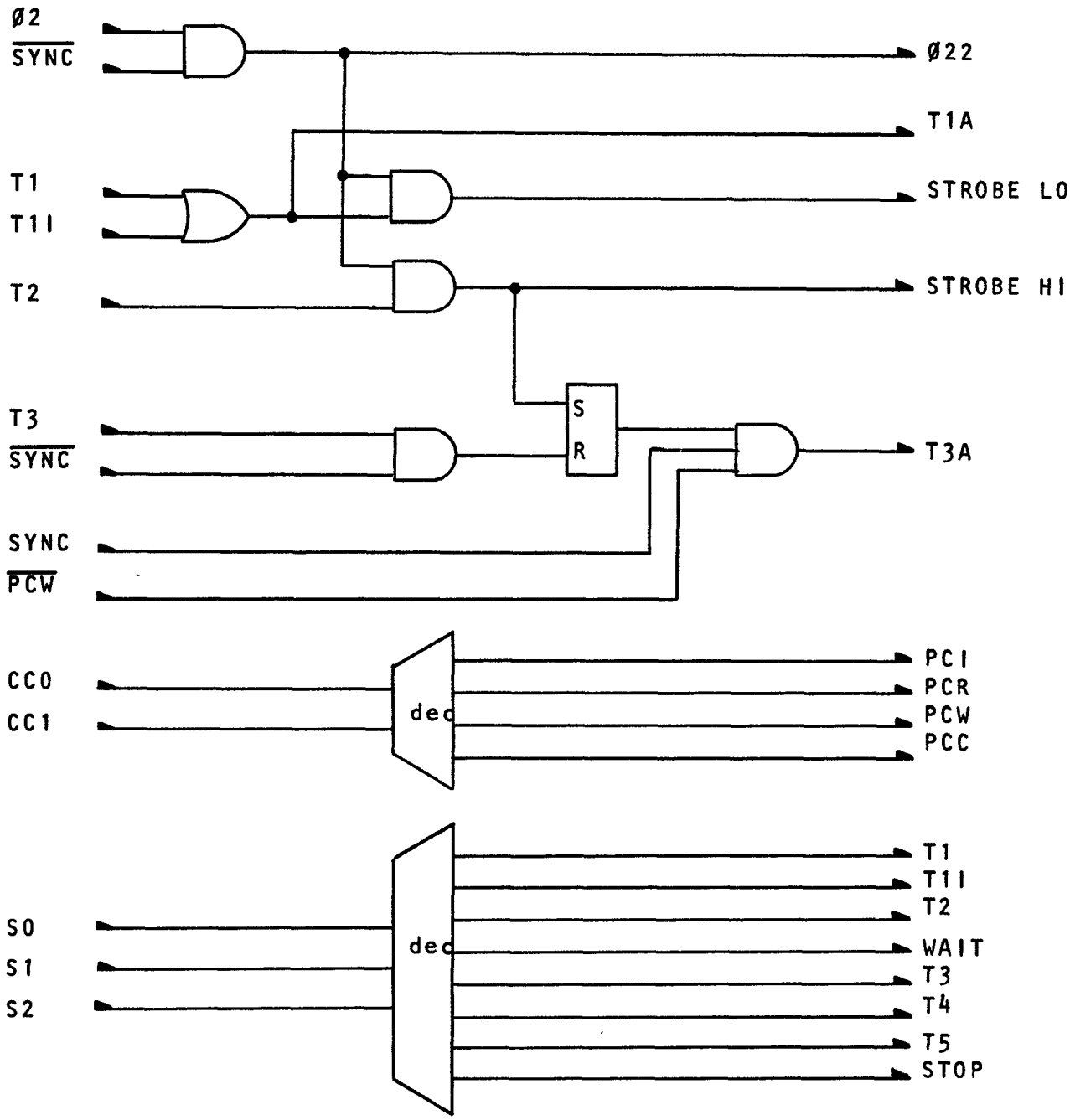
Ø2
SYNC
Ø22
T1A
T1
T11
STROBE LO
T2
STROBE HI
S
R
T3
SYNC
T3A
SYNC
PCW
CC0
CC1
dec
PCI
PCR
PCW
PCC
dec
T1
T11
T2
WAIT
T3
T4
T5
STOP
S0
S1
S2

fig. 1.2

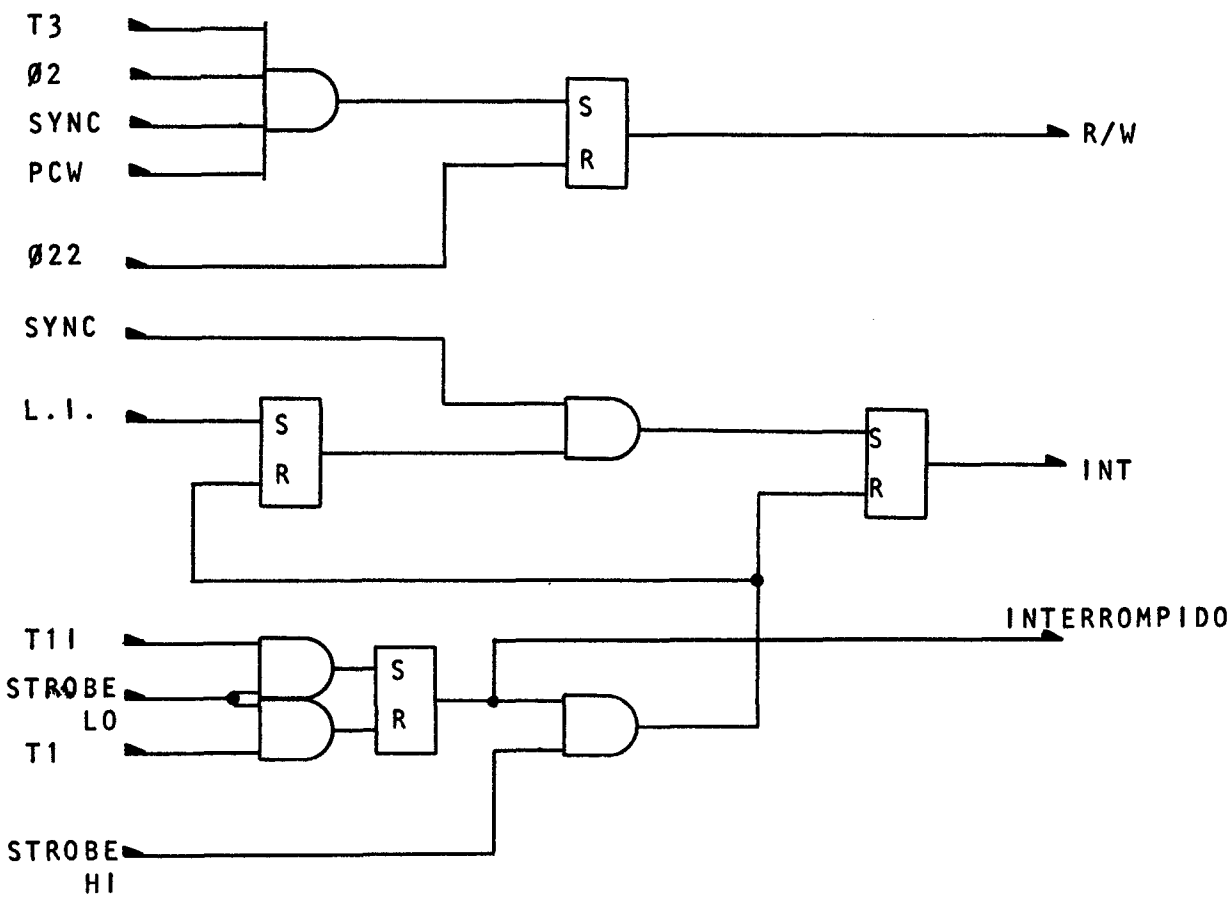
T3
Ø2
SYNC
PCW
Ø22
S
R
R/W
SYNC
L.I.
S
R
S
R
INT
INTERROMPIDO
T11
STROBE
LO
T1
S
R
STROBE
HI

fig. 1.3

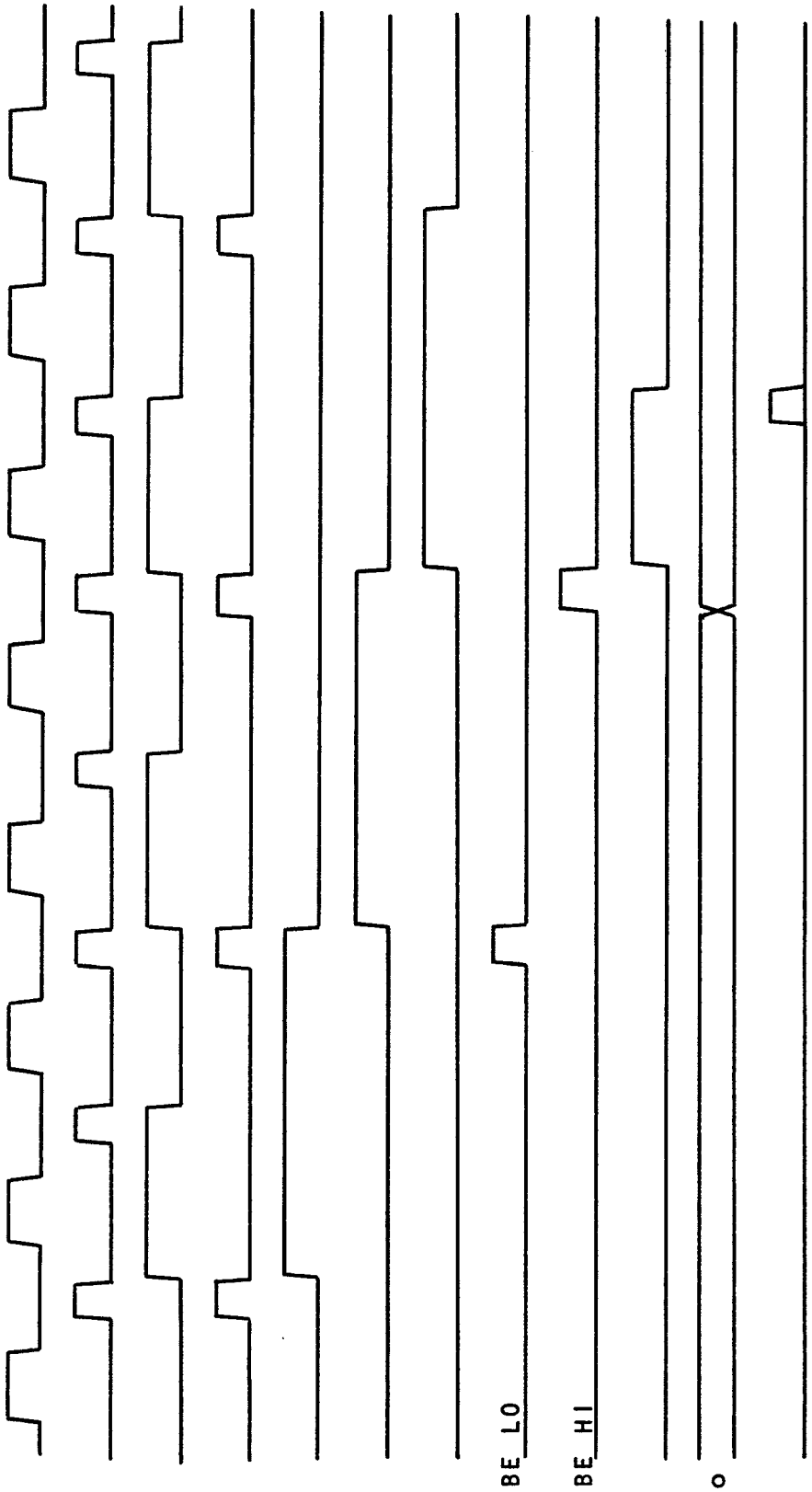

fig. 1.4

I.3 - MEMÓRIA

A memória principal do microcomputador é organizada segundo as necessidades do usuário e geralmente será do tipo a semicondutor. Normalmente haverá uma divisão por módulos (considera-se módulo de memória aquela que já tiver circuito próprio de decodificação de enderêço) com a parte de ROM nos endereços mais baixos. A decodificação da parte alta do enderêço é função do tamanho dos módulos de memória utilizados; a parte alta do enderêço é usada para selecionar o módulo. No caso de uma subdivisão uniforme da memória, ie, módulos de mesmo tamanho, a organização será a da figura 1.5.

A palavra lida em memória só será colocada na barra E durante LIB MEM, já que todos os módulos são isolados desta barra por uma porta lógica (figura 1.6). No caso de RAM (Random Access Memory) o dado a ser escrito está na barra M.

O sinal LIB MEM existirá quando se verificarem as seguintes condições:

- ciclos PCI ou PCR
- processador não interrompido

No caso de memórias lentas há a necessidade de sincronização do processador a elas. Para isto existe o sinal de RDY que poderá estar em 1 ou 0 lógicos, significando que o processador tende a passar de T2 a T3 diretamente ou tende a entrar

no estado de WAIT após T2, respectivamente. No primeiro caso, o módulo de memória lenta deverá forçar RDY a 0, inibindo assim a passagem de T2 para T3, e mantê-lo nesse nível até o fim da operação de leitura ou escrita. No segundo caso, RDY deverá ser levado a 1 quando o acesso tiver sido terminado, fazendo com que o processador deixe o estado de WAIT passando a T3.

A capacidade de memória pode ser aumentada por meio de instruções de entrada e saida. Pode-se concatenar registros de 8 bits aos 14 bits já existentes, carregando-os por meio de instruções OUT. Isto permite uma expansão da memória praticamente ilimitada.

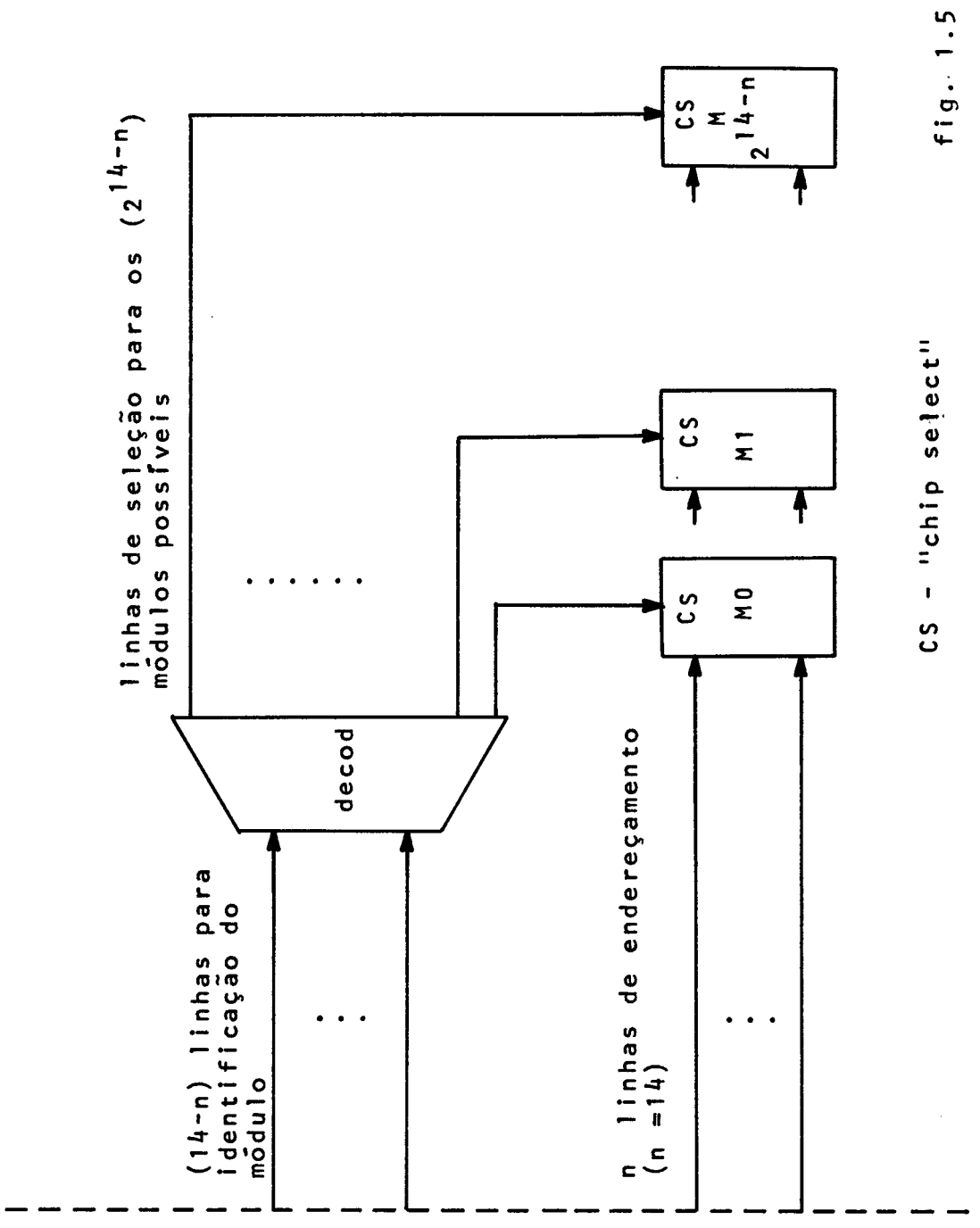
endereço
RH□ RL
(14 bits)
(14-n) linhas para identificação do módulo
n linhas de endereçamento (n =14)
decod
linhas de seleção para os (2^{14-n}) módulos possíveis
CS M0
CS M1
CS M 2^{14-n}
CS - "chip select"

fig. 1.5

I.4 - ENTRADA E SAIDA

Toda comunicação de entrada e saida é feita durante o ciclo PCC. Os periféricos de entrada estão ligados à barra E através de portas lógicas que serão amostradas pelos sinais LIB INP correspondentes. Os periféricos de saida ligados à barra S serão carregados pelos sinais CARR OUT correspondentes (figura 1.6).

O sinal LIB INP 'número do periférico', existirá quando se verificarem as seguintes condições:

- ciclo PCC
- chamada ao periférico de entrada específico (resulta da decodificação do código contido na instrução INP)

O sinal CARR OUT 'número do periférico', existirá quando se verificarem as seguintes condições:

- ciclo PCC
- chamada ao periférico de saida específico (resulta da decodificação do código contido na instrução OUT)

A decodificação do código de periférico, presente em RH, tem duas soluções extremas: total decodificação do código junto ao processador com saida de linhas individuais ou decodificação local, ie, no próprio periférico. A escolha de uma solução entre esses dois extremos dependerá do número e tipo

dos periféricos, bem como da distância entre êles e o processador.

Como no caso da memória, o processador pode ser sincronizado com os periféricos. As explicações dadas anteriormente sôbre o sinal de RDY são válidas aqui.

Para os periféricos de entrada, numerados de 0 a 7, a palavra presente na barra S pode ser usada como comando. O dado a ser enviado ao processador será colocado na barra E. Para os periféricos de saida, numerados na base octal de 10 a 37, o dado está na barra S, enquanto que a barra E permanecerá neutra.

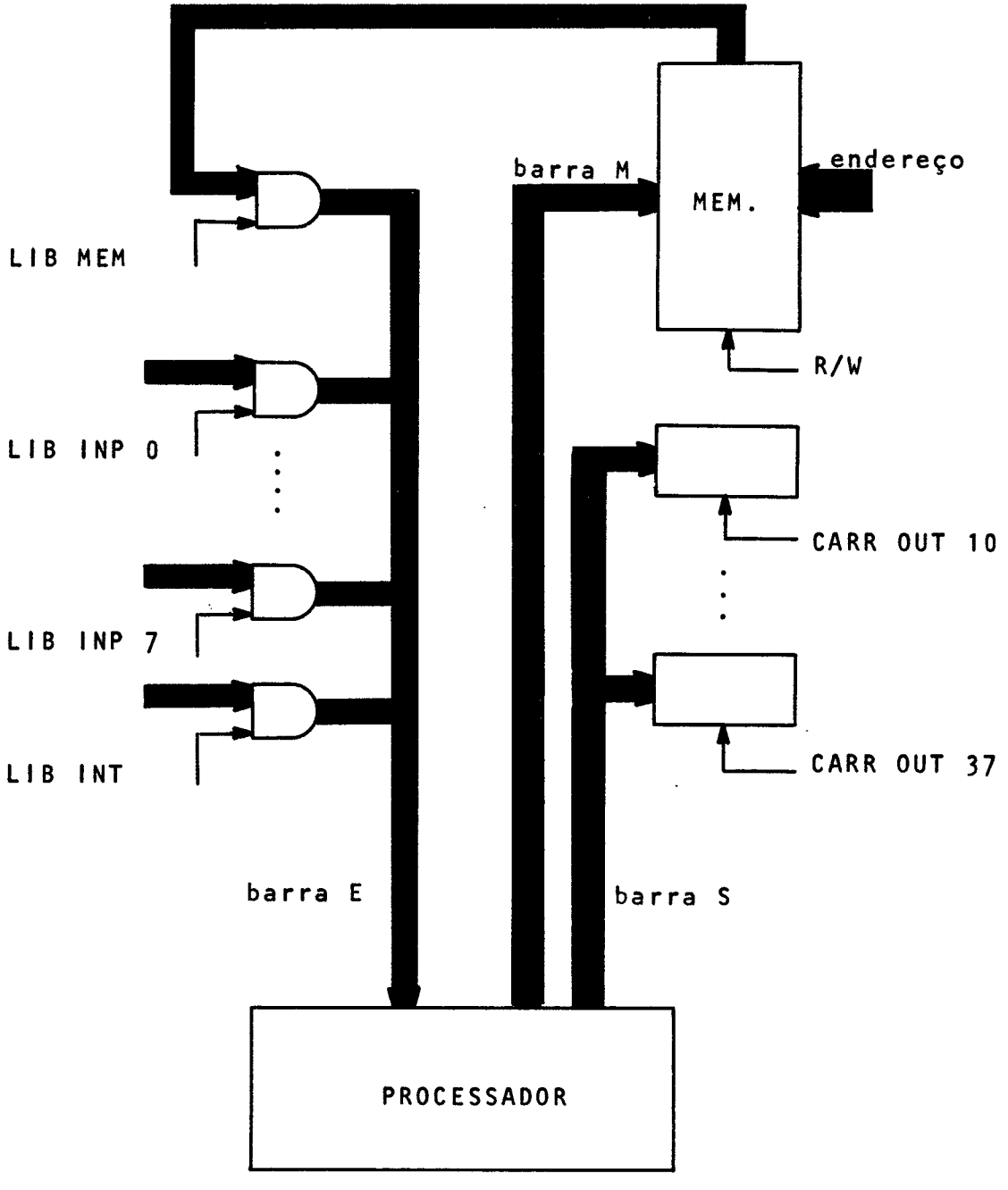
barra M
endereço
MEM.
LIB MEM
R/W
LIB INP 0
CARR OUT 10
LIB INP 7
LIB INT
CARR OUT 37
barra E
barra S
PROCESSADOR

fig. 1.6

I.5 - SISTEMA DE INTERRUPÇÃO

Para o projeto do sistema de interrupção dispõe-se de uma linha de interrupção (L.I.), por onde será enviado o sinal que irá interromper a 8008, e a barra E onde será colocada a instrução. A ligação a essa barra é feita através de porta lógica (figura 1.6), amostrada por LIB INT, que ocorre nas seguintes condições:

- ciclo PCI, ou PCR no caso de instruções longas (duas ou três palavras)
- quando o processador estiver interrompido

Em alguns casos, tipicamente quando a instrução for longa, poderá haver a necessidade de mais outra condição, além das duas citadas acima, gerada pelo próprio sistema de interrupção, a fim de diferenciar cada octeto da instrução longa.

Um sinal poderá ser colocado em L.I. sempre que a última interrupção tiver sido processada (uma interrupção é considerada já processada durante ou após os estados T3, T4 ou T5 do último ciclo da instrução relativa a esta interrupção). Os sinais que não respeitarem êste limite serão ignorados pelo processador.

A primeira palavra da instrução deve estar presente na barra E durante o primeiro ciclo PCI após a colocação do sinal em L.I.. Os eventuais endereços ou dados imediato (caso das ins-

truções longas) deverão ser colocados a sua vez na barra E nos sucessivos ciclos PCR. A figura 1.7 ilustra a sequência de eventos.

Em termos da 8008 há a necessidade de sincronização dos sinais de interrupção, que poderão chegar aleatoriamente no tempo. Essa função será atribuida a um circuito sequencial que armazenará o sinal (de qualquer largura) enviando-o no momento certo para a 8008.

O reconhecimento da interrupção pela 8008 implica na substituição de T1 por T1I em todos os ciclos da instrução (isto é válido para qualquer uma das 48 instruções da 8008). Além disso, o processador fornecerá um sinal (INTERROMPIDO) que permanecerá ligado durante todo o processamento da interrupção.

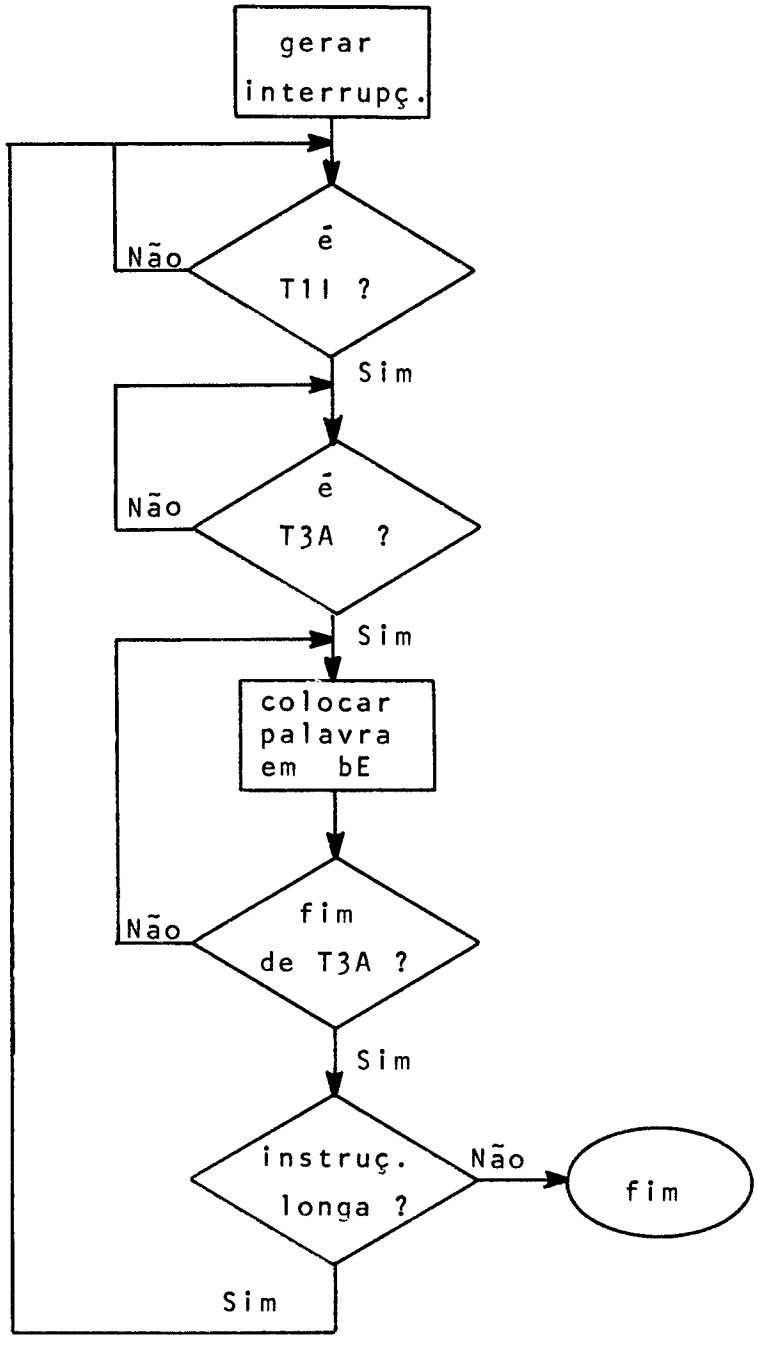
gerar
interrupç.
é
T1I ?
Não
Sim
é
T3A ?
Não
Sim
colocar
palavra
em bE
fim
de T3A ?
Não
Sim
instruç.
longa ?
Não
fim
Sim

fig 1.7

I.6 - CONCLUSÃO

A utilização da barra do tipo coletor aberto com a 8008 oferece como vantagem a modularidade, ie, a colocação de um novo circuito sôbre a barra E não implica em alterações no sistema original. A desvantagem é a limitação inerente ao coletor aberto quanto ao número de circuitos que podem ser conectados à barra (normalmente 25 para o integrado 7401 - 2 input positive nand gate with open-collector output - quando se garantir que apenas um circuito estará ativo de cada vez e que a barra termina numa porta lógica com FAN-IN igual a 1). Porisso é aconselhável ligar as saidas de todos os módulos de memória na entrada de uma só porta lógica com saida coletor aberto (figura 1.6). Isto, e o fato de só existirem no máximo 8 periféricos de entrada, permitirá ao sistema de interrupção o acesso à barra E através de várias portas lógicas, cada qual com seu sinal de amostragem LIB INT.

A barra S não precisa ser do tipo coletor aberto, porém, como nela estarão ligados todos os periféricos de saida e todo o endereçamento da memória, é absolutamente necessário que o registro RL tenha alto FAN-OUT.

Cuidado também deve ser exercido na barra M, pois a soma dos FAN-IN dos módulos de memória (na entrada de dados) não deve ultrapassar o FAN-OUT do circuito que a alimenta.

O FAN-OUT normal de 10 (familia TTL) para o RH geralmente será suficiente.

Os esquemas apresentados para o processador foram projetados para o caso geral de utilização da 8008. Em aplicações particulares, alguns circuitos poderão ser eliminados. A sincronização do sinal de interrupção, por exemplo, poderá ser desnecessária se o sinal em L.I. tiver largura suficiente para interromper a 8008. Também, o circuito que gera R/W só é necessário em sistemas que utilizem memória RAM.

A organização da memória merece cuidados especiais devido ao carregamento em tempos diferentes dos registros RL e RH , e ao fato que nas memórias MOS o tempo de acesso após a mudança das linhas de endereçamento (supondo o módulo de memória já selecionado) é bem maior que o tempo de acesso após a seleção do módulo de memória (supondo o enderêço já presente nas linhas de endereçamento). Portanto, deve-se primeiramente gerar os níveis para as linhas de endereçamento e depois o sinal que seleciona o módulo. Isso pode ser realizado com o processador, se os bits de endereçamento estiverem totalmente contidos em RL, quando então as linhas terão tempo (entre STROBE LO e STROBE HI) para se estabilizar até a seleção do módulo de memória, obtida pela decodificação (que é rápida no caso) dos bits presentes em RH, e mais aqueles de RL que não participam da ativação das linhas de endereçamento. Isto implica num tamanho ótimo do módulo de memória

de 256 palavras (n=8 na figura 1.5), já que a utilização de módulos menores representa um aumento na área ocupada pela memória.

A utilização, nas instruções INP, do valor do acumulador, é particularmente útil para a leitura de condições relativas a um periférico ou para comandos especiais. Por exemplo, pode-se saber se uma fita está ocupada, comandar o enrolamento rápido da fita, etc. Isto é feito precedendo a instrução de INP de um LAI que carrega um código que será interpretado pelo periférico, e eventualmente introduzindo uma instrução de teste sôbre o octeto lido após a instrução de INP.

A existência de diversas partes do sistema com capacidade de interrupção cria o problema da coincidência de sinais em L.I.. Um esquema de prioridade deverá ser estabelecido, se necessário, entre os elementos capazes de interromper a 8008; pode ser uma simples fila (FIFO) ao lado do processador, que armazena as instruções que chegam para tratamento sob interrupção, ou alguma solução mais complexa que envolva uma hierarquia entre êsses elementos.

Uma das particularidades do processador é o fato de não fazer distinção durante o ciclo PCR entre instruções imediatas e de referencia memória, ie, RL e RH não sabem se o enderêço recebido no ciclo de leitura é proveniente do PC ou de H concatenado com L (os dois últimos registros da 8008).

Durante o processamento normal isto não causa problema, pois é uma leitura na memória que é envolvida nos dois casos. Mas sob interrupção implica que se deve optar durante PCR, ou pela leitura em memória (caso da instrução curta LrM), ou pela leitura do sistema de interrupção (como é o caso de instruções longas do tipo LrI). A primeira opção é inadequada, pois não permite instruções de chamada a subrotina, que, como as instruções de referência imediata, são instruções longas (o sistema de interrupção enviaria à 8008 o primeiro octeto da instrução CAL durante PCI, e depois a 8008 leria a memória, endereçada pelo PC, para obter o enderêço da subrotina, com resultados imprevisíveis). Só resta a opção de se restringir as instruções que podem ser resolvidas sob interrupção ao conjunto que não inclui instruções de referência memória com leitura (a escrita é permitida, pois acontece no ciclo PCW). Isto é perfeitamente tolerável para a maioria dos casos. Instruções associadas a interrupções são tipicamente CALL, RESTART e HALT. A eliminação das instruções LrM do repertório daquelas que podem ser resolvidas sob interrupção poderia ter sido evitado pelos engenheiros da INTEL se durante a resolução da instrução LrM o ciclo PCR aparecesse com T1 em vez de T1I. Infelizmente êste não é o caso.

C A P Í T U L O I I

O SIMULADOR

II.1 - INTRODUÇÃO

O simulador escrito em PL/1 tem como finalidade o desenvolvimento e correção de programas para o microcomputador. Uma sequencia de instruções em linguagem de máquina da 8008 é fornecida ao simulador que lista como resposta o estado da 8008 a cada passo da resolução do programa.

Entradas e saidas, e interrupções podem ser definidos pelo programador dentro do programa principal em posições preestabelecidas.

O estabelecimento da imagem da memória é feita por meio de cartões de controle colocados antes dos dados.

Como outra opção pode-se controlar a impressão dos resultados informando ao simulador quais as instruções que aparecerão na listagem final.

Neste capítulo, ao contrário do primeiro, dividir-se-á a memória em páginas. Em princípio êste novo conceito e módulo de memória se confundem; um módulo seria uma página. Mas se no projeto do computador não se pode abstrair dos tamanhos dos módulos por causa da decodificação, no simulador se fará uma

divisão uniforme do vetor memória a fim de facilitar a programação PL/1. As páginas maiores serão portanto subdivididas em menores com consequencia que um módulo da máquina real corresponderá a uma ou mais páginas dentro do simulador.

Os tempos envolvidos na simulação, como o de instrução, de acesso à memória e outros, serão medidos com a unidade de tempo do simulador - o período do sinal de sincronismo gerado pela 8008 (entre 2 e 3 microsegundos). Não serão admitidos tempos com parte fracionária, devendo normalmente se arredondar o valor para o inteiro imediatamente superior.

Inicialmente descreve-se as características principais do simulador e em seguida a sua utilização. A listagem do programa principal se encontra no apêndice C .

A fim de evitar confusão usar-se-á a seguinte convenção:

programa principal ou simulador - lista de instruções PL/1 que realizam a simulação do microcomputador.

vetor memória - vetor de 16384 palavras de 8 bits utilizado como memória pelo microcomputador simulado. Este vetor é subdividido em páginas.

programa - lista de instruções 8008 armazenada na memória.

imagem da memória - máscara do vetor memória para a definição de

.tipo de cada página, ie, se é ROM ou RAM

.tempos de acesso, leitura ou escrita, a cada página

.comportamento do sistema no caso de endereçamento de página inexistente

ponto de quebra - par de endereços que definem o início e fim da impressão dos resultados.

II.2 - PROGRAMA PRINCIPAL

A primeira função importante do simulador é a medição do "tempo real" de processamento da 8008. Isto é feito através da variável NUMERO_DE_ESTADOS.

Inicializada em zero, NUMERO_DE_ESTADOS será incrementada a medida que instruções forem resolvidas (tempos fixos dentro do programa principal) e a cada chamada à memória, às entradas e saidas, e ao sistema de interrupção (tempos definidos pelo usuário). A introdução desses tempos no programa principal se faz com instruções PL/1 do tipo

```
NUMERO_DE_ESTADOS = NUMERO_DE_ESTADOS + tempo ;
```

II.2.1 - Módulos

O programa principal é subdividido em módulos que consistem numa série de instruções PL/1 que realizam uma função específica dentro do simulador. Há dois conjuntos de módulos: os processados uma só vez e os utilizados repetidas vêzes durante o processamento.

O primeiro conjunto, que faz a inicialização do simulador, é formado pelos seguintes módulos:

<u>módulo de declaração e inicialização de variáveis</u>

nêle estão declaradas tôdas as variáveis utili-

```
/*************************************************************************/SIMU5300
/* ************** M O D U L O   A R I T M E T I C O   R E G I S T R O*/SIMU5301
/*************************************************************************/SIMU5302
MODULO_ALU_R:                                                            SIMU5304
TIPO_DE_REFERENCIA=NOME_DO_REGISTRO(D2D1D0);                             SIMU5308
        REGB=REGINDEX(D2D1D0);                                           SIMU5312
        NUMERO_DE_ESTADOS=NUMERO_DE_ESTADOS+5;                           SIMU5314
        GOTO MODULO_ALU;                                                 SIMU5316

/*************************************************************************/SIMU5350
/* ************** M O D U L O  A R I T M E T I C O   M E M ************/SIMU5351
/*************************************************************************/SIMU5352
MODULO_ALU_M:                                                            SIMU5354
TIPO_DE_REFERENCIA='M'; NUMERO_DE_ESTADOS=NUMERO_DE_ESTADOS+8;           SIMU5356
LABEL2=MODULO_ALU; GOTO MODULO_M;                                        SIMU5358

/*************************************************************************/SIMU5400
/* ************** M O D U L O   A L U   I M E D I A T O ************/SIMU5401
/*************************************************************************/SIMU5402


DECODIFICACAO_DE_D2D1D0(4): MODULO_ALU_I:                                SIMU5404
TIPO_DE_REFERENCIA='I'; NUMERO_DE_ESTADOS=NUMERO_DE_ESTADOS+8;           SIMU5406
LABEL1=MODULO_ALU; GOTO MODULO_I;                                        SIMU5408
```