1. Publicação nº	2. Versão	3. Data	5. Distribuição
INPE-3265-RTR/062		*Set., 1984*	☐ Interna ☐ Externa ☒ Restrita
4. Origem *DCA/DEA*	Programa *LASIDA/PSDA*		

6. Palavras chaves - selecionadas pelo(s) autor(es)

TRADUTOR
MICROPROGRAMAÇÃO
GERAÇÃO DE MICROCÓDIGOS
ESTRUTURA DE DADOS

7. C.D.U.:

8. Título *INPE-3265-RTR/062*

TRADUTOR LMP VERSÃO 1.0: MANUAL DE MANUTENÇÃO

10. Páginas: *67*

11. Última página: *B.8*

12. Revisada por

Ricardo G.

P/ *Marcos A. Cardoso Cruz*

9. Autoria *Wilson Yamaguti*

Assinatura responsável

13. Autorizada por

Parada

Nelson de Jesus Parada
Diretor Geral

14. Resumo/Notas

Este relatório apresenta a estrutura de dados utilizada na construção do tradutor LMP versão 1.0, disponível no computador Burroughs B6800, visando basicamente sua manutenção. Este tradutor aceita um subconjunto da linguagem de microprogramação LMP desenvolvida neste Instituto.

15. Observações

ABSTRACT

This report presents the data structures used in the construction of the LMP version 1.0 translator available on the Burroughs B6800 computer for maintenance purpose. This translator implements a subset of the LMP microprogramming language developed at this Institute.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1. INTRODUÇÃO

O tradutor LMP versão 1.0 é um tradutor cruzado escrito em ALGOL (Burroughs, 1974) e residente no computador B6800, cuja finalidade básica é traduzir um microprograma escrito na linguagem LMP (Yamaguti, 1981) em microcódigos a ser posteriormente armazenados no sistema microprogramado em desenvolvimento.

A estrutura básica do tradutor LMP é apresentada na Figura 1. Este tradutor é essencialmente orientado pela sintaxe da linguagem. Assim, o Algoritmo de Análise Sintática é o núcleo do processo de tradução e controla os Analisadores Léxico ("scanner") e Semântico e também o Gerador de Códigos na Forma Intermediária.

O Analisador Léxico é ativado pelo Analisador Sintático para adquirir a sequência de caracteres correspondentes ao microprograma fonte e fornecer os símbolos da linguagem.

O Analisador Sintático efetua a análise sintática do microprograma fonte e, quando necessário, requisita ao Analisador Semântico e ao Gerador de Códigos na Forma Intermediária a função de verificar e armazenar as informações semânticas numa estrutura de tabelas e, ao mesmo tempo, gerar uma forma intermediária, o que constitui o 1º passo de tradução.

A partir da forma intermediária, o Gerador de Códigos CDX, (Passo # 2) gera os microcódigos na forma semi-final, os quais contêm bits dos tipos 0 (zero), 1 (um) e o ponto (".") que representa os bits "don't care".

A descrição detalhada de cada um dos módulos mencionados é realizada nas Seções seguintes, onde são apresentadas essencialmente a estrutura de dados e a filosofia de projeto, tendo em vista a manutenção do tradutor LMP versão 1.0.

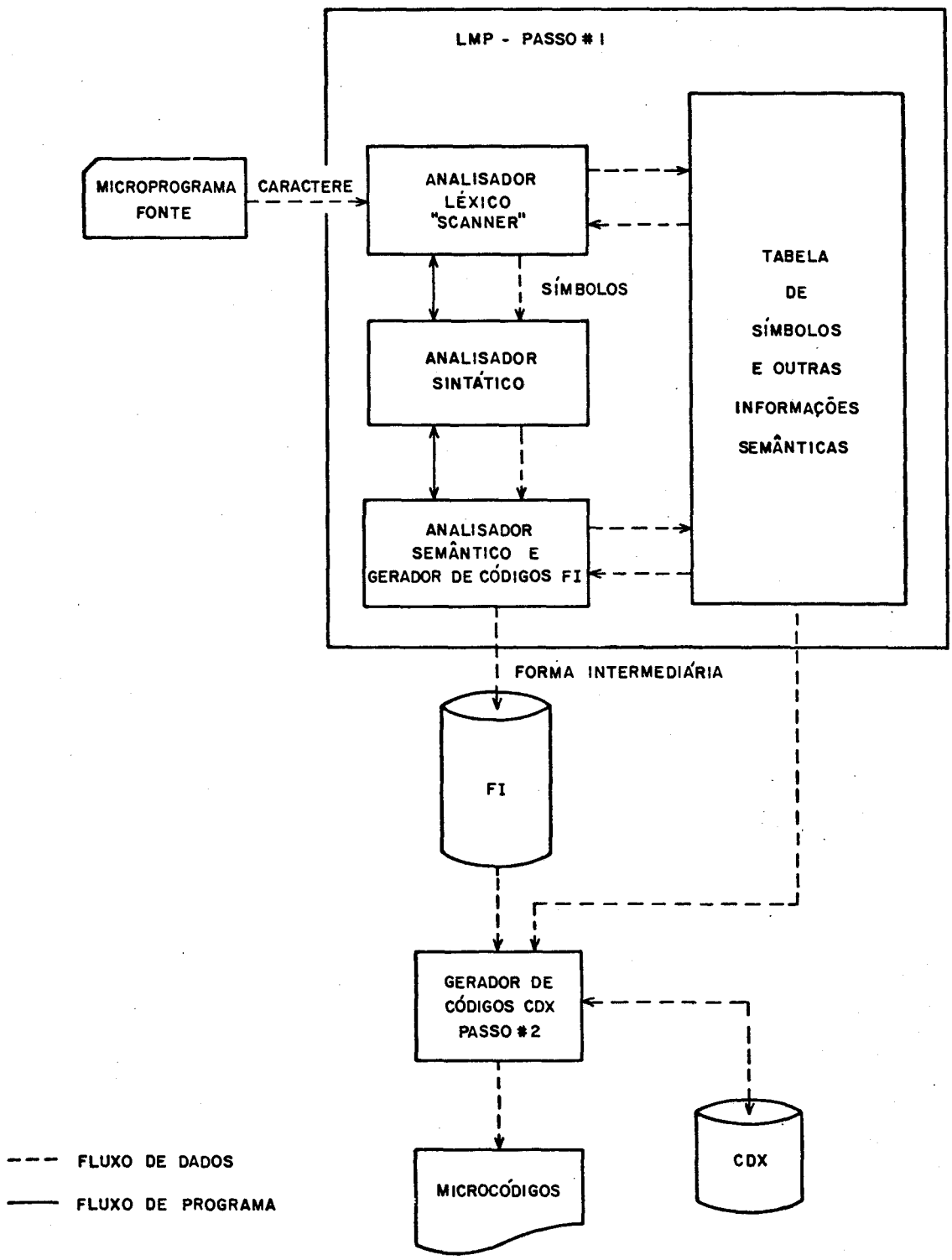


Fig. 1 - Estrutura básica do tradutor LMP versão 1.0.

É apresentada ainda na Seção 7 a estrutura do programa fonte do tradutor LMP.

2. ESTRUTURA DE DADOS DA TABELA DE SÍMBOLOS

Na estruturação da Tabela de Símbolos procurou-se atingir os objetivos que permitem armazenar informações estruturadas em blocos e que tentem minimizar os tempos de acesso às tabelas que a compõem.

Devido às peculiaridades da linguagem LMP, acredita-se que num microprograma usual far-se-á utilização intensa de identificadores, resultando em tabelas relativamente grandes, o que poderia tornar o acesso do tipo sequencial dispendioso em tempo.

Optou-se então pelo endereçamento do tipo HASH que, com dispêndio adicional relativamente pequeno de memória em relação às outras técnicas, permitiu um acesso rápido, com poucas comparações.

A organização da Tabela de Símbolos é apresentada nas Figuras 2 e 3 onde se verifica a existência de cinco tabelas lógicas que a compõe. Esta organização foi baseada nos esquemas apresentados por Aho e Ullman (1978) e por Wulf et alii (1975).

Na implementação dessas tabelas procurou-se utilizar os recursos oferecidos pela linguagem Algol, de modo a minimizar o espaço ocupado por elas.

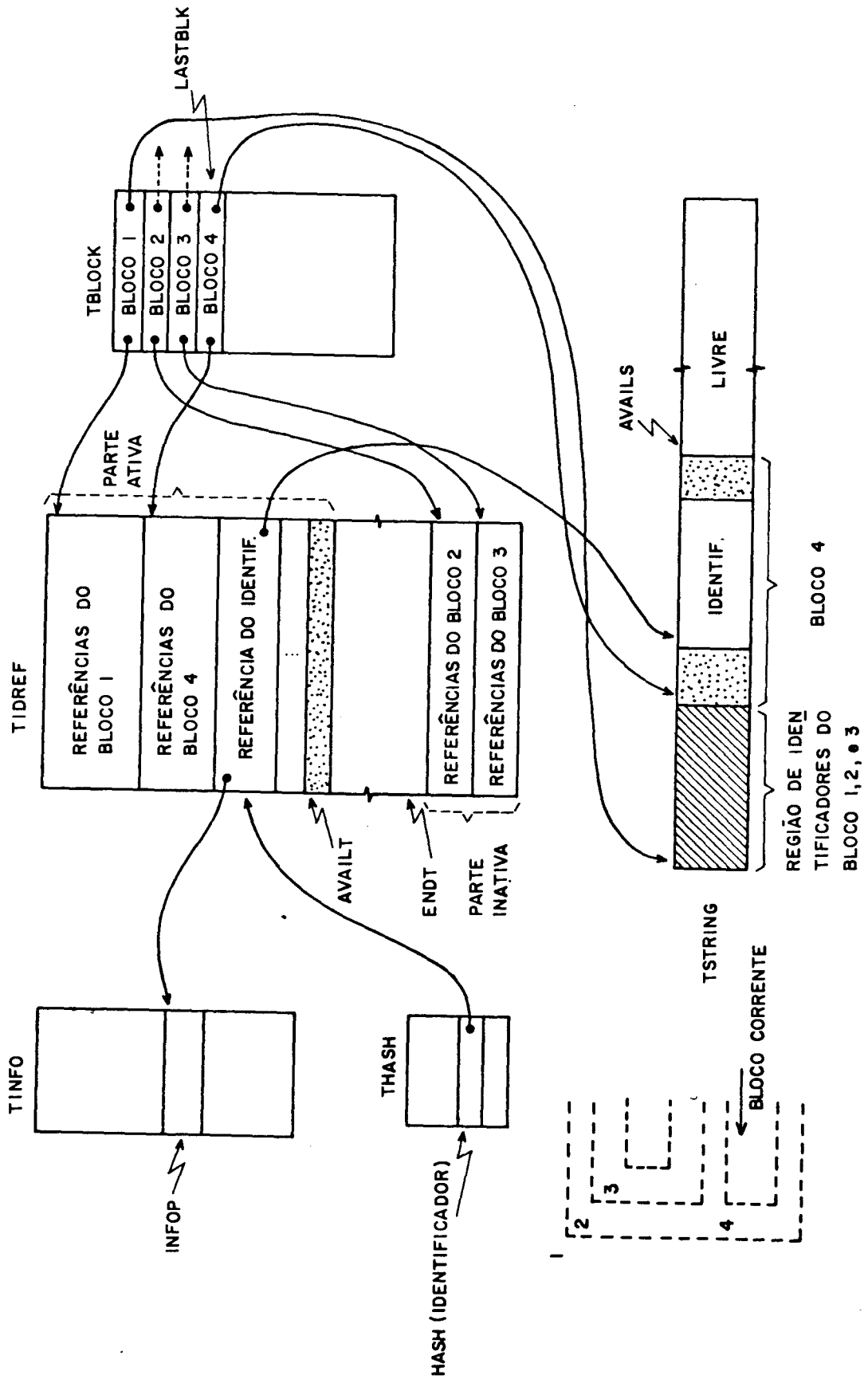


Fig. 2 - Tabela de Símbolos LMP.

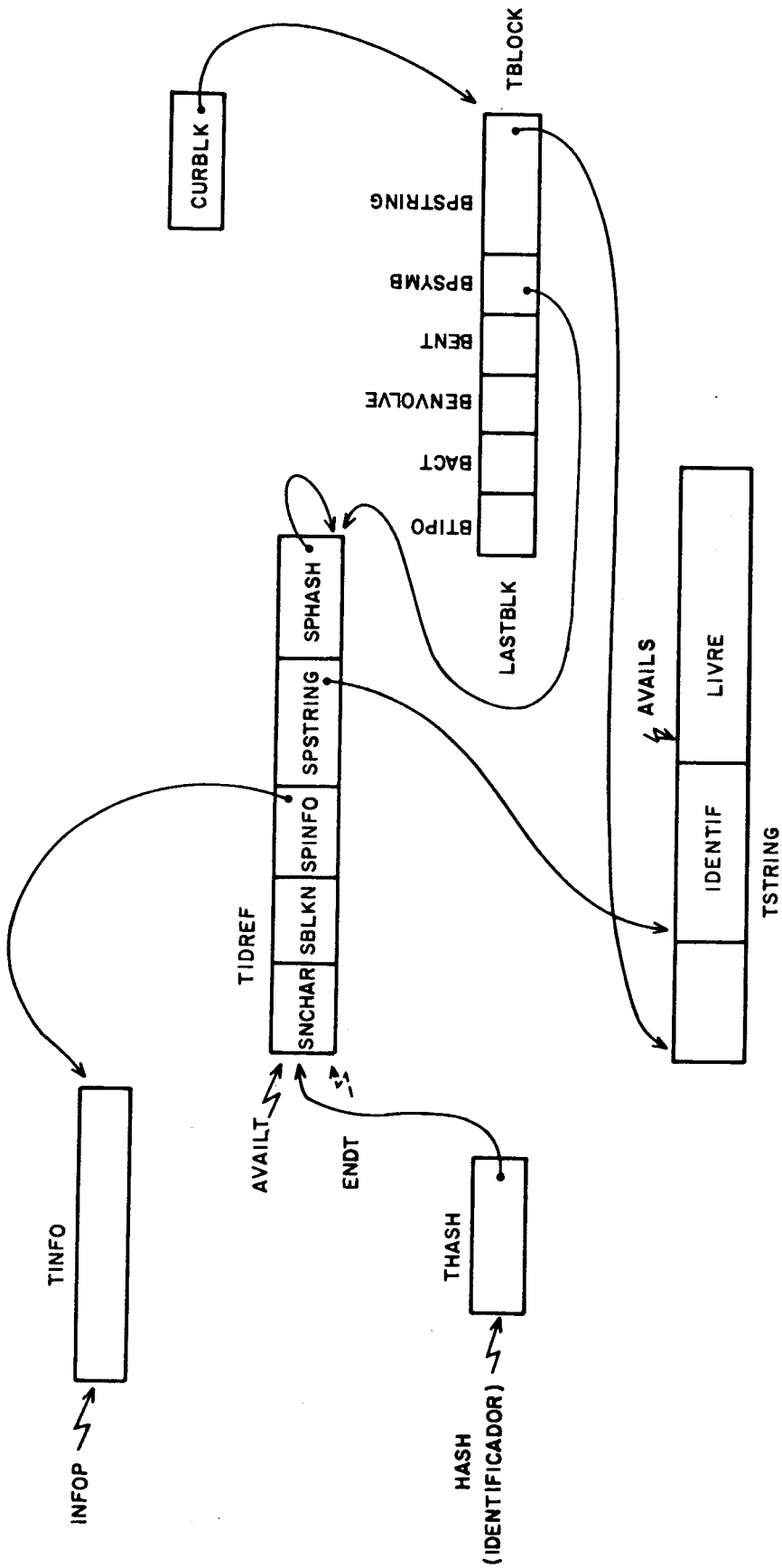


Fig. 3 - Estrutura dos nós da Tabela de Símbolos LMP.

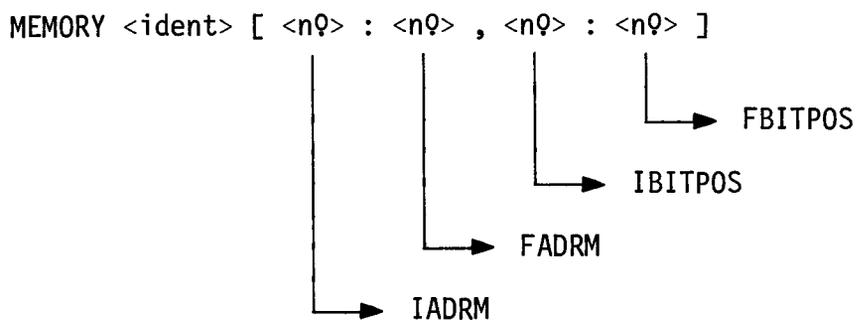
3. INFORMAÇÕES SEMÂNTICAS DAS DECLARAÇÕES NA TABELA DE SÍMBOLOS

Nesta Seção são apresentadas as informações semânticas armazenadas em tabelas, que constituem a Tabela de Símbolos, relacionadas com as declarações da linguagem LMP.

A sintaxe das declarações é apresentada em forma conveniente para a implementação.

3.1 - DECLARAÇÃO DE MEMÓRIA

Sintaxe simplificada:



Os valores numéricos que caracterizam as dimensões da memória de controle são armazenados nas variáveis globais abaixo:

IADRM: endereço inicial da memória de controle,

FADRM: endereço final da mêmoria de controle,

IBITPOS: posição inicial na palavra de controle,

FBITPOS: posição final na palavra de controle,

TAMPAL = FBITPOS - IBITPOS + 1: tamanho da palavra.

A Figura 4 apresenta a estrutura de armazenamento utilizada para armazenamento de informações semânticas relacionadas com a declaração de memória.

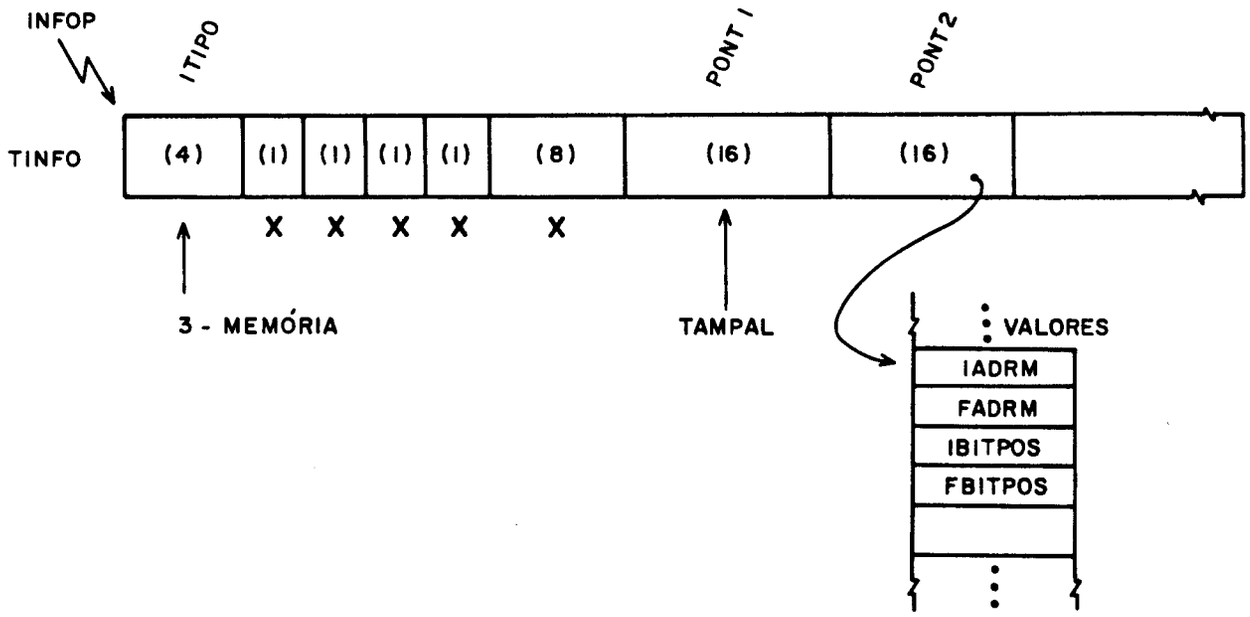


Fig. 4 - Estrutura da declaração de memória.

Na Figura 4 os Xs representam campos não utilizados de TINFO e podem ocorrer as seguintes situações de erro:

a) FADRM - IADRM > MAXCDXADR;

b) IADRM > FADRM;

c) TAMPAL > 24 * MAXCDXLARG.

Na implementação realizada foram supostas as seguintes condições:

a) MAXCDXADR = 2047,

b) MAXCDXLARG = 7.

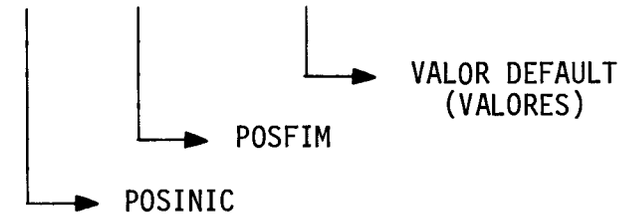
Isto permite manipular uma memória de tamanho máximo de 2048 palavras por 168 bits. Caso sejam necessárias memórias de tamanho maior, basta alterar os valores acima.

3.2 - DECLARAÇÃO DE CAMPO

Sintaxe modificada:

FIELD <ident> = <ident> [<nº> : <nº>] {(<cte>)} {, <ident> =

<ident> [<nº> : <nº>] {(<cte>)}}



POSINIC: posição inicial do campo na palavra,

POSFIM: posição final do campo na palavra,

cte ≡ constante opcional armazenada em VALORES.

A Figura 5 apresenta a estrutura da declaração de campos.

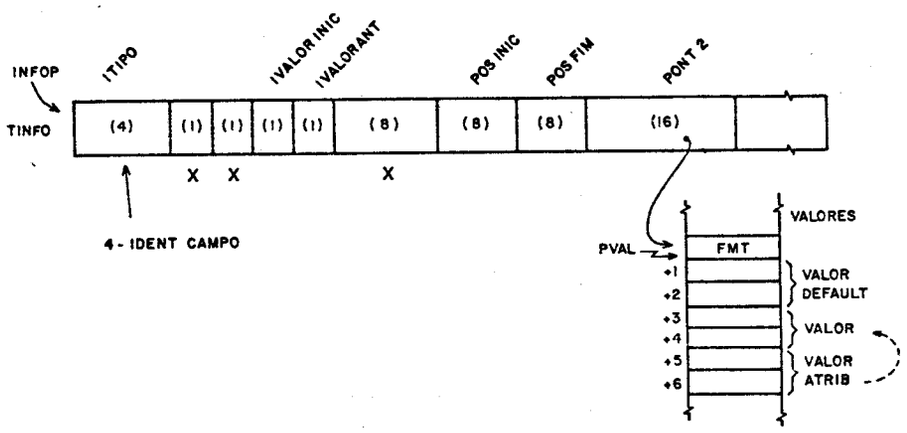


Fig. 5 - Estrutura da declaração de campos.

Significado dos campos desta estrutura:

1) IVALORINIC = Ø se não tem valor inicial,

1 se tem constante na declaração de campo;

2) IVALORANT = Ø se não houve nenhuma atribuição ao campo até o momento,

1 se tem valor anterior atribuído ao campo. Observar que IVALORANT = 1 quando IVALORINIC = 1;

3) PONT2: ponteiro de VALORES para armazenar:

- formato do qual o campo faz parte (ver declaração de formato);
- valor "default", isto é, a constante opcional da declaração de campo;

- valor atual do campo;

- valor em atribuição ao campo na microinstrução corrente. O valor do campo é atualizado somente no final da microinstrução.

Situações de erro:

1) POSFIM - POSINIC + 1 < TAMCONF, então o tamanho do campo é menor que o tamanho da configuração de bits da constante;

2) VALCTE > 2** (POSFIM - POSINIC + 1) - 1, então a constante não pode ser representada com os bits disponíveis para o campo.

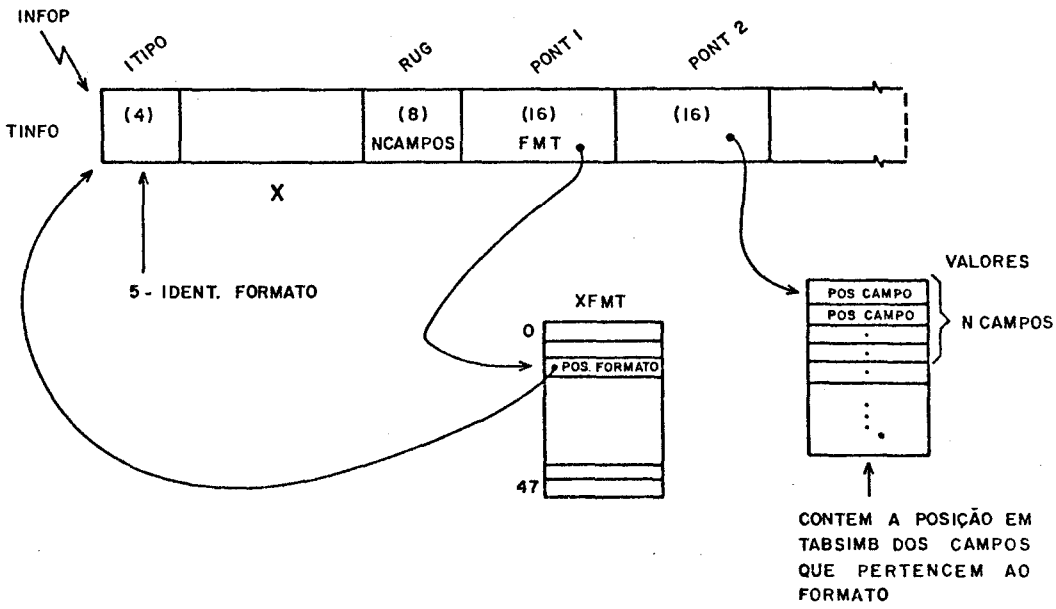
3.3 - DECLARAÇÃO DE FORMATO

Sintaxe modificada:

FORMAT <ident> = (<ident> {,<ident>}) {,<ident> = (<ident> {,<ident>})}.

A estrutura desta declaração é mostrada na Figura 6.

Na Figura 6 a idéia básica utilizada na implementação da declaração de formato é a seguinte. Para cada identificador de campo é associado um conjunto de formatos possíveis. Ao final de cada microinstrução, é realizada a intersecção dos conjuntos possíveis de formato para todos os campos da microinstrução. Se o resultado não for um único formato, então uma mensagem de aviso é emitida.

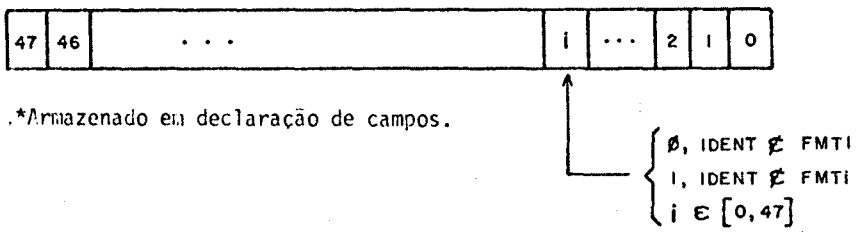


FMT: variável contador de formato [0,47]

XFMT(I) = AFMT [(I)DIV 3].[47 - ((I)MOD 3)*16:16]

FMTP(I) = VALORES [PONT2 (SPINFO (POS))]. [(I):1]

VALORES [PONT2 (SPINFO(POS))]*



*Armazenado em declaração de campos.

Fig. 6 - Estrutura de declaração de formatos.

Exemplificando, e com as seguintes condições:

1) ADR → F1, F2, F3, F4, F5,

2) COUT → F1, F3, F6,

3) ALU → F3, F5, F7.

verifica-se através da Figura 7 que o formato resultante é único e igual a F3.

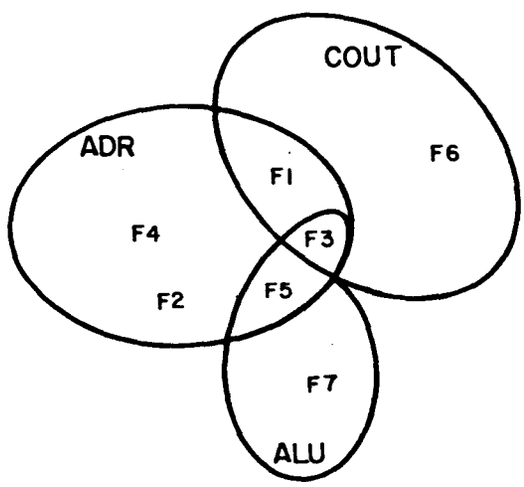


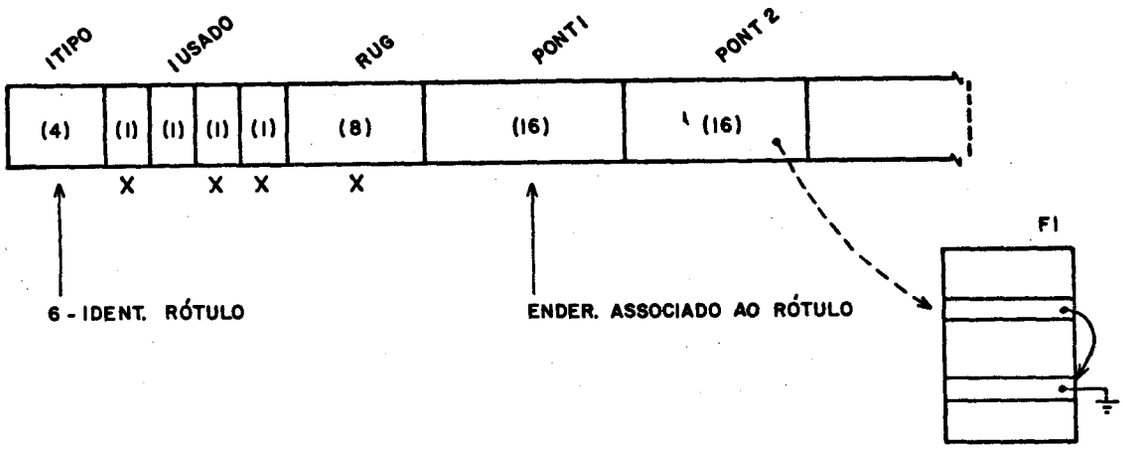
Fig. 7 - Intersecção dos formatos.

3.4 - DECLARAÇÃO DE RÓTULO

Sintaxe modificada:

LABEL <ident> {,<ident>}

A estrutura desta declaração pode ser vista na Figura 8.



IUSADO = Ø se ident. de rótulo não utilizado;

= 1 se já utilizado.

Fig. 8 - Estrutura da declaração de rótulos.

Na Figura 8 tem-se:

1) na inicialização IUSADO é igual a 0;

2) a referência ao rótulo é controlado da seguinte maneira:

   - se IUSADO = 1 então PONT1 contém a variável MILC;

   - se IUSADO = 0 então a forma intermediária armazena o encadeamento das referências ao rótulo;

3) PONT2 = 0 é indicação de fila vazia.

3.5 - DECLARAÇÃO DE DEFINIÇÃO

Sintaxe modificada:

DEFINE <ident> = <texto> # {, <ident> = <texto> # }

A utilização de recursos de macro como o DEFINE se processa em dois passos distintos:

1) montagem da definição, ou seja, o texto que segue o identificador é armazenado em tabelas internas para posterior utilização (Array TEXTOS);

2) invocação de definição (Expansão), ou seja, substituição do identificador de definição pelo texto associado. Somente neste passo é realizada a verificação da validade do texto associado.

A estrutura do DEFINE utilizada se baseia em GRIES (1971). Todo identificador de definição é armazenado na Tabela DEFNOMES e também na Tabela de Símbolos, as quais obedecem às regras de validade de acordo com o nível de bloco. Uma declaração de definição permite a utilização de identificadores de definição já declarados em seu texto associado.

A Figura 9 apresenta a estrutura de dados utilizada para o armazenamento de informações associadas à declaração de definição.

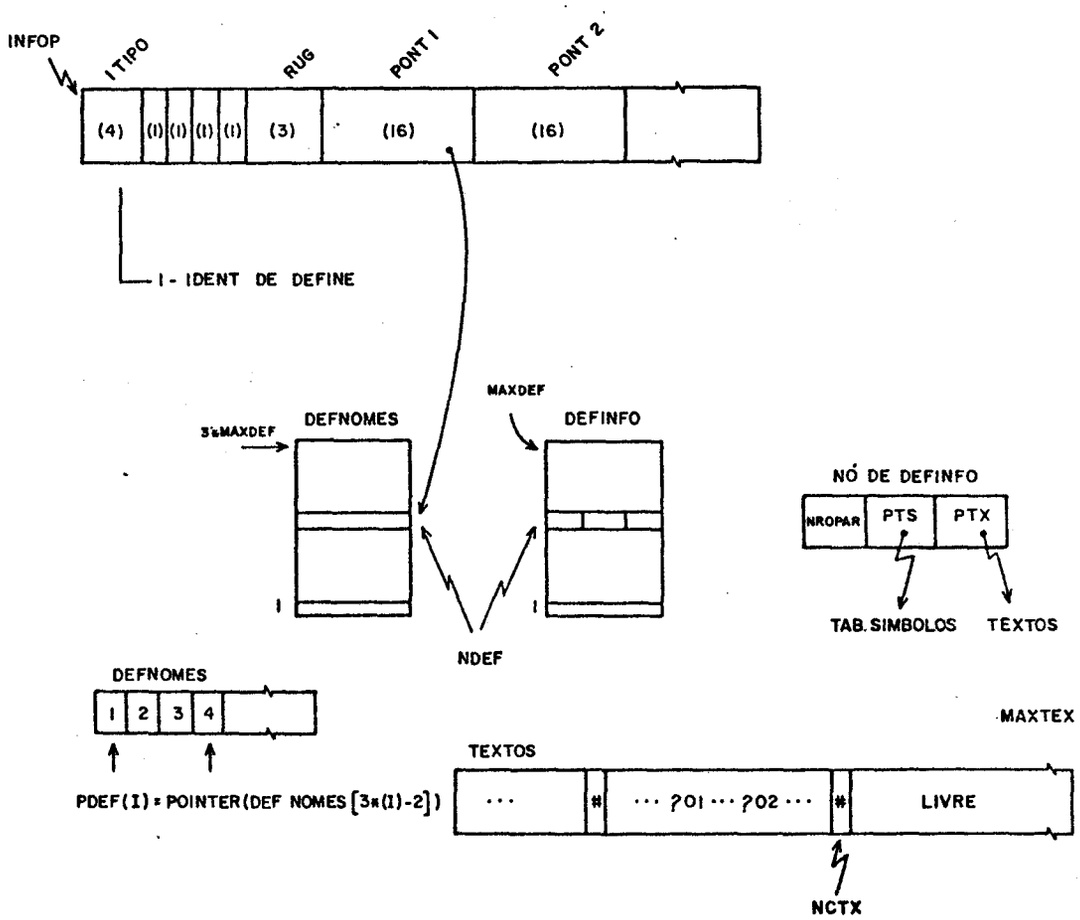


Fig. 9 - Estrutura de dados para armazenamento de declarações de definição.

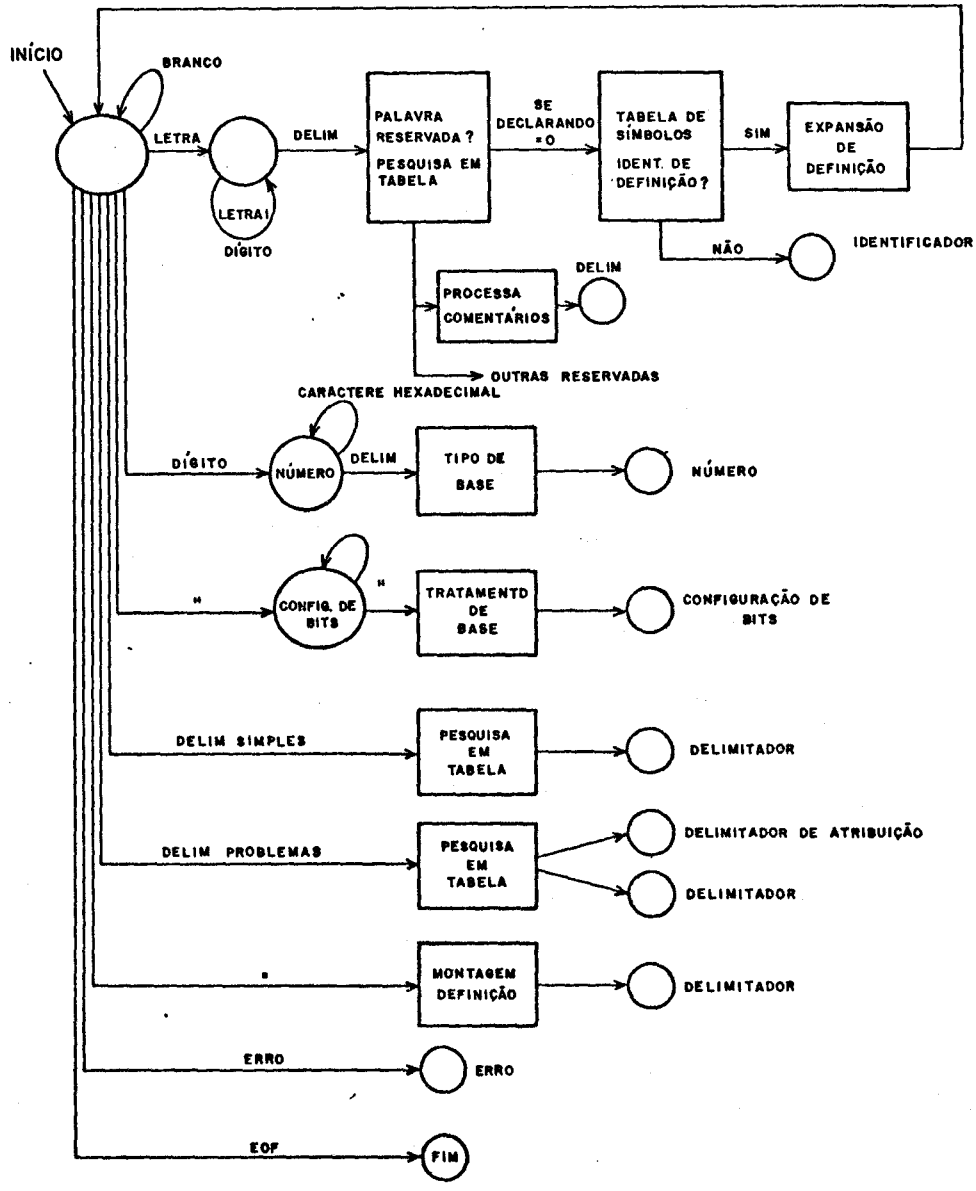


Fig. 10 - Estrutura do analisador Léxico LMP.

```
VALUE ARRAY TABSIMB(="          "4"00", % 0  SERRO
                    ="          "4"01", % 1  SNRO
                    ="          "4"02", % 2  SID
                    ="          "4"03", % 3  SCONF
                    ="          "4"04", % 4
                    ="          "4"05", % 5
                    ="          "4"06", % 6
                    ="          "4"07", % 7
                    ="          "4"08", % 8  FIMCARTAO
                   ="BEGIN      "4"09", % 9  SBEGIN
                   ="END        "4"0A", % 10 SEND
                   ="TABLE      "4"0B", % 11 STABLE
                   ="MEMORY     "4"0C", % 12 SMEMORY
                   ="FIELD      "4"0D", % 13 SFIELD
                   ="FORMAT     "4"0E", % 14 SFORMAT
                   ="DEFINE     "4"0F", % 15 SDEFINE
                   ="MICRO      "4"10", % 16 SMICRO
                   ="LABEL      "4"11", % 17 SLABEL
                   ="PROCEDURE  "4"12", % 18 SPROCEDURE
                   ="COMMENT    "4"13", % 19 SCOMMENT
                   ="RESERVE    "4"14", % 20 SRESERVE
                   ="ORIGIN     "4"15", % 21 SORIGIN
                   ="SET        "4"16", % 22 SSET
                   ="LIST       "4"17", % 23 SLIST
                   ="RESET      "4"18", % 24 SRESET
                   ="SOURCE     "4"19", % 25 SSOURCE
                   ="INTER      "4"1A", % 26 SINTER
                   ="TABSIMB    "4"1B", % 27 STABSIMB
                   ="BIN        "4"1C", % 28 SBIN
                   ="OCT        "4"1D", % 29 SOCT
                   ="DEC        "4"1E", % 30 SDEC
                   ="HEX        "4"1F", % 31 SHEX
                   ="MAP        "4"20", % 32 SMAP
                   ="MSB        "4"21", % 33 SMSB
                   ="INVERT     "4"22", % 34 SINVERT
                   ="ALL        "4"23", % 35 SALL
                   ="DONTCARE   "4"24", % 36 SDONCARE
                   ="EQL        "4"25", % 37 SEQL
                   ="FOR        "4"26", % 38 SFOR
                   ="FMAG       "4"27", % 39 SFMAG
                   ="INTERNAL   "4"28", % 40 SINTERNAL
                   ="VAR        "4"29", % 41 SVAR
                   ="LSM        "4"2A", % 42 SLSM
                   ="           "4"2B", % 43
                   ="           "4"2C", % 44
                   ="           "4"2D", % 45
                   ="           "4"2E", % 46
                   =""          "4"3F", % 47 SASPAS
                   =";          "4"40", % 48 SPTVIRG
                   =",          "4"41", % 49 SVIRG
                   =":          "4"42", % 50 SDOISPTS
                   =":=         "4"43", % 51 SATRIB
                   ="#          "4"44", % 52 SJOGVELHA
                   ="'          "4"45", % 53 SAPOSTROFO
                   ="<          "4"46", % 54 SMENOR
                   =">          "4"47", % 55 SMAIOR
                   ="$          "4"48", % 56 SDOLAR
                   ="&          "4"49", % 57 SECOM
                   ="@          "4"4A", % 58 SARROBA
                   ="-          "4"4B", % 59 SMENOS
                   ="+          "4"4C", % 60 SMAIS
                   ="*          "4"4D", % 61 SMULT
                   ="/          "4"4E", % 62 SDIV
                   ="(          "4"4F", % 63 SABPEPAR
                   =")          "4"50", % 64 SFECHAPAR
                   ="[          "4"51", % 65 SABPECOL
                   ="]          "4"52", % 66 SFECHACOL
                   ="=          "4"53", % 67 SIGUAL
                   =            "4"54")); % 68
```

Fig. 11 - Palavras reservadas da linguagem LMP.

4.2 - EXPANSÃO DE DEFINIÇÃO

O recurso de definição no tradutor LMP se constitui num pré-processador, isto é, o Analisador Léxico (SCANNER) provê meios de expansão da definição quando encontra um identificador de definição.

A modificação inserida no SCANNER original é apresentada na Figura 12. Esta figura mostra basicamente a análise dos parâmetros associados ao identificador de definição.

Como na expansão de um recurso de definição poderá haver outra expansão de definição, a estrutura implementada utiliza uma pilha denominada DEFCALLS, conforme apresentada na Figura 13(a). E para cada expansão é montada uma das estruturas apresentadas na Figura 13(b) e Figura 13(c), em função do tipo de expansão, se é invocação de definição ou expansão de parâmetros formais (realizada na rotina PROXIMOCHAR ao encontrar o símbolo "?").

A rotina PROXIMOCHAR provê o retorno da definição em expansão ao encontrar o símbolo "# ", que sinaliza final de texto associado à definição (ver rotina PROXIMOCHAR).

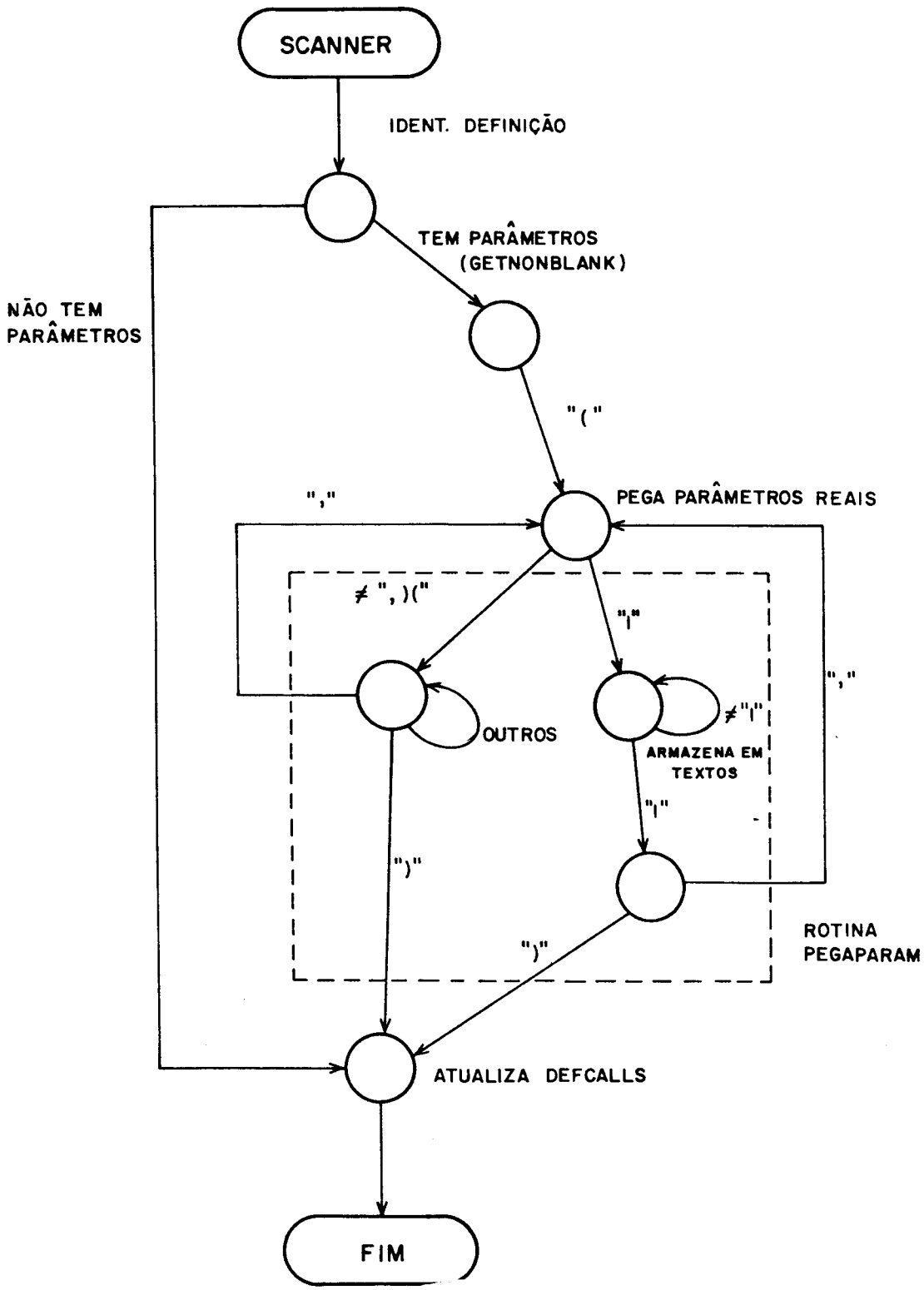
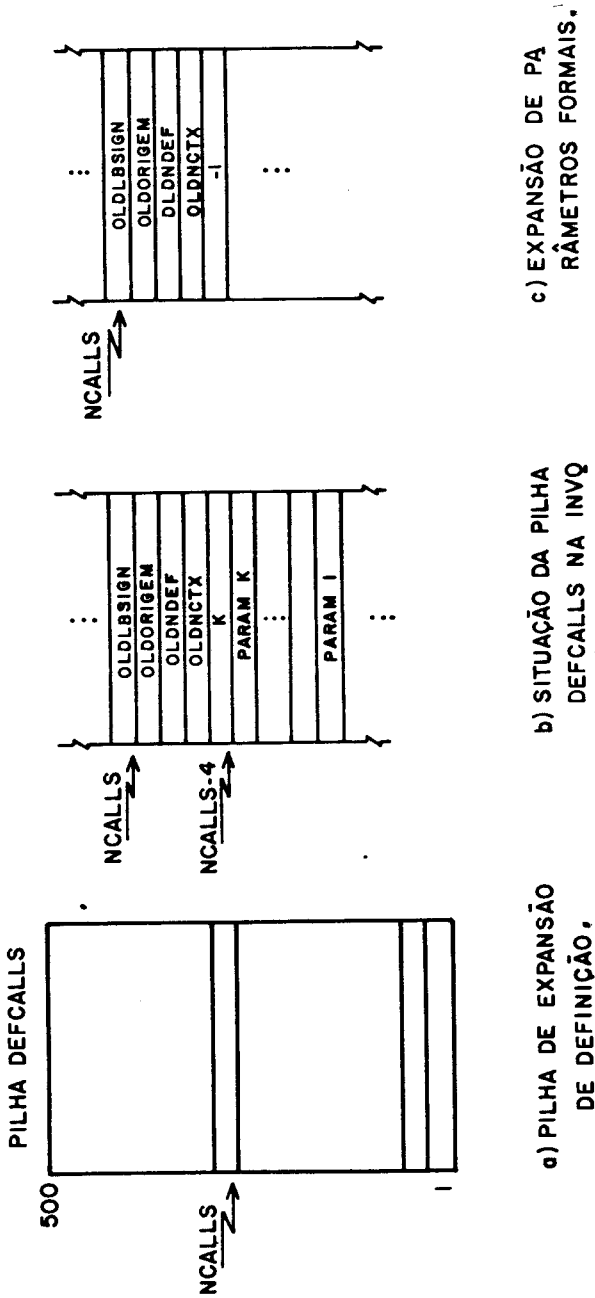


Fig. 12 - Modificação no SCANNER para tratamento de um identificador de definição.



a) PILHA DE EXPANSÃO DE DEFINIÇÃO.

b) SITUAÇÃO DA PILHA DEFCALLS NA INVOCAÇÃO DE UMA DEFINIÇÃO.

c) EXPANSÃO DE PARÂMETROS FORMAIS.

Fig. 13 - PILHA de expansão de definição DEFCALLS.

5. ANÁLISE SINTÁTICA E SEMÂNTICA

À medida que é realizada a Análise Sintática do programa fonte LMP, são geradas as informações semânticas, no caso de declarações, e a forma intermediária, no caso de microcomando e comandos.

Para cada uma das produções da linguagem LMP existe uma rotina correspondente que verifica a validade sintática e semântica do programa fonte.

5.1 - DECLARAÇÕES

As estruturas utilizadas para armazenamento de informações semânticas são apresentadas na Seção 3.

5.2 - COMANDOS E MICROCOMANDOS

A análise sintática dos comandos e microcomandos é realizada por rotinas que correspondem basicamente às produções da linguagem.

Dois tipos de microcomandos foram implementados:

1) microcomando de atribuição;

2) configuração de campo anterior.

Porém, a estrutura utilizada prevê introdução de novos comandos. Ver na listagem do programa fonte as rotinas CMD e MCMD, caso necessário.

6. FORMA INTERMEDIÁRIA

A forma intermediária, denominada FI, resulta do primeiro passo de tradução da linguagem LMP em microcódigos, e constitui dados de interface para o segundo passo.

A Tabela 1 mostra os significados associados a cada código utilizado, e as Tabelas 2 e 3 mostram a implementação dessa tabela.

TABELA 1

FORMA INTERMEDIÁRIA

# FI	MNEMÔNICO	FI GERADA	COMENTÁRIOS
0	CTL	CTL	Cartão de Controle.
1	VAGO		
2	ID	ID <pos>	Identificador simples. <pos> : posição na Tabela Símbolos.
3	SUBS	SUBS <pos><n><p1>...<pn>	Identificador subscrito: <pos> : posição na Tabela Símbolos, <n> : número de parâmetros, <pi> : parâmetro i.
4	VAGO		
5	VAGO		
6	$	$	Referência ao MILC.
7	NRO	NRO <PTCTE>	Número armazenado em TABCTE [PTCTE].
8	CONF	CONF <TAMCONF><PTCTE>	Configuração de bits, com DCARE = 0 e TABCTE [PTCTE] ← VALCTE.
9	CONFX	CONFX <TAMCONF><PTCTE>	Configuração de Bits incluindo "don't care" X. TABCTE [PTCTE] ← VALCTE TABCTE [PTCTE + 1] ← DCARE
10	INSERT	INSERT <pos>	Insere Macro (expansão futura).
11	BLK	BLK	Início de bloco.
12	BLKE	BLKE	Fim de bloco.
13	IMILC	IMILC <valor>	Inicializa MILC com <valor>.
14	ATRIB	ATRIB	Operador de atribuição. Transfere valor armazenado no topo da PILHAXQ para identificação apontado por IANT.
15	FVANT	FVANT	Atribui a ID ou SUBS o valor anterior do campo.
16	MIEND	MIEND	Fim de microinstrução.
17	VPIL	VPIL	Valor no topo da PILHAXQ.
18	FLDA	FLDA	Lado direito de expressão (Faz BLDA = "True").
19	VAGO		
20	*	*	Operador de multiplicação.
21	/	/	Operador de Divisão.
22	+	+	Operador de Soma.
23	-	-	Operador de Subtração.
24	--	--	Operador menos unário.
25	'	'	Complemento para um.
26	"	"	Complemento para dois.
27	&	&	Concatenação.
28	//	//	Truncamento.
29	<	<	Justificação à esquerda.

TABELA 2

DEFINIÇÃO DOS MNEMÔNICOS DA FORMA INTERMEDIÁRIA-POLONESA

```
DEFINE
        FCTL=0#,        % CARTAO DE CONTROLE
        FNRO=7#,        % CONSTANTE
        FLDA=18#,       % LADO DIREITO DE EXPRESSAO
        FID=2#,         % IDENTIFICADOR
        FSUBS=3#,       % VARIAVEL SUBSCRITA
        FMILC=6#,       % REFERENCIA AO MILC
        FCONF=8#,       % CONFIG. DE BITS
        FMIEND=16#,     % FIM DE MICROINSTRUCAO
        FINSERT=10#,    % GERA MICROINSTRUCAO
        FBLK=11#,       % BLOCO
        FBLKE=12#,      % FIM DE BLOCO
        FIMILC=13#,     % INICIALIZA MILC
        FATRIB=14#,     % ATRIBUICAO
        FVANT=15#,      % VALOR ANTERIOR
        FMULT=20#,      % MULTIPLICACAO
        FCONFX=9#,      % CONFIG. DE BITS COM DONTCARE OPCIONAL
        FDIV=21#,       % DIVISAO
        FSOMA=22#,      % SOMA
        FSUB=23#,       % SUBTRACAO
        FCONCAT=24#,    % CONCATENACAO
        FCPM1=25#,      % COMPLEMENTO PARA UM
        FCPM2=26#,      % COMPLEMENTO PARA DOIS
        FMENOS=27#,     % MENOS UNARIO
        FTRUNC=28#,     % TRUNCAMENTO
        FJUST=29#;      % JUSTIFICACAO A ESQUERDA
```

TABELA 3

NOMES DOS OPERANDOS E OPERADORES DA FORMA INTERMEDIÁRIA

```
VALUE ARRAY FINOMES("CTL    ",% 0 - CARTAO DE CONTROLE TABCIL
                    "       ",% 1 -
                    "ID     ",% 2 - IDENTIFICADOR
                   "SUBS    ",% 3 - VARIAVEL SUBSCRITA
                    "       ",% 4 -
                    "       ",% 5 -
                    "%      ",% 6 - REFERENCIA AO MILC
                    "NRO    ",% 7 - NUMERO ARMAZENADO EM MILC
                    "CONF   ",% 8 - CONFIG DE BITS (DCARE=0)
                    "CONFX  ",% 9 - CONFIG DE BITS X (DCARE#0)
                   "INSERT",% 10- OPERADOR INSERE MACRO
                   "BLK    ",% 11- BLOCO
                   "BLKE   ",% 12- FIM DE BLOCO
                   "IMILC  ",% 13- INICIALIZA MILC
                   "ATRIB  ",% 14- ATRIBUICAO
                   "EVANT  ",% 15- VALOR ANTERIOR DO CAMPO
                   "MIEND  ",% 16- FIM DE MICROINSTRUCAO
                   "       ",% 17-
                   "LDA    ",% 18- LADO DIREITO DE ATRIB
                   "       ",% 19-
                   "*      ",% 20- OPR MULTIPLICACAO
                   "/      ",% 21- OPR DIVISAO
                   "+      ",% 22- OPR MAIS
                   "-      ",% 23- OPR MENOS
                   "--     ",% 24- OPR MENOS UNARIO
                   "'      ",% 25- COMPLEMENTO PARA UM
                   "''     ",% 26- COMPLEMENTO PARA DOIS
                   "&      ",% 27- CONCATENACAO
                   "//     ",% 28- TRUNCAMENTO
                   "<      ",% 29- JUSTIFICACAO A ESQUERDA
                   "       ",% 30-
                   "       ",% 31-
                   "       ");% 32-
```

7. GERADOR DE CÓDIGOS CDX (Passo número 2)

O Gerador de Códigos CDX constitui o segundo passo de tradução da linguagem LMP, e utiliza como entrada a Forma Intermediária (FI) definida na Seção anterior e gera como saída a memória de controle na forma semi-final, denominada CDX.

Este passo utiliza ainda as informações armazenadas na Tabela de Símbolos e uma pilha para execução, denominada PILHAXQ. Cada nó desta pilha contém os seguintes campos:

1) VALOR (representa a configuração de bits);

2) DCARE (indica posição "don't care");

3) MTAM (máscara de tamanho da configuração armazenada em VALOR).

A Figura 14 mostra a estrutura de dados envolvida no Passo número 2. O programa GERACDX é basicamente um interpretador da forma Intermediária, que faz uso da PILHAXQ para executar expressões aritméticas ou comandos relacionados com essa forma.

As variáveis I e IANT sinalizam o andamento da interpretação da FI.

Devido às características da linguagem LMP, todas as atribuições realizadas durante a geração de uma microinstrução são armazenadas em variáveis temporárias e ao seu final, sinalizado pela Forma Intermediária MIEND, é verificado o formato utilizado, bem como é realizada a atribuição dos campos com valores por omissão ("default").

Para a realização dessas tarefas ao final de uma microinstrução, a estrutura de tabelas utilizada é apresentada na Figura 15.

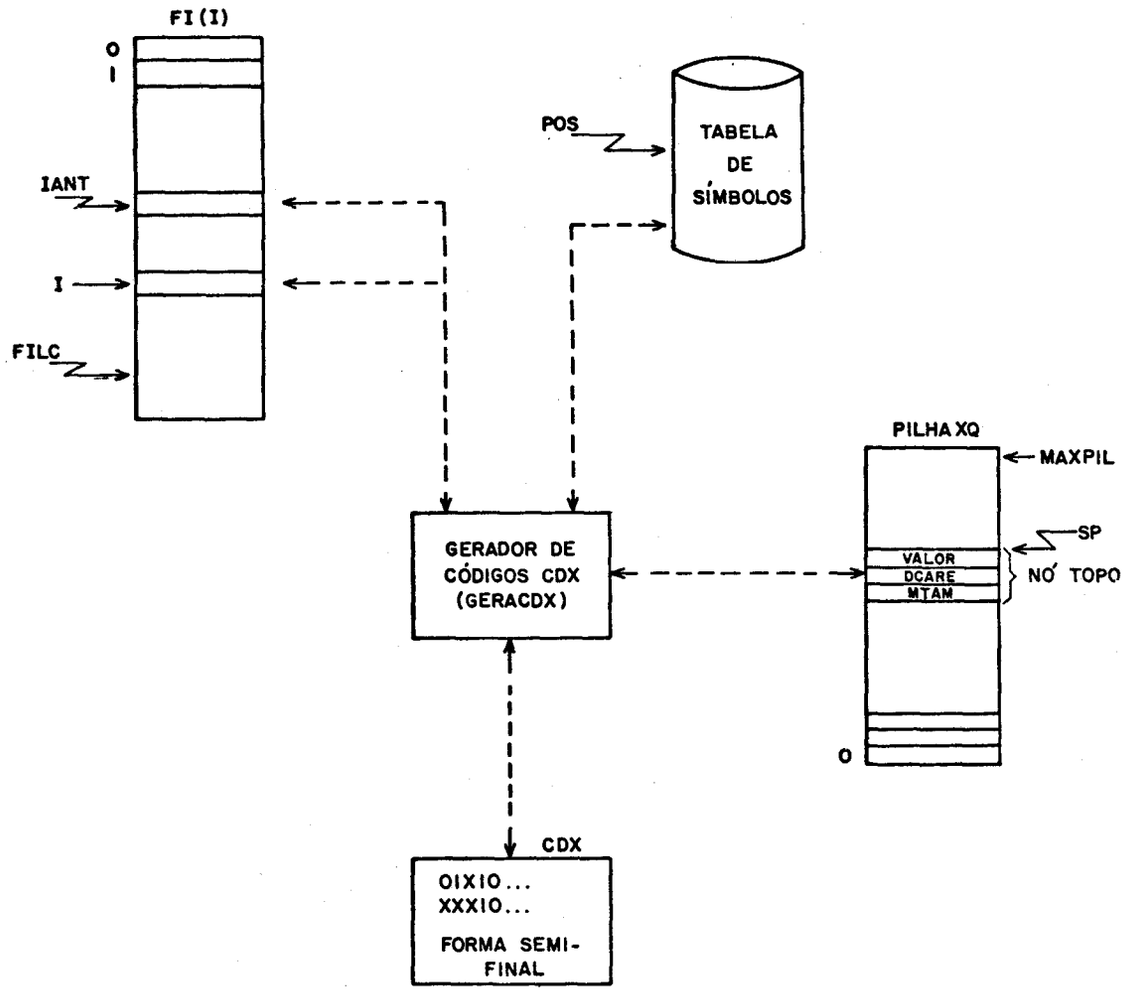
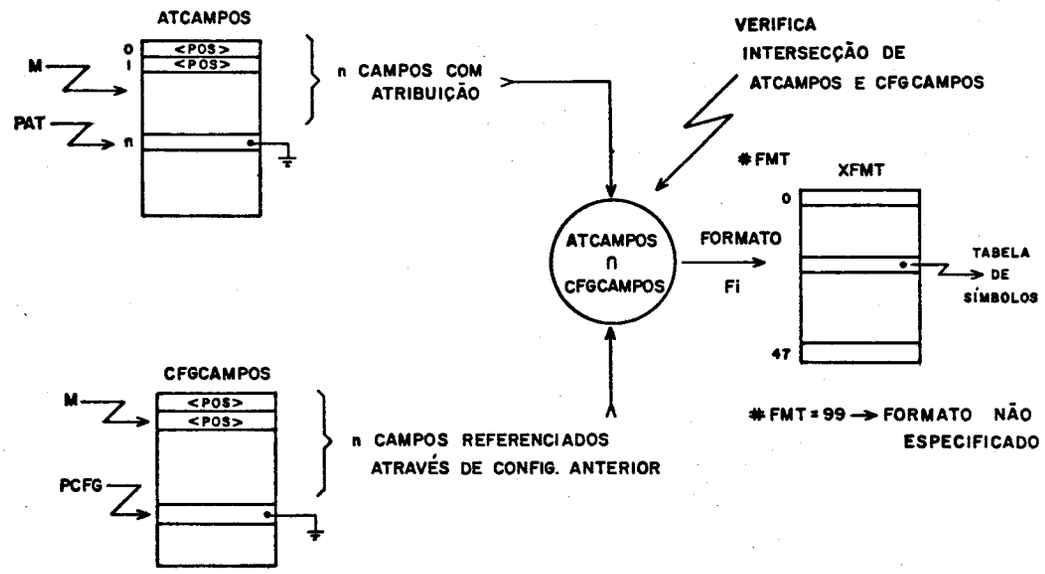


Fig. 14 - Estrutura de Dados do passo número 2.

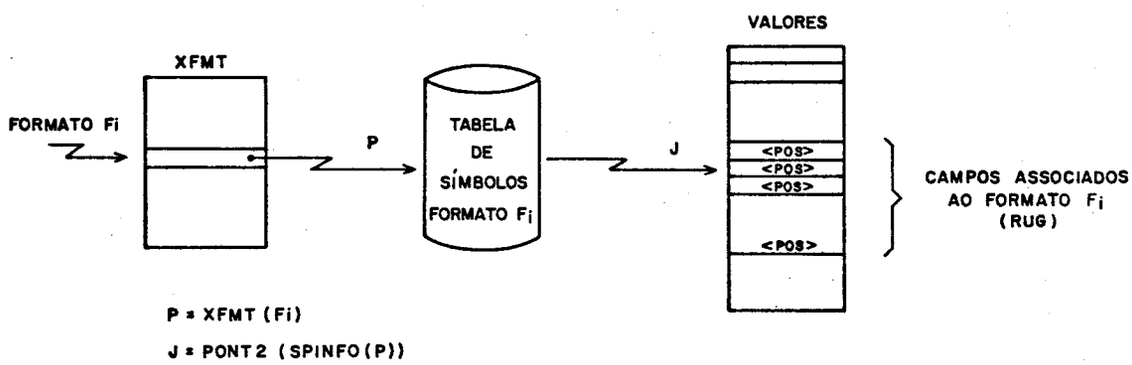
A tabela ATCAMPOS armazena as posições de Tabela de Símbolos associados aos identificadores de campo que receberam atribuição em uma microinstrução.

A tabela CFGCAMPOS armazena os campos referenciados por configuração anterior.

Estas duas tabelas permitem determinar o formato da microinstrução corrente, através de intersecção de campos referenciados em ATCAMPOS e em CFGCAMPOS (Seção 3.2). Se o formato não for único, haverá a emissão de mensagem correspondente.



(a) DETERMINAÇÃO DO FORMATO UTILIZADO.



(b) ATRIBUIÇÃO A CAMPOS COM VALORES POR OMISSÃO ("DEFAULT").

Fig. 15 - Estrutura de tabelas para determinação do formato e atribuição dos campos com valores por omissão.

tar aqueles campos que possuem valor por omissão associado. Para cada formato é associado um conjunto de campos correspondentes (Seção 3.3). Portanto, é verificado, para cada identificador de campo, se ele já recebeu atribuição ou se deve receber valor por omissão, através de consulta às Tabelas ATCAMPOS, CFGCAMPOS e VALORES.

Para esclarecimentos adicionais, ver o programa fonte LMP, passo número 2, onde são apresentadas as rotinas utilizadas, bem como as tabelas mencionadas.

8. RECUPERAÇÃO DE ERROS

A recuperação de erros sintáticos é efetuada pela rotina RECUP, onde basicamente é realizada a chamada ao SCANNER, enquanto o símbolo não for ";", "BEGIN" ou "END".

As rotinas LAVISO, ERRO, LAVISOP, ERROP auxiliam na emissão de mensagens de erro ou de aviso.

É disponível ainda a rotina RECLOC que opera de forma análoga à RECUP, porém aguarda a chegada do símbolo especificado como parâmetro para tentativa de recuperação localizada.

Devido à estrutura a ser adotada no microprograma fonte, houve a necessidade de implementar o diagrama apresentado na Figura 16, para recuperação efetiva e proibir a aceitação de declarações após a deteção de algum comando ou microcomando válido sintaticamente.

Como é possível uma expansão futura, utilizou-se um vetor de estados para caracterizar o estado corrente que pode ser um dos estados abaixo:

1) aceitando declarações (MPST [ ] = 0);

2) aceitando comandos (MPST [ ] = 1);

3) aceitando pós-processamento (MPST [[= 2).

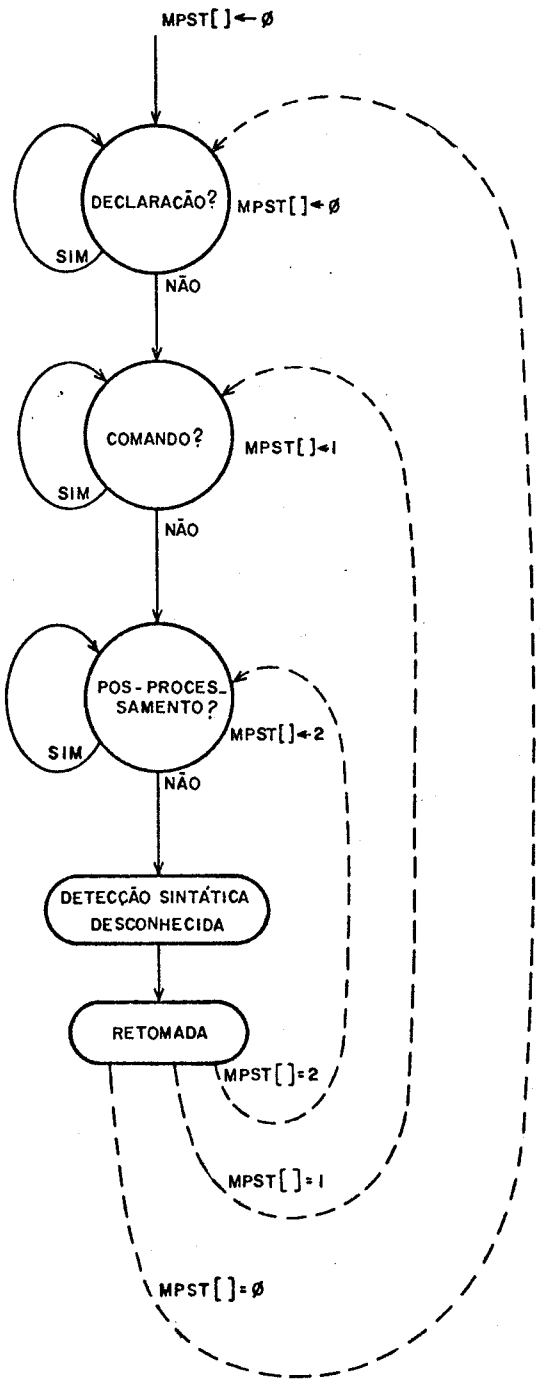


Fig. 16 - Retomafa após deteção de erro na rotina MICROPROGRAMA.

As mudanças de estado são realizadas quando se encontra alguma construção sintática válida. Na inicialização, o estado inicial é aceitando declarações. Uma vez que algum comando válido seja analisado, ocorrerá mudança de estado para o de aceitando comandos. Se uma declaração é analisada, então haverá emissão de mensagem de erro.

Desta forma, quando ocorre algum erro que resulte em não reconhecimento, o diagrama é percorrido até se atingir o estado de detecção sintática desconhecida, e a retomada é realizada no estado anterior à ocorrência de erro.

9. CONSIDERAÇÕES FINAIS

Procurou-se neste manual documentar principalmente as estruturas utilizadas com vistas na manutenção do tradutor LMP versão 1.0.

Com a utilização deste tradutor, erros de construção não detectados até o presente poderão surgir, porém o que poderá ocorrer com maior frequência em função do tamanho do programa fonte em LMP é a necessidade de redimensionar o tamanho de tabelas que integram a Tabela de Símbolos.

Atualmente, esforços têm sido colocados na expansão das facilidades de pós-processamento e introdução de novos comandos visando atender às necessidades coletadas de experiências acumuladas com esta versão.

O Apêndice A contém um exemplo de utilização da linguagem LMP, enquanto no Apêndice B são apresentadas as rotinas que compõem o tradutor LMP versão 1.0.

REFERÊNCIAS BIBLIOGRÁFICAS

AHO, A.V.; ULLMAN, J.D. *Principles of compiler design.* Reading, MA, Addison Wesley, 1978.

BURROUGHS B6600/B7700. *Algol language;* reference manual. Detroit, MI, 1974.

——— B7000/B6000 Series. *CANDE user's manual.* Detroit, MI, 1977.

GRIES, D. *Compiler construction for digital computers.* New York, NY, John Wiley, c 1971.

YAMAGUTI, W. *LMP, uma linguagem de microprogramação.* São José dos Campos, INPE, 1981. (INPE-2031-TDL/049).

WULF, W.; JOHNSON, R.K.; WEINSTOCK, C.B.; HOBBS, S.O.; GESCHKE, C.M. *The design of an optimizing compiler.* New York, American Elsevier, c 1975.

APÊNDICE A

EXEMPLO DE UTILIZAÇÃO DA LINGUAGEM LMP COM OPÇÃO DE LISTAGEM DA FORMA INTERMEDIÁRIA

```
=========================================================================

                    INSTITUTO DE PESQUISAS ESPACIAIS
  LMP - LINGUAGEM DE MICROPROGRAMACAO SISTEMA B6700/B6800 - VERSAO 1.0
      TRADUCAO EXECUTADA EM 30/08/83 AS 14 HORAS E 56 MINUTOS

=========================================================================

ARQUIVO FONTE : EXEMPLO.

BEGIN
COMMENT :- EXEMPLO DE UTILIZACAO DA LINGUAGEM LMP, BASEADO
           NA ARQUITETURA DO "KIT" EDUCATIVO AM2900 DA
           ADVANCED MICRO DEVICES;
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                   %
%                                                                   %
%                    KIT AM2900/EXEMPLO LMP                         %
%                   =====================                           %
%                                                                   %
%                   INPE - DCA / DEA                                %
%                   PROJETO : LASIDA/PSDA                           %
%                   EQUIPAMENTO : KIT AM2900                        %
%                   PLACA : EXEMPLO                                  %
%                   AUTOR : WILSON YAMAGUTI                         %
%                   DATA : 08/AGOSTO/1983                           %
%                                                                   %
%                                                                   %
%                                                                   %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                   %
%...................................................................%
%                                                                   %
%         DECLARACAO DA MEMORIA DE MICROCONTROLE                    %
%                                                                   %
%...................................................................%
%
MEMORY
           CS[0:15,0:31];    % 16 PALAVRAS DE 80 BITS.
%....................................................................
%
%  DECLARACAO DOS CAMPOS DE MICROCONTROLE
%
%....................................................................
%
%
FIELD
     BADR      = CS[0:3]("XXXX"B), % ENDER DE DESVIO
     NEXT.ADDR = CS[4:7]("0010"B), % CONTROLE PROX. ENDER
                                   % DEFAULT: CONT
     MUX1      = CS[8:8]("X"B),    % CONTROLE MUX BIT1
     DEST      = CS[9:11]("001"B), % CONTROLE DE DESTINO
     MUX0      = CS[12:12]("X"B),  % CONTROLE MUX BIT0
     FONTE    = CS[13:15]("XXX"B),% REG. FONTE
     CN        = CS[16:16]("X"B),  % CARRY IN 2901
     ALU       = CS[17:19]("XXX"B),% OPER NA ALU 2901
     SELA      = CS[20:23]("XXXX"B), % SELA/RAM 16 X 4
     SELB      = CS[24:27]("XXXX"B), % SELB/RAM 16 X 4
     DIN       = CS[28:31]("XXXX"B); % ENTRADA D2901
```

```
51 %
52 %
53 %.....................................................................
54 %
55 %   DECLARACAO DE FORMATO (INICIO)
56 %
57 %.....................................................................
58 %
59 COMMENT:- A DECLARACAO DE FORMATO, MESMO QUE UNICA E NECESSARIA;
60    FORMAT
61          F1( BADR,        % ENDER DE DESVIO
62              NEXT.ADDR,   % CTL PROX. ENDER
63              MUX1,        % MUX1/SHIFT 2901
64              DEST,        % DESTINO
65              MUXO,        % MUXO/SHIFT 2901
66              FONTE,      % FONTE R-S
67              CN,          % CARRY IN 2901
68              ALU,         % OPER. NA ALU
69              SELA,        % RAM A
70              SELB,        % RAM B
71              DIN);        % ENTRADA D 2901
72 %
73 %.....................................................................
74 %
75 %   DEFINICAO DE REGISTROS
76 %
77 %.....................................................................
78 %
79   DEFINE
80          R0  = "0"H#,
81          R1  = "1"H#,
82          R2  = "2"H#,
83          R3  = "3"H#,
84          R4  = "4"H#,
85          R5  = "5"H#,
86          R6  = "6"H#,
87          R7  = "7"H#,
88          R8  = "8"H#,
89          R9  = "9"H#,
90          R10 = "A"H#,
91          R11 = "B"H#,
92          R12 = "C"H#,
93          R13 = "D"H#,
94          R14 = "E"H#,
95          R15 = "F"H#;
96 %
97 %.....................................................................
98 %
99 %  DEFINICAO DE OPERANDOS PARA O AM2901
100 %
101 %.....................................................................
102 %
103   DEFINE                    %      R       S
104        AQ = "0"Q#,          %      A       Q
105        AB = "1"Q#,          %      A       B
106        ZQ = "2"Q#,          %     ZERO     Q
107        ZB = "3"Q#,          %     ZERO     B
108        ZA = "4"Q#,          %     ZERO     A
109        DA = "5"Q#,          %      D       A
110        DQ = "6"Q#,          %      D       Q
```

```
171   DEFINE
172         QREG   = DEST:=0#,  % F->Q              , Y=F
173         NOP    = DEST:=1#,  %                   , Y=F
174         RAMA   = DEST:=2#,  % F->B              , Y=A
175         RAMF   = DEST:=3#,  % F->B              , Y=F
176         RAMQD  = DEST:=4#,  % F/2->B, Q/2->Q    , Y=F     (DOWN)
177         RAMD   = DEST:=5#,  % F/2->B            , Y=F     (DOWN)
178         RAMQU  = DEST:=6#,  % 2F->B, 2Q->Q      , Y=F     (UP)
179         RAMU   = DEST:=7#;  % 2F->B,            , Y=F     (UP)
180 %
181 %.....................................................................
182 %
183 %  DEFINICAO DE OPERADORES DE CONTROLE PROXIMO ENDER
184 %
185 %.....................................................................
186 %
187   DEFINE
188         .BRFNO.     =       "0"H#,  % BRANCH REG. IF F#0
189         .BR.        =       "1"H#,
190         .CONT.      =       "2"H#,
191         .BM.        =       "3"H#,
192         .JSRFNO.      =       "4"H#,
193         .JSR.       =       "5"H#,
194         .RTS.       =       "6"H#,
195         .STKREF.    =       "7"H#,
196         .LOOPFNO.   =       "8"H#,
197         .PUSH.      =       "9"H#,
198         .POP.       =       "A"H#,
199         .LOOPCOUT.  =       "B"H#,
200         .BRF.EQ.0.  =       "C"H#,
201         .BRF3.      =       "D"H#,
202         .BROVR.     =       "E"H#,
203         .BRCOUT.    =       "F"H#;
204
205
206
207 %
208 %.....................................................................
209 %
210 %  DEFINICAO DA FUNCAO DE CONTROLE DO PROXIMO ENDERECO
211 %
212 %.....................................................................
213 %
214   DEFINE
215         BRFNO      = NEXT.ADDR:=.BRFNO.     , BADR:=?01#,
216         BR         = NEXT.ADDR:=.BR.        , BADR:=?01#,
217         CONT       = NEXT.ADDR:=.CONT.#     ,
218         BM         = NEXT.ADDR:=.BM.#       ,
219         JSRFNO     = NEXT.ADDR:=.JSRFNO.    , BADR:=?01#,
220         JSR        = NEXT.ADDR:=.JSR.       , BADR:=?01#,
221         RTS        = NEXT.ADDR:=.RTS.#      ,
222         STKREF     = NEXT.ADDR:=.STKREF.#   ,
223         LOOPFNO    = NEXT.ADDR:=.LOOPFNO.#,
224         PUSH       = NEXT.ADDR:=.PUSH.#     ,
225         POP        = NEXT.ADDR:=.POP.#      ,
226         LOOPCOUT   = NEXT.ADDR:=.LOOPCOUT.#,
227         BRF.EQ.0   = NEXT.ADDR:=.BRF.EQ.0.  , BADR:=?01#,
228         BRF3       = NEXT.ADDR:=.BRF3.      , BADR:=?01#,
229         BROVR      = NEXT.ADDR:=.BROVR.     , BADR:=?01#,
230         BRCOUT     = NEXT.ADDR:=.BRCOUT.    , BADR:=?01#;
```

```
231 %
232 %.....................................................
233 %
234 %  OUTRAS DEFINICOES
235 %
236 %.....................................................
237 %
238   DEFINE
239        CN0  = "0"B#,
240        CN1  = "1"B#,
241        LOW  = "0"B#,
242        HIGH = "1"B#,
243        ZERO = "0"B#,
244        ONE  = "1"B#;
245 %
246 %.....................................................
247 %
248 %  DEFINICAO DE MICROFUNCOES RELACIONADAS C/O ALGORITMO EXEMPLO
249 %
250 %.....................................................
251 %
252   DEFINE
253        LOAD   = RAMF,        % CARREGA REGISTRO COM VALOR IMEDIATO
254                 OR(DZ),      % CHAMADA LOAD(RI, VALOR)
255                 SELB:=?01,   % <==> RI:=VALOR
256                 DIN:=?02#,
257        IFCALL = JSRFNO(INCR3),        % DESVIO CONDICIONAL //
258                 RAMD,        % RI <= RI/2
259                 OR(ZB),
260                 SELB:=?01#;
261 %
262 %
263 %
264 %.....................................................
265 %
266 %  DECLARACAO DE ROTULOS
267 %
268 %.....................................................
269 %
270    LABEL LOOP.1, LOOP.2, INCR3;
271 %
272 %.....................................................
273 %
274 %  DECLARACAO DA SUBROTINA  INCR3
275 %
276 %.....................................................
277 %
278 %
279 %=====================================================
280 %
281 %     MICROPROGRAMA  PRINCIPAL
282 %
283 %=====================================================
284 %
285    $ORIGIN 0
286        LOAD(R0,15);     % V0 =15
287        LOAD(R1,9);      % V1 = 9
288        LOAD(R2,0);      % V2 = 0
289        LOAD(R4,4);      % R4 = 4
290        RAMF,AND(ZB), SELB:=R3;       % CLEAR R3
```

```
291 LOOP.1:AND(DA), SELA:=R0, SELB:=R0, DIN:="1"H;
292                          % R0 AND D E D=0001
293                          % 5
294 IFCALL( R0);      % CALL SUBROTINA INCR3 SE F#0
295                                % R0 <= R0/2
296                                % 6
297 AND(DA), SELA:=R1, SELB:=R1, DIN:="1"H;   % 7
298 IFCALL(R1);                  % 8     R1 <= R1/2
299 AND(DA), SELA:=R2, SELB:=R2, DIN:="1"H;   % 9
300 IFCALL(R2);                  % 10
301 SUBR(ZB),RAMF,SELB:=R4,CN:=CN0;          % 11
302 BRFND(LOOP.1);                           % 12
303 BR(LOOP.2);                             % 13
304 INCR3: RTS,ADD(ZB),SELB:=R3,CN:=CN1;     % ROTINA R3:=R3+1
305                                          % 14
306 LOOP.2: BR(LOOP.2), OR(ZB), SELB:=R3;    % 15
307 $SET MEMORY
308 $SET INTER
309 END
```

```
**=================================================================**
NUMERO DE ERROS DETETADOS =  0.     NUMERO DE AVISOS EMITIDOS =  0.
NUMERO DE REGISTROS DO MICROPROGRAMA FONTE =  309.
FORMA INTERMEDIARIA = 519.          MICROINSTRUCOES GERADAS=   16.
TRADUTOR LMP COMPILADO EM 06.07.83 NO SISTEMA B6800/INPE
PROJ.LASIDA/PSDA/INPE-DEPTO ENG.COMPUTACAO EM APLICACOES ESPACIAIS
**=================================================================**
```

LMP - VERSAO 1.0 - FORMA INTERMEDIARIARIA DO MICROPROGRAMA FONTE

CARTAO	POSICAO	FINTER	SIMBOLO/VALOR		
1	0	BLK			
285	1	IMILC		0	
286	3	ID	DEST		
	5	LDA			
	6	NRO		3	
	8	ATRIB			
	9	ID	ALU		
	11	LDA			
	12	NRO		3	
	14	ATRIB			
	15	ID	FONTE		
	17	LDA			
	18	CONF		3	7
	21	ATRIB			
	22	ID	SELB		
	24	LDA			
	25	CONF		4	0
	28	ATRIB			
	29	ID	DIN		
	31	LDA			
	32	NRO		15	
	34	ATRIB			
	35	MIEND			
287	36	ID	DEST		
	38	LDA			
	39	NRO		3	
	41	ATRIB			
	42	ID	ALU		
	44	LDA			
	45	NRO		3	
	47	ATRIB			
	48	ID	FONTE		
	50	LDA			
	51	CONF		3	7
	54	ATRIB			
	55	ID	SELB		
	57	LDA			
	58	CONF		4	1
	61	ATRIB			
	62	ID	DIN		
	64	LDA			
	65	NRO		9	
	67	ATRIB			
	68	MIEND			
288	69	ID	DEST		
	71	LDA			
	72	NRO		3	
	74	ATRIB			
	75	ID	ALU		
	77	LDA			
	78	NRO		3	
	80	ATRIB			

```
          81      ID          FONTE
          83      LDA
          84      CONF                3           7
          87      ATRIB
          88      ID          SELB
          90      LDA
          91      CONF                4           2
          94      ATRIB
          95      ID          DIN
          97      LDA
          98      NRO                 0
         100      ATRIB
         101      MIEND
289      102      ID          DEST
         104      LDA
         105      NRO                 3
         107      ATRIB
         108      ID          ALU
         110      LDA
         111      NRO                 3
         113      ATRIB
         114      ID          FONTE
         116      LDA
         117      CONF                3           7
         120      ATRIB
         121      ID          SELB
         123      LDA
         124      CONF                4           4
         127      ATRIB
         128      ID          DIN
         130      LDA
         131      NRO                 4
         133      ATRIB
         134      MIEND
290      135      ID          DEST
         137      LDA
         138      NRO                 3
         140      ATRIB
         141      ID          ALU
         143      LDA
         144      NRO                 4
         146      ATRIB
         147      ID          FONTE
         149      LDA
         150      CONF                3           3
         153      ATRIB
         154      ID          SELB
         156      LDA
         157      CONF                4           3
         160      ATRIB
         161      MIEND
291      162      ID          ALU
         164      LDA
         165      NRO                 4
         167      ATRIB
         168      ID          FONTE
         170      LDA
         171      CONF                3           5
         174      ATRIB
         175      ID          SELA
```

```
         275     CONF                    4               4
         278     ATRIB
         279     ID          BADR
         281     LDA
         282     ID          INCR3
         284     ATRIB
         285     ID          DEST
         287     LDA
         288     NRO                     5
         290     ATRIB
         291     ID          ALU
         293     LDA
         294     NRO                     3
         296     ATRIB
         297     ID          FONTE
         299     LDA
         300     CONF                    3               3
         303     ATRIB
         304     ID          SELB
         306     LDA
         307     CONF                    4               1
         310     ATRIB
         311     MIEND
299      312     ID          ALU
         314     LDA
         315     NRO                     4
         317     ATRIB
         318     ID          FONTE
         320     LDA
         321     CONF                    3               5
         324     ATRIB
         325     ID          SELA
         327     LDA
         328     CONF                    4               2
         331     ATRIB
         332     ID          SELB
         334     LDA
         335     CONF                    4               2
         338     ATRIB
         339     ID          DIN
         341     LDA
         342     CONF                    4               1
         345     ATRIB
         346     MIEND
300      347     ID          NEXT.ADDR
         349     LDA
         350     CONF                    4               4
         353     ATRIB
         354     ID          BADR
         356     LDA
         357     ID          INCR3
         359     ATRIB
         360     ID          DEST
         362     LDA
         363     NRO                     5
         365     ATRIB
         366     ID          ALU
         368     LDA
         369     NRO                     3
         371     ATRIB
```

```
         372     ID          FONTE
         374     LDA
         375     CONF                    3               3
         378     ATRIB
         379     ID          SELB
         381     LDA
         382     CONF                    4               2
         385     ATRIB
         386     MIEND
301      387     ID          ALU
         389     LDA
         390     NRO                     1
         392     ATRIB
         393     ID          FONTE
         395     LDA
         396     CONF                    3               3
         399     ATRIB
         400     ID          DEST
         402     LDA
         403     NRO                     3
         405     ATRIB
         406     ID          SELB
         408     LDA
         409     CONF                    4               4
         412     ATRIB
         413     ID          CN
         415     LDA
         416     CONF                    1               0
         419     ATRIB
         420     MIEND
302      421     ID          NEXT.ADDR
         423     LDA
         424     CONF                    4               0
         427     ATRIB
         428     ID          BADR
         430     LDA
         431     ID          LOOP.1
         433     ATRIB
         434     MIEND
303      435     ID          NEXT.ADDR
         437     LDA
         438     CONF                    4               1
         441     ATRIB
         442     ID          BADR
         444     LDA
         445     ID          LOOP.2
         447     ATRIB
         448     MIEND
304      449     ID          NEXT.ADDR
         451     LDA
         452     CONF                    4               6
         455     ATRIB
         456     ID          ALU
         458     LDA
         459     NRO                     0
         461     ATRIB
         462     ID          FONTE
         464     LDA
         465     CONF                    3               3
         468     ATRIB
```

```
              469      ID          SELB
              471      LDA
              472      CONF                    4          3
              475      ATRIB
              476      ID          CN
              478      LDA
              479      CONF                    1          1
              482      ATRIB
              483      MIEND
    306       484      ID          NEXT.ADDR
              486      LDA
              487      CONF                    4          1
              490      ATRIB
              491      ID          BADR
              493      LDA
              494      ID          LOOP.2
              496      ATRIB
              497      ID          ALU
              499      LDA
              500      NRO                     3
              502      ATRIB
              503      ID          FONTE
              505      LDA
              506      CONF                    3          3
              509      ATRIB
              510      ID          SELB
              512      LDA
              513      CONF                    4          3
              516      ATRIB
              517      MIEND
    309       518      BLKE

*PFI:
    0    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
    1    1    1    1    1    1    1    1    1    1
```

```
  1     1     1     1     1     1     1     1     1     1
  1     1     1     1     1     3    36    69   102   135
162   197   197   197   237   237   237   272   312   347
387   421   435   449   484   484   518   518   518     0

*FI:
 11    13     0     2     5    18     7     0    14     2
  9    18     7     1    14     2     7    18     8     3
  2    14     2    11    18     8     4     3    14     2
 12    18     7     4    14    16     2     5    18     7
  5    14     2     9    18     7     6    14     2     7
 18     8     3     7    14     2    11    18     8     4
  8    14     2    12    18     7     9    14    16     2
  5    18     7    10    14     2     9    18     7    11
 14     2     7    18     8     3    12    14     2    11
 18     8     4    13    14     2    12    18     7    14
 14    16     2     5    18     7    15    14     2     9
 18     7    16    14     2     7    18     8     3    17
 14     2    11    18     8     4    18    14     2    12
 18     7    19    14    16     2     5    18     7    20
 14     2     9    18     7    21    14     2     7    18
  8     3    22    14     2    11    18     8     4    23
 14    16     2     9    18     7    24    14     2     7
 18     8     3    25    14     2    10    18     8     4
 26    14     2    11    18     8     4    27    14     2
 12    18     8     4    28    14    16     2     3    18
  8     4    29    14     2     2    18     2   108    14
  2     5    18     7    30    14     2     9    18     7
 31    14     2     7    18     8     3    32    14     2
 11    18     8     4    33    14    16     2     9    18
  7    34    14     2     7    18     8     3    35    14
  2    10    18     8     4    36    14     2    11    18
  8     4    37    14     2    12    18     8     4    38
 14    16     2     3    18     8     4    39    14     2
  2    18     2   108    14     2     5    18     7    40
 14     2     9    18     7    41    14     2     7    18
  8     3    42    14     2    11    18     8     4    43
 14    16     2     9    18     7    44    14     2     7
 18     8     3    45    14     2    10    18     8     4
 46    14     2    11    18     8     4    47    14     2
 12    18     8     4    48    14    16     2     3    18
  8     4    49    14     2     2    18     2   108    14
  2     5    18     7    50    14     2     9    18     7
 51    14     2     7    18     8     3    52    14     2
 11    18     8     4    53    14    16     2     9    18
  7    54    14     2     7    18     8     3    55    14
  2     5    18     7    56    14     2    11    18     8
  4    57    14     2     8    18     8     1    58    14
 16     2     3    18     8     4    59    14     2     2
 18     2   106    14    16     2     3    18     8     4
 60    14     2     2    18     2   107    14    16     2
  3    18     8     4    61    14     2     9    18     7
 62    14     2     7    18     8     3    63    14     2
 11    18     8     4    64    14     2     8    18     8
  1    65    14    16     2     3    18     8     4    66
 14     2     2    18     2   107    14     2     9    18
  7    67    14     2     7    18     8     3    68    14
  2    11    18     8     4    69    14    16    12     0
```

*** LMP VERSAO 1.0 - INPE/B6800 ***

CARTAO FONTE	ENDERECO DEC	HEX	MICROCODIGOS NA FORMA SEMI-FINAL
286	0	0000	....0010.011.111.011....00001111
287	1	0001	....0010.011.111.011....00011001
288	2	0002	....0010.011.111.011....00100000
289	3	0003	....0010.011.111.011....01000100
290	4	0004	....0010.011.011.100....0011....
291	5	0005	....0010.001.101.100000000000001
294	6	0006	11100100.101.011.011....0000....
297	7	0007	....0010.001.101.100000100010001
298	8	0008	11100100.101.011.011....0001....
299	9	0009	....0010.001.101.100001000100001
300	10	000A	11100100.101.011.011....0010....
301	11	000B	....0010.011.0110001....0100....
302	12	000C	01010000.001.....
303	13	000D	11110001.001.....
304	14	000E	....0110.001.0111000....0011....
306	15	000F	11110001.001.011.011....0011....

APÊNDICE B

ROTINAS DO TRADUTOR LMP VERSÃO 1.Ø

1) DBGDEF

- Lista os conteúdos das tabelas DEFNOMES, DEFINFO, TEXTOS e DEFCALLS.

2) LISTFI

- Lista a forma intermediária - polonesa.

3) LISEM (I, LFMT, NROVAL)

- Lista a informação semântica armazenada na Tabela de Símbolos.
- I: posição na tabela TIDREF.
- LFMT: tipo de saída.
- NROVAL: número de posições em valores.

4) LTBLOCK

- Lista a tabela TBLOCK.

5) LTIDREF

- Lista a tabela TIDREF e o cordão de caracteres do identificador.

6) LTHASH

- Lista os conteúdos da tabela THASH.

7) DEBUG

- Lista as tabelas TBLOCK, TIDREF e THASH.

8) HASH (PNOME)

- Calcula a função HASH do "string" apontado pelo "pointer" PNOME, considerando somente os seis primeiros caracteres.
- retorna, em HASH, o valor da função ident. MOD 101.

9) INSERT (POS)

- Insere o identificador na Tabela de Símbolos.
- POS: endereço da tabela TIDREF onde foi criado um nó para o identificador.

10) LOOKUP (POS)

- Procura identificar em todos os blocos ativos.
- POS: é a posição na tabela TIDREF associada ao identificador procurado (se POS > 0).

11) LOOKCURBLK (POS)

- Procura identificar no bloco corrente.
- POS: idem ao anterior.

12) OPENBLOCK

- Abre novo bloco, criando nó em TCLOCK.

13) CLOSEBLOCK

- Transfere bloco corrente da parte ativa para a inativa.

14) CLASSE

- Indica a classe do caractere apontador por CHAR.

15) PROXIMOCHAR

- Fornece ao SCANNER cada caractere do microprograma fonte. Se origem = 0 pega caracteres de CARTÃO e, se for o caso, imprime cada cartão lido. Se origem > 0 é expansão de definição e pega caracteres do "array" TEXTOS.

16) PROCURATABELA (INÍCIO, FIM)

- Verifica se o símbolo está na tabela TABSIMB no intervalo indicado.

17) ERRO (N)

- Imprime as mensagens de erro indicando o ponto em que o SCANNER parou.

18) LAVISO (N)

- Idem para mensagens de aviso.

19) LAVISOP (N, POS, NC)

- Imprime mensagens de aviso com informação do cartão fonte e identificador associado.

20) ERROP (N, POS, NC)

- Idem à anterior. Mensagens de erro.

21) SCANNER

- Faz análise léxica do microprograma fonte, retornando em SÍMBOLO o "string" de caracteres identificados e, em NSIMB, o número de representação interna.

22) PEGAPARAM

- Rotina que reconhece e armazena um parâmetro real delimitado por vírgulas ou por barras (!).

23) SINRESERVE

- Interpreta e executa opção de controle.
- Incrementa MILC pelo valor dado.

24) SINORIGIN

- Interpreta e executa opção de controle.
- Inicializa valor de MILC.

25) OPCAOLIST (FLAG)

- Interpreta e executa opção de listagem,fonecida em um cartão de controle do tipo SET e RESET.

26) XQCTL

- Rotina que chama a execução de opções de controle.

27) RECUP

- Rotina que realiza a recuperação do erro sintático.

28) RECLOC (SSIMB)

- Análogo ao anterior, porém aguarda ainda a chegada do símbolo SSIMB.

29) VALOR

- Transforma o valor de constantes fornecidas em alfanumérico em valor interno de operação.

30) OCUPA (INÍCIO, FIM)

- Rotina que procura detetar a sobreposição de bits ou campos.

31) ATUALIZAMILC

- Atualiza o contador de alocação de microinstrução.
- Verifica se houve "overflow" em área de memória de controle.

32) LMODIF

- Faz a análise dos modificadores e execução.

33) LINVOCMICRO

- Faz análise sintática da lista de parâmetros de uma invocação de microinstrução parametrizada.

34) CONCAT

- Faz análise sintática do microcomando de concatenação.

35) PRIM

- Faz análise sintática do fator e o cálculo do valor correspondente.

36) FATOR

- Faz análise sintática do termo e o calculo do valor correspondente.

37) EXPR

- Faz análise sintática de expressão.

38) PARDELIM (PAR1, PAR2)

- Faz análise do par <constante> : <constante>

39) ESPCAMPO

- Faz análise sintática da produção

  <espec campo> :: = <ident> = <ident mem> [<par delim>]
  {<constante>}

40) SINFIELD

- Faz análise da declaração de campo.

41) ESPFORMAT

- Faz análise da produção

  <espec formato> :: = <ident> (<lista de ident>)

42) SINFORMAT

- Faz análise da declaração de formato.

43) SINLABEL

- Faz análise da declaração de rótulo.

44) SINPROCEDURE (expansão)

- Faz análise da declaração de "procedure".

45) SINMEMORY

- Faz análise da declaração de memória de controle.

46) SINDEFINE

- Faz análise da declaração de definição.

47) MONTADEF

- Armazena contexto do "define".

48) SINCOMMENT

- Trata cartão de comentários.

49) DECL

- Faz análise e chamada de rotinas específicas referentes a cada declaração.

50) MCMD

- Faz análise de um microcomando, bem como a geração da forma intermediária.

51) CMD

- Faz análise do comando rotulado ou não.

52) CMDCP (expansão)

- Faz análise do comando composto.

53) BLOCO (expansão)

- Faz análise do bloco.

54) LCMD

- Faz análise da lista de comandos, separados por ponto e vírgula.

55) MICROPROGRAMA

- Faz análise sintática de um microprograma.

56) FILLX (ENDEREÇO, NROPOS)

- Inicializa a palavra de controle a ser gerada com estados "don't care".

57) GERACAMPO (ENDEREÇO, TPOSINIC, TPOSFIM, TVALOR, TDCARE)

- Armazena a configuração de bits fornecida em TVALOR e TDCARE no campo especificado.

58) DUMPCDX (ADRM1, ADRM2)

- Rotina para descarga em impressora dos bits da memória de controle na forma semi-final.

59) LECAMPO (ENDEREÇO, TPOSINIC, TPOSFIM, TVALOR e TDCARE)

- Lê a configuração de bits armazenada na memória de controle colocando em TVALOR e TDCARE.

60) GERACDX

- Rotina que implementa o passo número 2 de tradução do programa fonte em LMP.