I

1.Classificação *INPE-COM.3/NTE* *C.D.U.- 681.322*	2.Período	4.Critério de Distribuição: interna [ ] externa [X]
3.Palavras Chave (selecionadas pelo autor)		
5.Relatório nº *INPE-1028-NTE/086*	6.Data *Maio de 1977*	7.Revisado por - *Ricardo C.O. Martins*
8.Título e Sub-Título *Um Simulador Reconfigurável para o Microprocessador INTEL 8080*		9.Autorizado por - *Dr.Nelson Jesus Parada* *Diretor*
10.Setor *CCI*	Código *460*	11.Nº de cópias *10*
12.Autoria *Wilson Yamaguti* *Ricardo Correa de Oliveira Martins* 13.Assinatura Responsável		14.Nº de páginas *131* 15.Preço

16.Sumário/Notas

O propósito deste trabalho é a implementação de um simulador reconfigurável para o microprocessador INTEL 8080, utilizando a linguagem ALGOL e o sistema B-6700 do INPE. A simulação é feita de maneira específica e a um nível de detalhe bem maior que o usual, nível este necessário principalmente em aplicações em tempo real, onde o projeto do sistema de atendimento de interrupções é uma das tarefas principais e de mais difíceis verificação.

17.Observações

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

CAPÍTULO I

INTRODUÇÃO

O advento dos microprocessadores reduziu sensivelmente os custos envolvidos no desenvolvimento do "hardware" de sistemas digitais. Porém, devido principalmente ao seu conjunto de instruções bastante primitivo, todo projeto que o utilize, necessita de um "software" de suporte. Este "software" de suporte se faz necessário tanto na fase de desenvolvimento do sistema como também na fase de programação do sistema final (o "microcomputador").

Essas ferramentas de suporte são desenvolvidas principalmente na forma de:

- "cross-assemblers" e "cross-compilers" - programas montadores que aceitam uma linguagem simbólica e geram código de máquina correspondente.

- "cross-simulators" - programas que aceitam os códigos de máquina gerados pelos "cross-assemblers" e "cross-compilers" e simulam sua execução no computador hospedeiro.

A parte de geração de código de máquina já foi abordada no INPE através do desenvolvimento e implementação de uma linguagem de alto nível para programação de microcomputadores. O "cross-compiler" ALGOL M,desenvolvido, permite a utilização de linguagem que apresenta uma série de facilidades na programação do microprocessadores atualmente existentes.

A utilização de simuladores possibilita certos diagnósti

cos que não são obtidos com o próprio microcomputador em ação. Além de permitirem certas manipulações e listagens de memória e registradores da CPU, os simuladores usualmente possuem capacidade de:

- "break points", onde o processamento pode ser parado num determinado endereço de programa ou quando o programa lê ou escreve numa posição especificada da memória.

- "tracing", em que o simulador imprime os conteúdos dos registradores e da memória, num intervalo de endereços especificado, à medida que as instruções são executadas.

- Informação de tempo, tais como número de instruções executadas, número de ciclos de máquina, número de pulsos de relógio, etc.

É indiscutível o poder desses simuladores como ferramentas de suporte, porém não se pode substituir completamente os testes de programa no próprio microcomputador, dado que o tempo específico e as condições externas de circuitação real nunca podem ser totalmente simuladas.

A simulação pode ser abordada de duas maneiras:

a) através da utilização de linguagem de Descrição de "Hardware" (C.H.D.L.).

b) ou através de desenvolvimento de um simulador específico.

O presente simulador, desenvolvido e implementado de forma específica para o microprocessador INTEL 8080, apesar de se destinar à simulação de um único microprocessador, oferece uma série de vantagens tanto no desenvolvimento de sistemas que utilizam o INTEL 8080 como também na verificação de programa de aplicação para o sistema final.

Em termos gerais, o sistema de simulação desenvolvido pode ser esquematizado como na Figura I.1 abaixo:

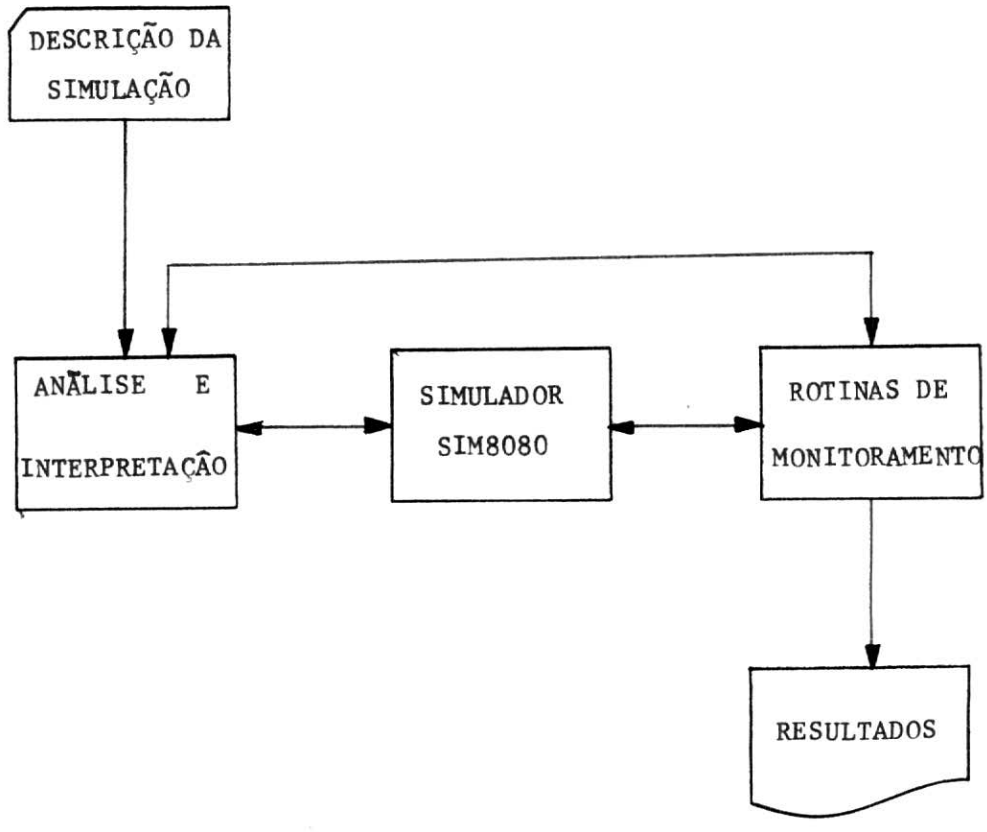


Fig. I.1 - Sistema de Simulação Implementado para o Microprocessador INTEL8080.

As características gerais do sistema de simulação, bem como a descrição dos analisadores sintáticos e léxicos, são apresentadas no Capítulo II.

No Capítulo III, é introduzida a descrição da unidade de processamento central do 8080 de acôrdo com os dados fornecidos no manual do fabricante, enquanto que a linguagem de contrôle de simulação é descrita no Capítulo IV. Os comentários finais, bem como a bibliografia utilizada são apresentados no Capítulo V.

CAPÍTULO II

DESCRIÇÃO GERAL DO SISTEMA

A simulação de um sistema deve, em princípio, se aproximar tanto quanto possível da realidade do simulado, a fim de que resultados úteis possam surgir estabelecendo, desta forma, a capacidade do simulador e suas características.

2.1 - CARACTERÍSTICAS GERAIS

A simulação do microprocessador INTEL 8080, de maneira específica, possibilita maior aproximação da realidade do simulado, sem aumento em demasia do grau de complexidade da implementação.

Escrito em linguagem ALGOL e implementado no sistema B-6700, o simulador reconfigurável para o 8080 apresenta, em linhas gerais, as características seguintes:

a) Estrutura Modular - simulação é feita a níveis de relógio ("clock"), possibilitando numa estrutura modular em estados assumidos pelo processador. A sincronização é obtida através de 3 rotinas identicas de incremento de tempo: pulsos de relógio (INCRCLOCK); ciclos (INCRCYCLE) e instrução (INCRINST). Dessa forma, a inclusão de subrotinas de simulação de interfaces de entrada/saída e periférico é facilitada.

b) Reconfigurável - estrutura do simulador organizada de modo que as instruções podem ser microprogramadas. A sequência de microoperações pode ser recarregada através de programa auxiliar.

c) Comandos de execução - o sistema de simulação apresenta um conjunto de "comandos de execução" que comandam e definem a simulação desejada.

d) "TRACING" - apresenta as facilidades de "tracing", especificados pelos comandos de execução, de posições de memória e de registradores do INTEL 8080.

e) Interrupção, "HOLD", "RESET" permitem a simulação e ativação, através do comandos de execução, do sistema de interrupções com prioridades associadas, do sistema de requisição de "HOLD" (DMA) e do sistema de "RESET".

f) Código de máquina - o simulador aceita código de máquina gerado pelo compilador da linguagem ALGOL M.

g) Memória e CPU - simula em "software" a memória do sistema e a unidade central constituida pelo INTEL8080.

O sistema de simulação pode ser apresentado como mostra o diagrama de blocos da Figura II.1

Na simulação de um dado programa, os comandos de execução são fornecidos,através de cartão,ao sistema simulador,que se encarrega

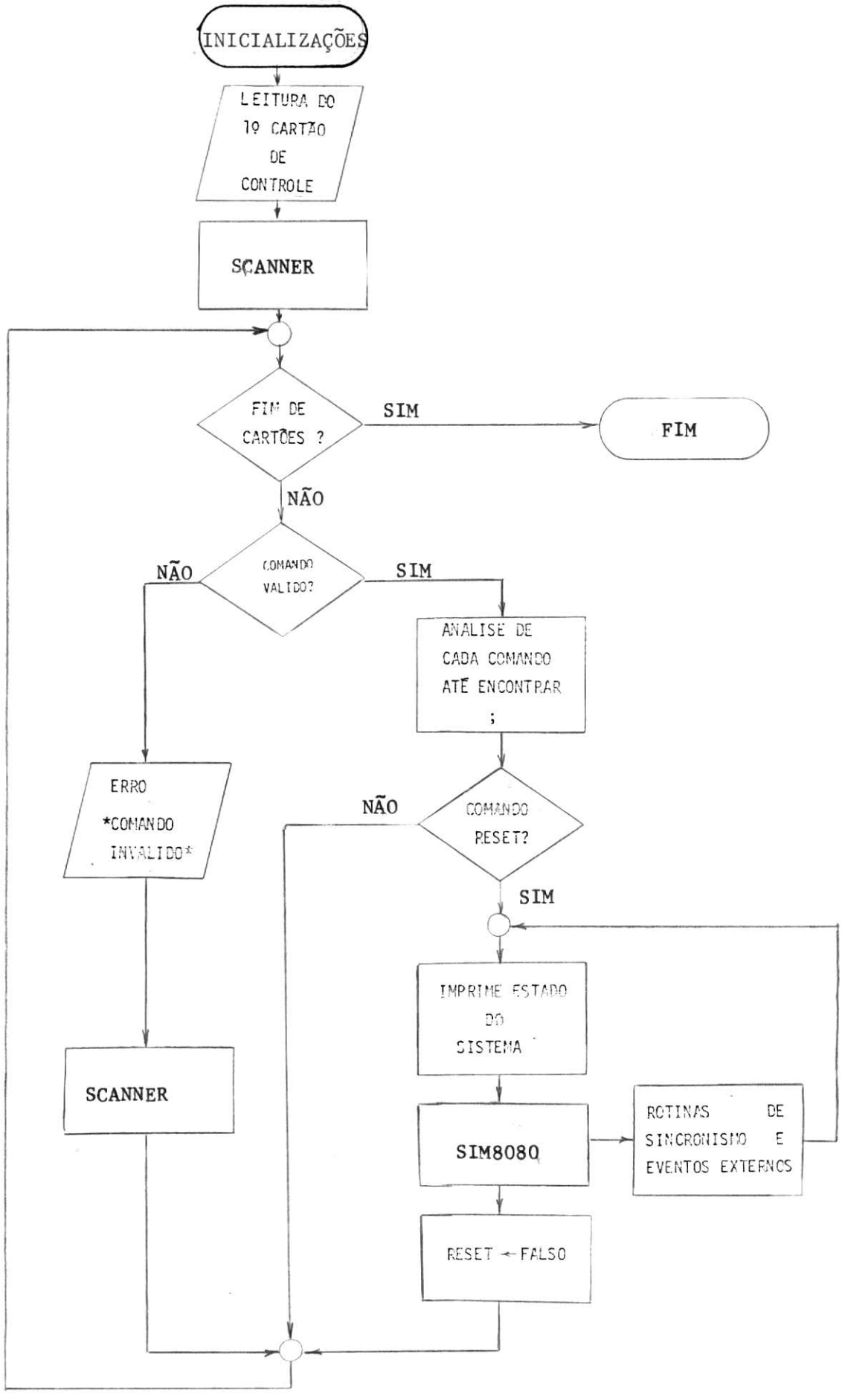


Fig. II.1 - Diagrama do Sistema de Simulação.

da análise léxica e sintática.

A análise léxica é feita pelo "SCANNER" (analisador léxico) que procura verificar se as entidades presentes num comando pertencem ao conjunto de simbolos da linguagem.

Caso ocorra erro de qualquer natureza, há indicação, na listagem do programa, onde o "SCANNER" parou, bem como as mensagens adequadas. O "SCANNER" é avançado até que o delimitador de fim de comando seja encontrado. Desta forma, a análise é feita até o último cartão.

A ativação do simulador propriamente dito (a subrotina SIM8080) é iniciada quando o comando "RESET" acaba de ser analisado. O estado inicial do sistema bem como os estados que assume ao longo da simulação são então impressos de acordo com os comandos de contrôle utilizados.

O simulador executa as instruções a partir do endereço de memória definido pelas condições iniciais do contador de programa ("PC") Esta simulação termina no tempo especificado pela instrução RESET encontrada, podendo iniciar uma nova simulação de acôrdo com um novo conjunto de comandos e por tempo determinado por um novo comando RESET.

2.2 - ANALISADOR LÉXICO ("SCANNER")

A análise léxica do programa é feita pela subrotina "SCANNER", que retorna nas variáveis globais:

⊳ ENTIDADE, o "cordão" de caracteres identificado.

⊳ SÍMBOLO, o número de classificação na tabela de símbolos. Caso seja encontrado erro, SÍMBOLO será igual a zero.

A Figura II.2, apresenta o diagrama de estados do "SCANNER". O Estado inicial (1) é mantido enquanto os caracteres de entrada forem brancos (GETNONBLANCK).

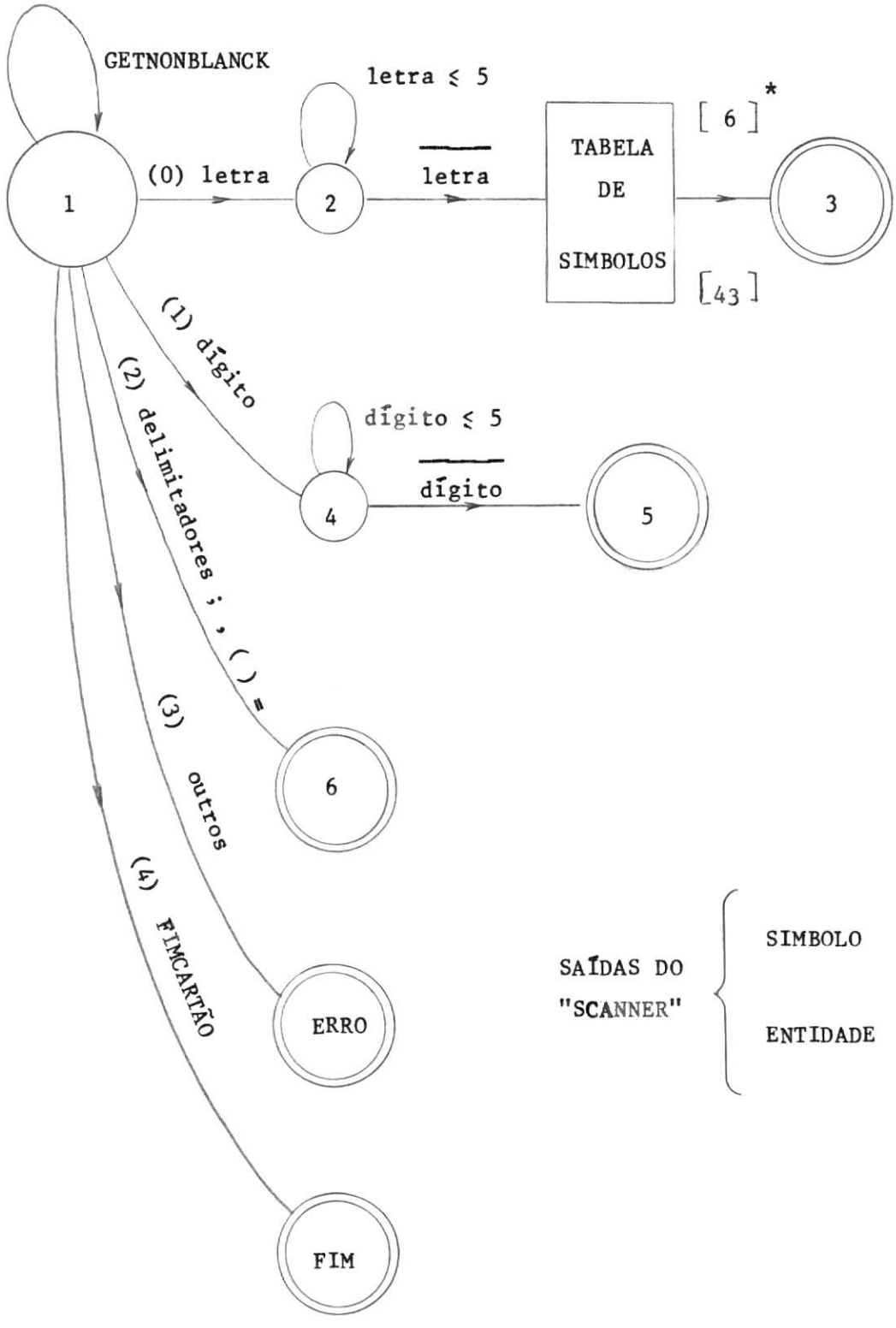
Os caracteres reconhecidos pelo "SCANNER" e que determinam a classe dos mesmos são os seguintes:

LETRAS:	A | B | C | D |. . . . . Z	Classe 0
DÍGITOS:	0 | 1 | 2 | 3 |. 9	Classe 1
DELIMITADORES:	; | , | (|) | = | %	Classe 2

Então em função da classe associada ao caracter, um novo estado é assumido.

Admitindo que ocaractere de entrada seja letra, tem-se o estado (2) e o mesmo é mantido enquanto os caracteres sucessivos pertencerem ao conjunto LETRAS. Haverá mensagem de erro, se o número desses caracteres fôr maior que 6.

Um caractere não pertencente a LETRAS, faz com que o "SCANNER", através de rotinas auxiliares, consulte sua tabela de símbolos



* [6] indica o intervalo na tabela de símbolos

[43]

Fig.II.2 - Diagrama de Estados do "SCANNER"

fornecendo sua classificação na mesma no estado terminal (3), bem como o "cordão" de caracteres identificado.

De modo análogo, os outros estados podem ser assumidos.

A Figura II.3, apresenta a tabela de símbolos das entidades reconhecidas e sua classificação.

2.3 - ANALISADOR SINTÁTICO

A análise sintática dos comandos de contrôle da simulação é feita pelo ANALISADOR SINTÁTICO, apresentado em forma de diagrama de estados da Figura II.4.

Os comandos aceitos são as constantes da tabela de simbolos, apresentada na Figura II.3, de 6 a 17, ressaltando os comandos INPUT, OUTPUT, COPY e SAVE que serão implementados futuramente.

Cabem,também, as observações seguintes,no diagrama de estados da Figura II.4:

- ▷ <*nº*> representa um numero de no máximo 6 dígitos
- ▷ No comando LOAD, <*id*> é um dos identificadores da tabela de símbolos de 26 a 43.
- ▷ <*id'*> é o conjunto associado a <*id*> excetuando o identificador M.

```
%
%
%  TABELA DE SIMBOLOS
   VALUE ARRAY SIMBOLOS("       ",

                        "       ",   % 1
                        "       ",   % 2
                        "       ",   % 3
                        "       ",   % 4
                        "       ",   % 5
                        "MEMORY",    % 6
                        "READY ",    % 7
                        "LOAD  ",    % 8
                        "INTER ",    % 9
                        "HOLD  ",    % 10
                        "RESET ",    % 11
                        "REFER ",    % 12
                        "WRITE ",    % 13
                        "INPUT ",    % 14
                        "OUTPUT",    % 15
                        "COPY  ",    % 16
                        "SAVE  ",    % 17
                        "TO    ",    % 18
                        "CYCLE ",    % 19
                        "CLOCK ",    % 20
                        "INST  ",    % 21
                        "WITH  ",    % 22
                        "DUMP  ",    % 23
                        "STATUS",    % 24
                        "AND   ",    % 25
                        "PC    ",    % 26
                        "SP    ",    % 27
                        "A     ",    % 28
                        "B     ",    % 29
                        "C     ",    % 30
                        "D     ",    % 31
                        "E     ",    % 32
                        "H     ",    % 33
                        "L     ",    % 34
                        "ZERO  ",    % 35
                        "CARRY ",    % 36
                        "SIGN  ",    % 37
                        "PARITY",    % 38
                        "ACARRY",    % 39
                        "M     ",    % 40
                        "INT   ",    % 41
                        "HOLD  ",    % 42
                        "INTE  ",    % 43
                        "      ",    % 44
                        "      ",    % 45
                        ";     ",    % 46
                        ",     ",    % 47
                        "(     ",    % 48
                        ")     ",    % 49
                        "=     ",    % 50
                        "      ",    % 51 <NUMERO>
                        "      ",    % 52 ERRO
                        "      ");   % 53 FIMCARTAO

%
```

Fig. II.3 - Tabela de Símbolos.

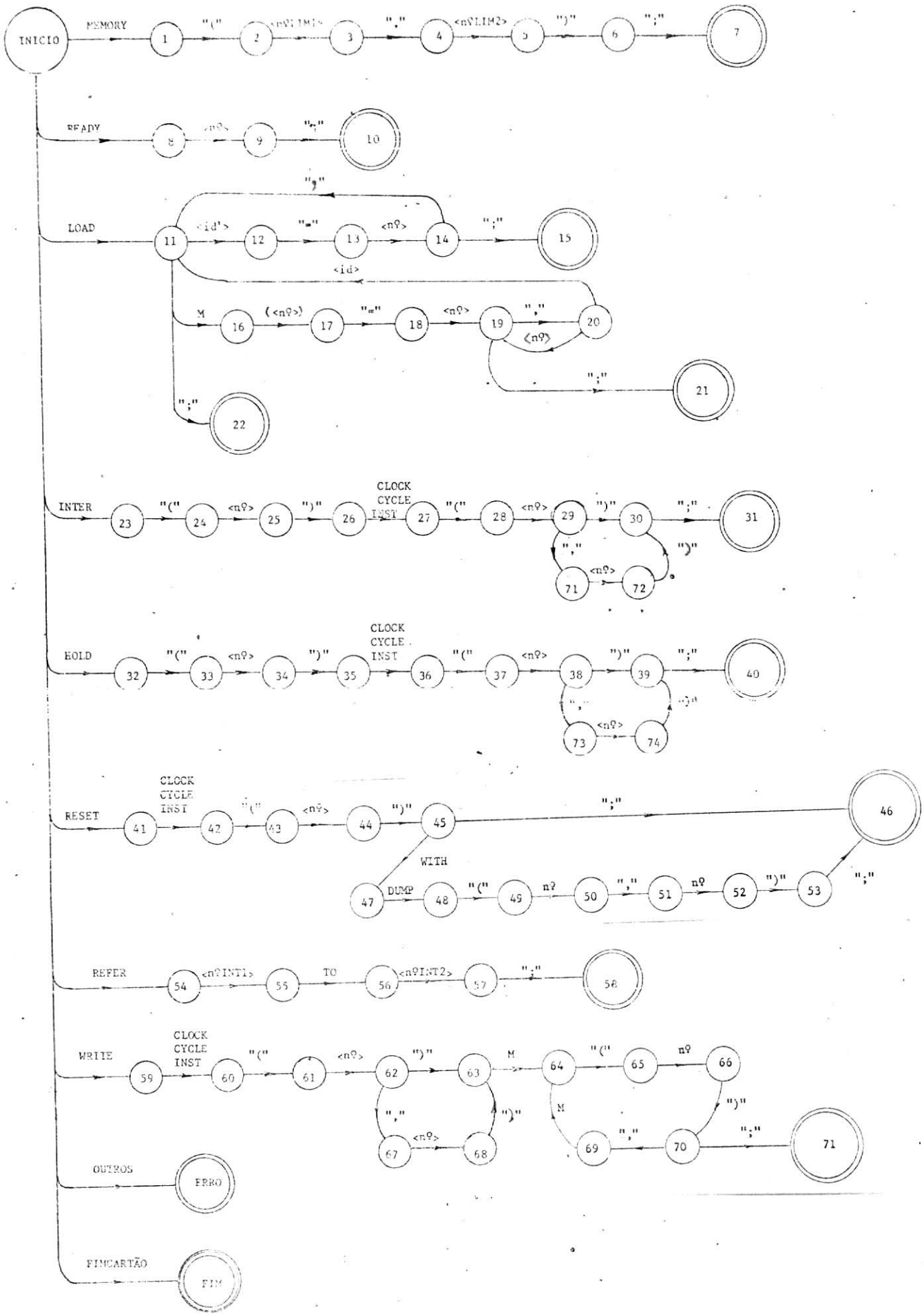


Fig.II.4 - Diagrama de Estados do Analisador Sintático

Os estados assinalados com duas circunferências concêntricas indicam fim de análise de cada comando, fazendo então o analisador retornar ao estado inicial. O estado assinalado em três circunferências indica estado final do analisador (fim do programa).

Os resultados da análise léxica determinam as transições do estado inicial a um dos estados terminais. Caso não ocorra o caracter esperado, em qualquer das transições, haverá mensagem de êrro, bem como o sinalização da posição em que o "SCANNER" parou, no programa de simulação analisado.

A Figura II.5, apresenta as possíveis mensagens de erro de sintaxe implementadas até o momento e, a Figura II.6, um programa teste de simulação com erros de sintaxe.

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
 SWITCH FORMAT ERRADOS:=
%1
 (X*,"MAIS DE 6 CARACTERES EM CONST. OU ID"),
%2
 (X*,"IDENTIFICADOR INEXISTENTE"),
%3
 (X*"CARACTER INVALIDO"),
%4
 (X*,"NUMERO EXCESSIVO DESTE COMANDO"),
%5
 (X*,"FALTOU O     ; "),
%6
 (X*,"FALTOU O ("),
%7
 (X*,"NUMERO INVALIDO"),
%8
 (X*,"FALTOU O )"),
%9
 (X*,"PAR DELIMITADOR INVALIDO"),
%10
 (X*,"COMANDO INVALIDO"),
%11
 (X*,"FALTOU O -TO- "),
%12
 (X*,"PERMITIDO APENAS-CYCLE,CLOCK OU INST"),

%13
 (X*,"NUMERO INVALIDO PARA INTERRUPCAO"),
%14
 (X*,"FALTOU O SINAL DE ATRIBUICAO"),
%15
 (X*,"CONSTANTE INVALIDA NESTA ATRIBUICAO"),

%16
 (X*,"ESPECIFICACAO DE MEMORIA INVALIDA"),
%17
 (X*,"MAIS DE 10 ELEMENTOS REFERENCIADOS"),

%18
 (X*,"ENDERECO INVALIDO");
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Fig. II.5 - Mensagens de Érro de Sintaxe.

```
==================================================================================================================
           SIM8080 -- SIMULADOR DO MICROPROCESSADOR INTEL 8080 (  8-87)0 -- VERSAO 1
                   SIMULACAO EXECUTADA EM 22/12/76 AS 14 HORAS E -51 MINUTOS
==================================================================================================================

          %
          %  SIMULACAO  PARA TESTAR O SIM8080
          %
         %
         % O PROGRAMA TESTE FAZ A ADICAO DECIMAL DOS CONTEUDOS DAS POSICOES    DE
         % MEMORIA M(30) A M(37) COM AS DE M(38) A M(45)
         % O RESULTADO EM COLOCADO NAS POSICOES DE MEMORIA M(30) A M(37)
         %
          MEMORY(01,100);
          %
          % INTERRUPCOES A SEREM SIMULADAS
          %
          INTER(6) INST(5);
          %
          HOLD(10) INST(15);
          %
          %
         WRITE(CLOCK)(1,1);
1>>>>>>       * <<<<<<
               PERMITIDO APENAS-CYCLE,CLOCK OU INST
         WRITE INST(1,1) M(30),M(31),M(32),M(33),M(34),M(35),M(36),M(37),M(38),
2>>>>>>       * <<<<<<
               NUMERO EXCESSIVO DESTE COMANDO
         M(39),M(40);
         M(39);
3>>>>>>   * <<<<<<
          COMANDO INVALIDO
          %
         LOAD M(1)= 021,      %       LXI     D,ALPHA           1
                    035,      %                                 2
                    000,      %                                 3
                    041,      %       LXI     H,BETA            4
                    046,      %                                 5
                    000,      %                                 6
                    016,      %       MVI     C,8               7
                    010,      %                                 8
                    257,      %       XRA                       9
                    032,      % LOOP:LDAX     D                10
                    216,      %       ADC     M                11
                    047,      %       DAA                      12
                    022,      %       STAX    D                13
                    043,      %       INX     H                14
                    023,      %       INX     D                15
                    015,      %       DCR     C                16
                    302,      %       JNZ     LOOP             17
                    012,      %
                    000,      %
                              %
              M(30)=001,002,003,004,005,006,007,000,    % ALPHA
                    001,002,003,004,005,006,007,000;    % BETA
         LOAD PC=000001, SP=000140,
         :
              INTE=1;
4>>>>>>           * <<<<<<
```

```
                COMANDO INVALIDO
LOAD M(48)= 373,    %       ET                48
            311;    %       RE                49
LOAD M(56)= 373,    %       ET                50
            311;    %       RT                51
                                              52
RESET INST(40);                               53
```

Fig. II.6 - Sinalização dos Erros de Sintaxe num Programa de Simulação.

CAPÍTULO III

DESCRIÇÃO DO MICROPROCESSADOR 8080 E DA ROTINA QUE O SIMULA (SIM 8080).

Nêste capítulo é apresentada, uma breve descrição do 8080, necessária à melhor compreensão da rotina de simulação SIM8080. Para a nomenclatura dos terminais do processador e de seus registros internos na simulação, utilizou-se aquela presente no diagrama funcional da Figura III.1 bem como para as variáveis da programação aquelas constantes no manual do Microprocessador INTEL 8080.

3.1 - DESCRIÇÃO DA UNIDADE DE PROCESSAMENTO CENTRAL (CPU)

O 8080 é um microprocessador de 8 bits para uso geral, fabricado pela INTEL numa pastilha LSI, empregando o processo "n - channel silicon gate MOS".

Os seus terminais são constituidos pelo barramento de dados (8 bits), pelo barramento de endereço (16 bits), pelo barramento de contrôle e tempo e pelos pinos de alimentação.

O barramento de dados (D7D0) é bidirecional, por meio do qual há transferência de dados entre a CPU e o resto do sistema. A transferência de endereços da memória e periféricos é feita através do barramento de endereço (A15A0). A CPU possui condições de forçar os barramentos de dados e de endereço para o estado de alta impedância, permitindo assim o contrôle desses barramentos por outros dispositivos (por ex. acesso

direto à memória (DMA)).

SYNC, DBIN, WAIT, $\overline{WR}$, HALDA e INTE constituem as saídas de contrôle e tempo enquanto que READY, HOLD, INT e RESET são as entradas.

Além dos terminais já mencionados, as entradas de alimentação + 12V, + 5V, - 5V, Terra e duas de relógio ϕ_1 e ϕ_2 completam um rápido apanhado dos pinos do microprocessador 8080.

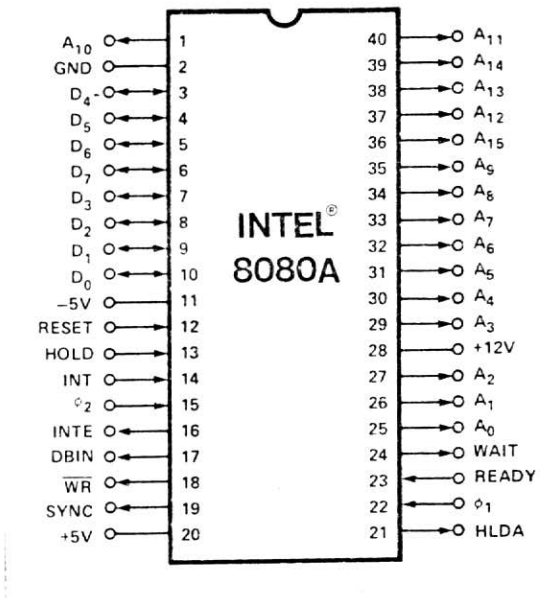


Fig. III.1 - Configuração dos Terminais da CPU 8080.

É apresentada em forma de tabela, a descrição funcional de cada terminal do 8080.

TABELA III.1

NOMENCLATURA E FUNÇÃO DOS TERMINAIS DO 8080

NOMENCLATURA	F U N Ç Ã O
A15A0	Barramento de endereço - constituido por 16 bits onde A0 é o menos significativos
D7D0	Barramento de dados - que provê comunicação bidirecional entre a CPU, memória e dispositivos de entrada e saída (I/O) para instruções e dados. D0 é bit menos significativo.
SYNC	Sinal de Sincronismo - fornecido pela CPU que indica o começo de um novo ciclo de máquina.
DBIN	Sinal que indica ao circuito externo o modo de operação do barramento de dados. Nível alto significa modo de entrada (ciclos de busca, leitura de memória , leitura de PILHA e interrupção).
READY	Sinal que indica à CPU que os dados estão disponíveis no barramento D7D0. Este sinal é utilizado para sincronizar a memória ou periféricos. Se depois de mandar um endereço, o 8080 não recebe o sinal READY, o 8080 entra em estado de espera enquanto READY estiver em nivel baixo. O sinal READY também pode ser utilizado para contrôle da CPU por passos.

(continuação da Tabela III.1)

NOMENCLATURA	F U N Ç Ã O
WAIT	Sinal notificando que a CPU está no estado de espera, denominado estado WAIT.
$\overline{WR}$	Sinal utilizado para contrôle de escrita na memória ou I/O de periférico. Os dados estão estáveis no barramento D7D0 enquanto o sinal $\overline{WR}$ apresentar nível baixo (ciclos de escrita na memória, escrita na pilha e saída para periférico).
HOLD	Sinal para a CPU entrar em estado denominado de HOLD. Este estado possibilita o contrôle dos barramentos A15A0 e D7D0 por dispositivo externo, tão logo termine o ciclo de máquina corrente. A CPU anuncia que entrou em estado HOLD, pelo terminal HLDA. Para melhor compreensão vide o diagrama de estados, onde se pode observar que o sinal de HOLD é verificado no estado HALT e na entrada do estado T3.
HLDA	Sinal de resposta ao pedido de HOLD e indica que os barramentos entrarão em estado de alta impedância. O sinal HLDA poderá surgir em T3 no caso de leitura da memória ou dispositivo de entrada, ou no período seguinte ao estado T3 no caso de escrita em memória ou saída para periférico.
INTE	Indica o conteúdo de INTEFF que, quando em nível alto, habilita a CPU de receber interrupções.

(continuação da Tabela III.1)

NOMENCLATURA	F U N Ç Ã O
INT	Sinal de pedido de interrupção. No diagrama de estados fica evidente que a entrada INT é verificada no final da instrução corrente, ou enquanto a CPU está em estado HALT. Nota-se também que a CPU não aceitará o pedido de INT caso esteja em estado HOLD ou se o INTEFF estiver em nível zero.
RESET	Sinal que zera o contador de programa. Após o RESET, o programa começará a partir do endereço zero da memória, os "flip flops" INTE E HLDA são também levados ao nível zero. O Flag, o acumulador, o ponteiro da pilha, os registradores não são zerados.

3.1.1 - ARQUITETURA DO 8080

Pode-se dividir o microprocessador 8080 em cinco unidades funcionais:

a) Conjuntos de registradores e lógica de endereçamento.

b) Unidade Aritmética e Lógica (ALU).

c) Registrador de Instruções e Seção de Contrôle.

d) "Buffer" do barramento de dados.

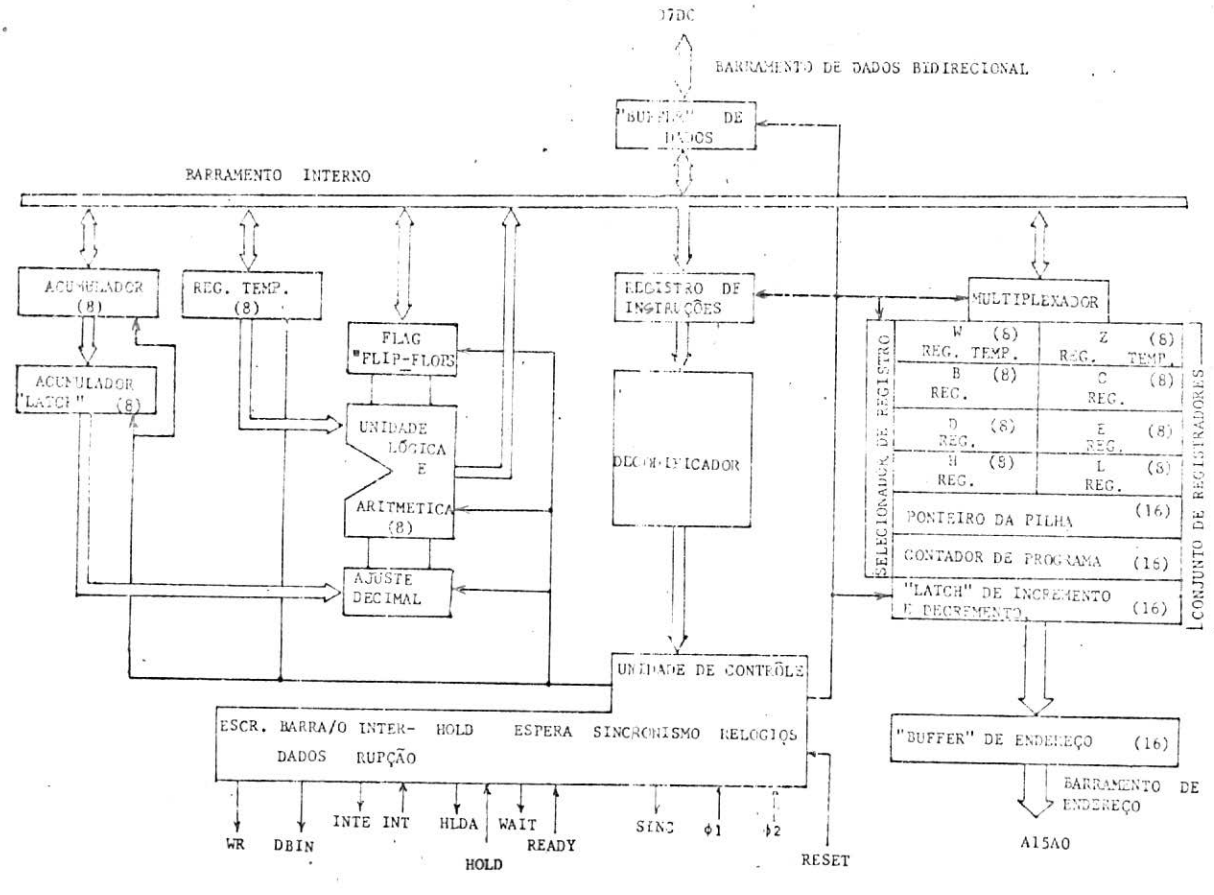


Fig. III.2 - Diagrama de Blocos Funcional da CPU.

3.1.1.1 - REGISTRADORES

Esta seção é constituída por conjunto de memórias estatí cas do tipo de acesso aleatório ("RAM"), organizadas em registradores de 16 bits.

Nomenclatura:

PC - Contador de Programa

SP - Ponteiro de PILHA

B, C, D, e H, L - Constituem os 6 registradores de proposito geral, os quais podem ser referidos em pares.

W, Z - Constituem o registro temporário.

O contador de programa mantém o endereço da próxima ins trução a ser executada. Isto é, no caso de ciclo de busca, o conteúdo de PC é colocado no barramento de endereço e, no período seguinte do relógio, é incrementado automaticamente.

Já o registrador SP mantém o ponteiro da PILHA que poderá ser inicializado com qualquer valor, de modo a utilizar parte da memória principal do sistema como estruturada numa pilha. É necessário ressaltar que a PILHA cresce no sentido de locações decrescentes da memória.

Os registros temporários, W, Z não são endereçáveis por programação e são usados somente para a execução interna das instruções.

Os 8 bits de um registrador podem ser transferidos para o barramento de dados interno, como se pode observar no diagrama funcional apresentado na Figura III.2, através do selecionador de registros. Já os 16 bits podem ser transferidos para o armazenador ("Latch") de endereço ou para o circuito de incremento/decremento. O armazenador de endereço recebe dados de qualquer dos pares de registradores e aciona o "buffer" de endereço, bem como o circuito de incremento/decremento. Este circuito recebe dados do armazenador de endereço e pode enviar seu conteúdo para um dos pares de registradores.

Os dados de 16 bits podem ser incrementados ou decrementados ou simplesmente transferidos entre os registradores.

3.1.1.2 -UNIDADE LÓGICA E ARITMETICA (ALU)

Esta unidade é constituida pelos seguintes registradores:

A - Acumulador de 8 bits

ACT - Acumulador Temporário de 8 bits.

FLAG - Registrador de 6 bits, onde é guardada informação dos sinais: zero, transporte ("carry"), sinal, paridade e transporte auxiliar ("auxiliar carry").

TMP - Registrador temporário de 8 bits.

As operações aritméticas, lógicas e operações de rotação são realizadas na ALU (Unidade Aritmética e Lógica). A ALU recebe de

TMP, de ACT e Transporte; e pode enviar os resultados de uma operação para o acumulador via barramento interno. No diagrama funcional são dadas as transferências possíveis, não havendo portanto necessidade de maiores comentários.

3.1.1.3 - REGISTRADOR DE INSTRUÇÕES E CONTRÔLES

Durante o ciclo de Busca ou Inicialização ("Fetch"), o primeiro byte da instrução código de operação ("OP-CODE") é transferido do "buffer" de dados, via barramento interno, para o registrador de Instrução IR. O código de operação fica então disponível ao circuito decodificador, que provê sinais, combinados com os de tempo, para coordenação da CPU. Além disso, os sinais do decodificador, mais os sinais de controle externo, alimentam a seção de tempos e contrôles que gera os sinais de tempo de estados e de ciclos.

3.1.1.4 - "BUFFER" DO BARRAMENTO DE DADOS

O "buffer" bidirecional de 8 bits, associado ao barramento de dados, é utilizado para se isolar o barramento interno do externo (D7D0). No modo de operação "output", o conteúdo do barramento interno é carregado no armazenador ("latch") que, por sua vez, aciona o "buffer" de saída. O "buffer" de saída é desligado durante as operações de entrada e operações que não necessitam de transferência.

Durante o modo "input", os dados do barramento externo são transferidos para o interno. O barramento interno é recarregado no início de cada estado interno, exceto para o estado de transferência T3, que será descrito posteriormente na seção 3.1.4.

3.1.2 - O CICLO DO PROCESSADOR

A uma instrução é associada uma fase de busca, onde o código de operação é extraído da memória e depositado no registro de instruções IR. Numa outra fase, a de execução, o código de operação em IR é decodificado para fornecer os sinais que comandarão as atividades do processador.

Cada instrução poderá consistir de um a cinco ciclos de máquina. Estes ciclos são necessários a todo acesso à memória ou periféricos. Na fase de busca serão necessário tantos ciclos de máquina quantos forem os bytes que compõem a instrução, enquanto que, na fase de execução, o número do ciclos de máquina é ditado pela instrução obtida. Há instruções que não requerem nenhum ciclo de máquina adicional além dos necessários à busca.

Cada ciclo de máquina poderá consistir de 3, 4 ou 5 estados. Um estado é a menor unidade de atividade do processador e corresponde a duas transições positivas e sucessivas de ϕ_1 (relógio). O 8080 necessita de dois pulsos de relógio ϕ_1 e ϕ_2 não sobrepostos conforme mostra a Figura III.3. O início de cada ciclo de máquina é noticiada pelo 8080, através do sinal SYNC, exceto no segundo e terceiro ciclo de máquina da instrução DAD ("DOUBLE AND") onde o SYNC não ocorre, uma vez que estes ciclos de máquina são usados para operação interna de adição entre os pares de registros especificados.

Há três exceções, em que o estado tem duração indeterminada, isto é, no estado WAIT, no estado HOLD e no estado HALT, que depen

dem exclusivamente de eventos externos à CPU. Porém a duração indeterminada destes estados é sincronizada ao relógio, de modo que todos os estados são multiplos do período de relógio.

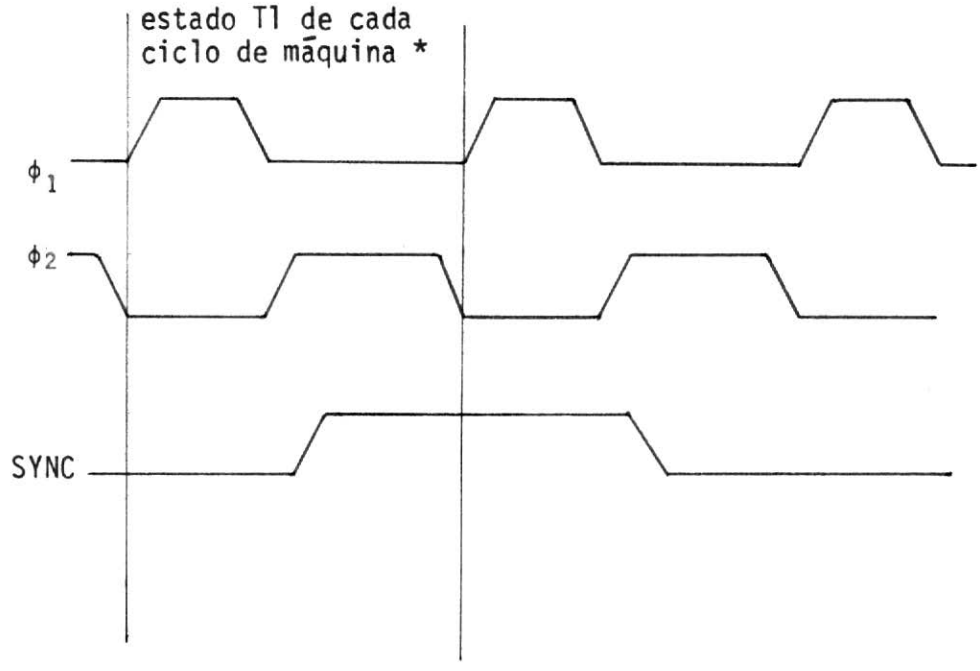


Fig. III.3 - Pulsos de Relógio ϕ_1 e ϕ_2.

3.1.3 - CICLOS DE MÁQUINA

Com a exceção da instrução DAD, um dado ciclo de máquina poderá ser identificado pela informação de estado ("status"), presente no barramento de dados quando o sinal de SYNC ocorrer, isto é, no estado T1 do ciclo.

Dentre os ciclos de máquina possíveis no conjunto de instruções do 8080, podemos classificá-los em:

1. Busca ("fetch")

2. Leitura de memória ("memory read")

3. Escrita em memória ("memory write")

4. Leitura da Pilha ("stack read")

5. Escrita na Pilha ("stack write")

6. Entrada ("input")

7. Saída ("output")

8. Interrupção ("interrupt")

9. Parada ("halt")

10. Interrupção em "halt" ("interrupt while halt")

11. Operação entre registros

Cabe aqui uma observação de que o ciclo (11) foi adicionado para incluir os dois ciclos de máquina da instrução DAD, que não apresentam o sinal SYNC.

A seguir são apresentadas na forma de tabelas, as informações de estado, bem como as definições das nomenclaturas utilizadas e suas correspondências com os bits do barramento de dados.

TABELA III.2

INFORMAÇÕES DE ESTADO DOS DIVERSOS TIPOS DE CICLOS DE MÁQUINA

BITS DO BARRAMENTO DE DADOS	INFORMAÇÕES DE ESTADOS	BUSCA	LEITURA DA MEMORIA	ESCRITA EM MEMÓRIA	LEITURA DA PILHA	ESCRITA NA PILHA	ENTRADA	SAIDA	INTERRUPÇÃO	PARADA	INTERRUPÇÃO EM HALT	OPERAÇÃO ENTRE REGISTROS
		1	2	3	4	5	6	7	8	9	10	11
D0	INTA	0	0	0	0	0	0	0	1	0	1	NÃO APRESENTA
D1	$\overline{w}$ 0	1	1	0	1	0	1	0	1	1	1	
D2	STACK	0	0	0	1	1	0	0	0	0	0	
D3	HLTA	0	0	0	0	0	0	0	0	1	1	
D4	OUT	0	0	0	0	0	0	1	0	0	0	
D5	M1	1	0	0	0	0	0	0	1	0	1	
D6	INP	0	0	0	0	0	1	0	0	0	0	
D7	MEMR	1	1	0	1	0	0	0	0	1	0	

TIPOS DE CICLOS DE MÁQUINA

TABELA III.3

NOMENCLATURA DOS BITS DE INFORMAÇÃO DE ESTADO

NOMEN-CLATURA	BIT DO BARRAMENTO DE DADOS	D E F I N I Ç Õ E S
INTA	DO	Reconhecimento de sinal de interrupção. Este sinal poderá ser utilizada para gatilhar a instrução RST no barramento de dados quando DBIN estiver ativo, isto é,DBIN=1 (nivel alto)
$\overline{WO}$	D1	Indica que a operação no ciclo de máquina corrente será escrita em memória ou saída para periférico quando $\overline{WO}$=0. Caso contrário, será uma leitura de memória ou entrada de periférico.
STACK	D2	Indica que o endereço em A15AO, é um endereço da pilha ("pushdown stack") dado pelo registrador SP, ponteiro da pilha.
HLTA	D3	Reconhecimento de instrução HALT.
OUT	D4	Indica que o endereço em A15AO é de um dispositivo de saída e o barramento de dados conterá os dados de saída quando $\overline{WR}$ estiver ativo.
M1	D5	É um sinal indicando que a CPU está num ciclo de busca ("fetch") do primeiro byte de instrução.
INP	D6	Indica que A15AO contém um endereço de dispositivo de entrada e que os dados de entrada poderão ser colocados no barramentos de dados quando DBIN for ativo.
MEMR	D7	Indica que é barramento de dados será utilizada para leitura de dados da memória.

3.1.4 - SEQUÊNCIA DOS ESTADOS DE TRANSIÇÃO

Cada ciclo de máquina poderá ter de três a cinco estados ativos, isto é, T1, T2, T3, T4, T5 e Tw. O número destes estados, num ciclo de máquina, dependerá da instrução particular que está sendo executada.

A diagrama da Figura III.4, apresenta onde os sinais de READY, HOLD e INT são amostrados e como eles influem na sequência básica dos estados. Posteriormente serão apresentadas as funções destes sinais.

Os estados mencionados anterioremnte não são diretamente indicados pela CPU, porém são fornecidos os sinais de contrôle INTE,HLDA, DBIN, $\overline{WR}$ e WAIT para uso do circuito externo.

O sinal de SYNC, já dito anteriormente, identifica o estado T1 de um ciclo de máquina (exceção feita à instrução DAD). Concomitantemente com o sinal de SYNC, é apresentado no barramento de dados D7D0, a informação de estado. Em T1 é fornecido, ao barramento A15A0, endereço de memória ou de algum periférico.

Em T2, o processador verifica os sinais READY e HOLD e a existência de instrução HALT.

No caso de o processador referenciar a memória ou periférico haverá oportunidade de pedido de espera (WAIT). Esta requisição é feita colocando-se o terminal READY em nível baixo.

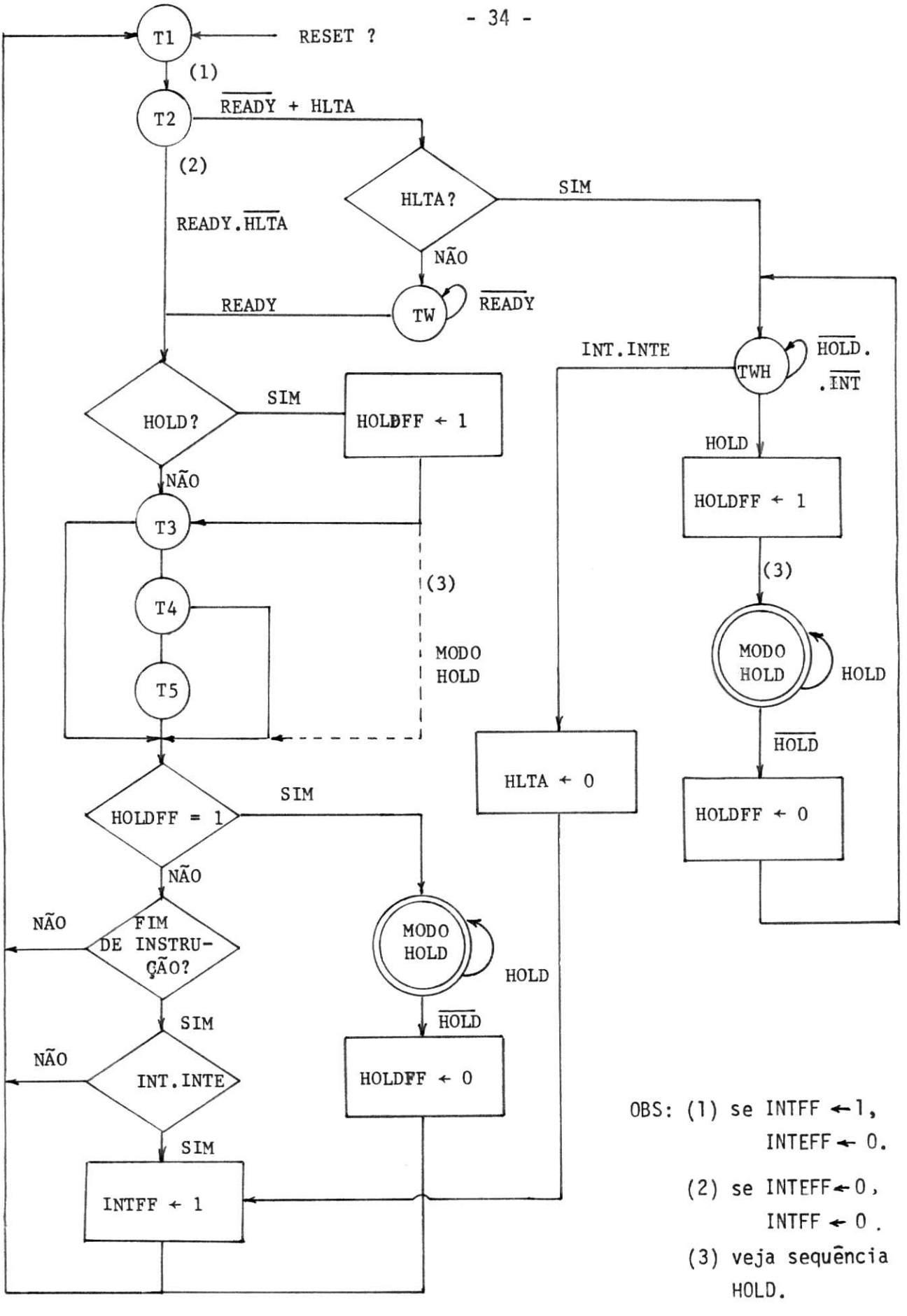


Fig. III.4 - Diagrama de Transição de Estados da CPU.

O processador responde ao pedido de espera entrando no estado Tw, onde permanecerá ocioso por tempo indeterminado, porém multiplo do período de relógio, até que o sinal de READY em nível alto apareça.

Na Figura III.5 é apresentado um ciclo básico do 8080.

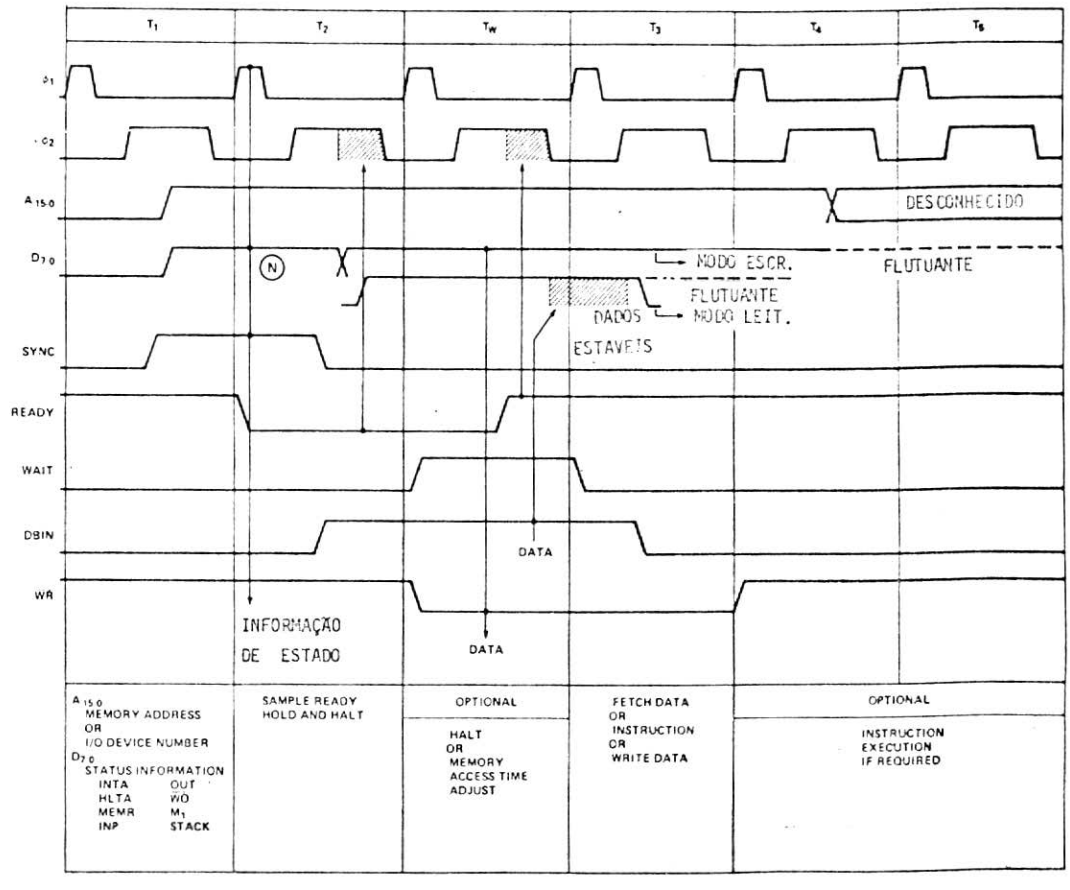


Fig. III.5 - Ciclo Básico do 8080.

Após o estado T3, fica muito difícil a generalização e todos os eventos que ocorrem, a partir de então, dependem do ciclo de maquina corrente. Sendo que os estados T4 e T5 são opcionais e somente utilizados em função da necessidade de cada instrução. Após o estado T5, o processador retorna diretamente ao estado T1 do ciclo de máquina seguinte.

TABELA III.4

ATIVIDADES ASSOCIADAS AOS ESTADOS DE UM CICLO DE MÁQUINA

ESTADO	ATIVIDADE ASSOCIADA
T1	É colocado no barramento de endereços A15A0, o endereço de alguma posição de memória ou de algum periférico. A informação de Estado é colocada no barramento de dados D7D0
T2	Verificação das entradas READY e HOLD, e a existência de instrução HALT
Tw (opcional)	O processador entra em estado de espera se READY tem nível baixo ou se ocorreu instrução de HALT
T3	Os dados entram no processador via barramento D7D0 nos ciclos de busca de leitura da memória, de leitura da pilha, de entrada ou de interrupção, enquanto que os dados são colocados como saida nos ciclos de escrita em memória, escrita na pilha ou de saida para periférico
T4 e T5 (opcionais)	Estes estados ficam a disposição das necessidades de cada instrução. Caso não sejam requisitadas estes estados podem ser saltados. São utilizadass somente para operações internas do processador

3.1.5 - A SEQUÊNCIA DA INTERRUPÇÃO

O 8080 tem capacidade inerente de manipular interrupções. O pedido de interrupção por um periférico é feito através da entrada INT.

A verificação do INT, como se pode observar no diagrama de estados na Figura III.4, é feita somente no final da instrução e só será aceita se INTE tem nível alto.

O ciclo de interrupção é muito semelhante ao ciclo de busca, na maioria dos aspectos. Na informação do estado ("status") é transmitido o bit M1 de maneira usual, porém é acrescentado o bit INTA, que notifica o circuito externo da aceitação da interrupção. É jogado em T1, no barramento de endereços, o conteúdo do contador de programa, porém, durante o ciclo de interrupção, o contador de programa não é incrementado.

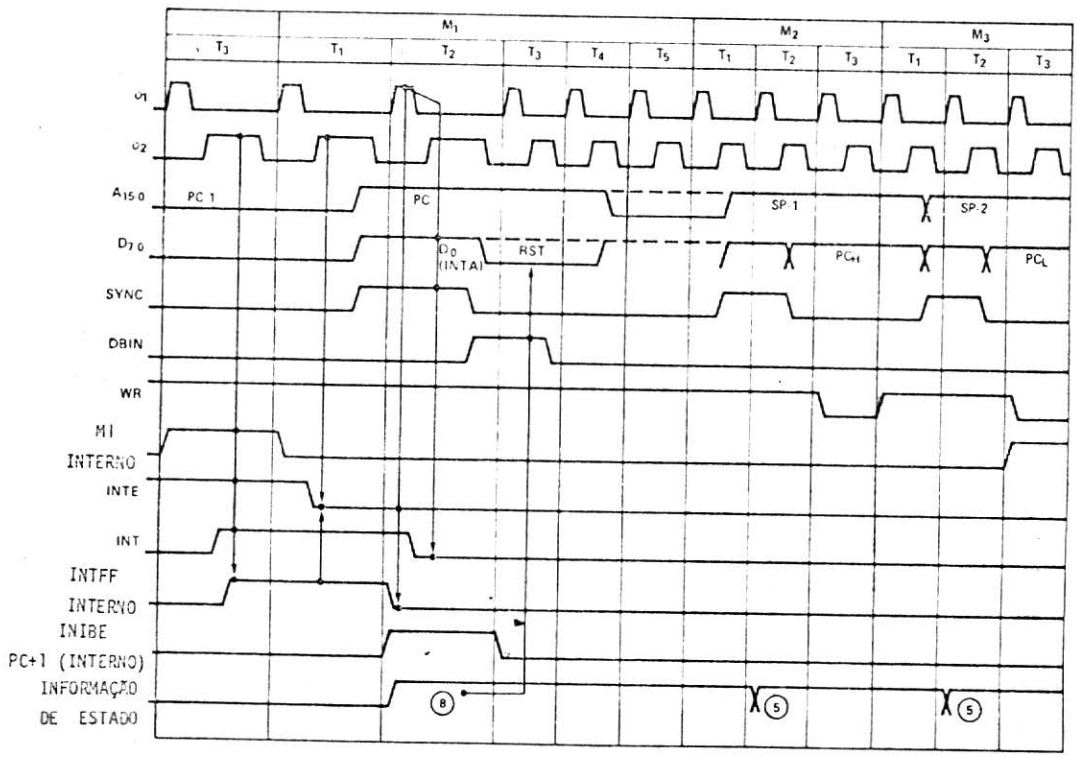
Deste modo o contador de programa é preservado, podendo ser utilizado posteriormente ao final do processamento da interrupção.

É função da lógica externa colocar, em T3, uma instrução de RST de máquina no barramento de dados. Isto, num sistema típico, significa que o barramento de dados será controlado pelo dispositivo que solicitou a interrupção.

No conjunto de instruções, RST possibilita a chamada de uma das oito posições fixas na memória, para alocar subrotinas de serviço de interrupções. Ainda nesta instrução RST, o valor anterior do contador de programa é armazenado na Pilha.

Os endereços dessas alocações em decimal são: 0, 8, 16,24, 32, 40, 48 e 56.

A Figura III.6 apresenta um pedido de interrupção enquanto INTE está em nível alto ("enable").



NOTA: (N) vide Tabela III.5.

Fig. III.6 - Diagrama de Tempo de uma Sequência de Interrupção.

3.1.6 - A SEQUÊNCIA DE HOLD

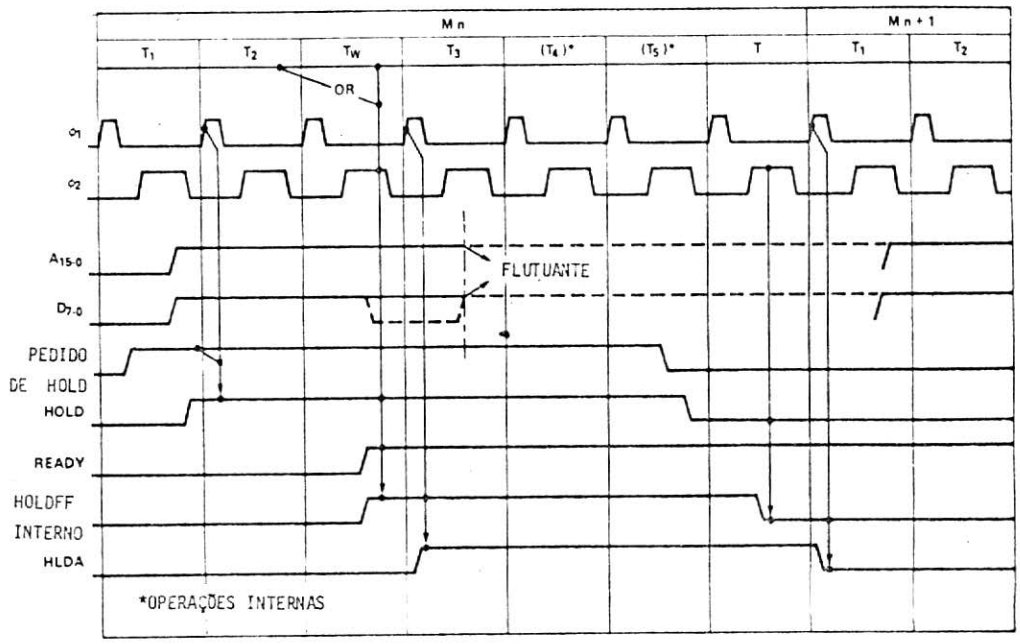
Aplicando-se nível alto em HOLD, um dispositivo externo pode suspender as atividades normais da CPU. O pedido dessa espécie é aceito pela CPU, que coloca os seus barramentos de dados e endereço em estado flutuante, de modo que o dispositivo requisitante de HOLD tenha total contrôle sobre os barramentos. A saída HLDA é colocada em nível alto possibilitando, destarte, o acesso direto a memória (DMA).

Aparentemente, o processador tem suas operações suspensas, uma vez que os barramentos de dados e endereços estão flutuantes. Porém nada impede a execução de operações internas a CPU. Esse processamento interno permite portanto um paralelismo com operações de transferência externa (DMA). A atividade da CPU não cessa enquanto não terminar de executar o ciclo de máquina corrente atingida pela HOLD.

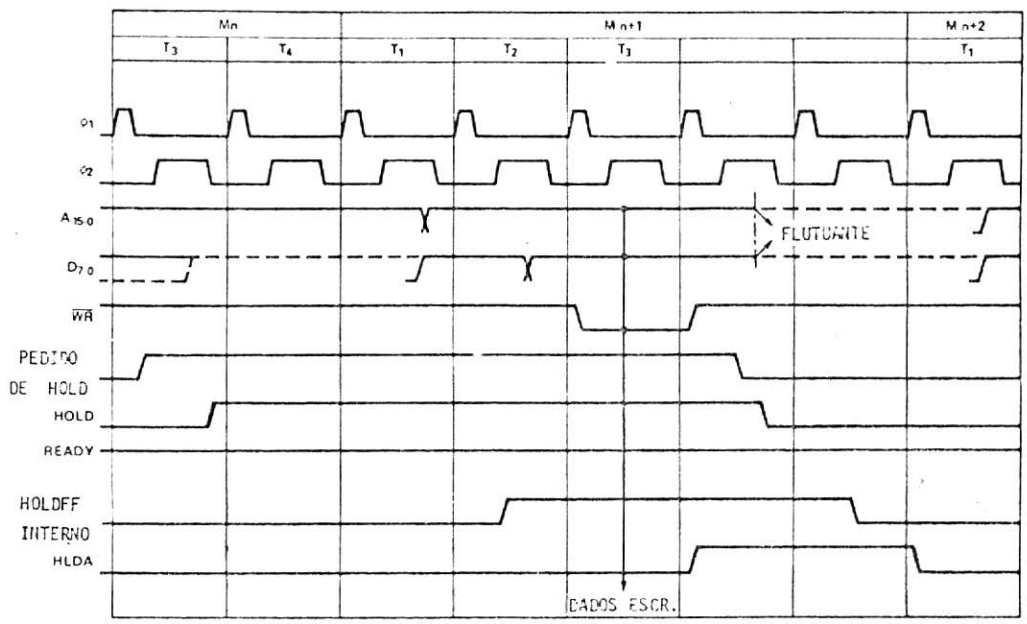
A saída do processador do estado HOLD é similar à maneira de sua entrada. O pedido de HOLD é terminado quando o dispositivo completou as transferências de dados. HLDA retorna então ao nível baixo e a CPU continua seu processamento normal, no próximo ciclo de máquina a ser executada.

3.1.7 - SEQUÊNCIA DE HALT

Se uma instrução de HALT for executada, o processador entra no estado HALT (TWH) após o estado T2 do próximo ciclo de máquina.



(a) - Sequência HOLD (MODO LEITURA)



(b) - Sequência HOLD (MODO ESCRITA)

Fig. III.7 - Diagramas de Tempo da Sequência HOLD.

Observando-se o diagrama de estados da Figura III.3.A, verifica-se que há somente três maneiras de sair deste estado de HALT:

1) Através de RESET, onde a CPU é jogada no estado T1, com o contador de programa zerado (vide Tabela III.1).

2) Através de HOLD, já descrito anteriormente. Após a saída do estado HOLD, o processador entra novamente em estado de HALT.

3) Através de Interrupção colocando-se INT em nível 1 enquanto INTE está habilitado, o que levará o processador ao estado T1, para execução da rotina de Interrupção. Note que ao entrar em estado de HALT, INTE deve estar no nível 1, caso contrário a saída de HALT, só será possível através de RESET.

A Figura III.8 relaciona as entradas HOLD, INT e HALT. E na Figura III.9 é apresentada um diagrama de blocos da sequência HALT. Verifica-se também,na Figura III.9, evidência de prioridade quanto à verificação dos sinais de RESET, HOLD e INT.

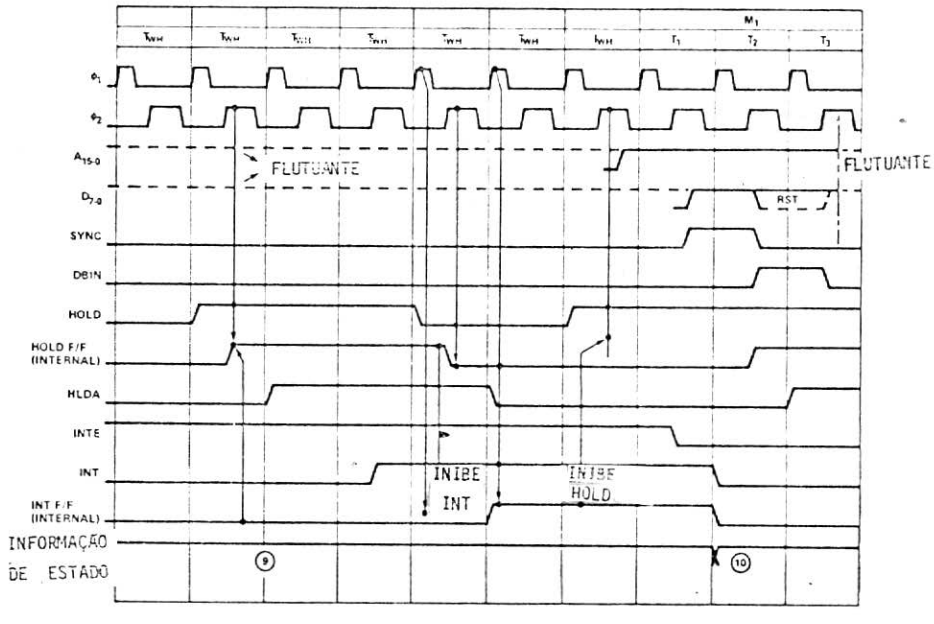


Fig.III.8 - Relação entre HOLD e INT no Estado HALT.

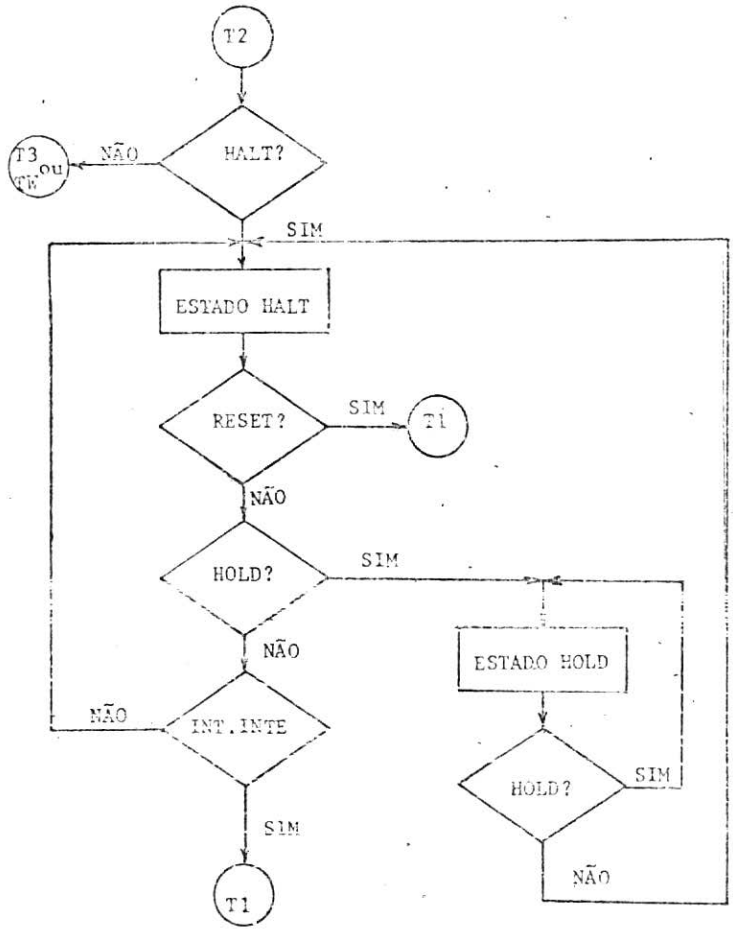


Fig. III.9 - Diagrama de Blocos da Sequência HALT.

3.1.8 - FORMATO DOS DADOS E DAS INSTRUÇÕES

Os dados na 8080 são armazenados em conjuntos de 8 bits conforme mostra a Figura III.10, onde o bit D7 é o mais significativo.

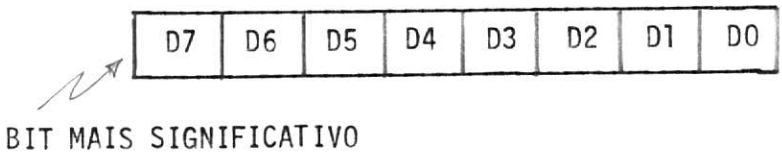


Fig. III.10 - Formato dos Dados.

Uma instrução pode ter 1, 2 ou 3 bytes de comprimento. Estes bytes são armazenados em posições sucessivos na memória, de acordo com o formato apresentado na Figura III.11.

Instrução 1 byte

D7	D6	D5	D4	D3	D2	D1	D0

Instrução de 2 bytes

D7	D6	D5	D4	D3	D2	D1	D0	código de operação
D7	D6	D5	D4	D3	D2	D1	D0	operando

Instrução de 3 bytes

D7	D6	D5	D4	D3	D2	D1	D0	código de operação
D7	D6	D5	D4	D3	D2	D1	D0	operando 1 (bits menos significativos)
D7	D6	D5	D4	D3	D2	D1	D0	operando 2 (bits mais significativos)

Fig. III.11 - Formato das Instruções.

3.1.9 - COMENTÁRIOS ADICIONAIS

Quando o sistema 8080 é ligado, devido aos fatores aleatórios, os conteúdos dos registradores, contador de programa e Ponteiro de Pilha são ignorados. Desta forma, é conveniente ao se ligar o sistema começar com um RESET. O sinal de RESET zera o contador de programa e leva ainda os "flip-flops" INTE e HLDA ao nível zero. Porém os "flags" de estado ("status") ou qualquer outro registrador (acumulador, registradores e ponteiro de pilha) não são afetados; seus conteúdos permanecem indeterminados até que sejam inicializadas por programação.

Há sistemas que requerem o estado de espera para o processador, após o seu acionamento. Para tanto , nas duas primeiras posições de memória devem ser colocadas as instruções respectivamente EI e HLT,uma vez que, dado o RESET, o programa é iniciado pela posição zero de memória, com INTE no estado 0. Assim, uma interrupção manual ou automática poderá ser utilizada para o início de processamento.

3.2 - SIM8080 - A ROTINA DE SIMULAÇÃO

A rotina SIM8080 é basicamente um implementação, em ALGOL, do diagrama de estados, em cada ciclo de máquina do microprocessador, conforme o diagrama apresentado na Figura III.4.

Desta maneira, a simulação do microprocessador é realizada a nível de cada estado, ou seja, cada estado (dos 5 possíveis: T1, T2, T3, T4 e T5) é simulado como uma unidade indivisível. Em consequência cada instrução de máquina é simulada através dos ciclos de máquina que lhe estão associados sendo que estes, por sua vez, são simulados através de uma sequência de estados no tempo. Assim sendo, a simulação permite o monitoramento do microprocessador a um nível de detalhe bem maior que o normal, nível este necessário principalmente em aplicações em tempo real, onde o projeto do sistema de atendimento de interrupções é uma das tarefas principais e de mais difícil verificação.

Assim, a partir deste enfoque, a rotina SIM8080 simula exatamente, estado a estado, o funcionamento do microprocessador INTEL 8080, de modo análogo a uma máquina de estados finita. As instruções de máquina são executadas através de uma sequência de ciclos no tempo, ciclos estes pertencentes a um conjunto de onze ciclos distintos. Por sua vez cada ciclo é executado através de uma sequência de estados, que varia entre um número mínimo de três estados a um número máximo de cinco estados. Cada um desses estados possíveis é "realizado" através de uma sequência de microoperações, sequência esta, em parte, função da instrução de máquina que está sendo realizada no momento.

Cada conjunto de microoperações, que "realiza" um estado da máquina, é executado num intervalo de tempo correspondente a um período do sinal de contrôle básico ("clock") da máquina. Assim sendo na rotina SIM8080, cada estado é simulado num intervalo de tempo lógico indivisível e as operações de sincronização com eventos externos (por ex. sinais de interrupção simulados) são realizadas no fim de cada estado.

Desta maneira, para cada instrução armazenou-se a sequência de ciclos correspondentes uma estrutura identificada por "TABELAIR".

Para os estados iniciais (T1 e T2) de cada ciclo de máquina, verificou-se que, com pequenas variações, apresentam características gerais comuns possibilitando, desta forma, a simulação destes estados por meio de subrotinas específicas.

Contudo, os estados T3, T4 e T5, não apresentam tais características. As operações associadas a cada um desses estados são funções

exclusivas da instrução simulada. Sendo assim, a sequência de microoperações, envolvidas nos estados T3, T4 e T5, associados a cada ciclo de uma instrução, foi armazenada numa estrutura denominada MICROPROGRAMA.

E no caso de certas instruções, que apresentam processamento paralelo, no qual aquelas transferências entre registros internos da CPU, que são realizadas nos estados T1 e T2 da instrução seguinte, foram simuladas colocando-as no estado T1 da próxima instrução.

Desta maneira, para cada instrução,num dado ciclo, o simulador (imitando o microprocessador) percorre os estados apresentados no diagrama da Figura III.12, de maneira simplificada.

No estado T1, o microprocessador coloca no seu barramento de endereço A15A0, o endereço de alguma posição de memória ou de algum periférico e, no seu barramento de dados D7D0, a informação de ESTADO ("status") do ciclo corrente. Excetuando a instrução DAD, o sinal SYNC identifica o estado T1 do ciclo.

Como dito anteriormente, para os estados T1 e T2 foram implementadas subrotinas específicas, as quais são descritas na seção 3.2.2

Ao final de T2 são realizados testes sôbre as entradas READY e HOLD e verificação de execução de instrução HLT.

Caso o ciclo seja de busca de instrução (CICLO = 0 ou CICLO = 11 ou CICLO = 13) é chamada a sub-rotina DECODIFICADOR, que procu

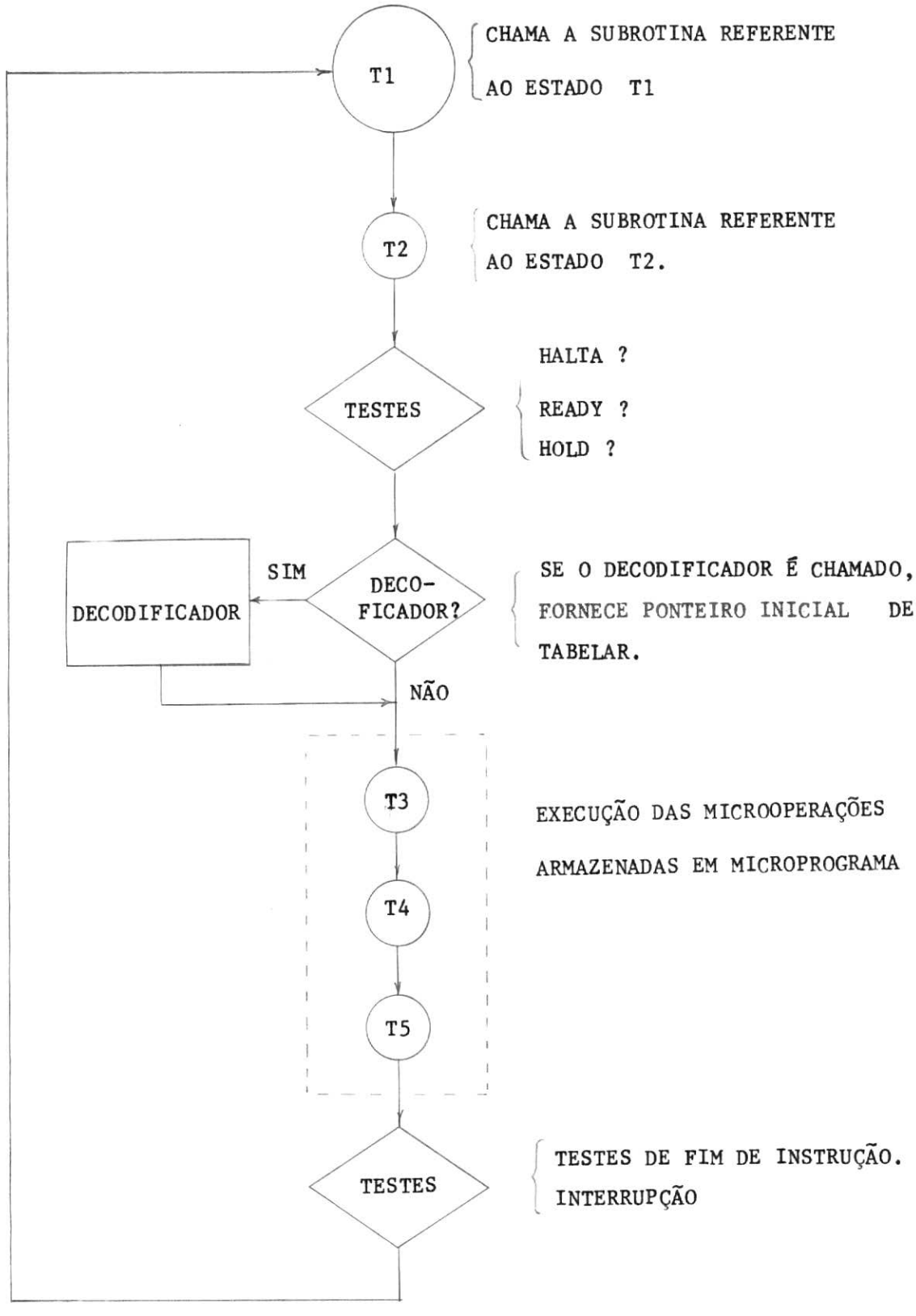


Fig.III.12 - Diagrama de Estados Simplificado.

ra identificar o código de operação colocado em IR Se o DECODIFICADOR reconhece esta configuração de bits, o mesmo fornece um ponteiro inicial para a estrutura de armazenamento de ciclos TABELAIR.

Para cada ciclo, é fornecido um endereço inicial da sequência de estados armazenada em microprograma executando, desta forma, as operações referentes aos estados T3, T4 e T5.

A sincronização de todo o simulador é feita a partir de rotinas especificas, que atuam na transição de um estado para outro (INCRCLOCK, INCRCYCLE e INCRINST). Nestas rotinas são realizados testes e verificação de pedido de interrupção, de pedido de "HOLD", etc.

Desta forma, através de transições sucessivas de estados e de ciclos, uma instrução é simulada.

3.2.1 - TIPOS DE CICLOS DE MÁQUINA

A numeração correspondente a cada ciclo de máquina, fornecida no manual do usuário, foi alterada para fins de simulação. Devido à possibilidade de se associar, aos estados T1 e T2 de um ciclo, subrotinas especificas, dadas as características comuns houve, para um dado tipo de ciclo, a necessidade de se implementar mais de uma subrotina, associada a cada um desses estados, e ativadas de acordo com a instrução de máquina a ser executada.

Na Tabela III.5 é apresentada a numeração associada a cada conjunto T1 e T2, de um ciclo de máquina, utilizada na simulação.

TABELA III.5

TIPOS DE CICLOS DE MAQUINA E SUBROTINAS DOS ESTADOS COMUNS

NUMERO DO CICLO	ESTADO	SUBROTINA ASSOCIADA	INFORMAÇÃO DE ESTADO	TIPO DE CICLO
			D7 D6 D5 D4 D3 D2 D1 D0	
0	T1	FETCHT1		Busca de instrução ("FETCH")
	T2	FETCHT2	0 1 0 0 0 1 0 1	
1	T1	MEMREADT1A		
	T2	MEMREADT2A		
2	T1	MEMREADT1B	1 0 0 0 0 0 1 0	Leitura de Memória
3	T1	MEMREADT1C		
2/3	T2	MEMREADT2BC		
4	T1	MEMWRITET1A		
5	T1	MEMWRITET1B	0 0 0 0 0 0 0 0	Escrita na memória
6	T1	MEMWRITET1C		
4/5/6	T2	MEMWRITET2ABC		
7	T1	STACKREADT1	1 0 0 0 0 1 1 0	Leitura da pilha
	T2	STACKREADT2		
8	T1	STACHWRITET1	0 0 0 0 0 1 0 0	Escrita na pilha
	T2	STACKWRITET2		
9	T1	INPUTREADT1	0 1 0 0 0 0 1 0	Leitura de periférico
	T2	INPUTREADT2		
10	T1	OUTPUTWRITET1	0 0 0 1 0 0 0 0	Escrita em periférico
	T2	OUTPUTWRITET2		
11	T1	INTERRUPTACKT1	0 0 1 0 0 0 1 1	Interrupção
	T2	INTERRUPTACKT2		
12	T1	HALTACKT1	1 0 0 0 1 0 1 0	Parada ("HALT")
	T2	HALTACKT2		
13	T1	INTERACKHALTT1	0 0 1 0 1 0 0 1	Interrupção no estado de parada
	T2	INTERACKHALTT2		
14	T1	REGISOPT1	não apresenta	Operação entre registros
	T2	REGISOPT2		

Na Tabela III.6 são apresentadas as microoperações realizadas nas subrotinas de simulação dos estados T1 e T2.

3.2.2 - TABELAIR

É uma estrutura,como mostra a Figura III.13,onde é armazenada a informação de sequência de ciclos para cada uma das instruções simuladas.

Uma vez obtido o código de operação de uma instrução, num ciclo de busca, o decodificador atua de modo a identificá-lo fornecendo o ponteiro PIR. Este ponteiro determina a posição inicial da sequência de ciclos da instrução reconhecida em TABELAIR.

TABELA III.6

MICROOPERAÇÕES REALIZADAS NAS SUBROTINAS DOS ESTADOS T1 E T2

CICLO	ESTADOT1	ESTADOT2
0	FETCHT1 1 : ADDRBUFFER ← PC 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"10100010" 4 : ALUREG ← 0 *5 : MACRO1	FETCHT2 1 : D7D0XXX - FLUTUANTE 2 : PC ← PC + 1 3 : DBIN ← 1 4 : SYNC ← 0
1	MEMREADT1A 1 : ADDRBUFFER ← PC 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"10000010" 4 : MACRO1	MEMREADT2A 1 : D7D0XXX-FLUTUANTE 2 : PC ← PC + 1 3 : DBIN ← 1 4 : SYNC ← 0
2	MEMREADT1B **1 : SE IR5 = 0 ENTAO SE IR54 00 : ADDRBUFFER ←BC 01 : ADDRBUFFER ←DE ***10 : ERROPROC 11 : ERROPROC CASO CONTRARIO ADDRBUFFER ← WZ 2: A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"10000010" 4 : MACRO1	MEMREADT2B ***1 : SE IR5 = 1 ENTAO SE HÁ PROXIMO CICLO : WZ ← WZ + 1 2 : D7D0XXX - FLUTUANTE 3 : DBIN ← 1 4 : SYNC ← 0
3	MEMREADT1C 1 : ADDRBUFFER ← HL 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"10000010" 4:: MACRO1	
4	MEMWRITET1A 1 : ADDRBUFFER ← HL 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"00000000" 4 : MACRO1	MEMWRITET2 1 : SE HA PROXIMO CICLO ENTAO WZ ← WZ + 1 2 : D7D0XXX - FLUTUANTE 3 : SYNC ← 0
5	MEMWRITET1B 1 : ADDRBUFFER ← WZ 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"00000000" 4 : MACRO1	

continuação de Tabela III.6

CICLO	ESTADOT1	ESTADOT2
6	MEMWRITET1C 1 : CASO IR54 00 : ADDRBUFFER ← BC 01 : ADDRBUFFER ← DE **10 : ERROPROC **11 : ERROPROC 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"00000000" 4 : MACRO1	
7	STACKREADT1 1 : ADDRBUFFER ← SP 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"10000110" 4 : MACRO1	STACKREADT2 1 : D700XXX - FLUTUANTE 2 : SE IR543210 NEQ 1"100011" ENTÃO SP ← SP + 1 3 : SYNC ← 0 4 : DBIN ← 1
8	STACKWRITET1 1 : ADDRBUFFER ← SP 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"00000100" 4 : MACRO1	STACKWRITET2 1 : SE HA PROXIMO CICLO ENTÃO SP ← SP - 1 2 : D7D0XXX - FLUTUANTE 3 : SYNC ← 0
9	INPUTREADT1 1 : ADDRBUFFER ← WZ 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"oloooolo" 4 : MACRO1	INPUTREADT2 1 : DBIN ← 1 2 : SYNC ← 0
10	OUTPUTWRITET1 1 : ADDRBUFFER ← WZ 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"00010000" 4 : MACRO1	OUTPUTWRITET2 1 : DBIN ← 1 2 : SYNC ← 0
11	INTERRUPTACKT1 1 : ADDRBUFFER ← PC 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"11000100" 4 : SE INTFF = 1 ENTÃO INTEFF ← 0 CASO CONTRARIO ERRO 5 : MACRO1	INTERRUPTACKT2 1 : DBIN ← 1 2 : SYNC ← 0 3 : SE INTEFF = 0 ENTÃO INTFF ← 0 4 : D7D0XXX - FLUTUANTE
12	HALTACKT1 1 : ADDRBUFFER ← PC 2 : A15A0 ← ADDRBUFFER 3 : D7D0 ← 1"10001010" 4 : MACRO1 5 : HLTA ← 1	HALTACKT2 1 : SYNC ← 0 2 : D7D0XXX - FLUTUANTE

continuação da Tabela III.6

CICLO	ESTADOT1	ESTADOT2
T3	INTERACKHALTT1 1 : ADDRBUFFER ← PC 2 : A15AO ← ADDRBUFFER 3 : D7DO ← 1"10010100" 4 : SE INTFF = 0 ENTÃO INTEFF ← 0 CASO CONTRARIO : ERRO ← SIMULAÇAO SUSPENSA	INTERACKHALTT2 1 : DBIN ← TRUE 2 : SE INTEFF = 0 ENTÃO INTFF ← 0 3 : SYNC ← 0 4 : D7DOXXX - FLUTUANTE
T4	REGISOPT1 1 : SE HÁ PROXIMO CICLO ENTÃO IR54 00 : ACT ← C 01 : ACT ← E 10 : ACT ← L 11 : ACT ← SPL CASO CONTRARIO IR54 00 : ACT ← B 01 : ACT ← D 10 : ACT ← H 11 : ACT ← SPH	REGISOPT2 1 : SE HA PROXIMO CICLO ENTÃO TMP ← L ALUAUX ← ACT + TMP ALUREG ← ALUAUX CASO CONTRARIO TMP ← H ALUAUX ← ACT + TMP + CY ALUREG ← ALUAUX

NOTA

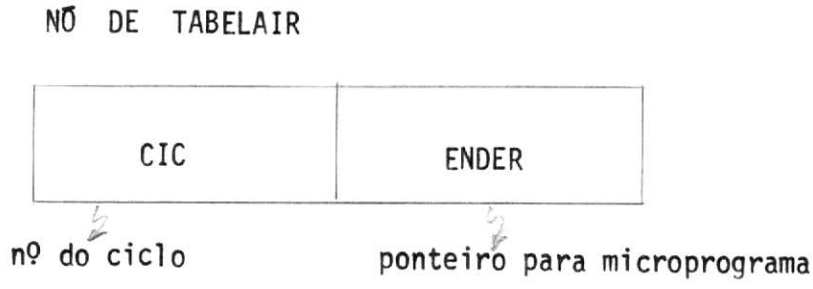
* MACRO1 é definida como um conjunto de operações comuns

MACRO1 { WRBARRA ← 1; DBIN ← 0; HLDA ← 0; WAIT ← 0; SYNC ← 1 }

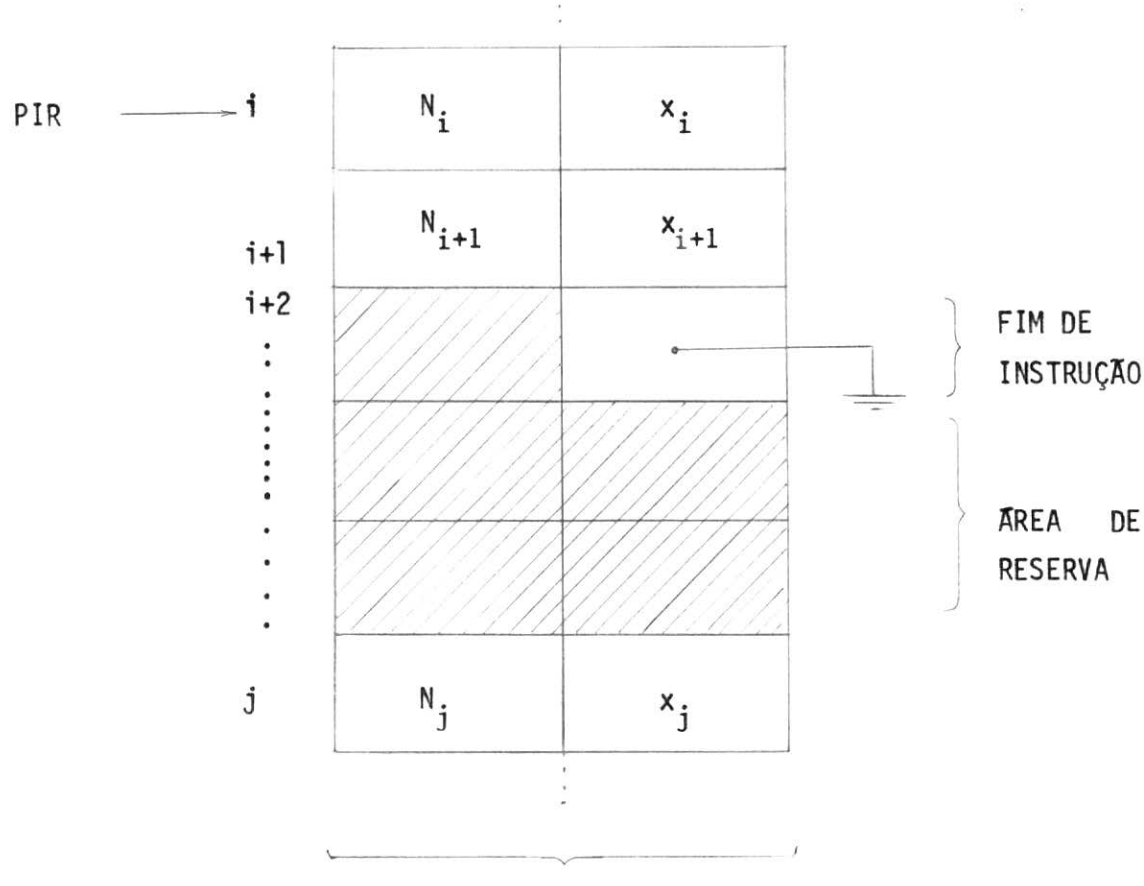
** IRS - Representa o bit numero 5 (D5) do IR

*** ERROPROC mensagem de erro na microprogramação da TABELAIR

A verificação da existência de próximo ciclo é feita pela subrotina PROXCIC(HACIC), onde o parametro formal HACIC é uma booleana. A subrotina consulta TABELAIR e,caso seja encontrado o próximo ciclo, HACIC é verdadeiro.



(a)



(b)

Fig. III.13 - (a) Representação de um nó de TABELAIR

(b) Estrutura TABELAIR

Na Figura III.13(a), é apresentado esquematicamente um nó de TABELAIR, na qual o campo CIC armazena o numero de um dos ciclos constante na Tabela III.5. O campo ENDER fornece um ponteiro para a sequência de estados em MICROPROGRAMA, descrita na seção 3.2.3.

Terminada a simulação de um ciclo do processador, o ponteiro PIR é incrementado, de modo que PIR aponta para o proximo ciclo a ser simulado. Desta maneira, a sequência de ciclos é percorrida e simulada.

A instrução termina quando o nó de sinalização de fim de instrução for apontado por PIR:

Como a estrutura de TABELAIR é sequencial, na implementação ao final da sequência de ciclos, deixou-se uma área de reserva constituída de alguns nós. Isto permite rapida alteração de TABELAIR, sem que seja necessária a modificação dos valores iniciais de PIR fornecida pelo Decodificador. A Figura III.13(b) mostra em TABELAIR essa área de reserva, bem como um nó de sinalização de fim de instrução.

A implementação de TABELAIR foi num "VALUE ARRAY", declaração da linguagem ALGOL(B-6700/B-7700 - ALGOL Language Reference Manual - Burroughs) que simula uma memória só de leitura ("ROM"). Além disso, como a palavra do B-6700 é de 48 bits, utilizou-se palavras parciais para constituir o nó de TABELAIR. O campo CIC é constituida de 8 bits, enquanto que o campo ENDER é de 16 bits. Deste modo, dois nós de TABELAIR foram acomodados numa palavra do B-6700.

Como o "VALUE ARRAY" requer uma forma adequada para o armazenamento desses dados, foi utilizado um programa auxiliar denominado MONTA, que, a partir de um formato mais livre dos cartões de dados, converte-o na forma adequada, gerando a tabela na forma compactada apresentada acima.

No Apêndice A, é apresentada a listagem dos cartões de entrada de MONTA, e o formato gerado para TABELAIR.

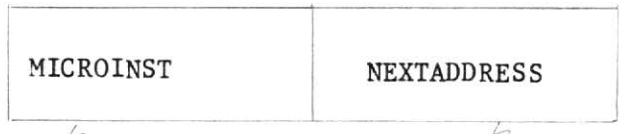
Assim, qualquer modificação ou acréscimo nas sequências de ciclos poderá ser introduzida no sistema, bastando, para tanto, utilizar a rotina MONTA e recompilar o programa de simulação.

3.2.3 - MICROPROGRAMA

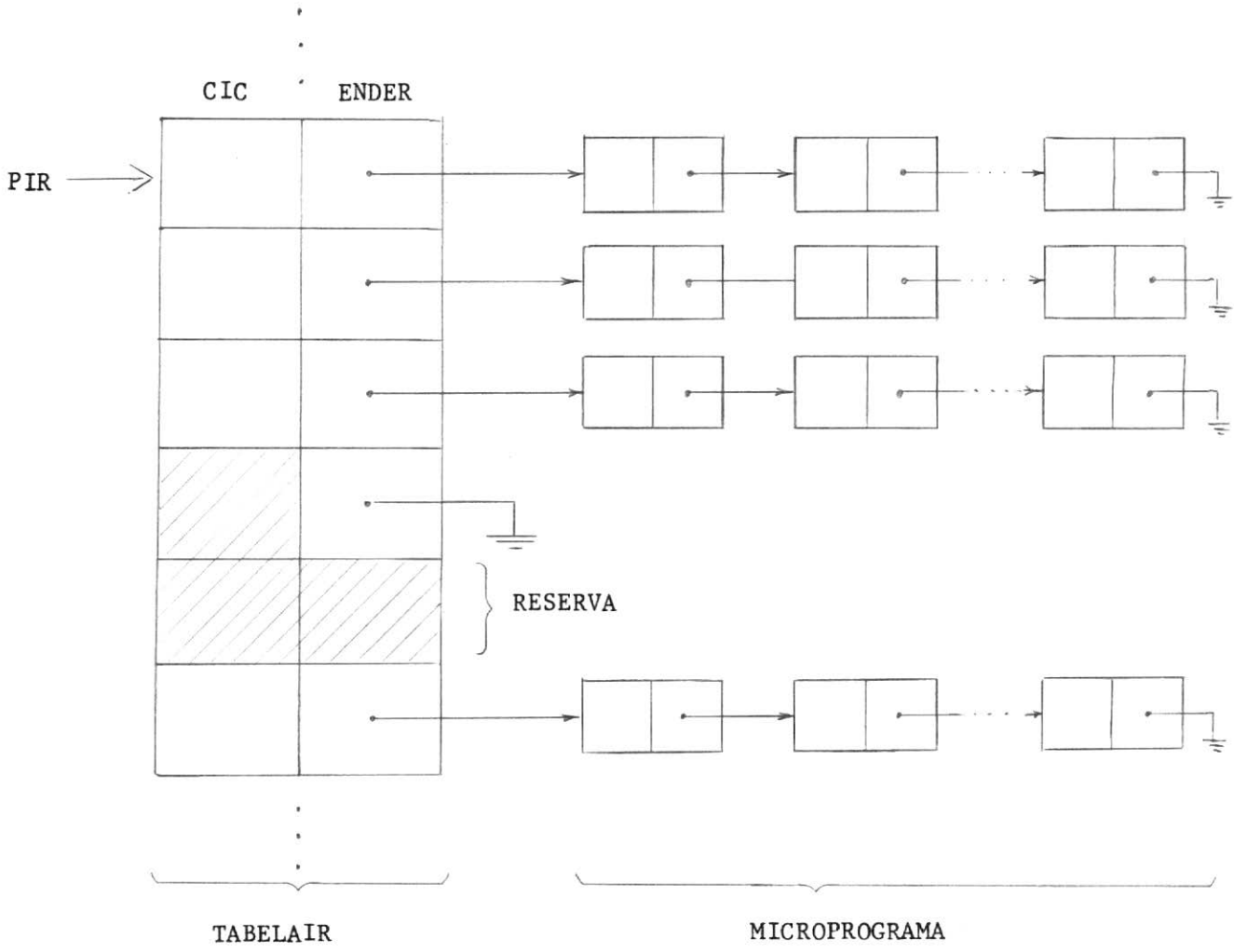
Microprograma é conjunto de listas associadas à TABELAIR, de modo que cada lista armazena a sequência de microoperações correspondentes a cada estado T3, T4 e T5 de um ciclo de máquina.

Uma vez determinada a sequência de ciclos de máquina referente a uma dada instrução, para cada ciclo corresponde um nó de TABELAIR, cujo campo ENDER, contém o ponteiro para o início da sequência de microoperações a serem realizadas no ciclo.

A Figura III.14(a) representa um nó de microprograma, com a designação de seus campos. O campo MICROINSTRUÇÃO contém o número de uma das microoperações apresentadas na tabela. O campo NEXTADDRESS, guarda o ponteiro para o nó, em microprograma que contém a próxima microope



(a) - Nó de Microprograma



(b) - Estrutura Microprograma e sua Relação com TABELAIR

Fig. III.14 - Microprograma

ração. O ciclo é simulado através do percorrimento da lista até que se encontre o nó de sinalização de final da sequência de microoperações. A sinalização é feita através do campo NEXTDDRESS que, quando for nulo, indica o fim do ciclo. Então o PIR é incrementado a um novo ciclo se inicia.

Em cada ciclo os estados são delimitados através de nós especiais que ativam microoperações responsáveis pela chamada das rotinas de sincronização.

A Figura III.14 apresenta a estruturação de microprograma e seu relacionamento com TABELAIR.

A Figura III.15 mostra a instrução SHLD, com suas sequências de ciclos e estados utilizadas para a sua simulação, bem como trecho de TABELAIR e MICROPROGRAMA em SHLD.

Procedimentos análogos à SHLD foram utilizados em cada uma das instruções, de modo a obter as estruturas finais de TABELAIR e MICROPROGRAMA, apresentadas nos Apêndice A e B respectivamente.

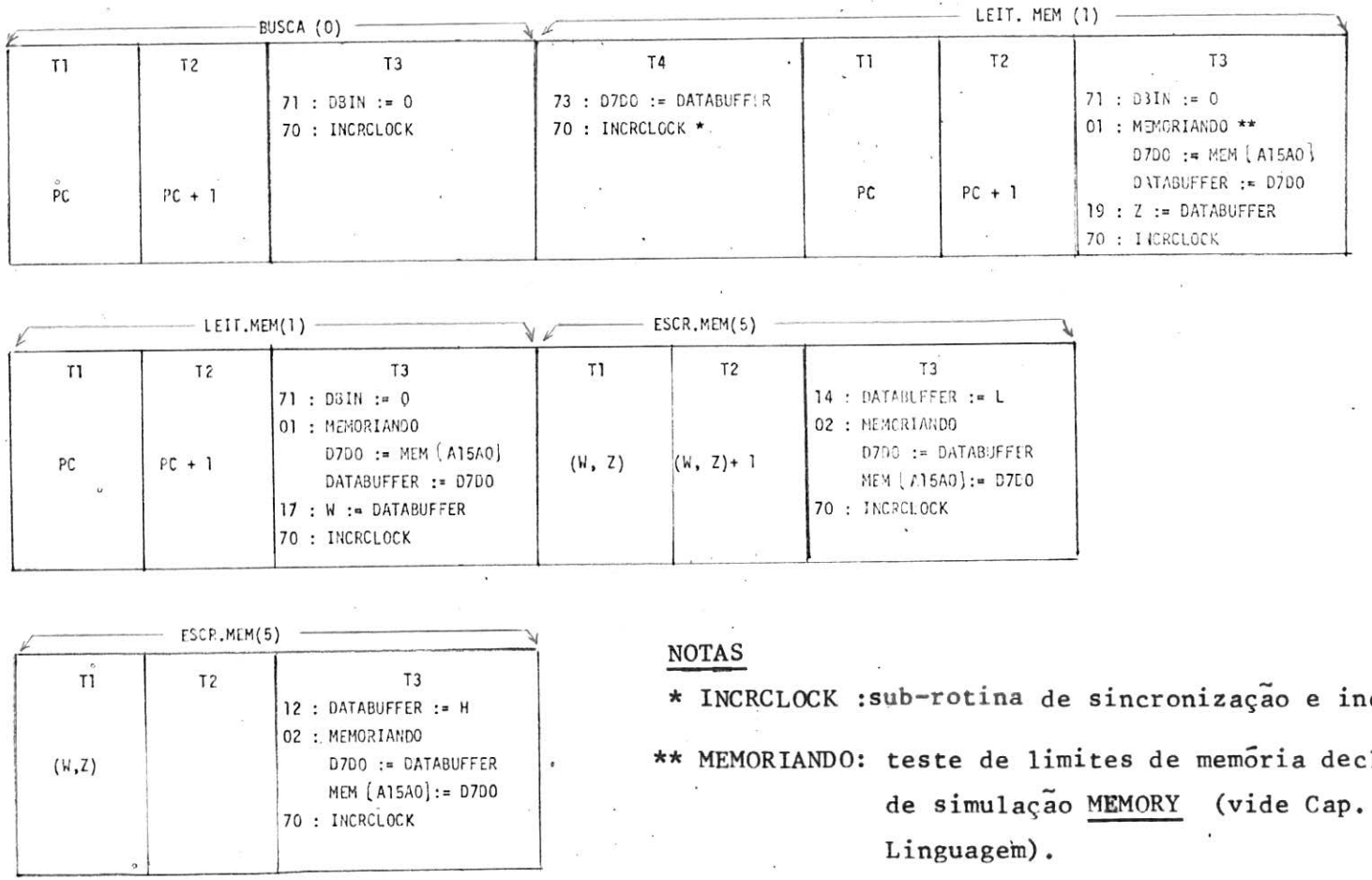
De maneira análoga a TABELAIR, MICROPROGRAMA foi armazenado num "VALUE ARRAY" e ainda, através de palavras parciais, dois nós de MICROPROGRAMA podem ser colocados numa palavra do computador B-6700.

Valem, também, as mesmas considerações, quanto à rotina MONTA mencionada em 3.2.2.

Instrução SHLD <B_2> <B_3> Armazena o conteúdo dos registradores H e L nas posições de memória endereçadas pelos bytes <B_2> e <B_3> da instrução. |<B_3><B_2>| ← (L), |<B_3><B_2>+ 1| ← (H)

Código de Operação: 00 100 010

PIR inicial dado pelo decodificador := 27



NOTAS

* INCRCLOCK :sub-rotina de sincronização e incremento de tempo

** MEMORIANDO: teste de limites de memória declarado no comando de simulação MEMORY (vide Cap. IV - Descrição da Linguagem).

Fig. III.15 (a) - Instrução SHLD.

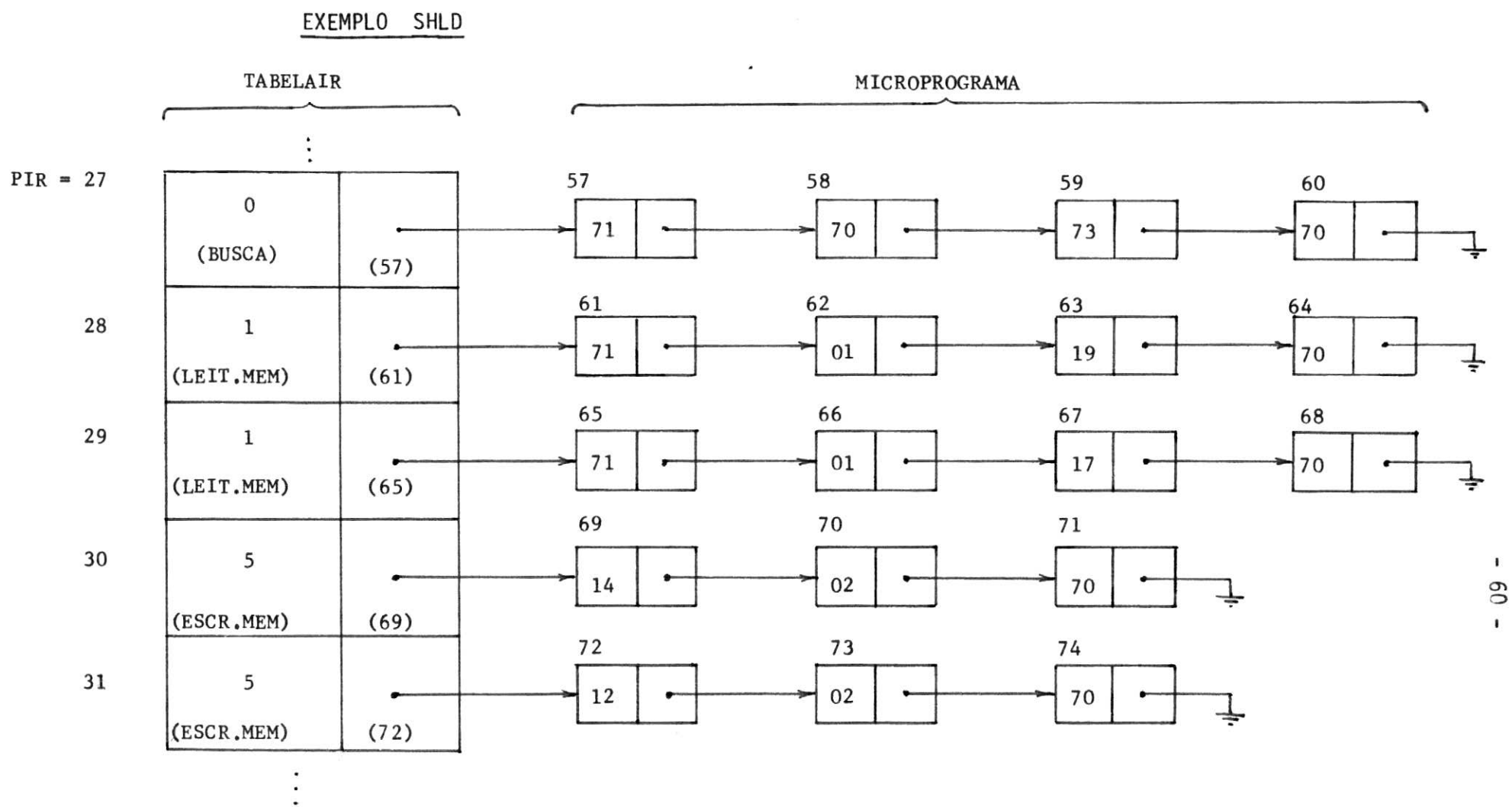


Fig. III.15(b) - TABELAIR e MICROPROGRAMA em SHLD

No Apêndice B, é apresentada a listagem correspondente a MICROPROGRAMA. MICROINST e NEXTADDRESS, correspondem respectivamente ao primeiro e segundo campo do cartão de entrada.

A separação entre os estados T3, T4 e T5 é feita através da microinstrução MICRO70, que chama a rotina INCRCLOCK,que incrementa o relógio da simulação e verifica a ocorrência de algum evento associado a interrupções, "HOLD" e "RESET".

3.2.4 - AS MICROOPERAÇÕES

Segue-se, uma lista das microoperações implementadas e a numeração correspondente para a simulação de cada instrução do repertório do 8080.

====>   DEFINE F = MICROOPERACOES

MICROOPERACOES UTILIZADAS NA EXECUCAO DA INSTRUCAO, DE ACORDO COM O DIAGRAMA DE BLOCOS FUNCIONAL DO CPU 8080

```
                                                          ;
MICRO0=              #,
MICRO1   = BEGIN
                MEMORIANDO;
                D7D0:=MEM[A15A0];
                DATABUFFER:=D7D0;
           END#,
%
MICRO2   = BEGIN
                MEMORIANDO;
                D7D0:=DATABUFFER;
                MEM[A15A0]:=D7D0
           END#,
MICRO3   = BEGIN
                D7D0:=DISP[BUS.[31:8]];
                DATABUFFER:=D7D0;
           END#,
MICRO4   = BEGIN
                D7D0:=DATABUFFER;
                DISP[BUS.[31:8]]:=D7D0;
           END#,
MICRO5   =Z:=8*IR543#,
MICRO6   = BEGIN
                D7D0XXX;
                A15A0XXX;
           END#,
MICRO7   = IR:=DATABUFFER #,
MICRO8   = DATABUFFER:=PCL #,
MICRO9   = PCL:=DATABUFFER #,
MICRO10  = DATABUFFER:=PCH #,
MICRO11  = PCH:=DATABUFFER #,
MICRO12  = DATABUFFER:=H #,
MICRO13  = H:=DATABUFFER #,
MICRO14  = DATABUFFER:=L #,
MICRO15  = L:=DATABUFFER #,
MICRO16  = DATABUFFER:=W #,
MICRO17  = W:=DATABUFFER #,
MICRO18  = DATABUFFER:=Z #,
MICRO19  = Z:=DATABUFFER #,
MICRO20  = DATABUFFER:=A #,
MICRO21  = A:=DATABUFFER #,
MICRO22  = BEGIN REG3.[45:1]:=0;  REG3.[43:1]:=0;
                 REG3.[41:1]:=1;
                 DATABUFFER:=FLAG; END #,
MICRO23  = FLAG:=DATABUFFER #,
MICRO24  = DATABUFFER:=CASE IR210 OF (B,C,D,E,H,L,M,A) #,
```

```
MICR025 = CASE IR543 OF
          BEGIN
               B:=DATABUFFER;
               C:=DATABUFFER;
               D:=DATABUFFER;
               E:=DATABUFFER;
               H:=DATABUFFER;
               L:=DATABUFFER;
               ERROM;
                A:=DATABUFFER;
          END #,
        %MICR026 EFETUA A TRANSFERENCIA DA PARTE MENOS SIG.
        %NIFICATIVA DO PAR DE REGISTROS ESPECIFICADO   PELO
        %IR54 PARA O DATABUFFER
MICR026 = DATABUFFER:=CASE IR54 OF (C,E,L,SPL) #,
        %
        %MICR027 EFETUA A TRANSFERENCIA DA PARTE MAIS SIGNI
        %FICATIVA DO PAR DE REGISTROS ESPECIFICADO    PELOS
        %BITS 5 E 4 DO IR  PARA O DATABUFFER
MICR027 = DATABUFFER:=CASE IR54 OF (B,D,H,SPH) #,
        %
        %MICR028 EFETUA A TRANSFERENCIA DO CONTEUDO DO DATA
        %BUFFER PARA A PARTE MENOS SIGNIFICATIVA DO PAR  DE
        %REGISTROS ESPECIFICADO POR IR54
MICR028 = CASE IR54 OF
          BEGIN
               C:=DATABUFFER;
               E:=DATABUFFER;
               L:=DATABUFFER;
               SPL:=DATABUFFER;
          END #,
        % MICR029 IDEM A MICR028 COM A PARTE MAIS SIGNIFICA
        % TIVA DO PAR DE REGISTROS
MICR029 = CASE IR54 OF
          BEGIN
               B:= DATABUFFER;
               D:= DATABUFFER;
               H:= DATABUFFER;
               SPH:= DATABUFFER;
          END  #,
        %
MICR030 = DATABUFFER:=TMP #,
        %
MICR031 = TMP:=DATABUFFER #,
        %
MICR032 = DATABUFFER:=ALUREG #,
        %
MICR033 = A:=ALUREG #,
        %
        %MICR034 EFETUA TRANSFERENCIA DO ALUREG PARA UM DOS
        %REGISTROS ESPECIFICADOS POR IR543

MICR034 = CASE IR543 OF
          BEGIN
               B:=ALUREG;
               C:=ALUREG;
               D:=ALUREG;
               E:=ALUREG;
               H:=ALUREG;
               L:=ALUREG;
               ERROM;
               A:=ALUREG;
          END #,
```

```
%
          %
          %MICRO35 ESPECIFICA O CONTEUDO DO FLAG EM FUNCAO DO
          %RESULTADO EM ALUREG
MICRO35 = BEGIN
               S:=ALUAUX7;
               BEGIN
                    REAL PTEST;
                    PTEST:=K:=0;
                    THRU 8 DO
                         BEGIN
                           PTEST:=*+BIT(K);
                           K:=*+1;
                         END;
                    P:=REAL(NOT BOOLEAN(PTEST));
                    ZER:=IF PTEST EQL 0 THEN 1 ELSE 0;
               END;
          END #,
          %
MICRO36  = BEGIN
            ALUAUX:=TMP+1;
            AC:=IF TMP4 NEQ BIT(4) THEN 1 ELSE 0;
          END#,
MICRO37  = BEGIN
            ALUAUX:=TMP+4"FF";
            AC:=IF TMP4 EQL BIT(4) THEN 1 ELSE 0;
          END#,
          %
MICRO38 =    #,
MICRO39 = CASE IR543 OF
            BEGIN
               TMP:=B;
               TMP:=C;
               TMP:=D;
               TMP:=E;
               TMP:=H;
               TMP:=L;
               ERROM;
               TMP:=A;
            END#,
          %
          %MICRO40 INCREMENTA OU DECREMENTA O PAR DE REGISTRO
          %ESPECIFICADO POR IR54
          %SE BIT 3 DO IR FOR 0 INCREMENTA
          %                   1 DECREMENTA
MICRO40 = INCDEC:=CASE IR543 OF (
                         BC+1,
                         BC+4"FF",
                         DE+1,
                         DE+4"FF",
                         HL+1,
                         HL+4"FF",
                         SP+1,
                         SP+4"FF") #,
          %
MICRO41 = CASE IR54 OF
             BEGIN
                  BC:=INCDEC;
                  DE:=INCDEC;
                  HL:=INCDEC;
                  SP:=INCDEC;
             END #,
```

```
          %
MICRO42 = BEGIN
               ALUAUX:=ACT+TMP ;
               CY:=MSBALUAUX;
               PY:=REAL(BOOLEAN(TMP4) EQV BOOLEAN(ACT4));
               AC:=IF BIT(4) EQL PY
                   THEN 1 ELSE 0;
          END #,
MICRO43 = BEGIN
               ALUAUX:=TMP+CY;
               AC:=IF TMP4 NEQ BIT(4) THEN 1 ELSE 0;
               TMP:=ALUAUX;
               CY:=MSBALUAUX;
               ALUAUX:=ACT+TMP;
               IF CY=0 THEN CY:=MSBALUAUX;
               IF AC=0 THEN
               PY:=REAL(BOOLEAN(TMP4) EQV BOOLEAN(ACT4));
               AC:=IF BIT(4) EQL PY
                       THEN 1 ELSE 0;
          END#,
MICRO44 = BEGIN
               ALUREG:=REAL(NOT BOOLEAN(TMP));
               TMP:=ALUREG+1;
               ALUAUX:=ACT+TMP;
               PY:=REAL(BOOLEAN(TMP4) EQV BOOLEAN(ACT4));
               AC:=IF BIT(4) EQL PY
                   THEN 1 ELSE 0;
               CY:=IF MSBALUAUX=0 THEN 1 ELSE 0;
          END#,
MICRO45 = BEGIN
               ALUAUX:=TMP+CY;
               TMP:=ALUAUX;
               TMP:=REAL(NOT BOOLEAN(TMP));
               TMP:=TMP+1;
               ALUAUX:=ACT+TMP;
               PY:=REAL(BOOLEAN(TMP4) EQV BOOLEAN(ACT4));
               AC:=IF BIT(4) EQL PY
                   THEN 1 ELSE 0;
               CY:=IF MSBALUAUX=0 THEN 1 ELSE 0;
          END#,
MICRO46 = BEGIN
               ALUREG:=REAL(BOOLEAN(ACT) AND  BOOLEAN(TMP));
               CY:=0;
          END#,
MICRO47 = BEGIN
            ALUREG:=REAL(NOT(BOOLEAN(ACT) EQV BOOLEAN(TMP)));
            CY:=0;
          END#,
MICRO48  = BEGIN
            ALUREG:=REAL(BOOLEAN(ACT) OR BOOLEAN(TMP)) ;

            CY:=0;
          END#,
          %
MICRO49 = ACT:=A #,
          %
          %MICRO50 EFETUA TRANSFERENCIA DO TMP PARA UM DOS
          %REGISTROS ESPECIFICADOS PELO IR543
```

```
MICR050 = CASE IR543 OF
          BEGIN
               B:=TMP;
               C:=TMP;
               D:=TMP;
               E:=TMP;
               H:=TMP;
               L:=TMP;
               ERROM;
               A:=TMP;
          END #,
          %
          %MICR051 EFETUA TRANSFERENCIA DO CONTEUDO DO REGIS-
          %TRO ESPECIFICADO POR IR210 PARA O TMP
MICR051 = CASE IR210 OF
          BEGIN
               TMP:=B;
               TMP:=C;
               TMP:=D;
               TMP:=E;
               TMP:=H;
               TMP:=L;
               ERROM;
               TMP:=A;
          END #,
          %
          %MICR052 COMPLEMENTA TODOS OS BITS DO ACUMULADOR
          %
MICR052 = A:=REAL   (NOT BOOLEAN(A)) #,
MICR053 = CY:=1"1" #,
          %
MICR054 = CY:=REAL   (NOT BOOLEAN(CY)) #,
          %
MICR055 = PC:=HL #,
          %
MICR056 = PC:=WZ #,
MICR057 = CONDICAO:=CASE IR543 OF
                    (
                      IF ZFR=0 THEN TRUE ELSE FALSE,   % NZ
                      IF ZFR=1 THEN TRUE ELSE FALSE,   % Z
                      IF CY=0 THEN TRUE ELSE FALSE,    % NC
                      IF CY=1 THEN TRUE ELSE FALSE,    % C
                      IF P=0  THEN TRUE ELSE FALSE,    % PO
                      IF P=1  THEN TRUE ELSE FALSE,    % PE
                      IF S=0  THEN TRUE ELSE FALSE,    % P
                      IF S=1  THEN TRUE ELSE FALSE     % M
                    )#,
MICR058 = SP:=HL #,
MICR059 = WZ:=HL #,
MICR060 = HL:=WZ #,
MICR062 = DE:=WZ #,
MICR063 = HL:=INCDEC#,
MICR061 = HL:=DE #;
          %
          %MICR064 EFETUA A SOMA DO PAR DE REGISTROS HL COM
          %O ESPECIFICADO POR IR54
MICR064 = WZ:=CASE IR54 OF
                 (HL+BC,
                  HL+DE,
                  HL+HL,
                  HL+SP) #,
```

```
MICRO65 = BEGIN
                TMP:=A;
                ALUAUX:=A;
                IF (Y GEQ 10) OR (AC=1)
                    THEN BEGIN ALUAUX:= TMP+6;
                               AC:=IF BOOLEAN(TMP4) EQV
                               BOOLEAN(BIT(4)) THEN 0 ELSE 1;
                         END;
                IF (X GEQ 10) OR (CY=1)
                   THEN BEGIN
                          ALUAUX:=ALUAUX+4"60";
                          CY:=MSBALUAUX;
                        END;
          END  #,
MICRO66 = BEGIN
                ALUAUX:=2*A;
                CY:=MSBALUAUX;
                ALUAUX0:=CY;
          END#,
MICRO67 = BEGIN
                ALUAUX:=A;
                CY:=ALUAUX0;
                ALUAUX:=ALUAUX/2;
                ALUAUX7:=CY;
          END  #,
MICRO68 = BEGIN
                ALUAUX:=2*A;
                ALUAUX0:=CY;
                CY:=MSBALUAUX;
          END  #,
MICRO69 = BEGIN
                ALUAUX:=A;
                MSBALUAUX:=CY;
                CY:=ALUAUX0;
                ALUAUX:=ALUAUX/2;
          END  #,
MICRO70 = BEGIN
                INCRCLOCK;
          END#,
MICRO71 =
          BEGIN
                DBIN:=FALSE;
          END#,
MICRO72  =
          BEGIN
                PROG:=FALSE;
          END#,
MICRO73 = D7D0:=DATABUFFER#,
MICRO74 = CY:=MSBALUAUX #,
MICRO75 = L:=ALUREG #,
MICRO76 = H:=ALUREG#,
MICRO77 = WAIT:=TRUE #,
MICRO78 = IF NOT CONDICAO THEN BEGIN PIR:=PIR+2;
                                     PC:=PC+2; END#,

MICRO79 = WRBARRA:=FALSE #,
MICRO80 = INTEFF:=TRUE #,
MICRO81 = INTEFF:=FALSE#,
MICRO82 = IF CONDICAO THEN SP:=SP-1#,
MICRO83 = SP:=SP-1#,
MICRO84 = W:=0#,
```

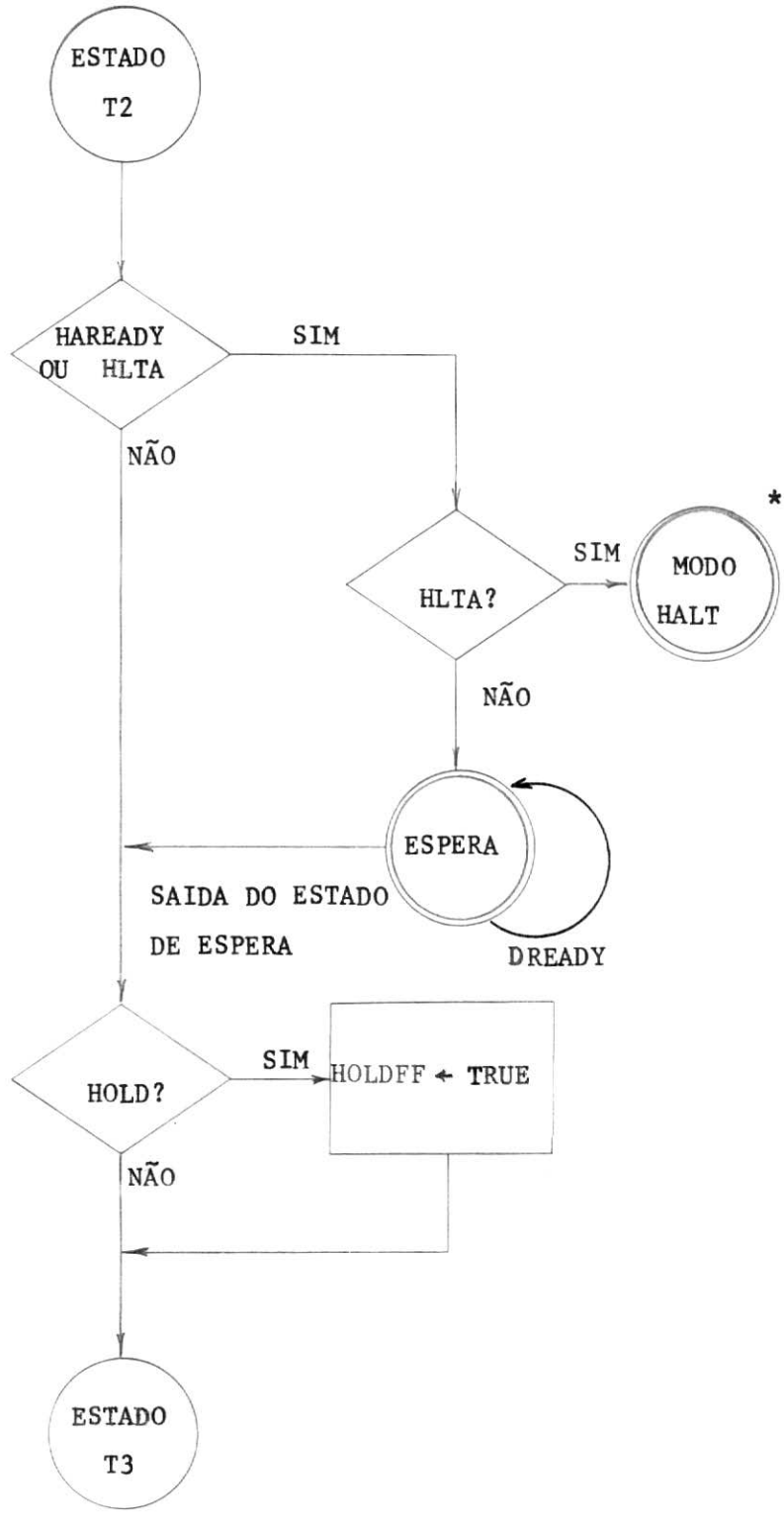
3.2.5 - SIMULAÇÃO DOS EVENTOS READY, HOLD, INTERRUPÇÃO E ESTADO HALT

Os eventos READY, HOLD e INTERRUPÇÃO são estabelecidos através dos comandos de contrôle de simulação correspondentes, enquanto que o estado HALT provêm da execução da instrução HLT, que ativa HLTA.

O comando de controle de simulação "READY", fornece ao simulador, na variável global "DREADY", o número de estados de espera e, na variável global booleana "HAREADY", a informação da existência de pedido de espera.

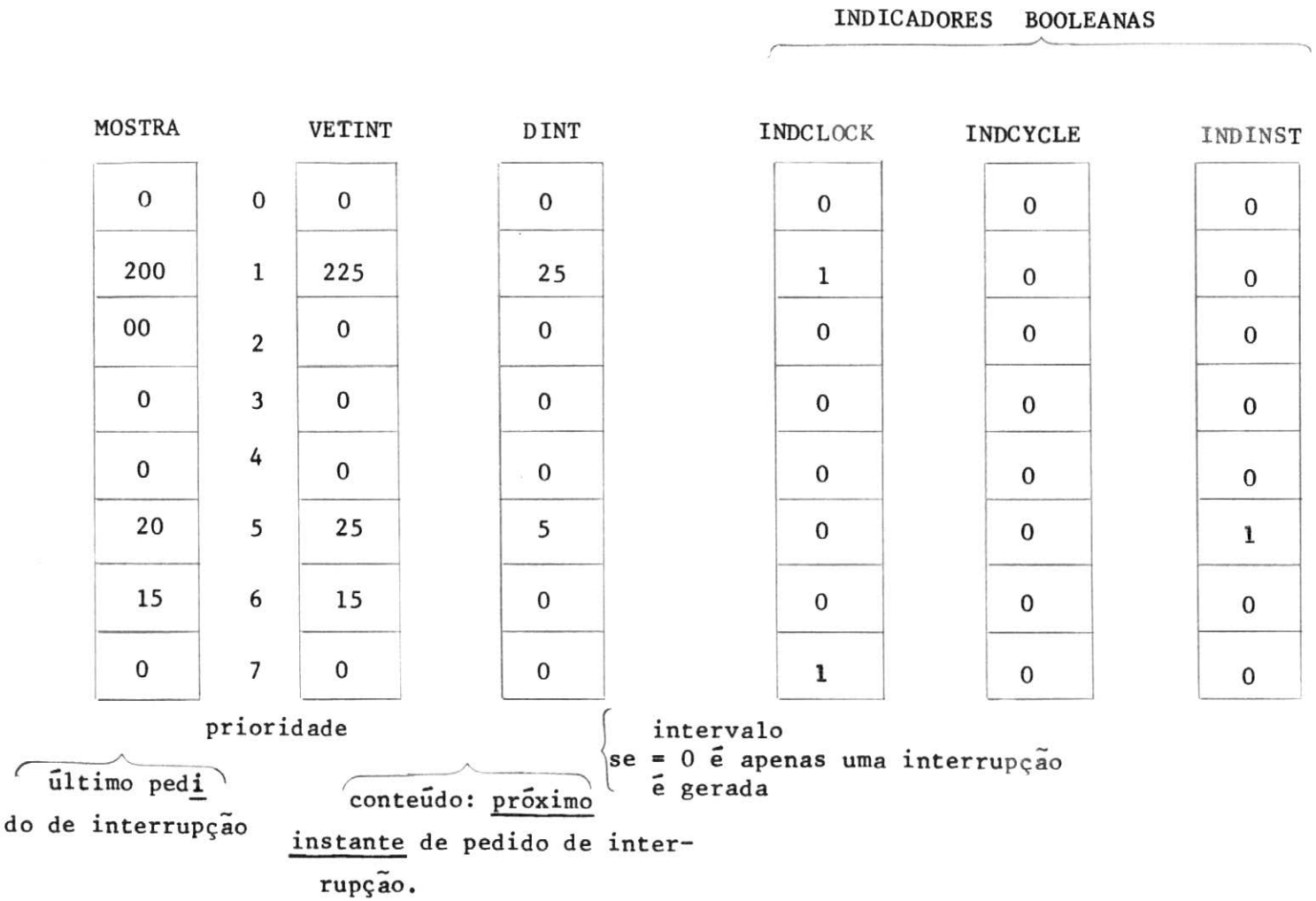
O comando de contrôle de simulação "HOLD" fornece a duração do evento HOLD na variável global "DURHOLD". O tempo e a frequência do pedido de HOLD também são dados pelo comando. A variável INSTHOLD determina o tempo do pedido em função das opções oferecidas CLOCK, CYCLE e INST (vide Cap. IV) e DHOLD; o período dos pedidos de HOLD. HAHOLD é uma variável booleana que indica a existência do evento. Os testes dos eventos READY e HOLD são dados na Figura III.16.

As interrupções são simuladas de modo análogo ao evento HOLD. Ao invés de se fornecer a duração de eventos, como no caso anterior, as prioridades associadas às interrupções são estabelecidas. Há 8 níveis de prioridades possíveis: 0, 1, 2, ..., 7. Como, também, as opções CLOCK, CYCLE e INST são disponíveis no comando INTER, foram criadas indicadores correspondentes. A Figura III.17 apresenta os indicadores mencionados (INDCYCLE, INDCLOCK, INDINST) bem como os outros parâmetros.



* Modo HALT é descrito na seção 3.2.5

Fig. III.16 - Testes de "READY", "HOLD" e "HLTA"



THAINTER indica se há interrupção por "clock"

CHAINTER indica se há interrupção por "cycle"

IHAINTER indica se há interrupção por "Inst".

Fig. III.17 - Interrupção com prioridade.

No diagrama de INCRCLOCK:

NROCLOCK é o contador do número de pulsos de relógio ("clock") ocorridos desde o início da simulação.

HARESET variável booleana que notifica a presença de RESET.

TVARIOS indicador booleano de existência de interrupção, de pedido de HOLD ou comando WRITE.

INST, INSTW, INSTHOLD e VETINT [I] estabelecidos pelos comandos RESET, WRITE, HOLD e INTER respectivamente, indicam o instante de ativação do evento correspondente.

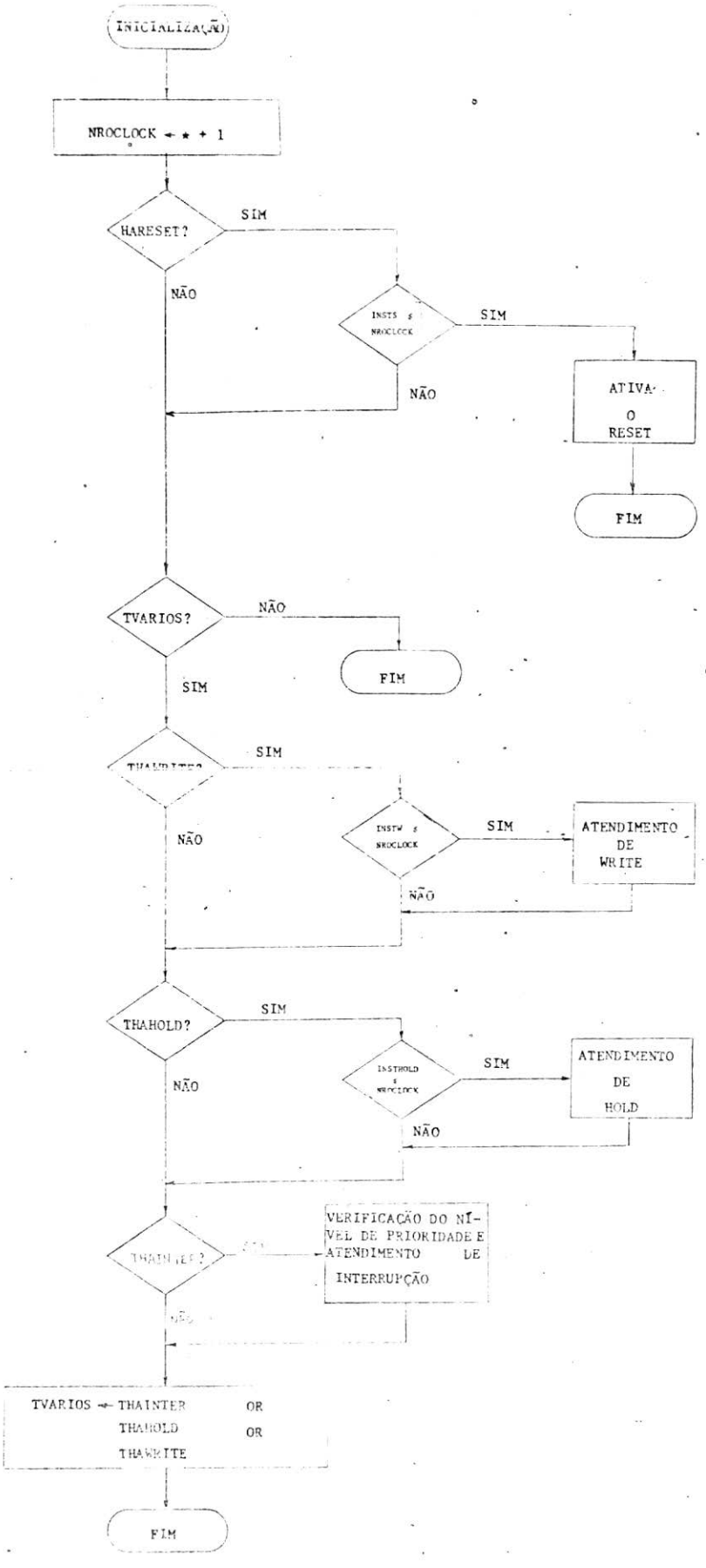
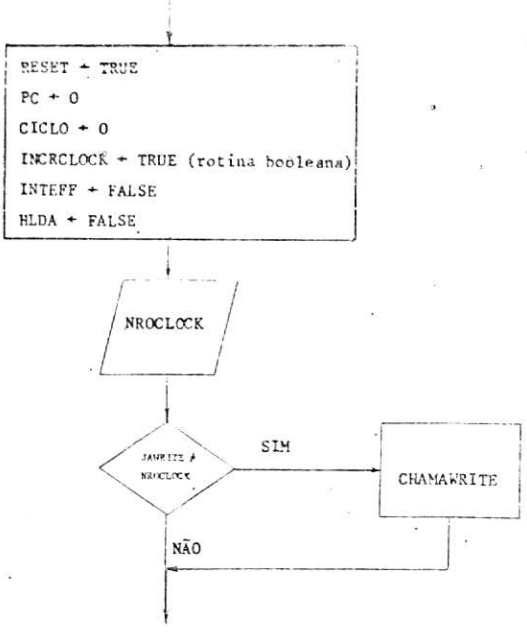
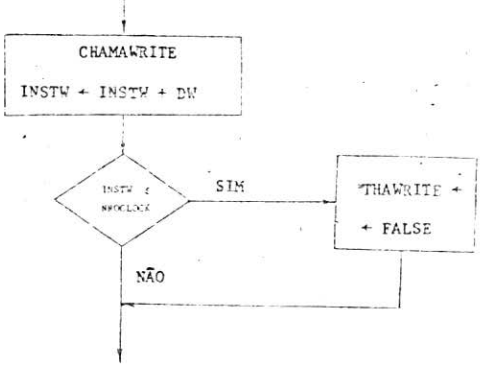


Fig. III.18 - Diagrama de Blocos de INCRINST.



(a)



(b)

Fig. III.19(a) - Ativação do RESET
(b) - Atendimento de WRITE

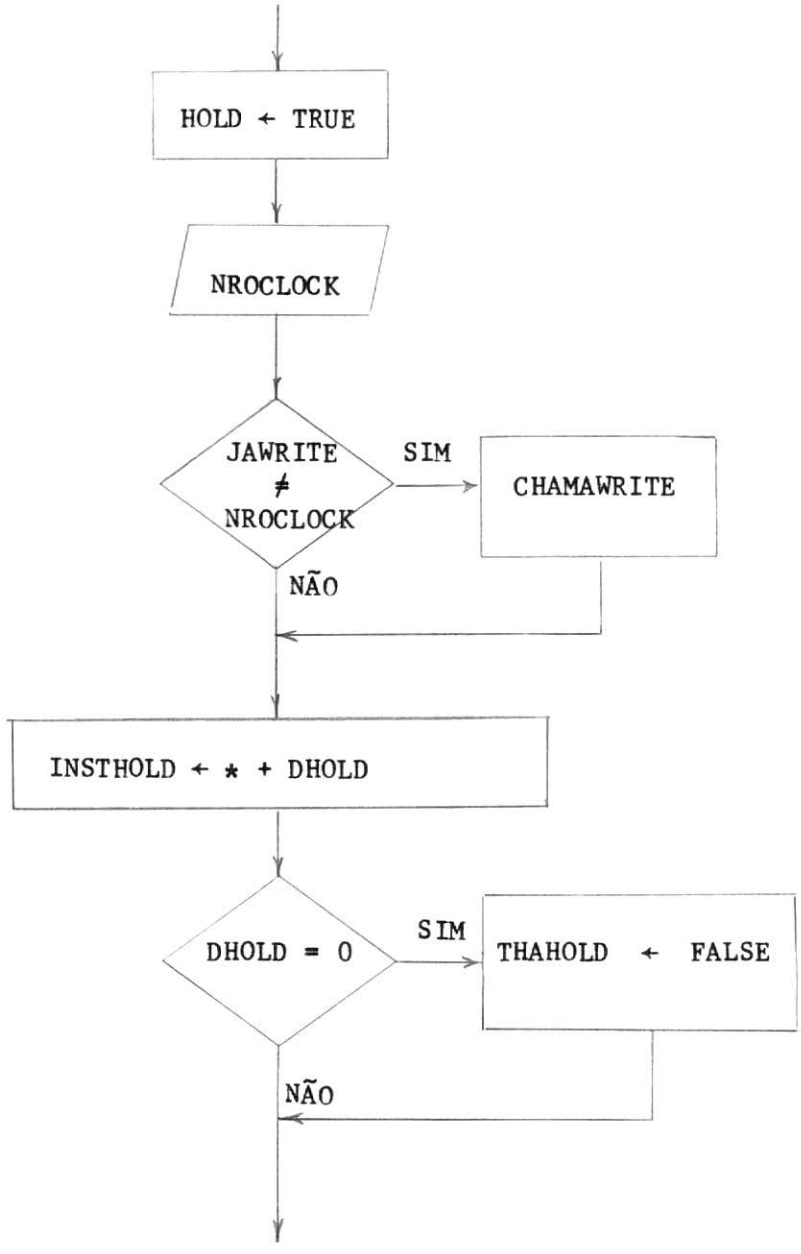


Fig. III.20 - Atendimento de HOLD

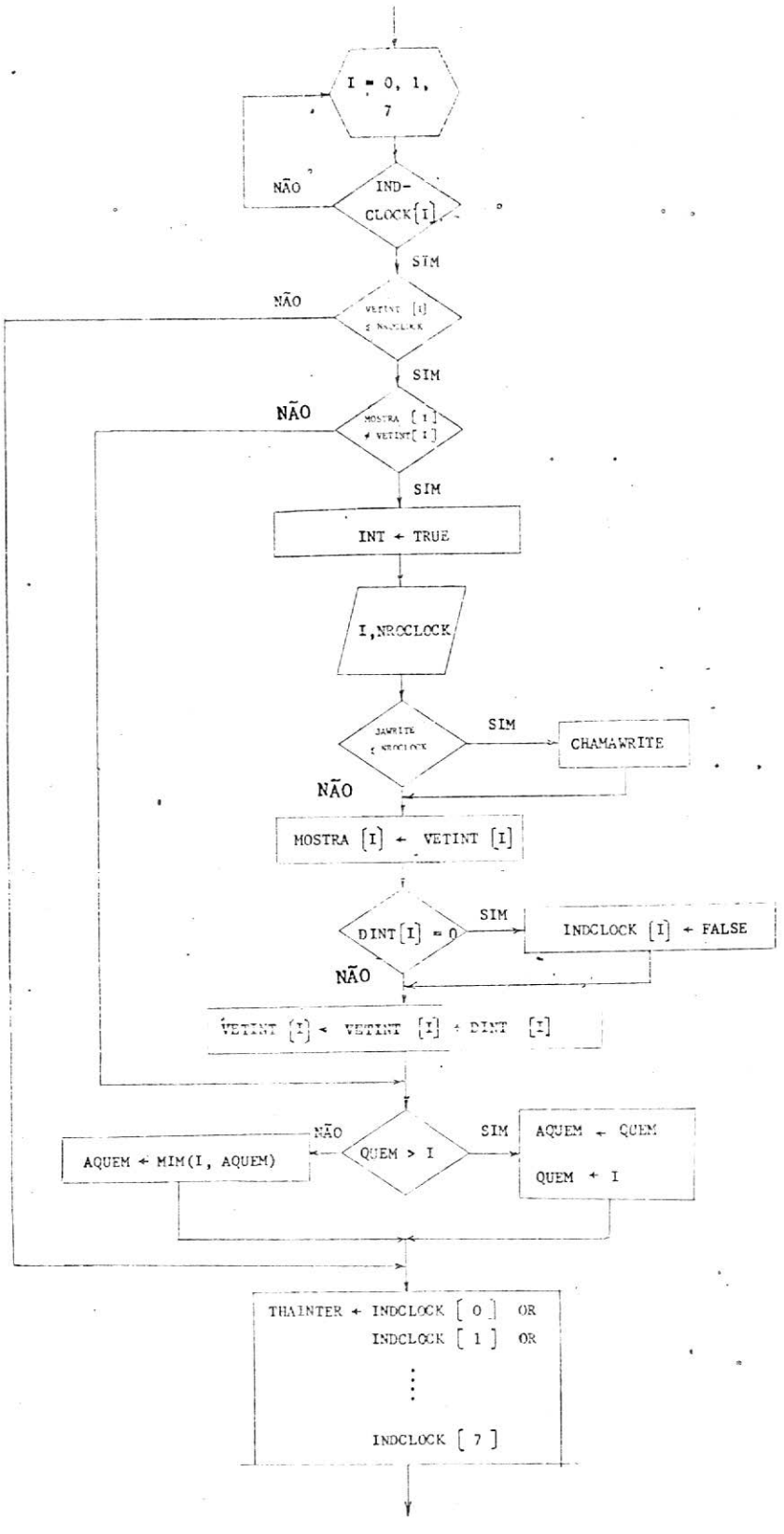


Fig. III.21 - Atendimento de Interrupção.

3.2.6 - DIAGRAMA DE BLOCOS DO SIMULADOR SIM8080

Nesta seção é apresentado o diagrama do blocos correspondente à implementação da sub-rotina SIM8080.

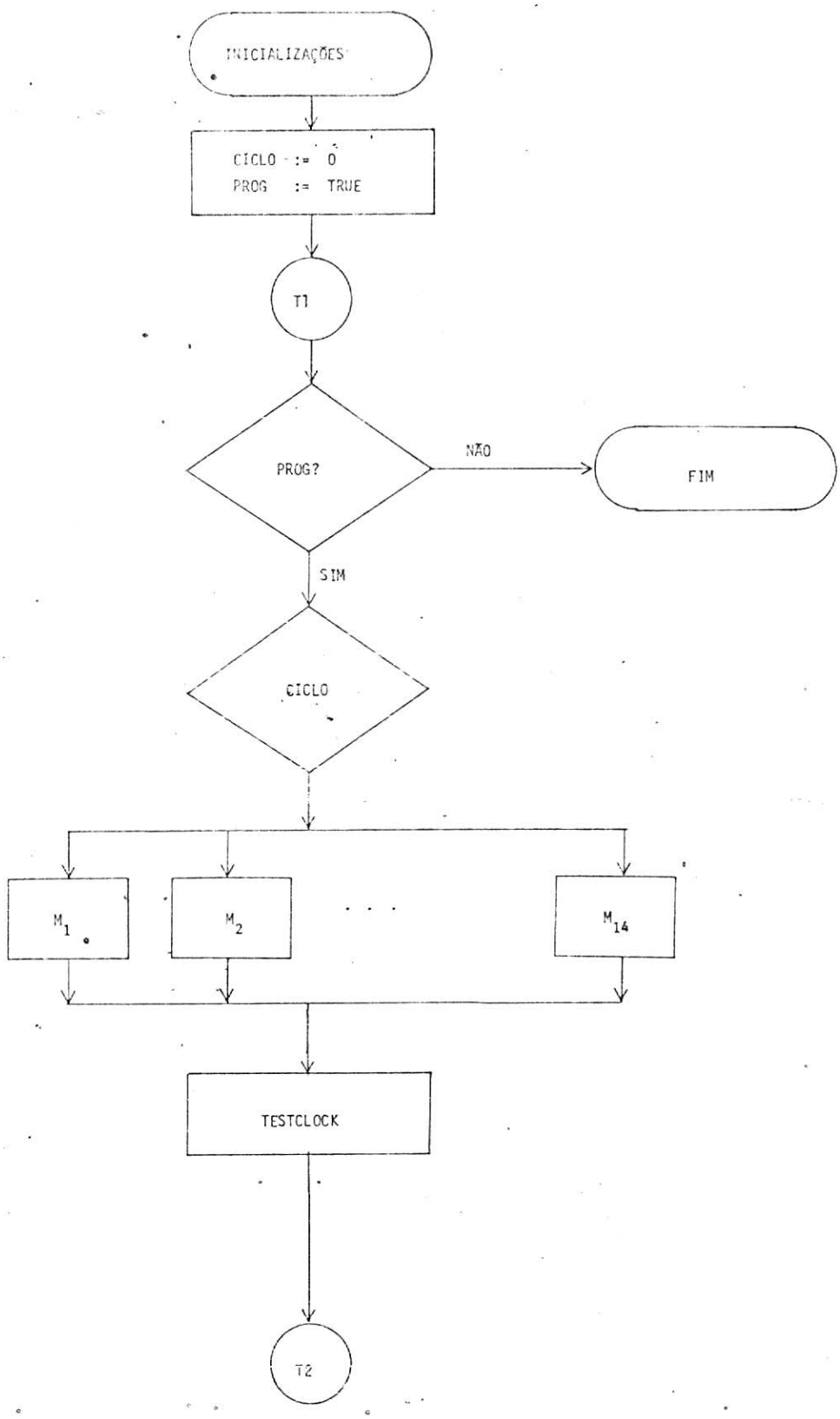


Fig. III.22 (a) - Simulação do Estado T1.

Fig.III.22 - Diagrama de Blocos de Simulador SIM 8080.

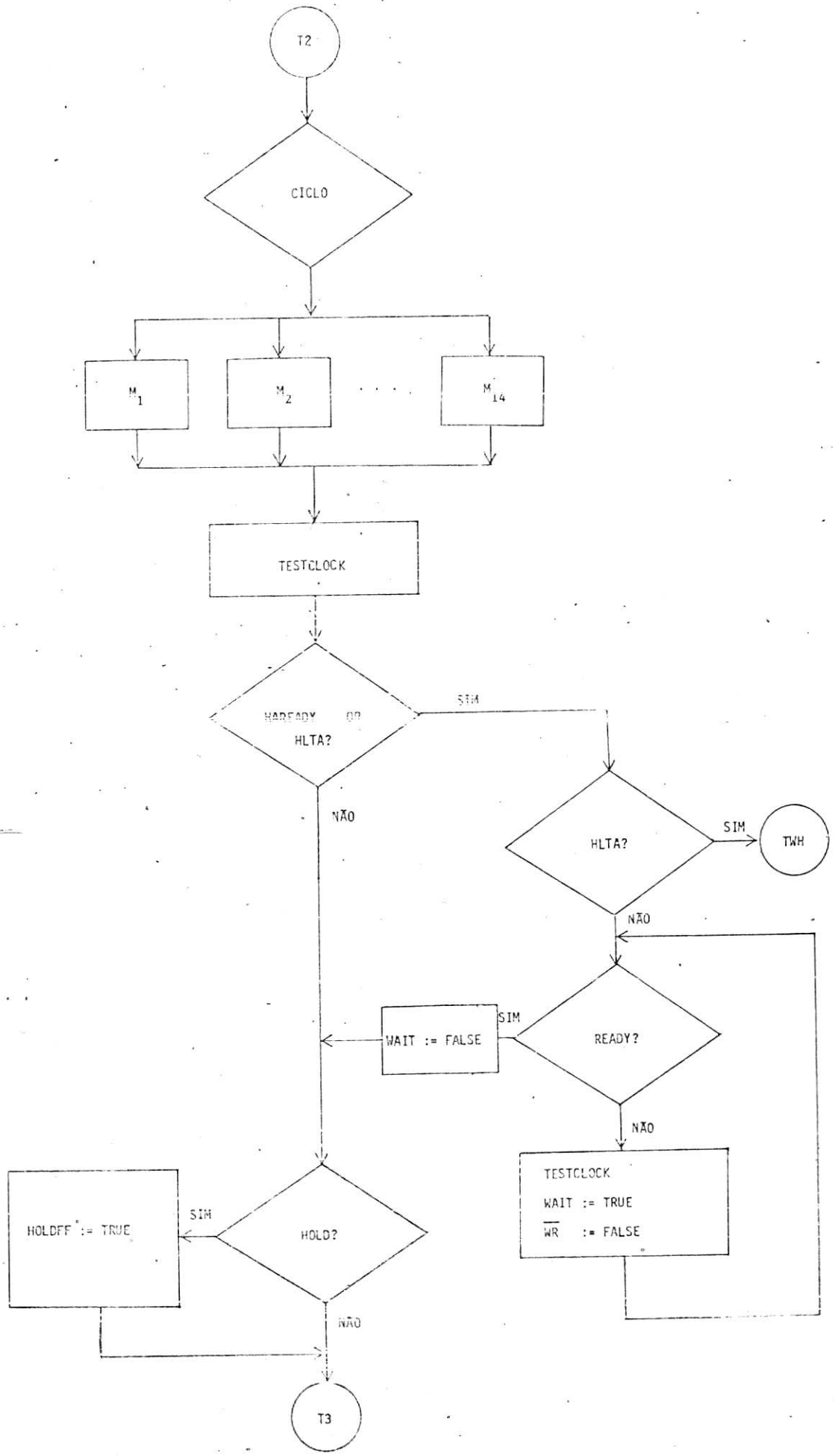


Fig. III.22 (b) - Simulação do Estado T2.

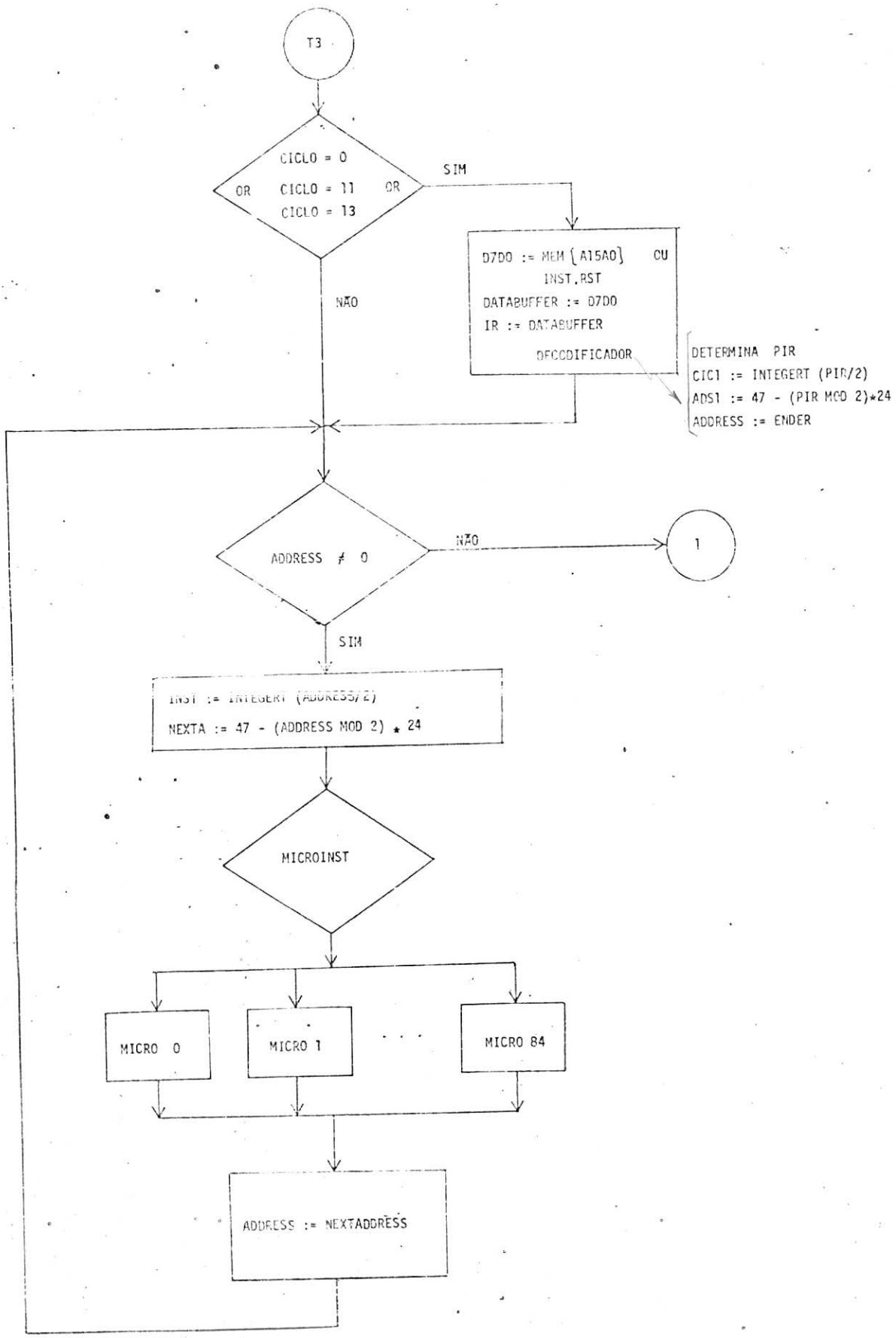


Fig. III.22 (c) - Simulação do Estado T3, T4 e T5.

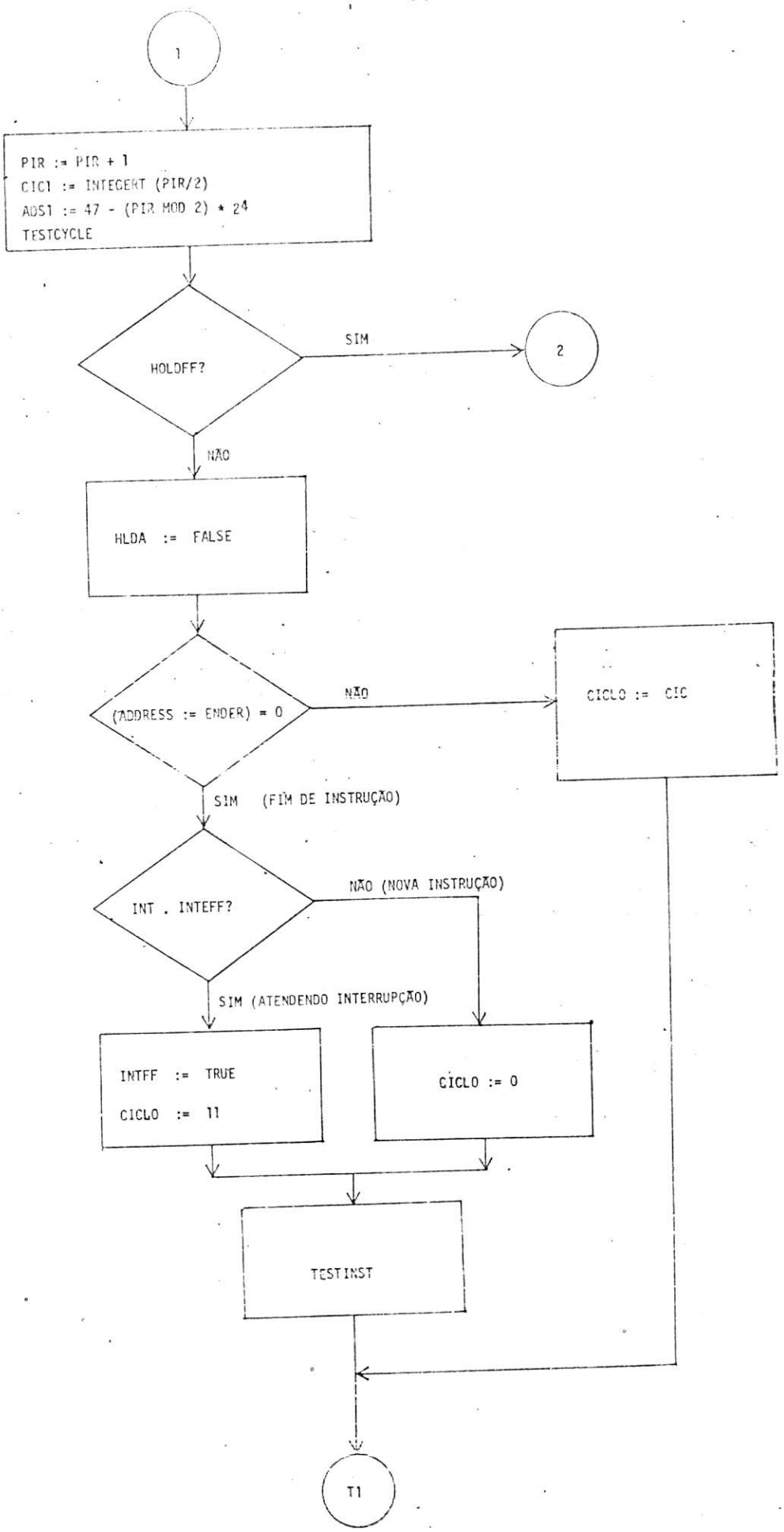


Fig. III.22 (d) - Teste de Fim de Instrução

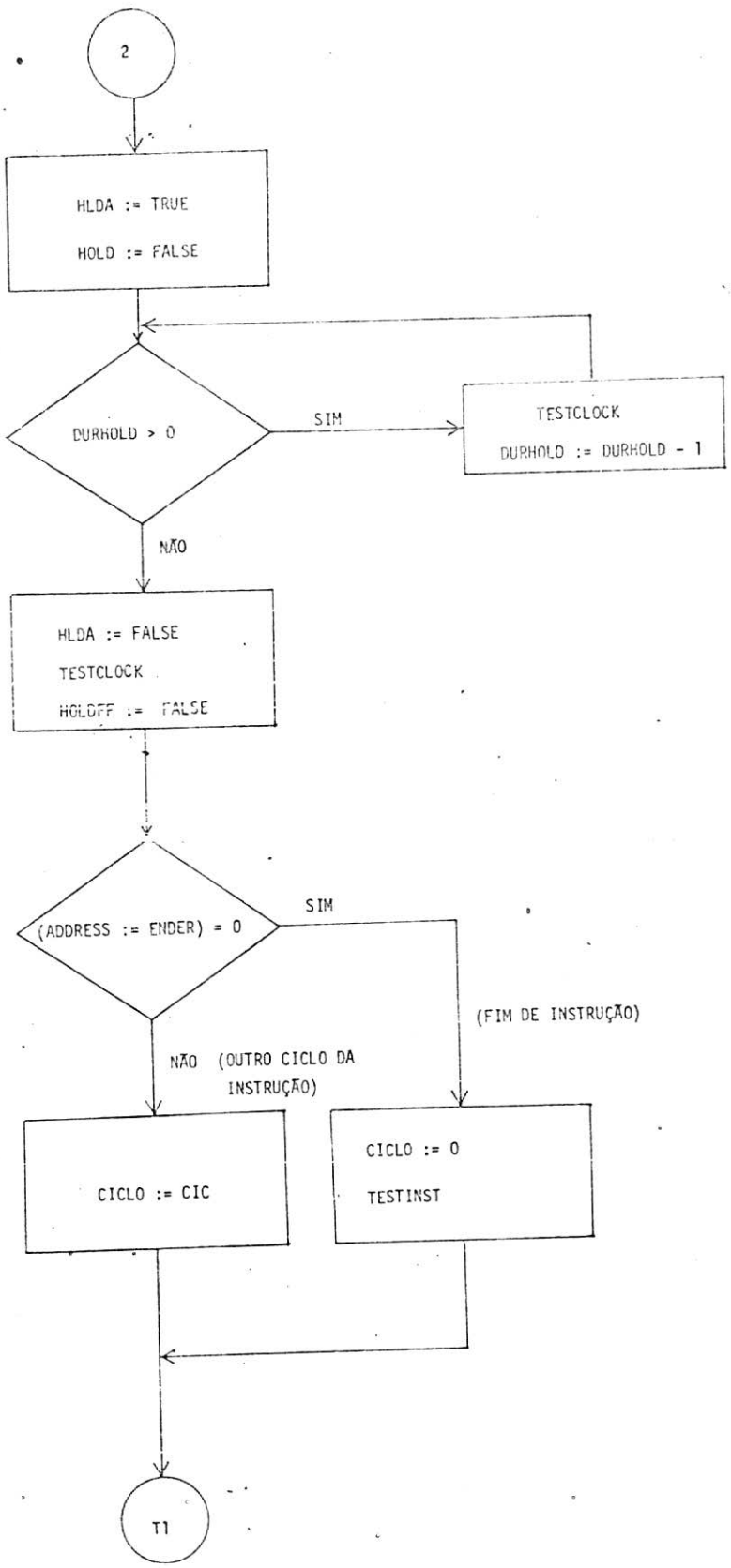


Fig. III.22 (e) - Atendimento de HOLD e Teste de Fim de Instrução.

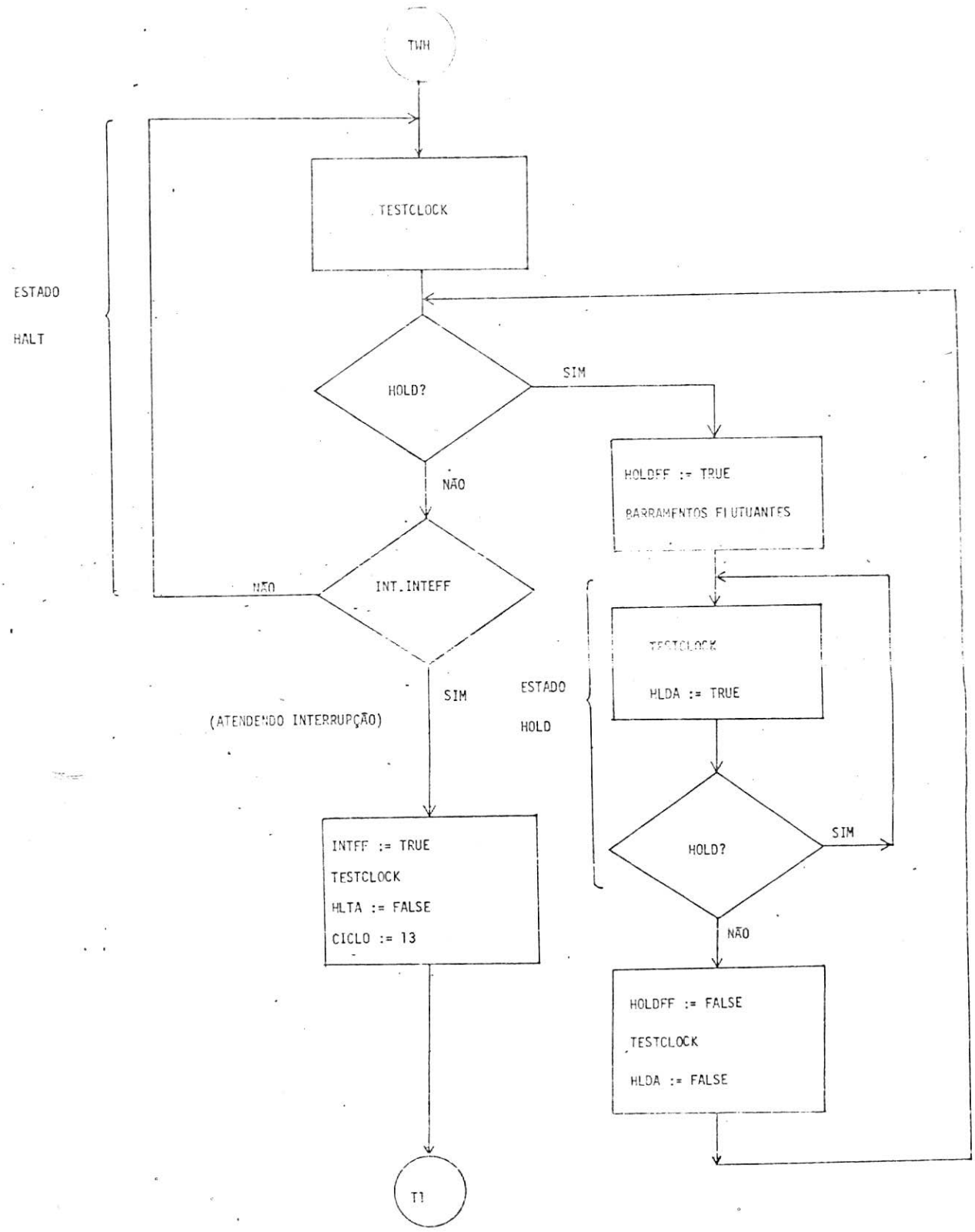


Fig. III.22 (f) - Simulação do Estado HALT. e HOLD

CAPÍTULO IV

A LINGUAGEM DE CONTROLE DE SIMULAÇÃO

A linguagem de controle de simulação do sistema especifica as condições iniciais do simulador, bem como as características do monitoramento desejado e os eventos externos que se deseja ativar. Como se trata de uma linguagem de contrôle específica de um simulador é constituida basicamente por um conjunto de comandos que controlam diretamente cada uma dessas condições, sendo totalmente condicionada pelas características do microprocessador.

4.1 - COMPONENTES DA LINGUAGEM

Sintaxe:

<*componentes da linguagem*> ::= <*simbolos básicos*>|
<*palavras reservadas*>|
<*constante*>|
<*comentários*>|
<*programa de controle*>

4.1.1 - SIMBOLOS BÁSICOS E PALAVRAS RESERVADAS

Sintaxe:

<*simbolos básicos*> ::= <*letras*>|

<*dígito*>|

<*vazio*>|

<*caracteres especiais*>

<*letra*> ::= A | B | C | D | E | F | G | H | I | J | K | L | M | N |
O | P | Q | R | S | T | U | V | W | X | Y | Z

<*dígito*> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<*caracteres especiais*> ::= (|) | , | ; | = | %

<*vazio*>::= {*conjunto vazio de caracteres*}

<*palavras reservadas*>::= <*palavra chave*>|
<*identificador especial*>|
<*conectivo*>

<*palavra chave*> ::= MEMORY | READY | LOAD | INTER | HOLD | RESET |
REFER | WRITE | INPUT | OUTPUT | COPY | SAVE

<*identificador especial*> ::= A | B | C | D | E | H | L | ZERO |
CARRY | SIGN | PARITY | ACARRY | INT |
HOLD | INTE

<*conectivo*> ::= <*indicador de tempo*>| TO | WITH | DUMP | STATUS |
AND| M

<indicador de tempo> ::= CLOCK | CYCLE | INST

Os símbolos básicos e as palavras reservadas constituem o "alfabeto" sobre o qual a linguagem de contrôle de simulações é definida.

4.1.2 - CONSTANTES

Sintaxe:

<caracter octal> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7

<constante> ::= *<constante octal>*|
<constante decimal>

<constante octal> ::= *<caracter octal>*|
<constante octal><caracter octal>

<constante decimal> ::= *<numero>*

<numero> ::= *<dígito>*|
<numero><digito>

Apenas dois tipos de constante podem ser usados na linguagem de controle, a *<constante octal>* e a *<constante decimal>*, sendo que o ponto do programa onde utilizar cada um deles será definido quando da definição dos comandos.

4.1.3 - COMENTÁRIOS

O comentário é uma forma de documentar o programa, podendo ser inserido em qualquer ponto, sempre de acordo com a sintaxe:

<*comentário*> ::= % {qualquer sequência de caracteres até o fim do cartão}.

Os caracteres à direita do sinal "%" são, então, ignorados pelo analisador.

4.1.4 - PROGRAMA DE CONTROLE DE SIMULAÇÃO:

A sintaxe de cada programa de controle de simulação tem a seguinte estrutura:

<*sequência de comandos*> ::= <*comandos simples*> |
<*sequência de comandos*>;<*comando simples*>

<*conjuntos de comandos*> ::= <*sequência de comandos*>;<*comando de finalização*>;

<*programa de controle*> ::= <*conjunto de comandos*> |
<*programa de controle*><*conjunto de comandos*>

O programa de controle de simulação pode ser constituido por uma sequência de <*conjunto de comandos*>, onde cada <*conjunto de co*

mandos> especifica as condições iniciais e as condições de monitoramento vigentes em cada ativação do simulador. Cada <*comando de finalização*>, ao ser analisado, provoca uma ativação do simulador. Desta maneira, em um mesmo programa de controle, poderemos especificar varias "simulações", sob condições totalmente distintas.

4.2 - OS COMANDOS DE CONTROLE DE SIMULAÇÃO

Sintaxe:

<*comandos de controle*> ::= <*comandos simples*> |
<*comando de finalização*>

<*comandos simples*> ::= <*comandos de inicialização*> |
<*comandos de monitoramento*> |
<*comandos de eventos*>

Os comandos de controle são divididos em duas classes gerias, sendo que a classe dos <*comandos simples*> é por sua vez dividia em 3 subclasses de acordo com as características funcionais dos comandos que as constituem.

4.2.1 - OS COMANDOS DE INICIALIZAÇÃO:

Sintaxe:

<*comandos de inicialização*> ::= <*comando* MEMORY> |

<comando READY> |

<comando LOAD> |

<comando COPY>

<comando MEMORY> ::= MEMORY (<numero>,<numero>)

<comando READY> ::= READY <numero>

<comando LOAD> ::= LOAD <sequência de inicializações>

<sequência de inicializações> ::= <inicialização simples> |
<sequência de inicializações> ,
<inicialização simples>

<inicialização simples> ::= <identificador-especial>=<constante-
octal> |
M (<endereço>)= <constante octal> |
M (<endereço>)= <sequência de constantes>

<endereço> ::= <numero>

<sequência de constantes> ::= <constante-octal> |
<sequência de constantes><constante
octal> |

<comando SAVE> ::= COPY M(<numero>,<numero>) |
COPY STATUS AND M (<numero>,<numero>)

OBSERVAÇÕES

- O <*comando* MEMORY> especifica o espaço de memória a ser utilizado na simulação (o par de numeros indica os endereços limites). Deve ser utilizado sempre como o primeiro comando do primeiro <*conjunto de comandos*> do programa de controle. O <*comando* READY> especifica o intervalo de tempo (em múltiplos do ciclo do "clock" básico) que a memória utiliza antes de fornecer o sinal READY no pino correspondente do microprocessador.

- O <*comando* LOAD> inicializa os registros e as posições de memória especificados com os valores indicados pelas constantes octais utilizadas.

- O <*comando* COPY> inicializa os registros do microprocessador e/ou as posições de memória, especificadas com os valores armazenados anteriormente num arquivo em disco (de nome "SALVADOR"), através do <*comando* SAVE>.

4.2.2 - COMANDOS DE MONITORAMENTO:

<*comandos de monitoramento*> ::= <*comando* REFER>|
<*comando* WRITE>|
<*comando* de E/S>|
<*comando* SAVE>

<*comando* REFER> ::= REFER <*numero*>TO <*numero*>

<*comando* WRITE> ::= WRITE <*indicador de tempo*>(<*ocasião*>) <*sequência de endereços*>

<*ocasião*> ::= <*número*>|<*numero*>,<*numero*>

<*sequência de endereços*>::= M (<*endereço*>)|
<*sequência de endereços*>,M (<*endereço*>)

<*comando* de E/S> ::= INPUT <*sequência de números*>|
OUTPUT <*sequência de numeros*>

<*comando* SAVE> ::= SAVE M (<*numero*>,<*numero*>)|
SAVE STATUS AND M (<*numero*>,<*numero*>)

<*sequência de numeros*> ::= <*numeros*>|
<*sequência de numeros*>,<*numeros* >

Os comandos de monitoramento controlam a simulação à medida que esta se processa.

O <*comando* REFER> significa que toda acesso à área de memória, por ele especificada, fará com que seja listado o estado de microprocessador e os conteudos das posições de memória, especificados no <*comando* WRITE>.

O <*comando* WRITE> especifica a partir de que instante da simulação e com que frequência devem ser listados o estado do microprocessador e os conteudos das posições de memória especificadas.

O <*comando de* E/S> indica, através da <*sequência de números*>, os endereços de periféricos que, quando referenciados na simulação de instruções de E/S do microprocessador, implicam em ações especiais por parte do simulador, a saber:

- os endereços especificados em "INPUT" indicam que, toda vez que for simulada uma instrução de entrada a eles se referindo, o simulador deve retirar sequencialmente de arquivos de cartão, fornecidos ao simulador (arquivos de nomes DADOS1, DADOS2,..... DADOS8), um caracter (codificado em ASCII (e colocá-lo na via de dados do microprocessador.

- os endereços especificados em "OUTPUT" indicam que, toda vez que for simulada uma instrução de saída a eles se referindo, o conteúdo da via de dados do microprocessador deve ser listado.

O <*comando* SAVE> é semelhante ao <*comando* COPY>, porem com o sentido oposto.

4.2.3 - COMANDOS DE GERAÇÃO DE EVENTOS EXTERNOS

Sintaxe:

<*comandos de eventos*> ::= <*comando* INTER> |
<*comando* HOLD>

<*comando* INTER> ::= INTER (<*nível de interrupção*>) <*indicador de tempo*> (<*ocasião*>)

<*nível de interrupção*> ::= <*numero*>

<*comando* HOLD> ::= HOLD (<*duração*>)<*indicador de tempo*>(<*ocasião*>)

<*duração*> ::= <*numero*>

O <*comando* INTER> controla a execução de interrupções externas no microprocessador. As interrupções serão atendidas de acordo com a prioridade indicada pelo <*nivel de interrupção*> (de 0 a 7, onde 0 é a maior prioridade).

O <*comando* HOLD> indica em que instante o microprocessador deve ser colocado no estado Hold e qual a duração deste estado.

4.2.4 - COMANDO DE FINALIZAÇÃO

<*comando de finalização*> ::= RESET <*indicador de tempo*>(<*numeros*>)
<*lista memória*>

<*lista memória*> ::= <*vazio*> |
WITH DUMP (<*numero*>,<*numero*>)

O comando "RESET" indica em que momento a simulação deve ser finalizada. Pode ser indicado tambem se deve ser efetuada um listagem do conteudo de uma area da memória especificada.

CAPÍTULO V

COMENTÁRIOS E CONCLUSÕES FINAIS

A performance do simulador pode ser considerada satisfatória, tendo em vista o nível de detalhe empregado ser bem maior que o usual. A simulação desce a níveis de pulsos de relógio ("clock"), para se obter maior flexibilidade, quanto a possíveis modificações e aumento de capacidade, e quanto a diminuição da complexidade, na simulação de dispositivos de E/S, bem como de eventos externos.

Uma vez que na transição de um estado para outro do processador, são ativadas rotinas de sincronização e verificação de eventos pendentes, tem-se, aí, um consumo relativamente grande do tempo de simulação. Principalmente no atendimento desses eventos (interrupções com prioridades, HOLD, WRITE e RESET).

Porém uma avaliação sistemática do simulador só poderá ser fornecida quando se fizer a comparação das instruções executadas, passo a passo, no simulador, com as mesmas instruções executadas no protótipo do microcomputador, que se encontra em fase de montagem.

AGRADECIMENTOS

Ao Instituto de Pesquisas Espaciais (INPE) e ao Instituto Tecnologico de Aeronáutica (ITA) de onde se destacam pela atuação neste trabalho:

Eng. Ricardo Corrêa de Oliveira Martins, M.Sc., orientador do INPE, com quem praticamente dividi os encargos deste trabalho, e que possibilitou a realização do mesmo.

Eng. Jacinto da Encarnação Cavaco Mendes, M.Sc., orientador no ITA.

Doroti Akico Tiba, que com grande dedicação datilografou êste trabalho.

José Benedito Soares Junior (Nino) que muito auxiliou na depuração de programas.

Os meus sinceros agradecimentos.

BIBLIOGRAFIA

BURROUGHS. *B.6700/B.7700 Algol Language: Reference Manual.* Detroit 1974.

CHU, Y. *Introduction to Computer Organization.* New Jersey, Prentice Hall, 1970.

ELECTRONICS. New York, V. 49, nº 8, Apr. 1976.

GEAR, C.W. *Computer Organization and Programming.* New York, McGraw-Hill, 1969.

GRIES, D. Compiler Construction for Digital Computers. New York, Wiley 1971. pp 50-83. Cap III.

IEEE Transactions on Industrial Electronics and Control Instrumentation. New York, V. 22, nº 3, Aug. 1975.

INTEL - 8080 Microprocessor User's Manual. Santa Clara, Ca., USA, 1975.

INTEL - 8080 Assembly Language Programming Manual. Santa Clara, Ca. USA, 1974.

INTEL - Intellec 8/Mod 80 Microcomputer Development System Reference Manual, Santa Clara, Ca., USA, 1974.

INTEL - Interp/80 User's Manual. Santa Clara, Ca., USA, 1975.

KNUTH, D.E. *The Art of Computer Programming.* 2 ed. Reading, Addison -Wesley, 1975, V.1

APÊNDICE A

LISTAGEM DE MONTAMEMORIA (MONTA)

TABELAIR

BURROUGHS B6700 ALGOL COMPILER, VERSION 2.8.001,   FRIDAY, 12/10/76, 04:42 PM,

```
                M O N T A M E M O R I A
                = = = = = = = = = = = = =

                                                                               000:0000:0
BEGIN                                                                 B.0000 IS SEGMENT 00003
                                                                1              003:0000:1
PROCEDURE  MONTAMEMORIA(PERFURA);                                              003:0000:1
BOOLEAN PERFURA;                                                               003:0000:1
BEGIN                                                                          003:0000:1
    FILE PENEIRA(KIND=PUNCH),                                   MONTAMEMORIA IS SEGMENT 00004
                                                                          DATA IS 0005 LONG
                                                                2              004:0000:1
        CARD(KIND=READER),                                                DATA IS 0005 LONG
                                                                               004:0000:1
        LINE(KIND=PRINTER),                                               DATA IS 0005 LONG
                                                                               004:0000:1
        LINHA(KIND=PRINTER);                                              DATA IS 0005 LONG
                                                                               004:0000:1
    INTEGER CONTA,CONTANT,MARGEM,P1,P2,I3;                                     004:0000:1
    REAL ARRAY CIN[0:13],  COUT[0:13], MEMORIA[1:4];                           004:0000:1
    POINTER PCOUT,PM;                                                          004:0000:1
    BOOLEAN FIM;                                                               004:0000:1
    %                                                                          004:0000:1
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%             004:0000:1
    %                                                                          004:0000:1
    PROCEDURE LERCARTAO;                                                       004:0000:1
    BEGIN                                                                      004:0000:1
        IF NOT FIM:=READ(CARD,14,CIN) THEN                      3              004:0005:4
            BEGIN                                               4              004:0006:1
                WRITE(LINE,<X10,A80>,POINTER(CIN));                            004:000E:2
                CONTANT:=READLOCK(INTEGER(POINTER(CIN)+11,4),CONTA);           004:0011:5
                IF CONTA LEQ CONTANT THEN                                      004:0012:4
                    WRITE(LINE,<"***  ERRO  ***  CARTOES FORA DE",            004:0015:0
                                "ORDEM">);                                DATA IS 0004 LONG
                                                                               004:0018:2
            END;                                                4              004:0018:2
    END;                                                        3              004:0019:4
    %                                                                          004:001A:4
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%             004:001A:4
    %                                                                          004:001A:4
    PROCEDURE  PERFURALISTA(I1,I2);                                            004:001A:4
    VALUE I1,I2;                                                               004:001A:4
    INTEGER I1,I2;                                                             004:001A:4
    BEGIN                                                                      004:001A:4
        INTEGER I;                                              PERFURALISTA IS SEGMENT 00008
                                                                3              008:0000:1
        REPLACE PCOUT:=POINTER(COUT) BY " " FOR 14 WORDS;                      008:0003:3
        PM:=POINTER(MEMORIA[I1],4);                                            008:0005:3
        PCOUT:=POINTER(COUT)+(16*(I1-1));                                      008:0008:3
        FOR I:=I1 STEP 1 UNTIL I2 DO                                           008:000C:3
            BEGIN                                               4              008:000C:3
                REPLACE PCOUT:PCOUT BY ",4"""" FOR 1,                          008:000F:4
                        PM:PM FOR 12 WITH HEXTOEBCDIC,                         008:0011:5
                        """ FOR 1;                                             008:0014:0
            END;                                                4              008:0014:3
        PCOUT:=POINTER(COUT)+67;                                               008:0016:4
        REPLACE PCOUT:PCOUT BY "%"FOR 1,
```

```
                        MARGEM+1 FOR 4 DIGIT:                                              008:0018:3
        IF PERFURA THEN  WRITE(PENEIRA,14,COUT);                                           008:001A:5
        WRITE(LINHA,14,COUT);                                                              008:0021:2
     END;                                                                                  008:0027:2
                                                                       PERFURALISTA(008)  IS 0029 LONG
  %                                                                      3                 004:001A:4
  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%                   004:001A:4
  %                                                                                        004:001A:4
  %  PROGRAMA PRINCIPAL                                                                    004:001A:4
  WRITE(LINE,<X10,120("=")>);                                                              004:001A:4
  WRITE(LINE,</,X40,"PROGRAMA FONTE",/>);                                                  004:001F:2
  WRITE(LINE,<X10,120("=")>);                                                              004:0024:2
  WRITE(LINHA,<X10,120("=")>);                                                             004:0029:2
  WRITE(LINHA,</X40,"PROGRAMA OBJETO",/>);                                                 004:002E:2
  WRITE(LINHA,<X10,120("=")>);                                                             004:0033:2
  %                                                                                        004:0036:2
  %  INICIALIZACOES                                                                        004:0038:2
  %                                                                                        004:0038:2
     LERCARTAO;                                                                            004:0038:2
  %                                                                                        004:0039:0
  WHILE NOT FIM DO                                                                         004:0039:0
     BEGIN                                                                                 004:0039:3
        %  ZERA A MEMORIA                                                3                 004:0039:5
        %                                                                                  004:0039:5
        FOR I3:=1 STEP 1 UNTIL 4 DO                                                        004:0039:5
              MEMORIA[I3]:=REAL(FALSE);                                                    004:003C:1
        MARGEM:=CONTA;                                                                     004:0041:3
        P2:=P1:=INTEGER((IF I3:=(CONTA MOD 8) = 0 THEN 8 ELSE I3)/2);                      004:0042:3
        MEMORIA[P2].[(47+24*(CONTA MOD 2)-24):8]:=                                         004:0048:0
                             INTEGER(POINTER(CIN),3);                                      004:004B:4
        MEMORIA[P2].[(47+24*(CONTA MOD 2)-32):16]:=                                        004:004E:0
                             INTEGER(POINTER(CIN)+5,1);                                    004:0052:0
        LERCARTAO;                                                                         004:0054:5
        WHILE NOT FIM  AND NOT(((CONTA-1)DIV 8) > ((MARGEM-1) DIV 8))                      004:0055:3
           DO  BEGIN                                                                       004:0057:5
                 P2:=INTEGER((IF I3:=(CONTA MOD 8)=0 THEN 8                      4         004:0059:1
                                   ELSE I3)/2);                                            004:005B:0
                 MEMORIA[P2].[(47+24*(CONTA MOD 2)-24):8]:=                                004:005E:0
                             INTEGER(POINTER(CIN),3);                                      004:0061:4
                 MEMORIA[P2].[(47+24*(CONTA MOD 2)-32):16]:=                               004:0064:0
                             INTEGER(POINTER(CIN)+5,1);                                    004:0068:0
                 LERCARTAO;                                                                004:006A:5
              END;                                                                         004:006B:3
        PERFURALISTA(P1,P2);                                             4                 004:006C:0
     END;                                                                                  004:006D:4
END;                                                                     3                 004:006E:1
                                                                       MONTAMEMORIA(004)  IS 0065 LONG
MONTAMEMORIA(FALSE);                                                     2                 003:0000:1
END.                                                                                       003:0002:1
                                                                          R,0000(003)  IS 0007 LONG
                                                                                 DATA  IS 0035 LONG
================================================================================================================
NUMBER OF ERRORS DETECTED = 0.
NUMBER OF SEGMENTS = 6.  TOTAL SEGMENT SIZE = 258 WORDS.  CORE ESTIMATE = 1620 WORDS.  STACK ESTIMATE = 31
PROGRAM SIZE = 92 CARDS, 515 SYNTACTIC ITEMS, 27 DISK SEGMENTS.
PROGRAM FILE NAME: (MICRO)MONTAMEMORIA.
COMPILATION TIME = 12.510 SECONDS ELAPSED; 1.083 SECONDS PROCESSING; 2.057 SECONDS I/O.
================================================================================================================
```

PROGRAMA OBJETO

```
,4"000000000000",4"000003010000",4"C10010000000",4"000000000017"   10002
,4"0E001B0F001F",4"000000000000",4"000024060028",4"020000000000"   10010
,4"00002F020032",4"000000000000",4"000000000000",4"000000000000"   10018
,4"000000000039",4"010030010042",4"050036050048",4"00000000004F"   10026
,4"010052010056",4"02005A02005E",4"000000000065",4"01006901006D"   10034
,4"05007100000C",4"000000000078",4"01007C010080",4"020084000000"   10042
,4"000000000000",4"00000000008B",4"000000000000",4"000000000094"   10050
,4"03009804009D",4"000000000000",4"0000000000A4",4"000000000000"   10058
,4"0000000000AE",4"0300B20400B7",4"000000000000",4"0000000000BF"   10066
,4"000000000000",4"0000000000C8",4"01000C040000",4"000000000000"   10074
,4"000000000007",4"0100D8000000",4"0000000000E2",4"000000000000"   10082
,4"0000000000EA",4"000000000000",4"0000000000F2",4"000000000000"   10090
,4"0000000000FA",4"000000000000",4"000000000102",4"000000000000"   10098
,4"00000000010A",4"000000000000",4"000000000111",4"000000000000"   10106
,4"000000000118",4"000000000000",4"000000000121",4"000000000000"   10114
,4"00000000012E",4"030120000000",4"000000000000",4"000133000137"   10122
,4"000000000000",4"00000000013D",4"000000000000",4"000000000146"   10130
,4"000000000000",4"000000000151",4"000000000000",4"00000000015C"   10138
,4"000000000000",4"000000000167",4"000000000000",4"000000000172"   10146
,4"000000000000",4"00000000017D",4"000000000000",4"000000000188"   10154
,4"000000000000",4"000000000193",4"000000000000",4"00000000019E"   10162
,4"0301A1000000",4"000000000000",4"00019C0301A8",4"000000000000"   10170
,4"00000000019C",4"0301B5000000",4"000000000000",4"00019D0301BF"   10178
,4"000000000000",4"00000000019D",4"0301C9000000",4"000000000000"   10186
,4"00019D0301D1",4"000000000000",4"00000000019D",4"0301DD000000"   10194
,4"000000000000",4"00019D0301E7",4"000000000000",4"0000000001F5"   10202
,4"0701FF070202",4"000000000000",4"00000000020A",4"0701FF070202"   10210
,4"000000000000",4"00000000020A",4"070211070215",4"000000000000"   10218
,4"00000000020A",4"07021C070220",4"000000000000",4"000000000227"   10226
,4"000000000000",4"00000000022F",4"000000000000",4"000000000237"   10234
,4"010240010244",4"000000000000",4"00000000024C",4"010240010244"   10242
,4"000000000000",4"00000000024C",4"0102530A0258",4"000000000000"   10250
,4"00000000024C",4"01025309025F",4"000000000000",4"000000000266"   10258
,4"000000000000",4"00000000026D",4"000000000000",4"00000000024C"   10266
,4"070274070278",4"08027C08027F",4"000000000000",4"000000000288"   10274
,4"000000000000",4"000000000291",4"010297010298",4"0802A00802A3"   10282
,4"000000000000",4"0000000002A7",4"0102970102AC",4"0802A00802A3"   10290
,4"000000000000",4"0000000002A7",4"0802B00802B3",4"000000000000"   10298
,4"0000000002A7",4"0802B80802BC",4"000000000000",4"0000000002C3"   10306
,4"0102C9000000",4"000000000000",4"0002C30102D3",4"000000000000"   10314
,4"0000000002C3",4"0102DD000000",4"000000000000",4"0002C30102E7"   10322
,4"000000000000",4"0000000002C3",4"0102F1000000",4"000000000000"   10330
,4"0002C30102FB",4"000000000000",4"0000000002C3",4"010305000000"   10338
,4"000000000000",4"0002C301030F",4"000000000000",4"000000000319"   10346
,4"0F031F080322",4"000000000000"                                   10354
```

```
                        1     <=== ACP
                        2
000  0008               3     <=== LXI RP
001  0012               4        =
001  0016               5
000  0000               6
                        7
000  0023               8     <=== DAD RP
014  0037               9        =
014  0130              10
006  0300              11
                       12
000  0036              13     <=== STAX RP
006  0240              14        =
000  0000              15
                       16
000  0024              17     <=== LDAX RP
002  0050              18        =
000  0000              19
                       20
                       21
                       22
                       23
                       24
                       25
000  0047              26     <=== SHLD
001  0091              27        =
001  0046              28
005  0040              29
005  0040              30
000  0010              31
000  0026              32     <=== LHLD
001  0052              33        =
001  0016              34
002  0030              35
002  0034              36
000  0000              37
000  0101              38     <=== STA
001  0105              39        =
001  0100              40
005  0113              41
000  0000              42
                       43
000  0120              44     <=== LDA
001  0124              45        =
001  0128              46
003  0132              47
000  0000              48
                       49
                       50
                       51
000  0139              52     <=== INX RP / DCRX RP
000  0000              53        =
                       54
                       55
000  0148              56     <=== INRM
```

```
003  0152     57        =
004  0157     58
000  0000     59
              60
              61
000  0164     62     <=== INRR
000  0000     63        =
              64
              65
000  0174     66     <=== DCRM
003  0177     67        =
004  0183     68
000  0000     69
              70
              71
000  0190     72     <=== DCRR
000  0000     73        =
              74
              75
000  0201     76     <=== MVIM
001  0202     77        =
004  0206     78
000  0001     79
              80
              81
000  0215     82     <=== MVIR
001  0216     83        =
              84
              85
000  0224     86     <=== RLC
000  0000     87        =
              88
              89
000  0234     90     <=== RRC
000  0000     91        =
              92
              93
000  0242     94     <=== RAL
000  0000     95        =
              96
              97
000  0250     98     <=== RAR
000  0000     99        =
             100
             101
000  0258    102     <=== DAA
000  0000    103        =
             104
             105
000  0266    106     <=== CMA
000  0000    107        =
             108
             109
000  0273    110     <=== STC
000  0000    111        =
             112
             113
000  0280    114     <=== CMC
000  0000    115        =
             116
             117
```

```
000  0219  118   <===  HALT
000  0000  119       =
           120
           121
000  0226  122   <===  MOVRM
003  0300  123       =
000  0000  124
           125
           126
000  0307  127   <===  MOVMR
004  0311  128       =
000  0000  129
           130
           131
000  0317  132   <===  MOVR1R2
000  0000  133       =
           134
           135
000  0326  136   <===  ADDR
000  0000  137       =
           138
           139
000  0337  140   <===  ADCR
000  0000  141       =
           142
           143
000  0348  144   <===  SUBR
000  0000  145       =
           146
           147
000  0359  148   <===  SBBR
000  0000  149       =
           150
           151
000  0370  152   <===  ANAR
000  0000  153       =
           154
           155
000  0381  156   <===  XRAR
000  0000  157       =
           158
           159
000  0392  160   <===  ORAR
000  0000  161       =
           162
           163
000  0403  164   <===  CMPR
000  0000  165       =
           166
           167
000  0413  168   <===  ADDM
003  0417  169       =
000  0000  170
           171
           172
000  0413  173   <===  ADCM
003  0427  174       =
000  0000  175
           176
           177
000  0413  178   <===  SUBM
```

```
003  0437   179        =
000  0000   180
            181
            182
000  0413   183   <===  SPRM
003  0447   184        =
000  0000   185
            186
            187
000  0413   188   <===  ANAM
003  0457   189        =
000  0000   190
            191
            192
000  0413   193   <===  XRAM
003  0467   194        =
000  0000   195
            196
            197
000  0413   198   <===  ORAM
003  0477   199        =
000  0000   200
            201
            202
000  0413   203   <===  CMPM
003  0487   204        =
003  0000   205
            206
            207
000  0501   208   <===  RET COND
007  0510   209        =
007  0514   210
000  0000   211
            212
            213
000  0502   214   <===  RET
007  0510   215        =
007  0514   216
000  0000   217
            218
            219
000  0502   220   <===  POP RP
007  0509   221        =
007  0533   222
000  0000   223
            224
            225
000  0502   226   <===  POP PSW
007  0540   227        =
007  0574   228
000  0000   229
            230
            231
000  0561   232   <===  PCHL
000  0000   233        =
            234
            235
000  0502   236   <===  SPHL
000  0000   237        =
            238
            239
```

```
000  0537   240     <===  JMP COND
001  0536   241       =
001  0580   242
000  0000   243
            244
            245
000  0538   246     <===  JMP
001  0536   247       =
001  0580   248
000  0000   249
            250
            251
000  0538   252     <===  OUT
001  0535   253       =
010  0600   254
000  0030   255
            256
            257
000  0538   258     <===  IN
001  0535   259       =
000  0607   260
000  0030   261
            262
            263
000  0614   264     <===  DI
000  0030   265       =
            266
            267
000  0621   268     <===  EI
000  0000   269       =
            270
            271
000  0588   272     <===  XTHL
007  0638   273       =
007  0632   274
008  0636   275
008  0639   276
000  0000   277
            278
            279
000  0648   280     <===  XCHG
000  0000   281       =
            282
            283
000  0657   284     <===  CALL COND
001  0663   285       =
001  0667   286
008  0672   287
008  0675   288
000  0000   289
            290
            291
000  0679   292     <===  CALL
001  0663   293       =
001  0684   294
008  0672   295
008  0675   296
000  0000   297
            298
            299
000  0679   300     <===  PUSH RP
```

```
018  0668   301        =
008  0661   302
003  0000   303
               304
               305
000  0629   306    <===  PUSH PSW
008  0667   307        =
008  0700   308
000  0000   309
               310
               311
000  0707   312    <===  ADI
001  0713   313        =
000  0000   314
               315
               316
000  0707   317    <===  ACI
001  0723   318        =
000  0030   319
               320
               321
000  0707   322    <===  SUI
001  0743   323        =
000  0000   324
               325
               326
000  0707   327    <===  SBI
001  0713   328        =
000  0030   329
               330
               331
000  0707   332    <===  ANI
001  0753   333        =
000  0030   334
               335
               336
000  0707   337    <===  XRI
001  0753   338        =
000  0030   339
               340
               341
000  0707   342    <===  ORI
001  0773   343        =
000  0006   344
               345
               346
000  0707   347    <===  CPI
001  0783   348        =
000  0005   349
               350
               351
000  0703   352    <===  RST
008  0706   353        =
008  0602   354
000  0000   355
```

APÊNDICE B

MICROPROGRAMA

PROGRAMA OBJETO

```
=====================================================================
,4"460003490004",4"460000000000",4"000000000000",4"47000946000A"   $0002
,4"490009460000",4"47000001000E",4"10000F460000",4"470011010012"   $0010
,4"100013460000",4"000000000000",4"000000470018",4"46001949001A"   $0018
,4"46000040001C",4"480010460000",4"4A001F4C0020",4"460000000000"   $0026
,4"000000000000",4"470025460026",4"490027460000",4"14002902002A"   $0034
,4"460000000000",4"000000000000",4"47002F460030",4"490031460000"   $0042
,4"470033010034",4"150035460000",4"000000000000",4"00000047003A"   $0050
,4"46003B49003C",4"46000047003E",4"01003F130040",4"460000470042"   $0058
,4"010043110044",4"4600000F0046",4"020047460000",4"00004902004A"   $0066
,4"460000000000",4"000000000000",4"47004F460050",4"490051460052"   $0074
,4"470053010054",4"130055460000",4"470057010058",4"110059460000"   $0082
,4"47005B01005C",4"0F005D460000",4"47005F010060",4"000061460000"   $0090
,4"000000000000",4"000000470066",4"460067490068",4"46000047006A"   $0098
,4"01006B13006C",4"46000047006E",4"01006F110070",4"460000470072"   $0106
,4"140073020074",4"460000000000",4"000000000000",4"47007946007A"   $0114
,4"49007B460000",4"47007D01007E",4"13007F460000",4"470081010082"   $0122
,4"110083460000",4"470085010086",4"150087460000",4"000000000000"   $0130
,4"00000047008C",4"46008D28008E",4"46008F290090",4"460000000000"   $0138
,4"000000000000",4"470095460096",4"490097460000",4"47009901009A"   $0146
,4"1F009B24009C",4"23009D46009E",4"47009F2000A0",4"0200A1460000"   $0154
,4"000000000000",4"4700A54600A6",4"2700A72400A8",4"2300A94600AA"   $0162
,4"2200AB460000",4"000000000000",4"4700AF4600B0",4"4900B1460000"   $0170
,4"4700B30100B4",4"1F00B52500B6",4"2300B7460000",4"4700B92000BA"   $0178
,4"0200BB460000",4"000000000000",4"4700BF4600C0",4"2700C12500C2"   $0186
,4"2300C34600C4",4"2200C5460000",4"000000000000",4"4700C94600CA"   $0194
,4"4900CB460000",4"4700CD0100CE",4"1F00CF460000",4"4700D11F00D2"   $0202
,4"0200D3460000",4"000000000000",4"0000004700D8",4"4600D94900DA"   $0210
,4"4600004700DC",4"0100DD1900DE",4"460000000000",4"000000000000"   $0218
,4"4700E34600E4",4"4200E52100E6",4"460000000000",4"000000000000"   $0226
,4"4700EB4600EC",4"4300ED2100EE",4"460000000000",4"000000000000"   $0234
,4"4700F34600F4",4"4400F52100F6",4"460000000000",4"000000000000"   $0242
,4"4700FB4600FC",4"4500FD2100FE",4"460000000000",4"000000000000"   $0250
,4"470103460104",4"410105210106",4"460000000000",4"000000000000"   $0258
,4"47010B46010C",4"34010D460000",4"000000000000",4"000000470112"   $0266
,4"460113350114",4"460000000000",4"000000000000",4"47011946011A"   $0274
,4"36011B460000",4"000000000000",4"000000000000",4"000000470122"   $0282
,4"460123490124",4"460000000000",4"000000000000",4"47012946012A"   $0290
,4"49012B460000",4"47012D01012E",4"19012F460000",4"000000000000"   $0298
,4"000000470134",4"460135330136",4"4600001B0138",4"020139460000"   $0306
,4"000000000000",4"00000047013E",4"46013F330140",4"460141320142"   $0314
,4"460143000000",4"000000000000",4"470147460148",4"33014931014A"   $0322
,4"46014B2A014C",4"23014D210000",4"000000000000",4"000000470152"   $0330
,4"460153330154",4"310155460156",4"2B0157230158",4"210000000000"   $0338
,4"000000000000",4"47015D46015E",4"33015F310160",4"460161200162"   $0346
,4"230163210000",4"000000000000",4"000000470168",4"46016933016A"   $0354
,4"31016B46016C",4"20016D23016E",4"210000000000",4"000000000000"   $0362
,4"470173460174",4"330175310176",4"4601772F0178",4"230179210000"   $0370
,4"000000000000",4"00000047017E",4"46017F330180",4"310181460182"   $0378
,4"2F0183230184",4"210000000000",4"000000000000",4"47018946018A"   $0386
,4"33018B31018C",4"46018D30018E",4"23018F210000",4"000000000000"   $0394
,4"000000470194",4"460195330196",4"310197460198",4"200199230000"   $0402
,4"000000000000",4"00000047019E",4"46019F3101A0",4"4600004701A2"   $0410
,4"0101A31F01A4",4"4601A52A01A6",4"230000210000",4"000000000000"   $0418
,4"0000004701AC",4"0101AD1F01AE",4"4601AF2B01B0",4"2301B1210000"   $0426
,4"000000000000",4"0000004701B6",4"0101B71F01B8",4"4601B92001BA"   $0434
,4"2301BB210000",4"000000000000",4"0000004701C0",4"0101C11F01C2"   $0442
```

,4"4601C3?001??",4"2301C5210000",4"000000000000",4"0000004701CA" 10450
,4"0101C901F01?",4"4601C0?F01CE",4"2301CF210000",4"000000000000" 10458
,4"0000004701??",4"0101D51F01D6",4"4601D72F01D8",4"2301D9210000" 10466
,4"000000000000",4"0000004701DE",4"0101DF1F01E0",4"4601E13001E2" 10474
,4"2301E3210000",4"000000000000",4"0000004701E8",4"0101E91F01EA" 10482
,4"4601EB2C01EC",4"230000000000",4"000000000000",4"000000000000" 10490
,4"000000000000",4"0000004701F6",4"4601F73301F8",4"4601F94001FA" 10498
,4"460000000000",4"000000000000",4"4701FFC10200",4"130201460000" 10506
,4"47020?010204",4"1102053B0206",4"460000000000",4"000000000000" 10514
,4"47020946020C",4"490200460000",4"000000000000",4"000000470212" 10522
,4"0102131A0214",4"460000470216",4"0102171B0218",4"460000000000" 10530
,4"000000000000",4"47021D01021E",4"17021F460000",4"470221010222" 10538
,4"150223460000",4"000000000000",4"000000470228",4"46022946022A" 10546
,4"37022B460000",4"000000000000",4"000000470230",4"460231460232" 10554
,4"340233460000",4"000000000000",4"000000470238",4"46023939023A" 10562
,4"46023B4F023C",4"460000000000",4"000000000000",4"470241010242" 10570
,4"130243460000",4"470245010246",4"110247390248",4"460000000000" 10578
,4"000000000000",4"47024D46024E",4"49024F460000",4"000000000000" 10586
,4"000000470254",4"010255130256",4"110257460000",4"14025904025A" 10594
,4"4F025B460000",4"000000000000",4"000000470260",4"030261150262" 10602
,4"460000000000",4"000000000000",4"470267460268",4"510269460000" 10610
,4"000000000000",4"00000047026E",4"46026F500270",4"460000000000" 10618
,4"000000000000",4"470275010276",4"130277460000",4"47027901027A" 10626
,4"11027B460000",4"00027D00027F",4"460000CF0280",4"020281460282" 10634
,4"460283300284",4"460000000000",4"000000000000",4"47028946028A" 10642
,4"3F028B30028C",4"3F028D460000",4"000000000000",4"000000470292" 10650
,4"460293390294",4"460295520296",4"460000470298",4"01029913029A" 10658
,4"46000047029C",4"01029D11029E",4"4F029F460000",4"0A02A10202A2" 10666
,4"4600000802A4",4"0202A54602A6",4"3800004702A8",4"4602A94602AA" 10674
,4"5202AB460000",4"4702AD0102AE",4"1102AF460000",4"1B02B10202B2" 10682
,4"4600001402B4",4"0202B5460000",4"000000000000",4"0000001402B4" 10690
,4"0202BB460000",4"1602BD0202BE",4"460000000000",4"000000000000" 10698
,4"0000004702C4",4"4602C53102C6",4"460000000000",4"0000004702CA" 10706
,4"0102CB1F02CC",4"4602CD2A02CE",4"2302CF210000",4"000000000000" 10714
,4"0000004702D4",4"0102D51F02D6",4"4602D72B02D8",4"2302D92102DA" 10722
,4"000000000000",4"0000004702DE",4"0102DF1F02E0",4"4602E12C02E2" 10730
,4"2302E3210000",4"000000000000",4"0000004702E8",4"0102E91F02EA" 10738
,4"4602EB2002EC",4"2302ED2102EE",4"000000000000",4"0000004702F2" 10746
,4"0102F31F02F4",4"4602F52F02F6",4"2302F72102F8",4"000000000000" 10754
,4"0000004702FC",4"0102FD1F02FE",4"4602FF2F0300",4"230301210000" 10762
,4"000000000000",4"000000470306",4"0103071F0308",4"46030930030A" 10770
,4"23030B210000",4"000000000000",4"000000470310",4"0103111F0312" 10778
,4"460313200314",4"230000000000",4"000000000000",4"00000047031A" 10786
,4"54031B46031C",4"46031D53031E",4"460000A00320",4"020321460000" 10794
,4"080323020324",4"050325460326",4"380000000000" 10802

```
==================================================================================================================================
                         PROGRAMA FONTE
==================================================================================================================================
070  0003  0001       =  INSTRUCAO NOP              CODIGO 00 000 000
073  0004  0002       =            ===                     == === ===
070  0000  0003    <===  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
000  0000  0004
000  0000  0005
000  0000  0006
071  0009  0007    <===  INSTRUCAO LXI RP           CODIGO 00 RP0 001
070  0010  0008       =            === ==                  == === ===
073  0011  0009       =  ESTADO T3  DO CICLO(=0) DE FETCH
070  0000  0010    <===  ESTADO T4
071  0013  0011    <===
001  0014  0012       =
029  0015  0013       =  ESTADO T3  DO CICLO(=1) DE MEMREAD
070  0000  0014    <===
071  0017  0015    <===
001  0018  0016       =
029  0019  0017       =  ESTADO T3  DO CICLO(=1) DE MEMREAD
070  0000  0018    <===
000  0000  0019
000  0000  0020
000  0000  0021
071  0024  0022    <===  INSTRUCAO DAD RP           CODIGO 00 RP1 001
070  0025  0023       =            === ==                  == === ===
073  0026  0024       =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0025    <===
074  0028  0026    <===
075  0029  0027       =  ESTADO T3 DO CICLO(=14) DE REGISTRO
070  0000  0028    <===
074  0031  0029    <===
076  0032  0030       =  ESTADO T3 DO CICLO(=14) DE REGISTRO
070  0000  0031    <===
000  0000  0032
000  0000  0033
000  0000  0034
071  0037  0035    <===  INSTRUCAO STAX RP          CODIGO 00 RP0 010
070  0038  0036       =            ==== ==                 == === ===
073  0039  0037       =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0038    <===
020  0041  0039    <===
002  0042  0040       =  ESTADO T3 DO CICLO(=6) DE MEMWRITE
070  0000  0041    <===
000  0000  0042
000  0000  0043
000  0000  0044
071  0047  0045    <===  INSTRUCAO LDAX RP          CODIGO 00 RP1 010
070  0048  0046       =            ==== ==                 == === ===
073  0049  0047       =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0048    <===
071  0051  0049    <===
001  0052  0050       =
021  0053  0051       =  ESTADO T3 DO CICLO(=2) DE MEMREAD
070  0000  0052    <===
000  0000  0053
000  0000  0054
000  0000  0055
071  0058  0056    <===  INSTRUCAO SHLD             CODIGO 00 100 010
```

```
070  0059  0057     =
073  0060  0058     =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0059  <===
071  0062  0060  <===
001  0063  0061     =  ESTADO T3 DO CICLO(=1) DE MEMREAD
019  0064  0062     =
070  0000  0063  <===
071  0066  0064  <===
001  0067  0065     =  ESTADO T3 DO CICLO(=1) DE MEMREAD
017  0068  0066     =
070  0000  0067  <===
014  0070  0068  <===
002  0071  0069     =  ESTADO T3 DO CICLO(=5) DE MEMWRITE
070  0000  0070  <===
012  0073  0071  <===
002  0074  0072     =  ESTADO T3 DO CICLO(=5) DE MEMWRITE
070  0000  0073  <===
000  0000  0074
000  0000  0075
000  0000  0076
071  0079  0077  <===  INSTRUCAO LHLD          CODIGO 00 101 010
070  0080  0078     =             ====               == === ===
073  0081  0079     =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0082  0080  <===
071  0083  0081  <===
001  0084  0082     =
019  0085  0083     =  ESTADO T3 DO CICLO(=1) DE MEMREAD
070  0000  0084  <===
071  0087  0085  <===
001  0088  0086     =  ESTADO T3 DO CICLO(=1) DE MEMREAD
017  0089  0087     =
070  0000  0088  <===
071  0091  0089  <===
001  0092  0090     =
015  0093  0091     =  ESTADO T3 DO CICLO(=2) DE MEMREAD
070  0000  0092  <===
071  0095  0093  <===
001  0096  0094     =
013  0097  0095     =  ESTADO T3 DO CICLO(=2) DE MEMREAD
070  0000  0096  <===
000  0000  0097
000  0000  0098
000  0000  0099
071  0102  0100  <===  INSTRUCAO STA           CODIGO 00 110 010
070  0103  0101     =            ===                 == === ===
073  0104  0102     =  ESTADOS T3 E T4 DO CICLO(=0)-DE FETCH
070  0000  0103  <===
071  0106  0104  <===
001  0107  0105     =
019  0108  0106     =  ESTADO T3 DO CICLO(=1) DE MEMREAD
070  0000  0107  <===
071  0110  0108  <===
001  0111  0109     =
017  0112  0110     =  ESTADO T3 DO CICLO(=1) DE MEMREAD
070  0000  0111  <===
071  0114  0112  <===
020  0115  0113     =
002  0116  0114     =  ESTADO T3 DO CICLO(=5) DE MEMWRITE
070  0000  0115  <===
000  0000  0116
000  0000  0117
```

```
071  0121  0119    <===  INSTRUCAO LDA              CODIGO 00 111 010
070  0122  0120       =                ===                  == === ===
073  0123  0121       =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH -
070  0000  0122    <===
071  0125  0123    <===
001  0126  0124       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
019  0127  0125       =
070  0000  0126    <===
071  0129  0127    <===
001  0130  0128       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
017  0131  0129       =
070  0000  0130    <===
071  0133  0131    <===
001  0134  0132       =
021  0135  0133       =  ESTADO T3 DO CICLO(=2) DE MEMREAD
070  0000  0134    <===
000  0000  0135
000  0000  0136
000  0000  0137
071  0140  0138    <===  INSTRUCAO INX RP           CODIGO 00 RP0 011
070  0141  0139       =  ********** DCRX RP                00 RP1 011
040  0142  0140       =             ==== ==                 == === ===
070  0143  0141       =  ESTADOS T3, T4 E T5 DO CICLO(=0) DE FETCH
041  0144  0142       =
070  0000  0143    <===
000  0000  0144
000  0000  0145
000  0000  0146
071  0149  0147    <===  INSTRUCAO INRM             CODIGO 00 110 100
070  0150  0148       =             ====                    == === ===
073  0151  0149       =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0150    <===
071  0153  0151    <===
001  0154  0152       =
031  0155  0153       =  ESTADO T3 DO CICLO(=3) DE MEMREAD
036  0156  0154       =
035  0000  0155       =
070  0158  0156    <===
071  0159  0157    <===
032  0160  0158       =  ESTADO T3 DO CICLO(=4) DE MEMWRITE
002  0161  0159       =
070  0000  0160    <===
000  0000  0161
000  0000  0162
071  0165  0163    <===  INSTRUCAO INRR             CODIGO 00 DDD 100
070  0166  0164       =             ====                    == === ===
039  0167  0165       =
036  0168  0166       =  ESTACOS T3, T4 E T5 DO CICLO(=0) DE FETCH
035  0169  0167       =
070  0170  0168       =
034  0171  0169       =
070  0000  0170    <===
000  0000  0171
000  0000  0172
071  0175  0173    <===  INSTRUCAO DCRM             CODIGO 00 110 101
070  0176  0174       =             ====                    == === ===
073  0177  0175       =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0176
071  0179  0177    <===
001  0180  0178       =
031  0181  0179       =  ESTADO T3 DO CICLO(=3) DE MEMREAD
```

```
037  0182  0180      =
035  0183  0181      =
070  0000  0182    <===
071  0185  0183    <===
032  0184  0184      =  ESTADO T3 DO CICLO(=4) DE MEMORIA
002  0187  0185      =
070  0000  0186    <===
000  0000  0187
000  0000  0188
071  0191  0189    <===  INSTRUCAO DCRR            CODIGO 01 000 101
070  0192  0190      =           ====                    =: === ===
039  0193  0191      =
037  0194  0192      =  ESTADOS T3, T4 E T5 DO CICLO(=0) DE FETCH
035  0195  0193      =
070  0196  0194      =
034  0197  0195      =
070  0000  0196    <===
000  0000  0197      -
000  0000  0198
071  0201  0199    <===  INSTRUCAO MVIM            CODIGO 01 110 110
070  0202  0200      =           ====                    =: === ===
073  0203  0201      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0202    <===
071  0205  0203    <===
001  0206  0204      =
031  0207  0205      =  ESTADO T3 DO CICLO(=1) DE MEMREAD
070  0000  0206    <===
071  0209  0207    <===
030  0210  0208      =
002  0211  0209      =  ESTADO T3 DO CICLO(=4) DE MEMWRITE
070  0000  0210    <===
000  0000  0211
000  0000  0212
000  0000  0213
071  0216  0214    <===  INSTRUCAO MVIR            CODIGO 00 000 110
070  0217  0215      =           ====                    =: === ===
073  0218  0216      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0217    <===
071  0220  0218    <===
001  0221  0219      =
025  0222  0220      =  ESTADO T3 DO CICLO(=1) DE MEMREAD
070  0000  0221    <===
000  0000  0222
000  0000  0223
000  0000  0224
071  0227  0225    <===  INSTRUCAO RLC             CODIGO 00 000 111
070  0228  0226      =           ===                     =: === ===
066  0229  0227      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
033  0230  0228      =
070  0000  0229    <===
000  0000  0230
000  0000  0231
000  0000  0232
071  0235  0233    <===  INSTRUCAO RRC             CODIGO 00 001 111
070  0237  0234      =           ===                     =: === ===
067  0238  0235      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
033  0239  0236      =
070  0000  0237    <===
000  0000  0238
001  0000  0239
000  0000  0240
```

```
071  0243  0241   <===  INSTRUCAO RAL              CODIGO 00 010 111
070  0244  0242      =              ===                   == === ===
069  0245  0243      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
033  0246  0244      =
070  0000  0245   <===
000  0000  0246
000  0000  0247
000  0000  0248
071  0251  0249   <===  INSTRUCAO RAR              CODIGO 00 011 111
070  0252  0250      =              ===                   == === ===
069  0253  0251      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
033  0254  0252      =
070  0000  0253   <===
000  0000  0254
000  0000  0255
000  0000  0256
071  0259  0257   <===  INSTRUCAO DAA              CODIGO 00 100 111
070  0260  0258      =              ===                   == === ===
065  0261  0259      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
033  0262  0260      =
070  0000  0261   <===
000  0000  0262
000  0000  0263
000  0000  0264
071  0267  0265   <===  INSTRUCAO CMA              CODIGO 00 101 111
070  0268  0266      =              ===                   == === ===
052  0269  0267      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0268   <===
000  0000  0269
000  0000  0270
000  0000  0271
071  0274  0272   <===  INSTRUCAO STC              CODIGO 00 110 111
070  0275  0273      =              ===                   == === ===
053  0276  0274      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0275   <===
000  0000  0276
000  0000  0277
000  0000  0278
071  0281  0279   <===  INSTRUCAO CMC              CODIGO 00 111 111
070  0282  0280      =              ===                   == === ===
054  0283  0281      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0282   <===
000  0000  0283
000  0000  0284
000  0000  0285
000  0000  0286
000  0000  0287
071  0290  0288   <===  INSTRUCAO HALT             CODIGO 01 110 110
070  0291  0289      =              ===                   == === ===
073  0292  0290      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0291   <===  PROXIMO CICLO(=12) HALTACK ===> HALT MODE
000  0000  0292
000  0000  0293
000  0000  0294
071  0297  0295   <===  INSTRUCAO MOVRM            CODIGO 01 000 110
070  0298  0296      =              =====                 == === ===
074  0299  0297      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0298   <===
071  0301  0299   <===
001  0302  0300      =
025  0303  0301      =  ESTADO T3 DO CICLO(=3) DE MEMREAD
```

```
070  0000  0302    <===
000  0000  0303
000  0000  0304
000  0000  0305
071  0308  0306    <===  INSTRUCAO MOVMR          CODIGO 01  10 SSS
070  0309  0307      =           =====                  ==  ===  ===
051  0310  0308      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0309    <===
024  0312  0310    <==
002  0313  0311      =  ESTADO T3 DO CICLO(=4) DE MEMWRITE
070  0000  0312    <===
000  0000  0313
000  0000  0314
000  0000  0315
071  0318  0316    <===  INSTRUCAO MOVR1R2        CODIGO 01  000  SSS
070  0319  0317      =           =======                ==  ===  ===
051  0320  0318      =
070  0321  0319      =  ESTADOS T3, T4 E T5 DO CICLO(=0) DE FETCH
050  0322  0320      =
070  0323  0321    <===
000  0000  0322
000  0000  0323
000  0000  0324
071  0327  0325    <===  INSTRUCAO ADDR           CODIGO 10  000  SSS
070  0328  0326      =           ====                   ==  ===  ===
051  0329  0327      =
049  0330  0328        ESTADOS T3 E T4 DO CICLO(=0) DE FE
070  0331  0329      =
042  0332  0330      =
035  0333  0331      =
033  0000  0332    <===
000  0000  0333
000  0000  0334
000  0000  0335
071  0338  0336    <===  INSTRUCAO ADCR           CODIGO 10  001  SSS
070  0339  0337      =           ====                   ==  ===  ===
051  0340  0338      =
049  0341  0339      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0342  0340      =
043  0343  0341      =
035  0344  0342      =
033  0000  0343    <===
000  0000  0344
000  0000  0345
000  0000  0346
071  0349  0347    <===  INSTRUCAO SUBR           CODIGO 10  010  SSS
070  0350  0348      =           ====                   ==  ===  ===
051  0351  0349      =
049  0352  0350      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0353  0351      =
044  0354  0352      =
035  0355  0353      =
033  0000  0354    <===
000  0000  0355
000  0000  0356
000  0000  0357
071  0360  0358    <===  INSTRUCAO SBBR           CODIGO 10  011  SSS
070  0361  0359      =           ====                   ==  ===  ===
051  0362  0360      =
049  0363  0361      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0364  0362      =
```

```
045  0365  0363      =
035  0366  0364      =
033  0000  0365    <===
000  0000  0366
000  0000  0367
000  0000  0368
071  0371  0369    <===  INSTRUCAO ANAR            CODIGO 10 100 SSS
070  0372  0370      =             ====                   == === ===
051  0373  0371      =
049  0374  0372      =   ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0375  0373      =
046  0376  0374      =
035  0377  0375      =
033  0000  0376    <===
000  0000  0377
000  0000  0378
000  0000  0379
071  0382  0380    <===  INSTRUCAO XRAR            CODIGO 10 101 SSS
070  0383  0381      =             ====                   == === ===
051  0384  0382      =
049  0385  0383      =   ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0386  0384      =
047  0387  0385      =
035  0388  0386      =
033  0000  0387    <===
000  0000  0388
000  0000  0389
000  0000  0390
071  0393  0391    <===  INSTRUCAO ORAR            CODIGO 10 110 SSS
070  0394  0392      =             ====                   == === ===
051  0395  0393      =
049  0396  0394      =   ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0397  0395      =
048  0398  0396      =
035  0399  0397      =
033  0000  0398    <===
000  0000  0399
000  0000  0400
000  0000  0401
071  0404  0402    <===  INSTRUCAO CMPR            CODIGO 10 111 SSS
070  0405  0403      =             ====                   == === ===
051  0406  0404      =
049  0407  0405      =   ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0408  0406      =
043  0409  0407      =
035  0000  0408    <===
000  0000  0409
000  0000  0410
000  0000  0411
071  0414  0412    <===                            CODIGO 10 XXX 110
070  0415  0413      =                                    == === ===
049  0416  0414      =   ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0415    <===
071  0418  0416    <===  INSTRUCAO ADDM            CODIGO 10 000 110
001  0419  0417      =             ====                   == === ===
031  0420  0418      =
070  0421  0419      =   ESTADO T3 DO CICLO(=3) DE MEMREAD
042  0422  0420      =
035  0000  0421      =
033  0000  0422    <===
           0423
```

```
071  0428   <===  INSTRUCAO ACCM          CODIGO 10 001 110
001  0429     =             ====                 == === ===
031  0430     =
070  0431     =   ESTADO T3 DO CICLO(=3) DE MEMREAD
043  0432     =
035  0433     =
033  0000   <===

071  0438   <===  INSTRUCAO SUBM          CODIGO 10 010 110
001  0439     =             ====                 == === ===
031  0440     =
070  0441     =   ESTADO T3 DO CICLO(=3) DE MEMREAD
044  0442     =
035  0443     =
033  0000   <===

071  0448   <===  INSTRUCAO SBBM          CODIGO 10 011 110
001  0449     =             ====                 == === ===
031  0450     =
070  0451     =   ESTADO T3 DO CICLO(=3) DE MEMREAD
045  0452     =
035  0453     =
033  0000   <===

071  0458   <===  INSTRUCAO ANAM          CODIGO 10 100 110
001  0459     =             ====                 == === ===
031  0460     =
070  0461     =   ESTADO T3 DO CICLO(=3) DE MEMREAD
046  0462     =
035  0463     =
033  0000   <===

071  0468   <===  INSTRUCAO XRAM          CODIGO 10 101 110
001  0469     =             ====                 == === ===
031  0470     =
070  0471     =   ESTADO T3 DO CICLO(=3) DE MEMREAD
047  0472     =
035  0473     =
033  0000   <===

071  0478   <===  INSTRUCAO ORAM          CODIGO 10 110 110
001  0479     =             ====                 == === ===
031  0480     =
070  0481     =   ESTADO T3 DO CICLO(=3) DE MEMREAD
048  0482     =
035  0483     =
033  0000   <===
```

```
071  0488  <===  INSTRUCAO CMPM             CODIGO 10 111 110
001  0489     =              ====                  == === ===
031  0490     =
070  0491     =  ESTADO T3 DO CICLO(=3) DE MEMREAD
044  0492     =
035  0000  <===

071  0502  <===  INSTRUCAO RETCCND          CODIGO 11 CCC 000
070  0503     =            =======
051  0504     =  ESTADOS T3,T4 E T5 DO CICLO(=0) DE FETCH
070  0505     =  CCC= 000 ==> RNZ       CCC= 100 ==> RPO
078  0506     =       001 ==> RZ             101 ==> RPE
070  0000  <===       010 ==> RNC            110 ==> RP
000  0000             011 ==> RC             111 ==> RM
000  0000
000  0000
071  0511  <===
001  0512     =  ESTADO T3 DO CICLO(=7) DE STACKREAD
019  0513     =  PARA RETURN CONDITION E RETURN SEGUNDO CICLO
070  0000  <===
071  0515  <===
001  0516     =  ESTADO T3 DO CICLO(=7) DE STACKREAD
017  0517     =
056  0518     =  PARA RETURN CONDITION E RETURN TERCEIRO CICLO
070  0000  <===

071  0523  <===  INSTRUCAO RET               CODIGO 1  001 001
070  0524     =  ********** POPRP,POPPSW          1  RP0 001
073  0525     =            ===== ======          =  === ===
070  0000  <===  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH

071  0530  <===  INSTRUCAO POPRP
001  0531     =            =====
026  0532     =  ESTADO T3 DO CICLO(=7) DE STACKREAD 2
070  0000  <===
071  0534  <===
001  0535     =  ESTADO T3 DO CICLO(=7) DE STACKREAD 3
027  0536     =
070  0000  <===

071  0541  <===  INSTRUCAO POPPSW
001  0542     =            ======
023  0543     =  ESTADO T3 DO CICLO(=7) DE STACKREAD 2
070  0000  <===
071  0545  <===
001  0546     =  ESTADO T3 DO CICLO(=7) DE STACKREAD 3
021  0547     =
```

```
070  0000  0546   <===
                  0547
                  0548
                  0549
071  0552  0550   <===  INSTRUCAO PCHL              CODIGO 11 101 001
070  0553  0551     =                 ====                 == === ===
070  0554  0552     =
055  0555  0553     =  ESTADOS T3, T4 E T5 DO CICLO(=0) DE FETCH
070  0000  0554   <===
                  0555
                  0556
                  0557
071  0560  0558   <===  INSTRUCAO SPHL              CODIGO 11 111 001
070  0561  0559     =                 ====                 == === ===
070  0562  0560     =
058  0563  0561     =  ESTADOS T3, T4 E T5 DO CICLO(=0) DE FETCH
070  0000  0562   <===
                  0563
                  0564
                  0565
071  0568  0566   <===  INSTRUCAO JUMPCOND          CODIGO 11 CCC 010
070  0569  0567     =                 =========            == === ===
057  0570  0568     =  ESTADOS T3, T4 E T5 DO CICLO(=0) DE FETCH
070  0571  0569     =  CCC= 000 ==> JNZ     CCC= 100 ==> JPO
070  0572  0570     =       001 ==> JZ              101 ==> JPE
070  0000  0571   <===     010 ==> JNC             110 ==> JP
000  0000  0572             011 ==> JC              111 ==> JM
000  0000  0573
000  0000  0574
071  0577  0575   <===
001  0578  0576     =  ESTADO T3 DO CICLO(=1) DE MEMREAD P2
019  0579  0577     =
070  0000  0578   <===
071  0581  0579   <===
001  0582  0580     =
017  0583  0581     =  ESTADO T3 DO CICLO(=1) DE MEMREAD P3
056  0584  0582     =
070  0000  0583   <===
                  0584
                  0585
                  0586
071  0589  0587   <===  INSTRUCAO JMP               CODIGO 11 000 011
070  0590  0588     =  ********* OUT/IN                    11 010 011
073  0591  0589     =            XTHL                     11 100 011
070  0000  0590   <===  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
                  0591
                  0592
                  0593
071  0594  0594   <===  INSTRUCAO OUT/IN
001  0597  0595     =                 === ==
019  0598  0596     =  ESTADO T3 DO CICLO(=1) DE MEMREAD
017  0599  0597     =
070  0000  0598   <===
020  0601  0599   <===
004  0602  0600     =  ESTADO T3 DO CICLO(=10) DE OUTPUT WRITE
079  0603  0601     =
070  0000  0602   <===
071  0608  0606   <===
003  0609  0607     =
021  0610  0608     =  ESTADO T3 DO CICLO(=9) DE INPUTREAD
070  0000  0609   <===
```

```
071  0615   <===  INSTRUCAO DI                CODIGO 11 110 011
070  0616      =              ==                      == === ===
081  0617      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000   <===

071  0622   <===  INSTRUCAO EI                CODIGO 11 111 011
070  0623      =              ==                      == === ===
080  0624      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000   <===

071  0629   <===
001  0630      =  ESTADO T3 DO CICLO(=7) DE STACKREAD 2
019  0631      =  XTHL
070  0000   <===
071  0633   <===
001  0634      =  ESTADO T3 DO CICLO(=7) DE STACKREAD 3
017  0635      =  XTHL
070  0000   <===
012  0637   <===
002  0638      =  ESTADO T3 DO CICLO(=8) DE STACKWRITE 4
070  0000   <===  XTHL
014  0640   <===
002  0641      =
070  0642      =  ESTADOS T3,T4 E T5 DO CICLO(=8) DE STACKWRITE 5
070  0643      =  XTHL
063  0644      =
070  0000   <===

071  0649   <===  INSTRUCAO XCHG              CODIGO 1 101 011
070  0650      =              ====                     = === ===
059  0651      =
061  0652      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
062  0653      =
070  0000   <===

071  0659   <===  INSTRUCAO CALL COND         CODIGO 11 CCC 100
070  0659      =              ==== ====                == === ===
057  0660      =  ESTADOS T3,T4 E T5 DO CICLO(=0) DE FETCH
070  0661      =
082  0662      =
070  0000   <===
071  0664   <===
001  0665      =  ESTADO T3 DO CICLO(=1) DE MEMREAD 2
019  0666      =
070  0000   <===
071  0665   <===
001  0669      =
017  0670      =  ESTADO T3 DO CICLO(=1) DE MEMREAD 3
079  0671      =
070  0000   <===
```

```
010  0673  0671    <===
002  0674  0672      =  ESTADO T3 DO CICLO(=8) DE STACKWRITE  4
070  0000  0673    <===
006  0676  0674    <===
002  0677  0675      =
070  0678  0676      =  ESTADO T3 DO CICLO(=8) DE STACKW ITE  5
056  0000  0677    <===
071  0680  0678    <===  INSTRUCAO CALL              CODIGO 1  001 101
070  0681  0679      =  ********* PUSH RP, PSW          1  RP0 101
070  0682  0680      =  ESTADOS T3, T4 E T5 DO CICLO(=0) DE FETCH
053  0683  0681      =
070  0000  0682    <===
071  0685  0683    <===
001  0686  0684      =  ESTADO T3 DO CICLO(=1) DE MEMREAD  3
017  0687  0685      =  DESCRICAO DOS OUTROS ESTADOS VIDE CALL COND
070  0000  0686    <===
027  0689  0687    <===  PUSH RP
002  0690  0688      =  ESTADO T3 DO CICLO(=8) DE STACKWRITE  2
070  0000  0689    <===
026  0692  0690    <===  PUSH RP
003  0693  0691      =  ESTADO T3 DO CICLO(=8) DE STACKWRITE  3
073  0000  0692    <===
               0693
               0694
               0695
020  0697  0696    <===  PUSH PSW
002  0698  0697      =  ESTADO T3 DO CICLO(=8) DE STACKWRITE  2
070  0000  0698    <===
022  0701  0699    <===  PUSH PSW
002  0702  0700      =  ESTADO T3 DO CICLO(=8) DE STACKWRITE  3
070  0000  0701    <===
               0702
               0703
               0704
               0705
071  0708  0706    <===  OPERACOES LOGICAS E ARITMETICAS ENTRE A E <RP>
070  0709  0707      =
049  0710  0709      =  ESTADOS T3 E T4 DO CICLO(=0) DE FETCH
070  0000  0709    <===
               0710
               0711
071  0714  0712    <===  INSTRUCAO ADI              CODIGO 11 000 110
001  0715  0713      =           ===                    ** === ===
031  0716  0714      =
070  0717  0715      =  ESTADO T3 DO CICLO(=1) DE MEMREAD
042  0718  0716      =
035  0719  0717      =
033  0000  0718    <===
               0719
               0720
               0721
071  0724  0722    <===  INSTRUCAO ACI              CODIGO 11 001 110
001  0725  0723      =           ===                    ** === ===
031  0726  0724      =
070  0727  0725      =  ESTADO T3 DO CICLO(=1) DE MEMREAD
043  0728  0726      =
035  0729  0727      =
033  0730  0728    <===
               0729
               0730
               0731
```

```
071  0734  0732    <===  INSTRUCAO SUI              CODIGO 11 010 110
001  0735  0733       =            ===                     == === ===
031  0736  0734       =
070  0737  0735       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
044  0738  0736       =
035  0739  0737       =
033  0000  0738    <===
           0739
           0740
           0741
071  0744  0742    <===  INSTRUCAO SBI              CODIGO 11 011 110
001  0745  0743       =            ===                     == === ===
031  0746  0744       =
070  0747  0745       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
045  0748  0746       =
035  0749  0747       =
033  0750  0748    <===
           0749
           0750
           0751
071  0754  0752    <===  INSTRUCAO ANI              CODIGO 11 100 110
001  0755  0753       =            ===                     == === ===
031  0756  0754       =
070  0757  0755       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
046  0758  0756       =
035  0759  0757       =
033  0760  0758    <===
           0759
           0760
           0761
071  0764  0762    <===  INSTRUCAO XRI              CODIGO 11 101 110
001  0765  0763       =            ===                     == === ===
031  0766  0764       =
070  0767  0765       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
047  0768  0766       =
035  0769  0767       =
033  0000  0768    <===
           0769
           0770
           0771
071  0774  0772    <===  INSTRUCAO ORI              CODIGO 11 110 110
001  0775  0773       =            ===                     == === ===
031  0776  0774       =
070  0777  0775       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
048  0778  0776       =
035  0779  0777       =
033  0000  0778    <===
           0779
           0780
           0781
071  0784  0782    <===  INSTRUCAO CPI              CODIGO 11 111 110
001  0785  0783       =            ===                     == === ===
031  0786  0784       =
070  0787  0785       =  ESTADO T3 DO CICLO(=1) DE MEMREAD
044  0788  0786       =
035  0000  0787    <===
           0788
           0789
           0790
           0791
071  0794  0792    <===  INSTRUCAO RST              CODIGO 11 AAA 111
```

```
084  0795  0793      =                ===                  == === ===
070  0796  0794      =
070  0797  0795      =   ESTADOS T3,T4 E T5 DO CICLO(=0) DE FETCH
083  0798  0796      =
070  0000  0797    <===
010  0800  0798    <===
002  0801  0799      =   ESTADO T3 DO CICLO(=8) DE STACKWRITE  2
070  0000  0800    <===
009  0803  0801    <===
002  0800  0802      =
005  0805  0803      =   ESTADO T3 DO CICLO(=8) DE STACKWRITE  3
070  0806  0804      =
056  0000  0805    <===
```

APÊNDICE C

EXEMPLO DE SIMULAÇÃO

```
==========================================================================================
        SIM8080 -- SIMULADOR DO MICROPROCESSADOR INTEL 8080 NO B-6700 -- VERSAO 1
                SIMULACAO EXECUTADA EM 21/12/76 AS 14 HORAS E 15 MINUTOS
==========================================================================================
  %
  %  SIMULACAO  PARA TESTAR O SIM8080
  %
  %
  % O PROGRAMA TESTE FAZ A ADICAO DECIMAL DOS CONTEUDOS DAS POSICOES    DE
  % MEMORIA M(30) A M(37) COM AS DE M(38) A M(45)
  % O RESULTADO EH COLOCADO NAS POSICOES DE MEMORIA M(30) A M(37)
  %
    MEMORY(01,100);
   %
   % INTERRUPCOES A SEREM SIMULADAS
   %
    INTER(6) INST(5);
    %
    HOLD(10) INST(15);
    %
    %
  WRITE INST(1,1) M(30),M(31),M(32),M(33),M(34),M(35),M(36),M(37),M(38),
  M(39);
    %
  LOAD M(1)= 021,    %      LXI    D,ALPHA         1
             036,    %                             2
             000,    %                             3
             041,    %      LXI    H,BETA          4
             046,    %                             5
             000,    %                             6
             016,    %      MVI    C,8             7
             010,    %                             8
             257,    %      XRA                    9
             032,    % LOOP:LDAX   D              10
             216,    %      ADC    M              11
             047,    %      DAA                   12
             022,    %      STAX   D              13
             043,    %      INX    H              14
             023,    %      INX    D              15
             015,    %      DCR    C              16
             302,    %      JNZ    LOOP           17
             012,    %
             000,    %
                     %
      M(30)=001,002,003,004,005,006,007,000,    % ALPHA
            001,002,003,004,005,006,007,000;    % BETA
  LOAD PC=000001, SP=000140,
       INTE=1;
  LOAD M(48)= 373,   %      EI
              311;   %      RE
  LOAD M(56)= 373,   %      EI
              311;   %      RT
    %
   RESET INST(80);


****** ESTADO INICIAL DO SISTEMA ******
```

```
 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000000  000000  00000  000  000001  000140  000   000  000  000  000  000  000  000  00000  000000  000  0010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000010  000003  00001  021  000004  000140  000   000  000  000  000  036  000  000  00000  000003  000  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000020  000006  00002  041  000007  000140  000   000  000  000  000  036  000  046  00000  000006  000  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000027  000008  00003  016  000011  000140  000   000  000  010  000  036  000  046  00000  000010  010  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000031  000009  00004  257  000012  000140  000   000  000  010  000  036  000  046  01010  000011  257  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000038  000011  00005  032  000013  000140  000   001  000  010  000  036  000  046  01010  000036  001  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------
****** INTERRUPT(6) = INSTRUCAO=5

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000045  000013  00006  216  000014  000140  000   002  000  010  000  036  000  046  00000  000046  001  1011000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------
****** ATENDENDO INTERRUPT(6) ==  CLOCK=47

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000047  000013  00006  216  000014  000140  000   002  000  010  000  036  000  046  00000  000014  347  1101000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000001   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010
-------------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINOS
0000056  000016  00007  367  000060  000136  014   002  000  010  000  036  000  046  00000  000136  014  1000000000
```

```
M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000001 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000060 000017 00008 373  000061 000136 014   002 000 010 000 036 000 046 00000 000060 373  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000001 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000070 000020 00009 311  000014 000140 000   002 000 010 000 036 000 046 00000 000137 000  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000001 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000074 000021 00010 047  000015 000140 000   002 000 010 000 036 000 046 00000 000014 047  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000001 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000081 000023 00011 022  000016 000140 000   002 000 010 000 036 000 046 00000 000036 002  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000086 000024 00012 043  000017 000140 000   002 000 010 000 036 000 047 00000 000016 043  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000091 000025 00013 023  000020 000140 000   002 000 010 000 037 000 047 00000 000017 023  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000096 000025 00014 015  000021 000140 000   002 000 007 000 037 000 047 00000 000020 015  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC     SP     M(SP) A   B   C   D   E   H   L   FLAGS ABUS   DBUS PINOS
0000107 000029 00015 302  000012 000140 000   002 000 007 000 037 000 047 00000 000023 000  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000010 00000011 00000100 00000101 00000110 00000111 00000000 00000001 00000010

****** HOLD - INSTRUCAO=15
```

```
****** ATENDENDO HOLD -- CLOCK=111

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000111  000030  00015  032  000013  000140  000  002  000  007  000  037  000  047  00000  000012  032  1010100000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000126  000031  00016  032  000013  000140  000  002  000  007  000  037  000  047  00000  000037  002  1010000000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000132  000033  00017  216  000014  000140  000  004  000  007  000  037  000  047  00000  000047  002  1010000000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000135  000034  00018  047  000015  000140  000  004  000  007  000  037  000  047  00000  000014  047  1010000000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000010   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000143  000036  00019  022  000016  000140  000  004  000  007  000  037  000  047  00000  000037  004  1010000000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000100   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000144  000037  00020  043  000017  000140  000  004  000  007  000  037  000  050  00000  000016  043  1010000000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000100   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000153  000038  00021  023  000020  000140  000  004  000  007  000  040  000  050  00000  000017  023  1010000000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000100   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000158  000039  00022  015  000021  000140  000  004  000  006  000  040  000  050  00010  000020  015  1010000000

  M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
  00000010   00000100   00000011   00000100   00000101   00000110   00000111   00000000   00000001   00000010

 CLOCK   CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS  ABUS  DBUS    PINS
0000169  000042  00023  302  000012  000140  000  004  000  006  000  040  000  050  00010  000023  000  1010000000
```

```
M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000011  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000176  000044  00024 032  000013  000140  000    003 000 006 000 040 000 050 00010   000040 003  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000011  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000183  000046  00025 216  000014  000140  000    006 000 006 000 040 000 050 00010   000050 003  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000011  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000187  000047  00026 047  000015  000140  000    006 000 006 000 040 000 050 00010   000014 047  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000011  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000194  000049  00027 022  000016  000140  000    006 000 006 000 040 000 050 00010   000040 006  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000199  000050  00028 043  000017  000140  000    006 000 006 000 040 000 051 00010   000016 043  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000204  000051  00029 023  000020  000140  000    006 000 006 000 041 000 051 00010   000017 023  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000207  000052  00030 015  000021  000140  000    006 000 006 000 041 000 051 00010   000020 015  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
0000220  000055  00031 302  000012  000140  000    006 000 006 000 041 000 051 00010   000023 000  1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A   B   C   D   E   H   L   FLAGS   ABUS  DBUS  PINS
```

```
0000227  000057  00032  032  000013  000140  000  004  005  005  000  041  000  051  00010  000041  004  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000234  000059  00033 216  000014  000140  000    010  004  005  000  041  000  051  00000  000051  004   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000239  000060  00034 047  000015  000140  000    010  004  005  000  041  000  051  00000  000014  047   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00000100  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000245  000062  00035 022  000016  000140  000    010  004  005  000  041  000  051  00000  000041  010   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000250  000063  00036 043  000017  000140  000    010  004  005  000  041  000  052  00000  000016  043   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000255  000064  00037 023  000020  000140  000    010  004  005  000  042  000  052  00000  000017  023   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000260  000065  00038 015  000021  000140  000    010  004  004  000  042  000  052  00000  000020  015   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000271  000068  00039 302  000012  000140  000    010  004  004  000  042  000  052  00000  000023  000   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00000101  00000110  00000111  00000000  00000001  00000010

 CLOCK    CYCLE   INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
 0000278  000070  00040 032  000013  000140  000    005  004  004  000  042  000  052  00000  000042  005   1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00000101  00000110  00000111  00000000  00000001  00000010
```

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000285	000072	00041	216	000014	000140	000	012	000	004	000	042	000	052	00010	000052	005	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00000101	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000289	000073	00042	047	000015	000140	000	020	000	004	000	042	000	052	00010	000014	047	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00000101	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000296	000075	00043	022	000016	000140	000	020	000	004	000	042	000	052	00010	000042	022	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00010000	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000301	000076	00044	043	000017	000140	000	020	000	004	000	042	000	053	00010	000016	043	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00010000	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000306	000077	00045	023	000020	000140	000	020	000	004	000	043	000	053	00010	000017	023	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00010000	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000311	000078	00046	015	000021	000140	000	020	000	003	000	043	000	053	00010	000020	015	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00010000	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000322	000081	00047	302	000012	000140	000	020	000	003	000	043	000	053	00010	000023	000	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00010000	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000329	000083	00048	032	000013	000140	000	006	000	003	000	043	000	053	00010	000043	006	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)
00000010	00000100	00000110	00001000	00010000	00000110	00000111	00000000	00000001	00000010

CLOCK	CYCLE	INST	IR	PC	SP	M(SP)	A	B	C	D	E	H	L	FLAGS	ABUS	DBUS	PINS
0000336	000085	00049	216	000014	000140	000	014	000	003	000	043	000	053	00010	000053	006	1010000000

M(00030)	M(00031)	M(00032)	M(00033)	M(00034)	M(00035)	M(00036)	M(00037)	M(00038)	M(00039)

```
00000010  00000100  00000110  00001000  00010000  00000110  00000111  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000340 000086 00050 047  000015  000140 000  022 000 003 000 043 000 053 00010 000014 047  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00000110 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000347 000088 00051 022  000016  000140 000  022 000 003 000 043 000 053 00010 000043 022  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00010010 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000352 000089 00052 043  000017  000140 000  022 000 003 000 043 000 054 00010 000015 043  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00010010 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000357 000090 00053 023  000020  000140 000  022 000 003 000 044 000 054 00010 000017 023  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00010010 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000362 000091 00054 015  000021  000140 000  022 000 002 000 044 000 054 00000 000020 015  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00010010 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000373 000094 00055 302  000012  000140 000  022 000 002 000 044 000 054 00000 000023 000  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00010010 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000380 000096 00056 032  000013  000140 000  007 000 002 000 044 000 054 00000 000044 007  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00010010 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000387 000098 00057 216  000014  000140 000  016 000 002 000 044 000 054 00000 000054 007  1010000000

M(00030) M(00031) M(00032) M(00033) M(00034) M(00035) M(00036) M(00037) M(00038) M(00039)
00000010 00000100 00000110 00001000 00010000 00010010 00000111 00000000 00000001 00000010

CLOCK   CYCLE  INST  IR   PC      SP    M(SP) A   B   C   D   E   H   L   FLAGS ABUS  DBUS  PINS
0000391 000099 00058 047  000015  000140 000  024 000 002 000 044 000 054 00000 000014 047  1010000000
```

```
M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00000111  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000398 000101 00059 022  000016  000140  000    024  000  002  000  044  000  054  00000  000044  024   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000403 000102 00060 043  000017  000140  000    024  000  002  000  044  000  055  00000  000016  043   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000408 000103 00061 023  000020  000140  000    024  000  002  000  045  000  055  00000  000017  023   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000413 000104 00062 015  000021  000140  000    024  000  001  000  045  000  055  00000  000020  015   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000424 000107 00063 302  000012  000140  000    024  000  001  000  045  000  055  00000  000023  000   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000431 000109 00064 032  000013  000140  000    000  000  001  000  045  000  055  00000  000045  000   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000438 000111 00065 216  000014  000140  000    000  000  001  000  045  000  055  01010  000055  000   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010

CLOCK   CYCLE  INST  IR   PC      SP      M(SP)  A    B    C    D    E    H    L    FLAGS  ABUS    DBUS  PINS
0000442 000112 00066 047  000015  000140  000    000  000  001  000  045  000  055  01010  000014  047   1010000000

M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
```

```
 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000449  000114  00067  022  000016  000140  000   000  000  001  000  045  000  055  01010  000045  000  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000454  000115  00068  043  000017  000140  000   000  000  001  000  045  000  056  01010  000016  043  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000459  000116  00069  023  000020  000140  000   000  000  001  000  046  000  056  01010  000017  023  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000464  000117  00070  015  000021  000140  000   000  000  000  000  046  000  056  01010  000020  015  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000469  000118  00071  302  000024  000140  000   000  000  000  000  046  000  056  01010  000021  302  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000473  000119  00072  000  000025  000140  000   000  000  000  000  046  000  056  01010  000024  000  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000477  000120  00073  000  000026  000140  000   000  000  000  000  046  000  056  01010  000025  000  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000481  000121  00074  000  000027  000140  000   000  000  000  000  046  000  056  01010  000026  000  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
-------------------------------------------------------------------------------------------------------

 CLOCK    CYCLE   INST   IR    PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS  DBUS     PINS
0000485  000122  00075  000  000030  000140  000   000  000  000  000  046  000  056  01010  000027  000  1010000000

 M(00030)  M(00031)  M(00032)  M(00033)  M(00034)  M(00035)  M(00036)  M(00037)  M(00038)  M(00039)
 00000010  00000100  00000110  00001000  00010000  00010010  00010100  00000000  00000001  00000010
```

```
 CLOCK    CYCLE   INST   IR     PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS   DBUS     PINOS
0000489  000123  00076  000  000031  000140  000   000  00   000  000  046  000  056  01010  000030  000  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000010   00000100   00000110   00001000   00010000   00010010   00010100   00000000   00000001   00000010

 CLOCK    CYCLE   INST   IR     PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS   DBUS     PINOS
0000493  000124  00077  000  000032  000140  000   000  000  000  000  046  000  056  01010  000031  000  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000010   00000100   00000110   00001000   00010000   00010010   00010100   00000000   00000001   00000010

 CLOCK    CYCLE   INST   IR     PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS   DBUS     PINOS
0000497  000125  00078  000  000033  000140  000   000  000  000  000  046  000  056  01010  000032  000  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000010   00000100   00000110   00001000   00010000   00010010   00010100   00000000   00000001   00000010

 CLOCK    CYCLE   INST   IR     PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS   DBUS     PINOS
0000501  000126  00079  000  000034  000140  000   000  001  000  000  046  000  056  01010  000033  000  1010000000

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000010   00000100   00000110   00001000   00010000   00010010   00010100   00000000   00000001   00000010

******RESET = INSTRUCAO=80

 CLOCK    CYCLE   INST   IR     PC      SP    M(SP)  A    B    C    D    E    H    L   FLAGS   ABUS   DBUS     PINOS
0000505  000127  00080  000  000000  000140  000   000  000  000  000  046  000  056  01010  000034  000  1000000001

 M(00030)   M(00031)   M(00032)   M(00033)   M(00034)   M(00035)   M(00036)   M(00037)   M(00038)   M(00039)
 00000010   00000100   00000110   00001000   00010000   00010010   00010100   00000000   00000001   00000010
```