

1. Publicação nº <i>INPE-2479-NTE/189</i>	2. Versão	3. Data <i>Julho, 1982</i>	5. Distribuição <input type="checkbox"/> Interna <input checked="" type="checkbox"/> Externa <input type="checkbox"/> Restrita
4. Origem <i>DCA/DEA</i>	Programa <i>SISMAG/SDA</i>		
6. Palavras chaves - selecionadas pelo(s) autor(es) <i>MICROPROGRAMAÇÃO</i> <i>CONTROLES DIGITAIS MICROPROGRAMADOS</i>			
7. C.D.U.: <i>681.326.32</i>			
8. Título <i>INPE-2479-NTE/189</i>  <i>IDÉIAS BÁSICAS PARA A IMPLEMENTAÇÃO</i> <i>DE CONTROLES MICROPROGRAMADOS</i>		10. Páginas: <i>46</i>	
		11. Última página: <i>39</i>	
9. Autoria <i>Almir Cavalcanti Lemos Filho</i>		12. Revisada por <i>Ricardo C.O. Martins</i>	
		13. Autorizada por <i>Nelson de Jesus Parada</i> Diretor	
Assinatura responsável <i>Almir Cavalcanti Lemos Filho</i>			
14. Resumo/Notas  <i>Discute-se a implementação da parte de controle de sistemas digitais, utilizando-se microprogramação. Apresentam-se as vantagens, bem como algumas estruturas que incorporam as idéias básicas relativas à aplicação deste método de projeto.</i>			
15. Observações			

ABSTRACT

*The implementation of digital systems control part by microprogramming is discussed. The advantages and the disadvantages are presented as well as some control structures that incorporate basic ideas related to the application of this method of design.*



## SUMÁRIO

	<u>Pág.</u>
LISTA DE FIGURAS .....	v
<u>CAPÍTULO 1 - INTRODUÇÃO</u> .....	1
<u>CAPÍTULO 2 - A PARTE DE CONTROLE DE UM SISTEMA DIGITAL</u> .....	3
2.1 - A parte de controle .....	3
2.2 - Classificação da parte de controle .....	5
2.2.1 - Controle classe 0 .....	5
2.2.2 - Controle classe 1 .....	6
2.2.3 - Controle classe 2 .....	6
2.2.4 - Controle classe 3 .....	7
2.2.5 - Controle classe 4 .....	8
<u>CAPÍTULO 3 - IMPLEMENTAÇÃO DA PARTE DE CONTROLE</u> .....	9
3.1 - Metodologias de implementação da parte de controle .....	9
3.1.1 - Controle implementado com lógica aleatória .....	9
3.1.2 - Controle microprogramado .....	9
3.2 - Lógica aleatória x microprogramação .....	9
3.2.1 - Rapidez de projeto .....	9
3.2.2 - Rapidez de teste .....	10
3.2.3 - Velocidade do circuito .....	10
3.2.4 - Flexibilidade do projeto .....	11
3.2.5 - Custo do desenvolvimento .....	11
3.2.6 - Custo do circuito .....	11
3.2.7 - Confiabilidade .....	12
<u>CAPÍTULO 4 - O CONTROLE MICROPROGRAMADO</u> .....	13
4.1 - Estrutura de controle classe 4 .....	13
4.2 - Estrutura de controle classe 3 - Redução da MMC .....	16
4.2.1 - Controle com a influência de uma só entrada na determinação do próximo estado .....	17
4.2.2 - Controle com uma das opções de próximo estado preestabelecida .....	18
4.2.3 - Controle com formato variável .....	20

	<u>Pág.</u>
4.2.4 - Codificação de sinais de controle .....	22
4.2.5 - Consideração sobre a minimização da MMC .....	24
4.3 - Estruturas de controle classe 3 - Aumento de velocidade ..	25
4.3.1 - Controle com registro "pipeline" .....	25
4.3.2 - Controle com registro "pipeline" e previsão da próxima microinstrução .....	28
4.4 - Estruturas de controle classe 3 - Aumento da eficiência na determinação da próxima microinstrução .....	31
4.4.1 - Controle com registro "pipeline" e registro auxiliar ...	31
4.4.2 - Controle com PROM de mapeamento .....	34
<u>CAPÍTULO 5 - CONCLUSÕES FINAIS</u> .....	37
REFERÊNCIAS BIBLIOGRÁFICAS .....	39

## LISTA DE FIGURAS

	<u>Pág.</u>
1.1 - Partes básicas de um sistema digital .....	1
2.1 - Sistema Digital com a parte de controle detalhada .....	4
2.2 - Representação geral da parte de controle de um sistema di gital .....	4
2.3 - Controle classe 0 .....	5
2.4 - Controle classe 1 .....	6
2.5 - Controle classe 2 .....	7
2.6 - Controle classe 3 .....	7
2.7 - Controle classe 4 .....	8
4.1 - Estruturas de um controle classe 4 microprogramado .....	13
4.2 - Componente 82S104/105 da Signetics Corp., usado na imple mentação de controles classe 4 .....	15
4.3 - Estrutura de um controle microprogramado com influência de uma só entrada na determinação do próximo estado .....	17
4.4 - Estrutura de um controle microprogramado com uma das opções de próximo estado preestabelecida .....	19
4.5 - Estrutura de um controle microprogramado com formato variã vel .....	21
4.6 - Estrutura com codificação dos sinais de controle de um blo co funcional da parte controlada .....	22
4.7 - Estrutura com codificação na geração de pulsos de controle mutuamente exclusivos .....	23
4.8 - Atrasos envolvidos no cálculo de frequência máxima da estru tura da Figura 4.4. ....	26
4.9 - Estrutura de um controle microprogramado com registro "pi line" .....	27
4.10 - Estrutura de um controle microprogramado com registro "pi peline" e previsão do endereço da próxima $\mu$ I .....	29
4.11 - Formas de onda do controle da Figura 4.10 .....	30
4.12 - Algoritmo com trechos idênticos repetidos (a) e com o tre cho repetido implementado só uma vez na MMC (b) .....	32
4.13 - Estrutura de um controle microprogramado com registro "pi peline" e registro auxiliar (R) .....	33
4.14 - Estrutura de um controle microprogramado com PROM de mapea mento .....	35

•••

## CAPÍTULO 1

### INTRODUÇÃO

O conceito de microprogramação foi inicialmente proposto por Wilkes (1951) como um método para implementar as instruções em linguagem de máquina de um processador. Com o desenvolvimento tecnológico de memórias rápidas e o surgimento de componentes LSI, a microprogramação passou a ser encarada genericamente nos dias de hoje como uma maneira rápida, barata e flexível de se projetar e construir a parte de controle de um sistema digital (Figura 1.1), especialmente a dos mais complexos (Rausher and Adams, 1980).

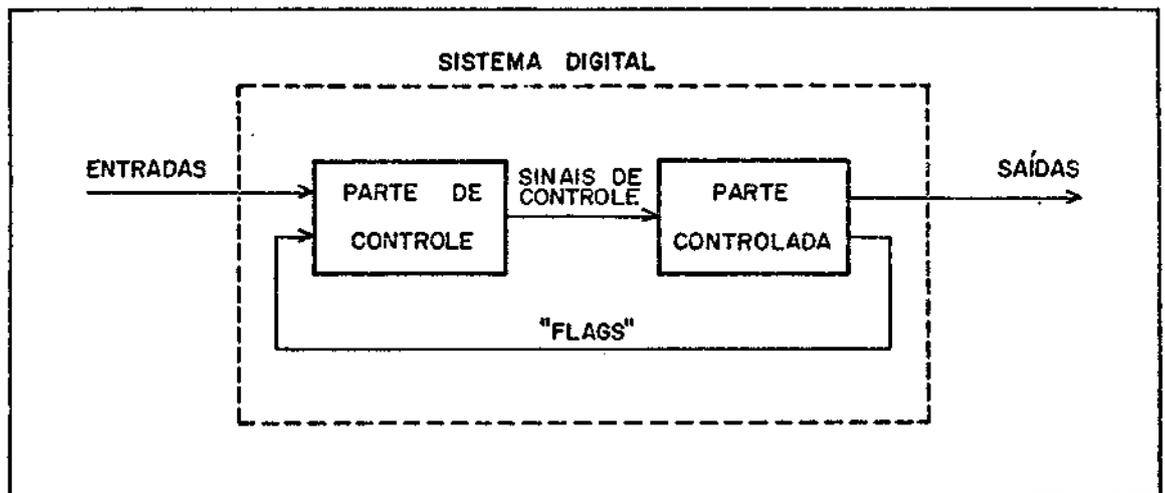


Fig. 1.1 - Partes básicas de um sistema digital.

Este trabalho objetiva a apresentação de idéias básicas envolvidas na implementação da parte de controle de sistemas digitais, utilizando-se microprogramação. Inicialmente (Capítulo 2) a parte de controle é classificada segundo o proposto por Clare (1973). No Capítulo 3 são discutidas as vantagens e desvantagens de duas metodologias distintas, aplicadas na implementação da parte de controle: o projeto tradicional com lógica aleatória e o microprogramado. A seguir (Capítulo 4) são

apresentadas e comentadas estruturas de partes de controle classe 3 (autômato de Moore) microprogramadas, enfocando-se a redução da memória de microcontrole e sua influência na velocidade do sistema digital em questão.

## CAPÍTULO 2

### A PARTE DE CONTROLE DE UM SISTEMA DIGITAL

#### 2.1 - ELEMENTOS QUE COMPÕEM A PARTE DE CONTROLE

A parte de controle de um sistema digital é uma *máquina de estados finita*, com maior ou menor complexidade, de acordo com o sistema a que se destina, podendo sempre ser reduzida à configuração da Figura 2.1, onde se tem:

- a) A memória que armazena o ESTADO da máquina, ou seja, a história passada da parte de controle suficiente para determinar o comportamento futuro.
- b) Uma lógica de DETERMINAÇÃO DO PRÓXIMO ESTADO que é função do estado da máquina, dos "flags" originados na parte controlada e das entradas do sistema.
- c) Uma lógica de GERAÇÃO DOS SINAIS DE CONTROLE para a parte controlada que depende do estado da máquina, das entradas do sistema e dos "flags" da parte controlada.

Vistas da parte de controle, as entradas do sistema digital e os "flags" originados na parte controlada podem ser ambos considerados simplesmente "entradas" da parte de controle. Assim, na Figura 2.2 mostra-se uma representação geral da parte de controle.

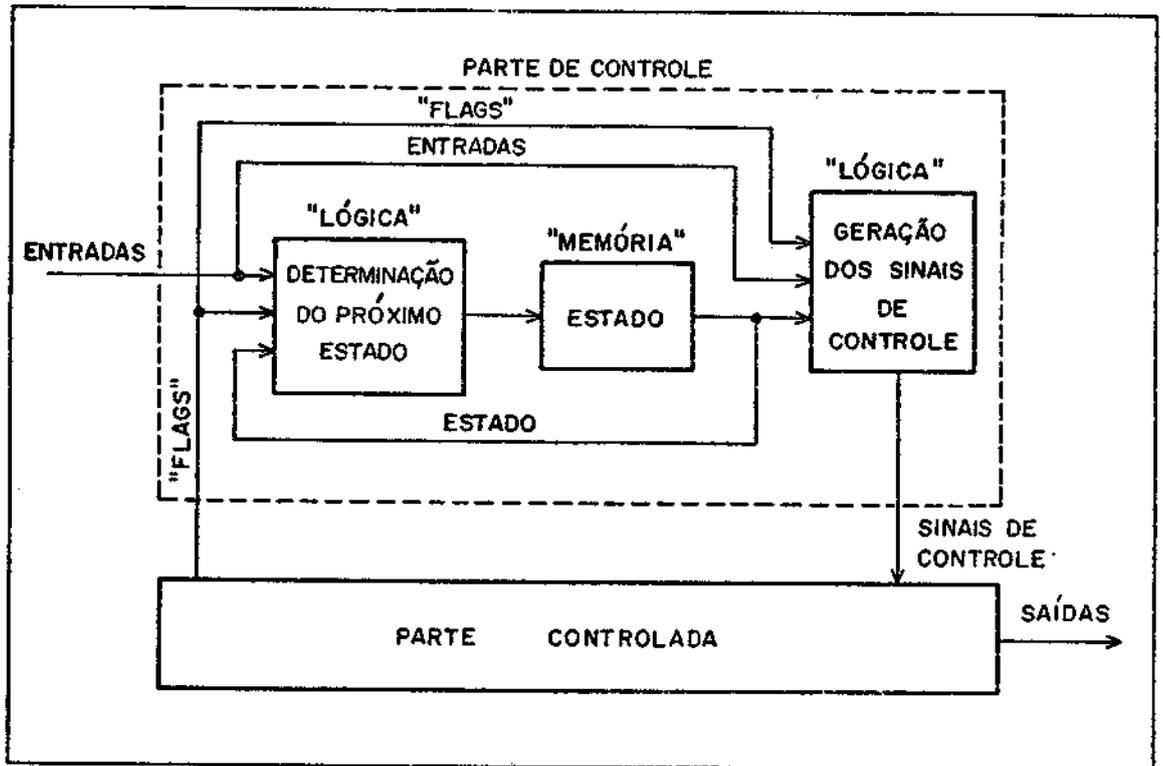


Fig. 2.1 - Detalhamento da parte de controle do sistema digital.

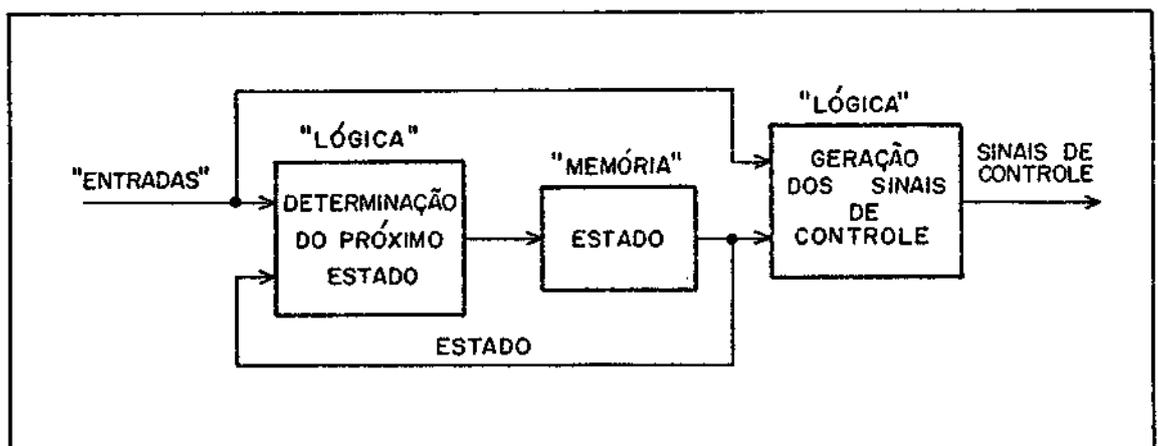


Fig. 2.2 - Representação geral da parte de controle de um sistema digital.

## 2.2 - CLASSIFICAÇÃO DA PARTE DE CONTROLE

Como a parte de controle (ou simplesmente "controle") de um sistema digital é uma máquina de estados, ela pode ser classificada segundo as classes dessa última, apresentadas por Clare (1973).

Convém ressaltar que nas Figuras 2.3 a 2.6 as linhas tracejadas não são parte das respectivas classes de controle.

### 2.2.1 - CONTROLE CLASSE 0

O controle classe 0 (Figura 2.3) gera os sinais de controle como função apenas das entradas, sendo uma lógica combinacional pura. Pode também ser considerada como um controle de um único estado.

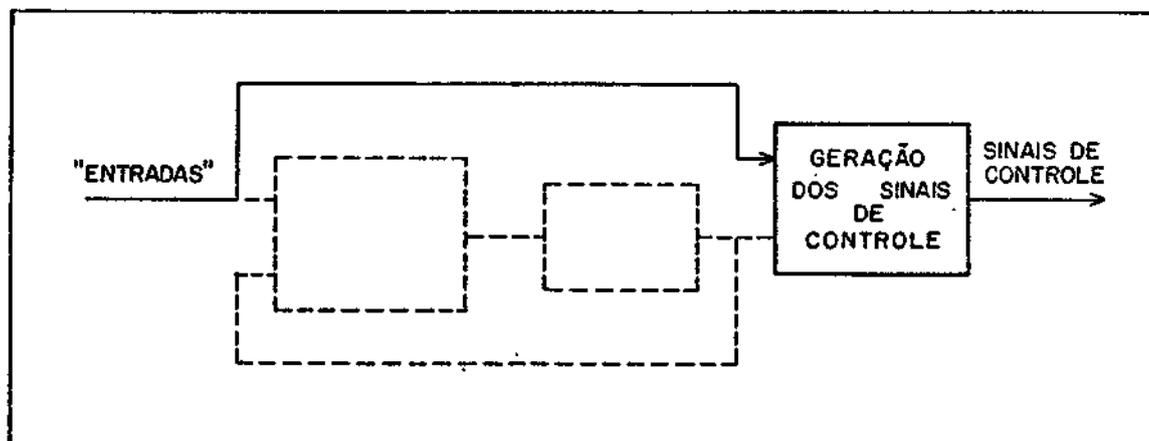


Fig. 2.3 - Controle classe 0.

### 2.2.2 - CONTROLE CLASSE 1

Nessa classe (Figura 2.4) os sinais de controle gerados dependem somente do estado da máquina, enquanto o próximo estado é função s̄o das entradas. Portanto, as saídas do controle são uma função com binacional das entradas atrasadas.

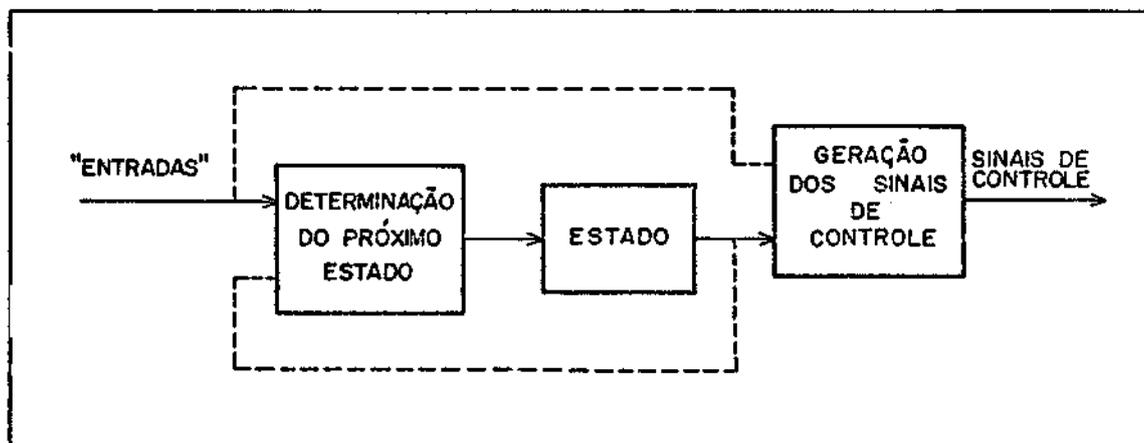


Fig. 2.4 - Controle classe 1.

### 2.2.3 - CONTROLE CLASSE 2

No controle classe 2 (Figura 2.5), tanto a geração dos sinais de controle como o próximo estado s̄o dependem do estado da máquina. Com isso, os sinais de controle são gerados em uma seqüência repetitiva, independente de entradas.

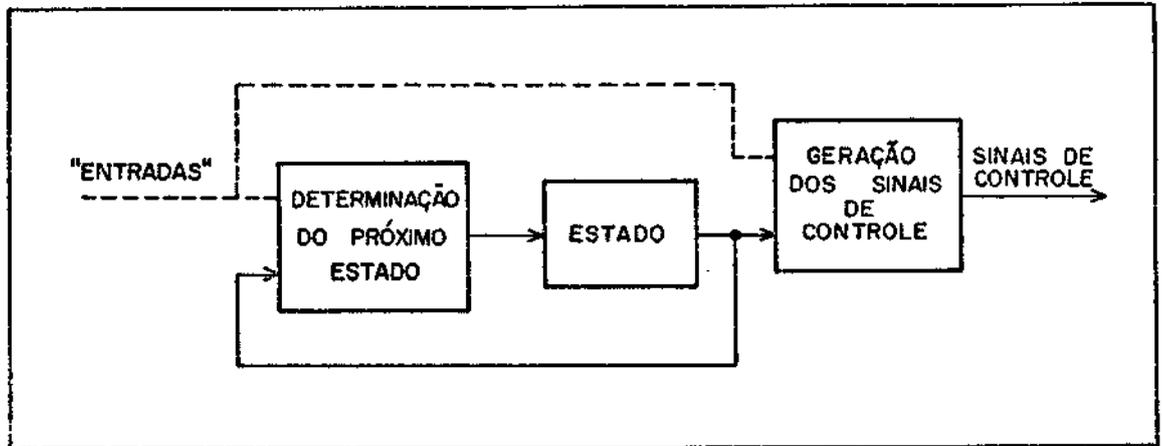


Fig. 2.5 - Controle classe 2.

#### 2.2.4 - CONTROLE CLASSE 3

Aqui (Figura 2.6), os sinais de controle gerados são função apenas do estado da máquina, enquanto o próximo estado depende também das entradas (autômato de Moore). Essa classe de controle é capaz de realizar qualquer algoritmo.

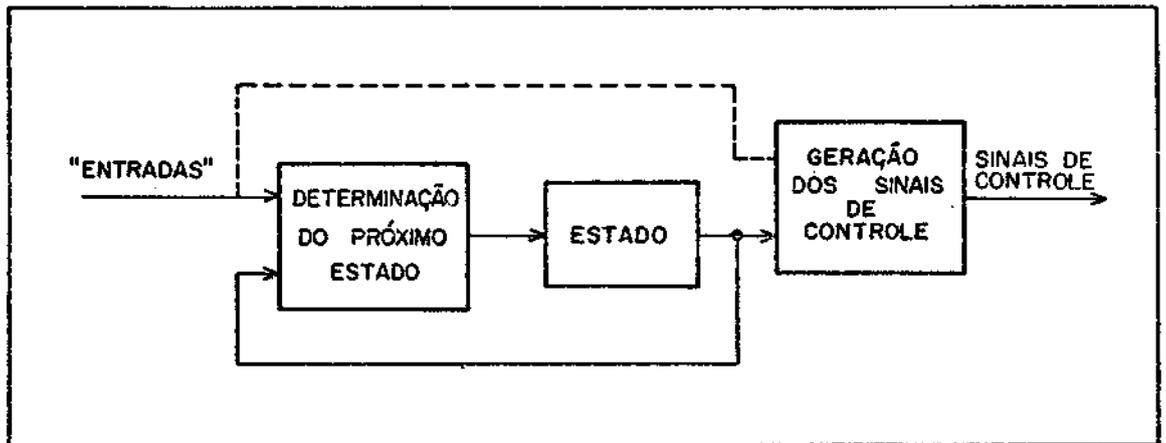


Fig. 2.6 - Controle classe 3.

#### 2.2.5 - CONTROLE CLASSE 4

Difere do controle classe 3 pelo fato de os sinais de controle gerados dependerem também das entradas, além do estado da máquina (autômato de Mealy). O controle classe 4 (Figura 2.7) engloba as outras classes de controle, podendo realizar também qualquer algoritmo, com algumas possíveis simplificações em relação à realização desse algoritmo com um controle classe 3.

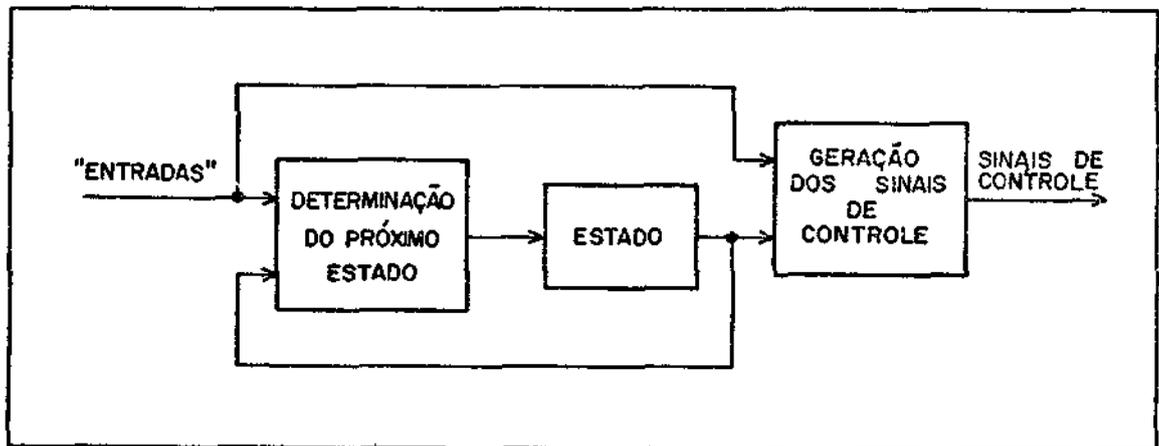


Fig. 2.7 - Controle classe 4.

## CAPÍTULO 3

### IMPLEMENTAÇÃO DA PARTE DE CONTROLE

#### 3.1 - METODOLOGIAS DE IMPLEMENTAÇÃO DA PARTE DE CONTROLE

A implementação do controle de um sistema digital pode ser feita com a aplicação de dois métodos distintos, como segue.

##### 3.1.1 - CONTROLE IMPLEMENTADO COM LÓGICA ALEATÓRIA

Nesse tipo de implementação, as lógicas para determinação do próximo estado e para a geração dos sinais de controle são realizadas com o método tradicional de projeto de circuitos sequenciais, com portas lógicas NAND, OR, AND, etc, como exposto, por exemplo, por Kohavi (1970).

##### 3.1.2 - CONTROLE MICROPROGRAMADO

No controle microprogramado, a implementação das lógicas para geração dos sinais de controle e para a determinação do próximo estado é feita com memórias PROMs (Programmable Read Only Memory e/ou FPLAs (Field Programmable Logic Array), que realizam, nesse tipo de aplicação, uma *Memória de Microcontrole* (MMC).

#### 3.2 - LÓGICA ALEATÓRIA X MICROPROGRAMAÇÃO

Apresentam-se a seguir algumas considerações sobre as duas metodologias de implementação da parte de controle, citadas anteriormente.

##### 3.2.1 - RAPIDEZ DE PROJETO

O projeto da parte de controle de um sistema digital microprogramado é, sem sombra de dúvidas, mais rápido do que o projeto com lógica aleatória. Uma vez decidida a estrutura mais conveniente do con

trole microprogramado (Capítulo 4), o trabalho se reduz à determinação do conteúdo de memórias PROMs e/ou FPLAs. No caso de grandes sistemas, esta tarefa pode ser realizada com o auxílio de ferramentas tais como o LMP - Linguagem de Microprogramação (Yamaguti, 1981).

Por outro lado, o projeto com lógica aleatória envolve a síntese de funções lógicas por meio de Mapas de Karnaugh, o que é difícil e, em funções acima de 8 variáveis, facilmente sujeito a erros.

### 3.2.2 - RAPIDEZ DE TESTE

Sem ferramentas de teste adequadas que possibilitem a emulação de memórias PROMs ou FPLAs, por exemplo o EMMAC - Emulador de Memórias de Microcontrole (Amaral, 1979), o teste de controles microprogramados se tornaria proibitivo, já que o erro no preenchimento de uma memória PROM ou FPLA fatalmente inutiliza o componente. Um paliativo seria a utilização de memórias EPROMs (Erasable Programmable Read Only Memory), caso isso não comprometesse a velocidade do sistema na fase de testes.

Mas, com ferramentas de teste adequadas, o teste de um controle microprogramado é tão mais rápido do que o teste de um controle com lógica aleatória, quanto mais complexo for o controle em questão.

### 3.2.3 - VELOCIDADE DO CIRCUITO

A grande desvantagem do controle microprogramado é a limitação por ele imposta à velocidade do sistema digital, devido ao tempo gasto com o acesso à memória de microcontrole. Tanto que, após ter seus fundamentos propostos por Wilkes em 1951, a microprogramação aguardou uma década e meia que o desenvolvimento tecnológico de memórias viabilizasse o seu emprego comercial em escala significativa. Isto ocorreu apenas em meados da década de 60 com os módulos IBM/360 (Agrawala and Rauscher, 1976).

Mas vale ressaltar que hoje estão disponíveis, no mercado, memórias PROMs cada vez mais densas e rápidas, como por exemplo a Am 27S184A/185A (Advanced Micro Devices, 1981) de 2 K x 8 bits com tempo de acesso máximo da ordem de 35 ns; e que técnicas de "pipeline" empregadas na estrutura do controle microprogramado (Seção 4.3) levam ao menoramento, senão ao desaparecimento, do efeito do tempo de acesso à memória do microcontrole na frequência máxima do relógio do sistema digital.

#### 3.2.4 - FLEXIBILIDADE DO PROJETO

Uma modificação em um controle com lógica aleatória, por erro de projeto ou aperfeiçoamento do sistema digital, tem grande chance de levar o controle a ser praticamente todo refeito, envolvendo a nova síntese de um bom número de variáveis lógicas e modificações em partes consideráveis do circuito.

No controle microprogramado, as modificações por erro ou aperfeiçoamento geralmente causam apenas alteração no conteúdo das memórias de microcontrole. Vale ressaltar que no caso de um sistema digital em teste, com o uso de emuladores de memórias, a MMC não chegou sequer a ser gravada.

#### 3.2.5 - CUSTO DO DESENVOLVIMENTO

Como o projeto em si e o teste com ferramentas adequadas são mais rápidos no controle microprogramado, obviamente o seu custo de desenvolvimento é menor do que o do controle com lógica aleatória.

#### 3.2.6 - CUSTO DO CIRCUITO

A implementação do controle microprogramado resulta na utilização sensível de menos componentes (o que implica economia de placas de circuito impresso, conectores, soquetes, fontes, etc.), porém de com

ponentes caros como PROMs. Mas, na redução do custo do circuito, a minimização do número de componentes é mais relevante que o alto custo deles, como é enfatizado por Blakeslee, (1979).

### 3.2.7 - CONFIABILIDADE

O controle microprogramado possui menos componentes do que a implementação de um controle análogo com lógica aleatória, apresentando, portanto, maior confiabilidade (Blakeslee, 1979).

## CAPÍTULO 4

### O CONTROLE MICROPROGRAMADO

Os controles classe 0, 1 e 2 são muito simples e não justificam uma discussão maior sobre eles. Portanto, neste capítulo serão analisadas apenas estruturas de controles microprogramados classes 3 e 4, direcionando-se a implementação da memória de microcontrole para o uso de PROMs.

#### 4.1 - ESTRUTURA DE CONTROLE CLASSE 4

Ao substituir as lógicas de geração dos sinais de controle e de determinação do próximo estado representado na Figura 2.7 por uma memória de microcontrole (MMC), obtém-se o controle classe 4 microprogramado (Figura 4.1).

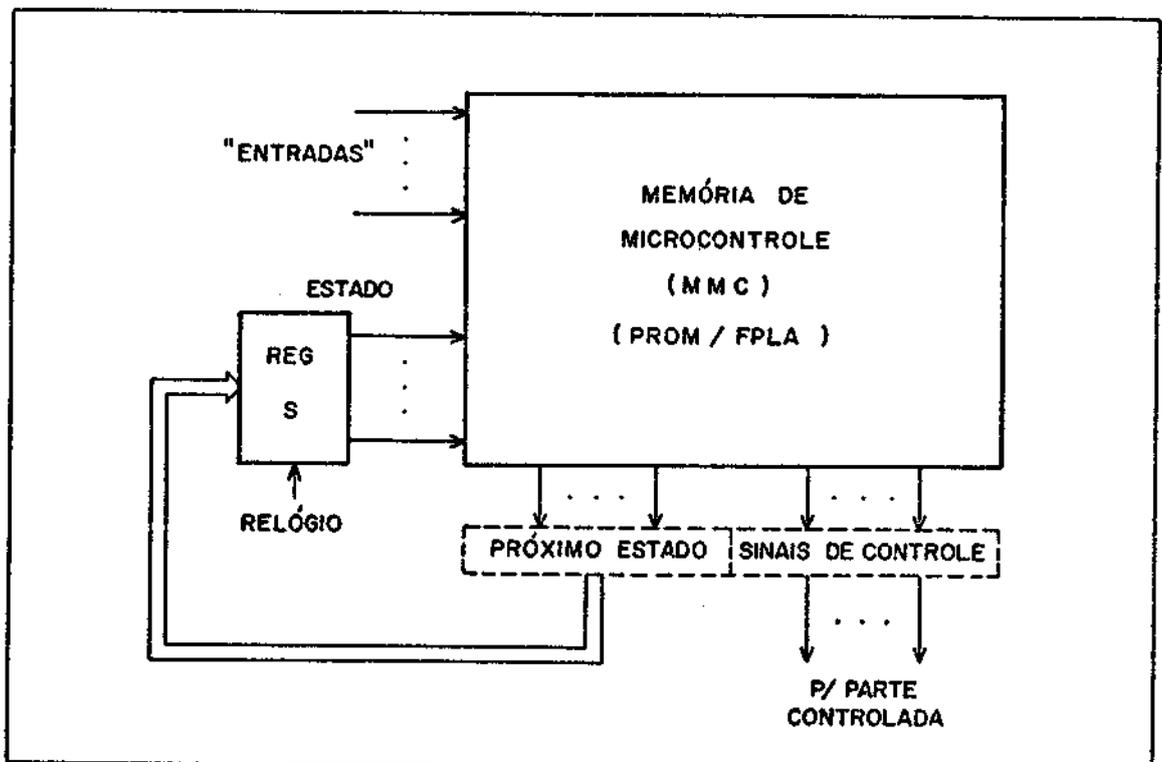


Fig. 4.1 - Estrutura de um controle classe 4 microprogramado.

Tanto os sinais de controle gerados como o próximo estado são função das entradas e do estado da máquina (armazenado no registro S). Essa estrutura é muito versátil: o estado seguinte é influenciado por todos os sinais de entrada, mas só é aplicável a casos onde os números de estados, sinais de controle gerados, entradas e termos mínimos requeridos na FPLA, se for o caso, são relativamente pequenos.

Existem disponíveis no mercado componentes que englobam o registro S, uma FPLA e um registro para a amostragem dos sinais de controle gerados. Um exemplo disto seria o 82S104/105 da Signetics Corp. (Figura 4.2), cuja utilização em controles classe 4 microprogramados é descrita por Cavlan e Durham (1979a e b).

O emprego de memórias PROMs para realizar a MMC em um controle classe 4 é impraticável. Por exemplo, um controle pequeno com:

- a) 8 sinais de entrada,
- b) 16 sinais de controle gerados, e
- c) 16 estados para a realização de um dado algoritmo,

implementado segundo a estrutura da Figura 4.1 com memórias PROM, necessitaria de uma MMC com  $2^{(8+4)} = 4.096$  palavras de 20 bits =  $4(\text{PROX. ESTADO}) + 16(\text{SINAIS DE CONTROLE})$ , o que totaliza 81.920 bits.



#### 4.2 - ESTRUTURAS DE CONTROLE CLASSE 3 - REDUÇÃO DA MMC

Conforme foi mostrado na secção anterior, a utilização de memórias PROMs para implementar a MMC de um controle classe 4 requer uma MMC de tamanho considerável. Faz-se necessário, então, o estudo de técnicas que reduzam a MMC de um controle microprogramado e viabilizem o uso de PRMs na sua implementação.

A diminuição no número de sinais que endereçam uma memória leva a uma redução, de forma geométrica, no tamanho desta memória (cada bit de endereço a menos reduz a memória pela metade). Por isso, uma solução eficaz para reduzir a memória de microcontrole, na implementação com PROMs de um controle microprogramado, é retirar os sinais de entrada ("entradas" na Figura 4.1) do endereçamento da MMC e permitir a influência de *uma entrada por vez* em uma lógica de determinação do próximo estado.

Ao retirar as "entradas" do endereçamento da MMC, elas não atuam mais diretamente na determinação dos sinais de controle enviados para a parte controlada, que ficam dependentes apenas do estado da máquina; passa-se a ter a estrutura de um controle classe 3.

Como a determinação do próximo estado é influenciada por uma entrada por vez, uma decisão que envolva mais de uma entrada tem que ser agora implementada com vários estados: um que realiza trabalho útil com a parte controlada, dito *estado útil*, seguido de outros que apenas testam entradas para determinar o *estado útil* seguinte. A velocidade de execução de um algoritmo (e por conseguinte do sistema digital) pode vir a ser prejudicada. O que antes podia ser executado em um único estado pode requerer agora vários estados para que seja determinado o *estado útil* seguinte. Geralmente, a decisão pelo próximo *estado útil* envolve duas, uma ou nenhuma entrada, além de se poder gerar as funções lógicas de duas ou mais entradas, que são mais frequentemente utilizadas no algoritmo a ser implementado no controle, e usar essas funções geradas diretamente como entrada da parte de controle.

#### 4.2.1 - CONTROLE COM A INFLUÊNCIA DE UMA SÕ ENTRADA NA DETERMINAÇÃO DO PRÓXIMO ESTADO

A Figura 4.3 apresenta a estrutura de um controle classe 3 (os sinais de controle gerados dependem sō do estado da máquina) que considera uma sō entrada em cada estado para a determinação do estado seguinte.

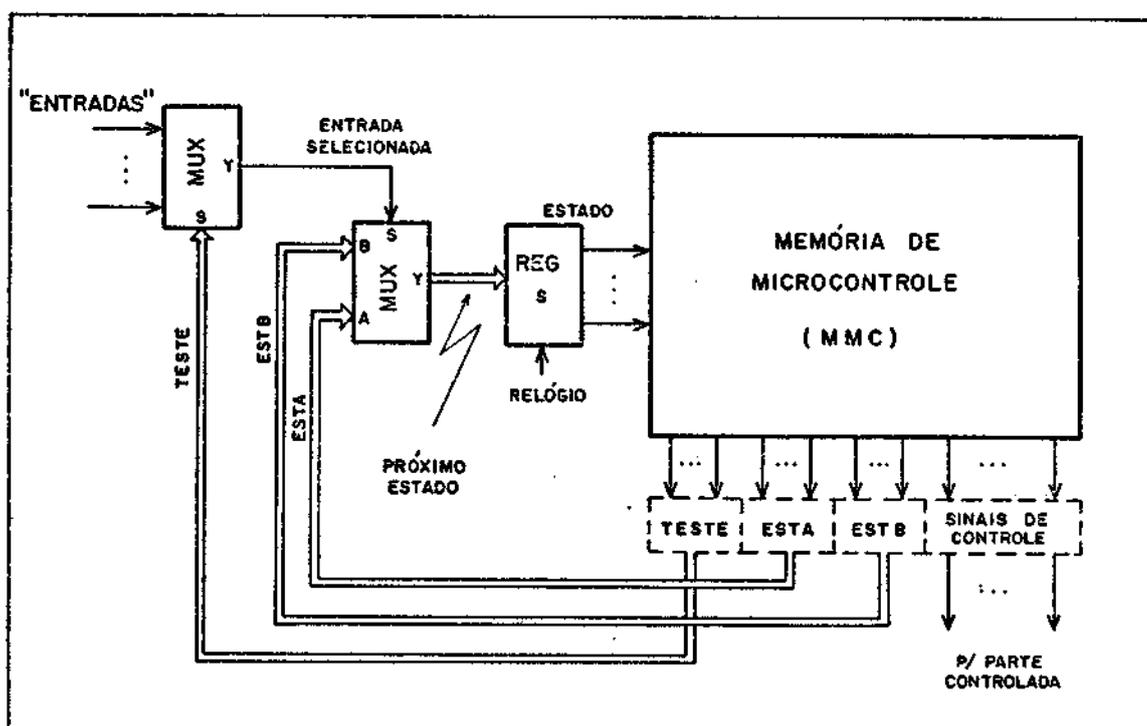


Fig. 4.3 - Estrutura de um controle microprogramado com influência de uma sō entrada na determinação do próximo estado.

A cada estado  $\bar{e}$  associada uma palavra da MMC que, junto com os sinais de controle enviados para a parte controlada, possui três outros campos envolvidos na determinação do estado seguinte: ESTA e ESTB são as duas opções para o próximo estado, e a escolha entre estes  $\bar{e}$  feita pela entrada selecionada pelo campo TESTE. Em relação ao controle discutido na Seccão 4.1, a palavra da MMC possui um número maior de bits, mas a profundidade da memória se reduz ao número de estados requeridos na implementação de um algoritmo.

Considere-se o mesmo exemplo da Secção 4.1 . Para implementar um algoritmo análogo, na estrutura da Figura 4.3, serão necessários, no máximo, um número de estados 8 vezes maior (caso extremo de o próximo estado útil depender das 8 entradas) e a MMC terá  $8 \times 16 = 128$  palavras com 33 bits =  $3(\underline{\text{TESTE}})+7(\underline{\text{ESTA}})+7(\underline{\text{ESTB}})+16(\underline{\text{SINAIS DE CONTROLE}})$ , dando um máximo de 4.224 bits (redução de 95%).

Caso o algoritmo análogo necessitasse de apenas o dobro de estados, a MMC seria de  $2 \times 16 = 32$  palavras com 29 bits =  $3(\underline{\text{TESTE}}) + 5(\underline{\text{ESTA}})+5(\underline{\text{ESTB}})+16(\underline{\text{SINAIS DE CONTROLE}})$ , o que totaliza apenas 928 bits.

Se o estado seguinte independe das entradas, basta preencher os campos ESTA e ESTB com a configuração de bits desse próximo estado e seleccionar, no MUX, uma entrada qualquer.

#### 4.2.2 - CONTROLE COM UMA DAS OPÇÕES DE PRÓXIMO ESTADO PREESTABELECIDAS

A estrutura de controle apresentada na secção anterior pode ser simplificada, objetivando uma maior redução no tamanho da MMC. Ao invés de se terem dois estados quaisquer como opções livres para a determinação do próximo estado, uma dessas opções é preestabelecida como o estado da máquina incrementado de um. A Figura 4.4 apresenta uma estrutura desse tipo.

A função da porta OU-EXCLUSIVO é inverter ou não a polaridade da entrada seleccionada pelo campo TESTE. Em um estado i pode-se desejar que o estado seguinte seja o conteúdo do campo EST, se a entrada seleccionada for "1"; caso contrário, i+1. Em outro estado j pode ser necessário o inverso: se a entrada for "0", o próximo estado será o conteúdo do campo EST; caso contrário, j+1. Para evitar a necessidade de gerar também as entradas invertidas, no MUX, faz-se o controle de polaridade com o sinal do campo PLD, que deverá ser "0" e "1", respectivamente, nos estados i e j mencionados acima.

Quando o próximo estado é determinado de forma incondicional às entradas, basta seleccionar a entrada "0" e ajustar convenientemente a polaridade.

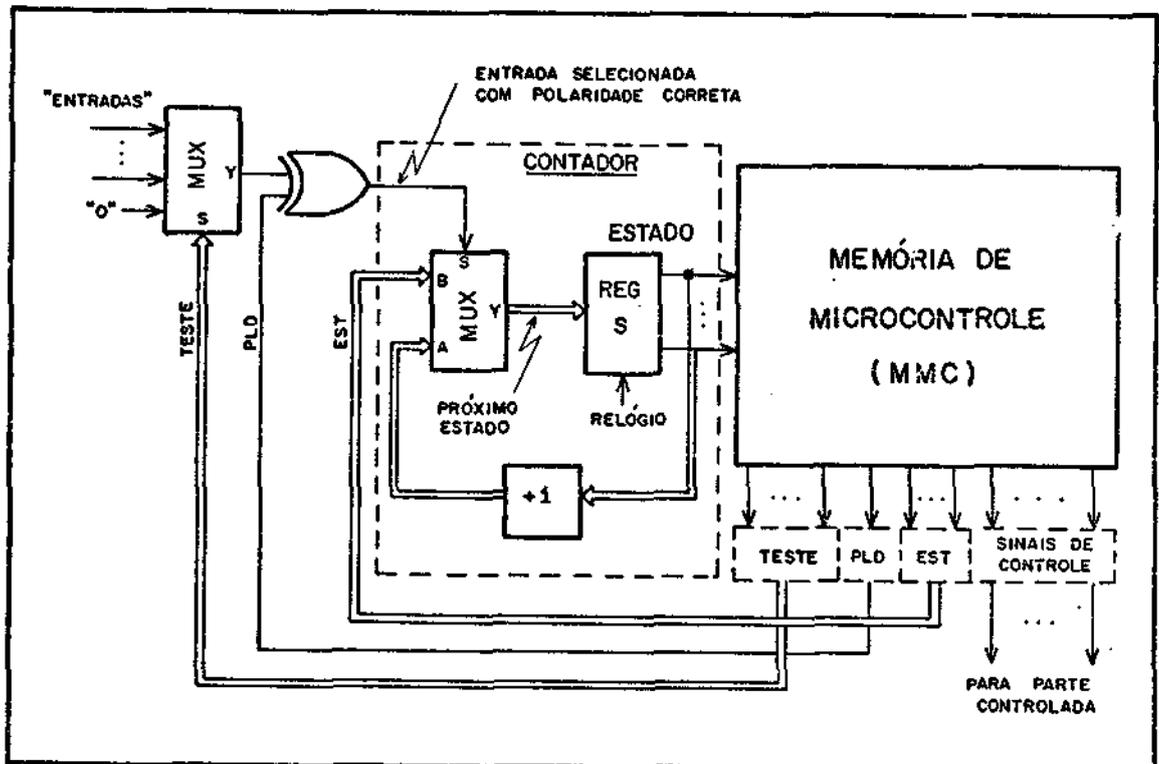


Fig. 4.4 - Estrutura de um controle microprogramado com uma das opções de próximo estado preestabelecida.

A designação de estados em um controle microprogramado, com uma das opções de próximo estado preestabelecida, deve ser feita de maneira a que estados adjacentes, sejam, dentro do possível, o estado anterior mais um. Isso evita perda desnecessária de tempo na determinação de um estado futuro, enquanto a parte controlada não executa trabalho útil.

Na implementação de um algoritmo análogo ao do exemplo da secção 4.1, com a estrutura da Figura 4.4, vão ser precisos no máximo um número de estados 8 vezes maior (por razões análogas às da Secção 4.2.1), o que resulta em uma MMC com  $8 \times 16 = 128$  palavras de 27 bits =  $3(\text{TESTE}) + 1(\text{PLD}) + 7(\text{EST}) + 16(\text{SINAIS DE CONTROLE})$ , resultando um total máximo de

3.456 bits e redução de 20% em relação à implementação do controle com a estrutura da Figura 4.3. Se a quantidade de estados apenas dobrasse, a MMC passaria a ser de 32 palavras com 25 bits, o que totaliza 800 bits (redução de 15% em relação à implementação da seção anterior).

Vale ressaltar que a parte do controle da Figura 4.4 que engloba o registro S, o MUX e o somador de um, pode ser realizada com um contador que tenha "load" síncrono, por exemplo o 74163. (Texas Instruments, 1976).

#### 4.2.3 - CONTROLE COM FORMATO VARIÁVEL

Não é raro que os algoritmos realizados com controles microprogramados apresentem um grande número de estados, onde a determinação do próximo estado útil independe das entradas. Se a designação de estados for executada com os cuidados mencionados na Seção 4.2.2, a implementação do controle com a estrutura da Figura 4.4 apresentará uma MMC com um número razoável de palavras, onde os campos TESTE, PLD e EST são pouco utilizados em sua total potencialidade de determinação do estado seguinte, já que estariam selecionando o conteúdo do registro S mais um.

A estrutura com formato variável (Figura 4.5) leva em conta a característica desses algoritmos. Se a palavra acessada na MMC apresentar o bit mais à esquerda (bit de formato) igual a "0", será implicitamente considerado que o estado seguinte é o estado corrente mais um. Se esse bit for "1", então o relógio para a parte controlada será bloqueado e a palavra restante da MMC será usada para determinar o próximo estado sob a influência de alguma entrada selecionada.

A implementação do algoritmo do exemplo da Seção 4.1 nessa estrutura requer, no máximo, um número de estados 9 vezes maior (um que comanda a parte controlada e oito para o caso extremo de o próximo estado útil depender das 8 entradas), o que acarreta uma MMC com  $9 \times 16 = 144$  palavras de 17 bits =  $1(\text{FORMATO}) + 16(\text{SINAIS DE CONTROLE})$  - Formato 1 maior do que o Formato 2, com 13 bits =  $1(\text{FORMATO}) + 3(\text{TESTE}) + 1(\text{PLD}) + 8$

(EST) - , o que totaliza 2.448 bits (redução de 30% em relação à estrutura da Figura 4.4). Se o número de estados fosse apenas duplicado a MMC necessitaria de 32 palavras de 17 bits, num total de 544 bits (redução de 30% em relação à implementação do controle com a estrutura da Figura 4.4).

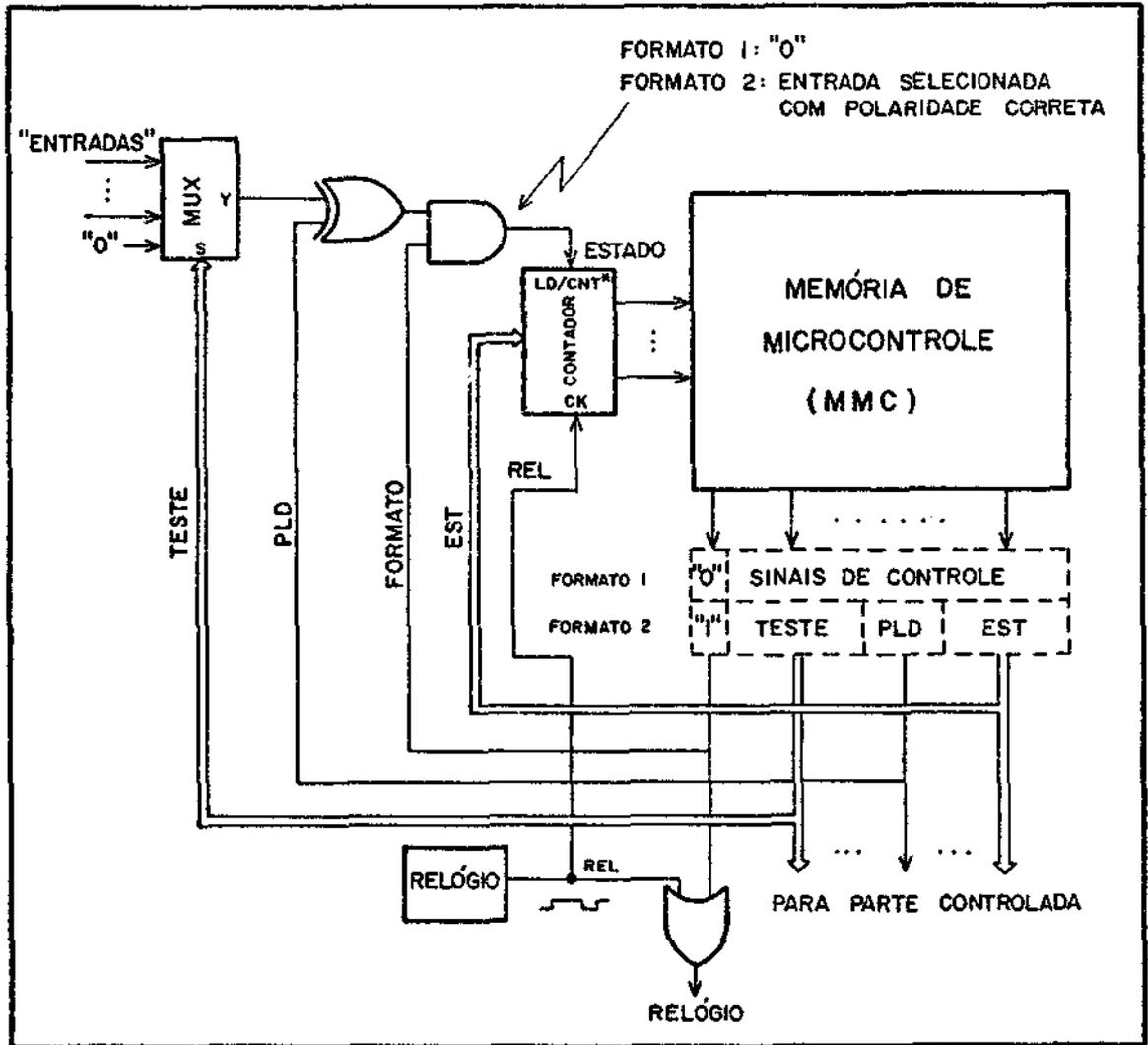


Fig. 4.5 - Estrutura de um controle microprogramado com formato variável.

#### 4.2.4 - CODIFICAÇÃO DE SINAIS DE CONTROLE

Uma abordagem que pode trazer bons resultados na redução da MMC é a codificação dos sinais de controle, que expressa a mesma quantidade de informação útil com um número menor de bits. Um método é a codificação dos sinais de controle de um bloco funcional da parte controlada (Figura 4.6).

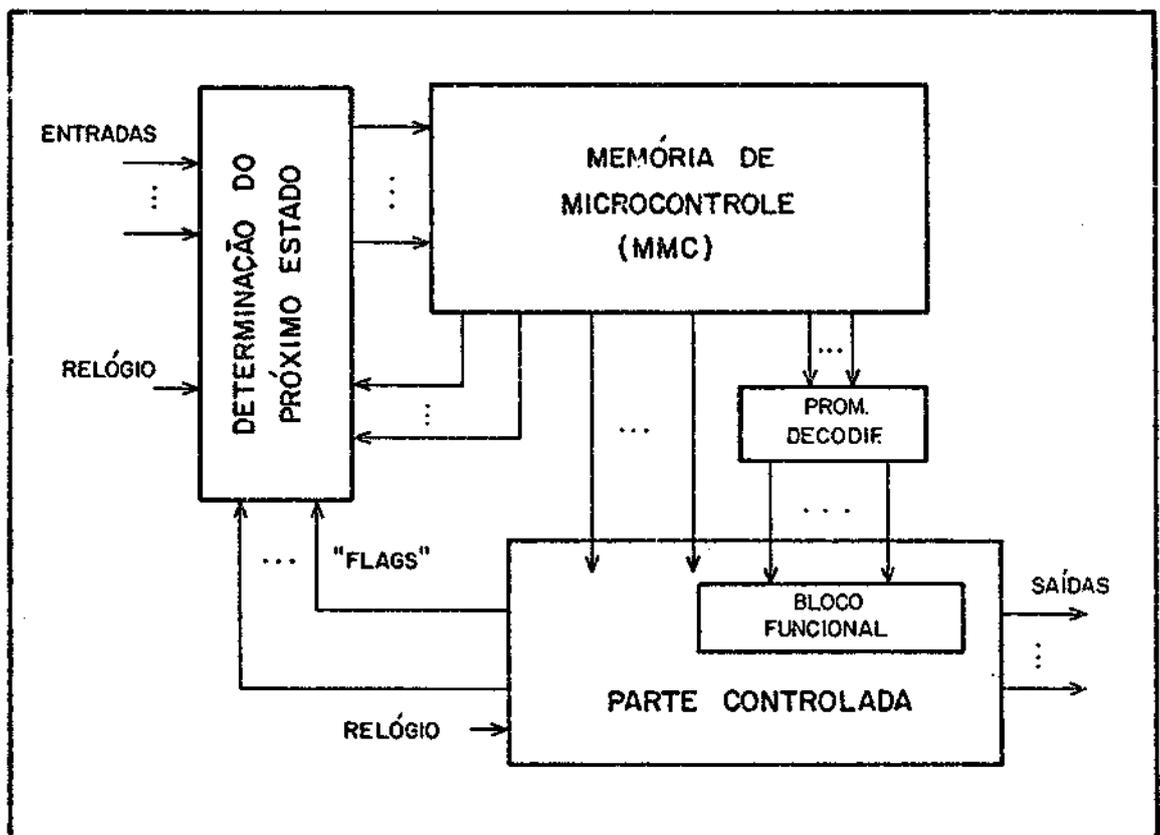


Fig. 4.6 - Estrutura com codificação dos sinais de controle de um bloco funcional da parte controlada.

Por exemplo, se em um bloco funcional são utilizadas apenas 14 combinações diferentes dos 8 sinais que o controlam, o uso de uma PROM decodificadora no exemplo da Seção 4.1, implementado com a estrutura

tura da Figura 4.5, necessitaria agora de uma MMC com as mesmas 144 palavras, porém de 13 bits = 1(FORMATO)+8(SINAIS DE CONTROLE)+4(SINAIS DE CONTROLE CODIFICADOS) - Formato 1 igual ao Formato 2, que permanece inalterado - , o que totaliza 1.984 bits = 144x13+122 bits da PROM decodificadora e reduz a MMC em 20%. No caso de o número de estados apenas do brar, a MMC passaria a ser de 528 bits (32 x 13 + 112).

Outro método é a codificação de sinais de controle que nunca são ativados juntos, como os responsáveis pela seleção e armazenamento do conteúdo de barramentos, ou a geração de pulsos que ocorrem um por vez na execução do algoritmo implementado (Figura 4.7).

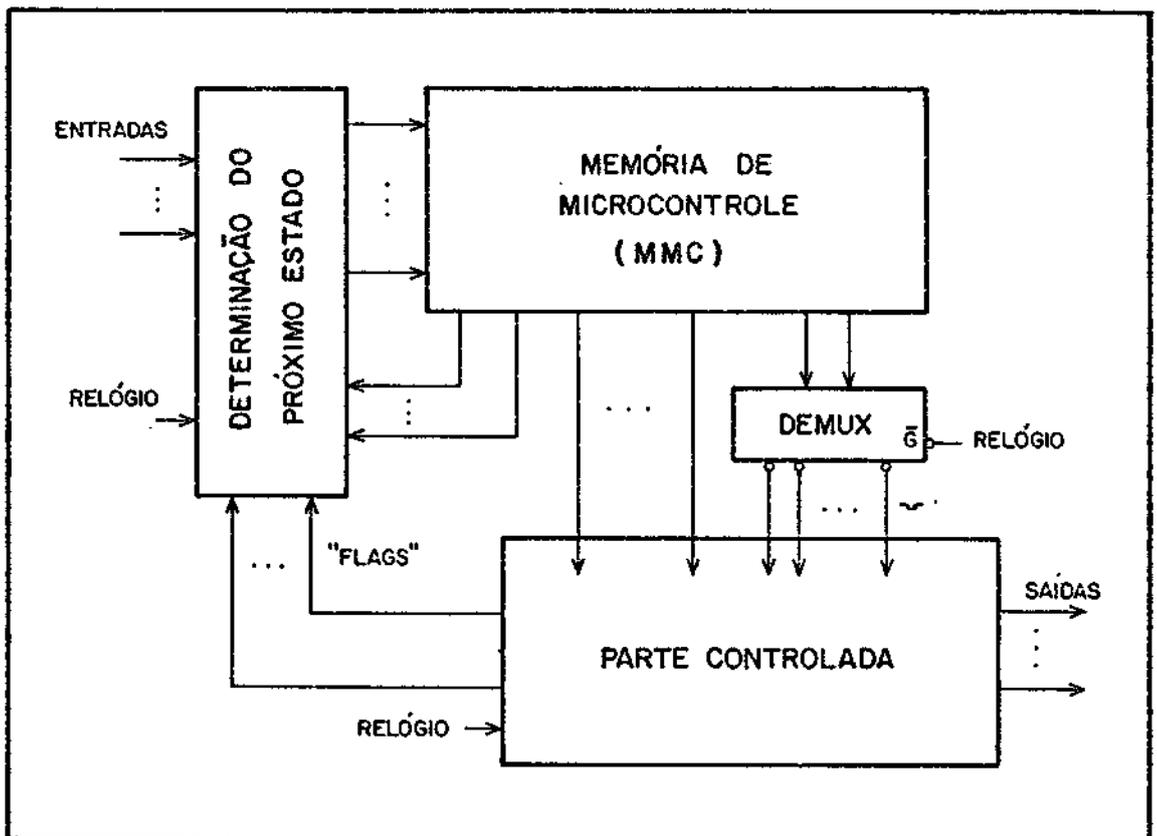


Fig. 4.7 - Estrutura com codificação na geração de pulsos de controle mutuamente exclusivos.

#### 4.2.5 - CONSIDERAÇÕES SOBRE A MINIMIZAÇÃO DA MMC

Na Tabela 4.1 estão condensados os resultados obtidos com o exemplo da Secção 4.1, implementando-se o controle com as cinco estruturas apresentadas até agora. A decisão de qual dessas estruturas (ou outra qualquer) é a melhor depende do que é exigido de cada controle específico e do algoritmo que ele deve implementar.

TABELA 4.1

TAMANHO DA MEMÓRIA DE MICROCONTROLE, NECESSÁRIA PARA A IMPLEMENTAÇÃO DO ALGORITMO DO EXEMPLO DA SECÇÃO 4.1

TIPO DE CONTROLE	Nº DE ESTADOS	TAMANHO DA MMC		
		PROFUNDIDADE (PALAVRAS)	LARGURA (BITS)	TOTAL (BITS)
Controle classe 4	16	4.096	20	81.920
Controle com influência de uma só entrada na de terminação do próximo estado (classe 3)	128 <sup>(1)</sup>	128	33	4.224
	32 <sup>(2)</sup>	32	29	928
Controle com uma das opções de próximo estado preestabelecida (classe 3).	128 <sup>(1)</sup>	128	27	3.456
	32 <sup>(2)</sup>	32	25	800
Controle com formato variável (classe 3)	144 <sup>(1)</sup>	144	17	2.448
	32 <sup>(2)</sup>	32	17	544
Controle com formato variável e codificação de sinais de controle (classe 3)	144 <sup>(1)</sup>	144	13	1.948 <sup>(3)</sup>
	32 <sup>(2)</sup>	32	13	528 <sup>(3)</sup>

(1) - Número máximo de estados necessários.

(2) - Supondo-se apenas a duplicação do número de estados.

(3) - Incluindo-se a PROM decodificadora.

Como PROMs são componentes caros, a tentativa em minimizar a memória de microcontrole deve ser uma preocupação constante, mas não deve comprometer a velocidade do sistema digital como um todo. Não deve ser esquecido que PROMs são disponíveis somente com 4 ou 8 bits, e 32, 256, 512, 1024 ou 4096 palavras, o que foi desconsiderado nos exemplos deste capítulo. Portanto, não convém sofisticar o controle, em detrimento da velocidade do sistema digital, para reduzir a largura da palavra da MMC, por exemplo, de 30 para 25 bits (no caso de ter disponíveis PROMs de 8 bits), ou o número de estados necessários na implementação do algoritmo de 200 para 140, porque ocorre diminuição da MMC e não do número de PROMs usadas.

#### 4.3 - ESTRUTURAS DE CONTROLE CLASSE 3 - AUMENTO DE VELOCIDADE

##### 4.3.1 - CONTROLE COM REGISTRO "PIPELINE"

Nos controles classe 3 anteriormente apresentados, o tempo de acesso à MMC é um componente importante no cálculo da frequência máxima do sistema digital.

Tomando-se como exemplo a estrutura de controle da Figura 4.4, reproduzida na Figura 4.8, o período mínimo do relógio será:

$$T_{\min} = \max \left\{ (t_S + \underline{t_M} + t_O), (t_S + \underline{t_M} + t_F + t_X + t_P + t_Z) \right\}$$

Observe que  $\underline{t_M}$  está presente nas duas expressões usadas para o cálculo de  $T_{\min}$ .

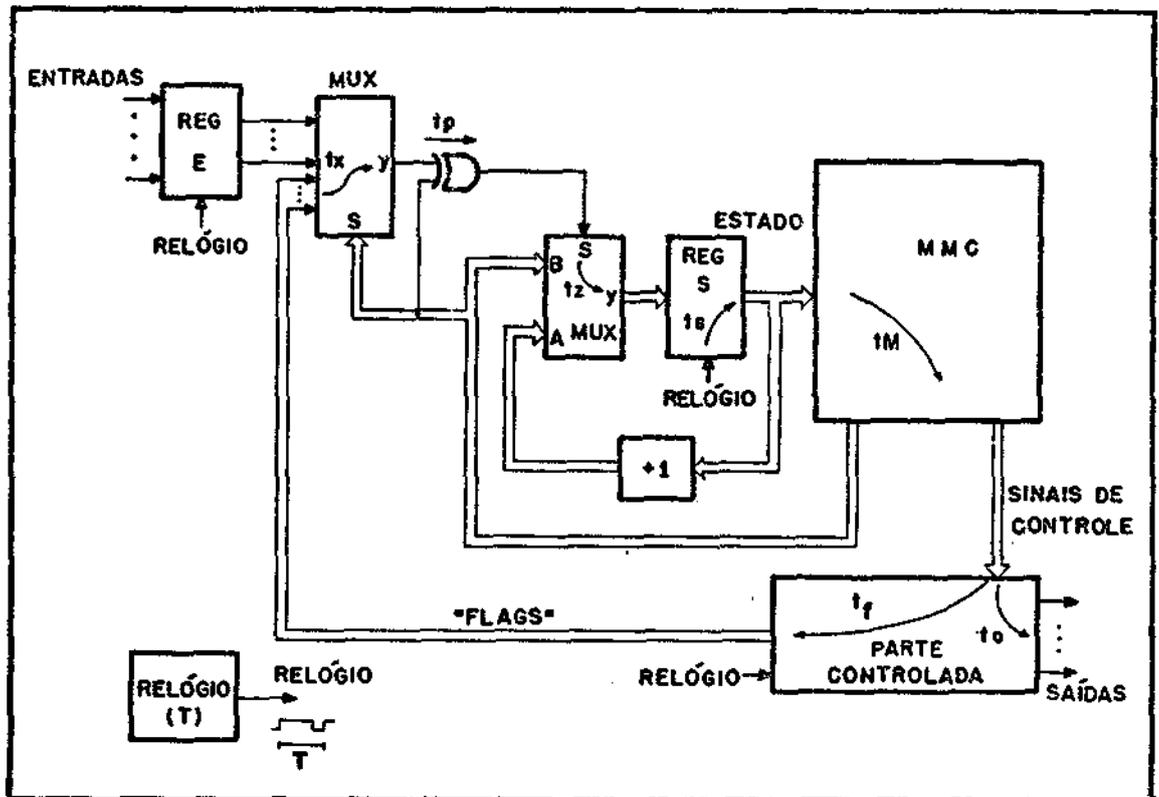


Fig. 4.8 - Atrasos envolvidos no cálculo da frequência máxima da estrutura da Figura 4.4 .

Uma solução para reduzir  $T_{min}$  é lançar mão da técnica de "pipeline" no acesso à MMC, o que é feito na estrutura da Figura 4.9 . Essa estrutura difere fundamentalmente daquela da Figura 4.8 pelo fato de o estado presente da máquina não ser o endereço da MMC, mas a palavra da MMC referente a esse endereço, armazenada em um registro "pipeline" colocado na saída da MMC.

As palavras da MMC atuam efetivamente no circuito controlado e são chamadas *palavras de microcontrole* ou *microinstrução*, doravante denotado por  $\mu I$ . A lógica de determinação do próximo estado que endereça a MMC faz o sequenciamento das microinstruções, armazenadas no registro "pipeline" para serem executadas, recebendo a denominação de *sequenciador de microinstruções*, ou simplesmente *sequenciador*.

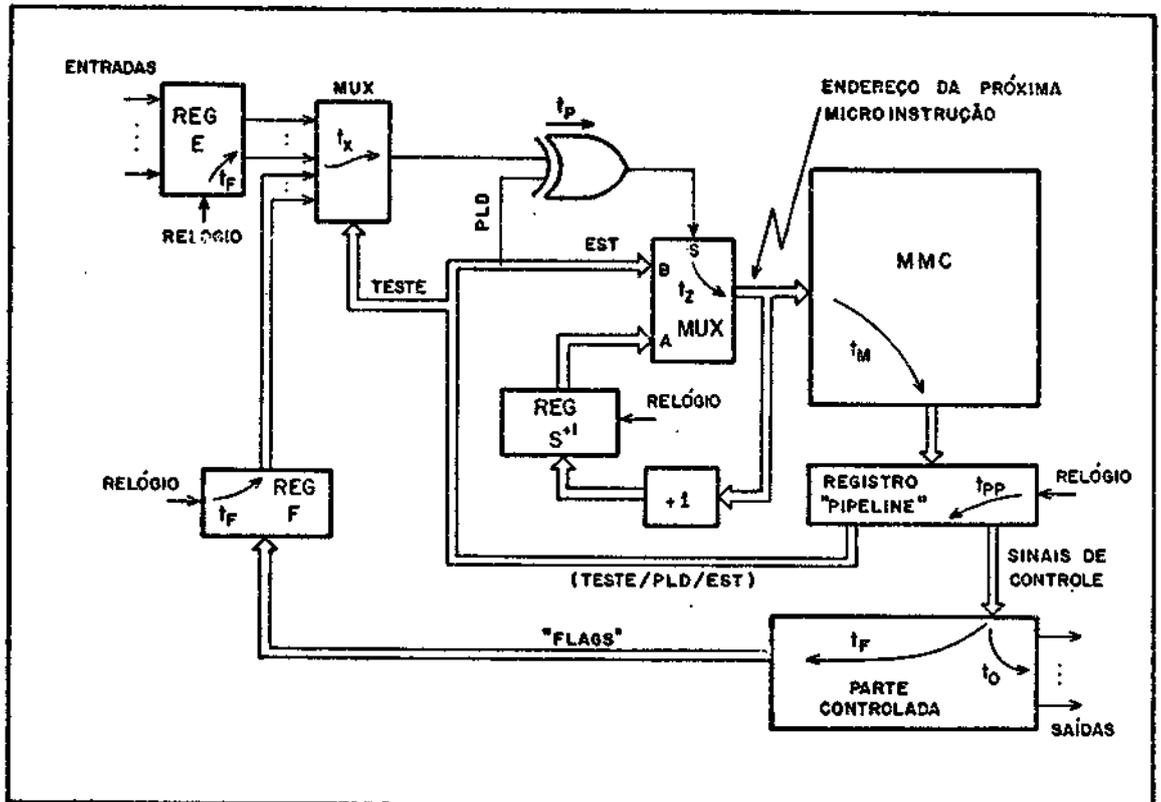


Fig. 4.9 - Estrutura de um controle microprogramado com registro "pipeline".

O endereço, da última microinstrução acessada, é incrementado de 1 (um) e guardado no registro  $S^{+1}$ , para ser usado como opção preestabelecida no acesso à MMC da microinstrução seguinte. O registro F contém os "flags" da parte controlada, gerados na execução da microinstrução anterior, já que eles devem estar estáveis durante a determinação do endereço da próxima  $\mu I$  pelo sequenciador.

Na estrutura da Figura 4.9,

$$T_{\min} = \max \left\{ (t_{pp} + t_0), (t_{pp} + t_F), (t_R + t_X + t_P + t_Z + t_M) \right\}.$$

Agora,  $t_M$  aparece apenas em uma das expressões usadas no cálculo de  $T_{min}$ , e sem se somar a  $t_0$  ou  $t_F$ , porque o acesso a  $\mu I$  seguinte é feito em paralelo com a execução da  $\mu I$  atual.

Dependendo de  $t_0$ ,  $t_F$  e  $t_M$ , é possível que  $T_{min}$  independa de  $t_M$  e, nesse caso, o controle microprogramado pode ter velocidade maior do que a de um controle equivalente, realizado com lógica aleatória, porque a velocidade passa a ser determinada pela parte controlada.

#### 4.3.2 - CONTROLE COM REGISTRO "PIPELINE" E PREVISÃO DA PRÓXIMA MICROINSTRUÇÃO

A estrutura com registro "pipeline", discutida na seção anterior, aumenta sensivelmente a velocidade do controle microprogramado e, por consequência, a do sistema digital. Mas, ela apresenta uma característica que pode não ser conveniente na implementação de certos algoritmos: a determinação da  $\mu I$  seguinte é função de "flags" oriundos da parte controlada na execução da  $\mu I$  anterior. Como solucionar esse problema se, para a busca da próxima  $\mu I$ , na MMC, em paralelo com a execução da  $\mu I$  atual, requerem-se "flags" estáveis?

Uma solução é a apresentada na Figura 4.10 .

Considere-se inicialmente a saída do Flip-Flop JK (INVERT) igual a "0" e o bit SEL do registro "pipeline" que seleciona o *endereço mais provável* da  $\mu I$  seguinte. O sinal  $\overline{OK}$ , gerado pelos campos TESTE e PLD e pelo sinal selecionado no MUX (que pode ser um "flag" originado pela parte controlada durante a execução da  $\mu I$  atual), deve estar estável antes da descida do relógio, indicando que a previsão de endereço feita por SEL, com INVERT = "0", foi confirmada ou não. Se nesse instante  $\overline{OK}$  = "0", então a previsão estava correta e a palavra da MMC referente a  $\mu I$  seguinte já se encontra (ou quase) na entrada do registro "pipeline". O sistema digital pode receber então o pulso negativo do relógio e o sinal INVERT se mantém igual a "0". Caso contrário ( $\overline{OK}$  = "1"), a previsão não se confirmou. Na descida do relógio, INVERT passa para "1" e bloqueia o

pulso positivo do relógio enviado para o sistema digital, a fim de dar tempo a um novo acesso à MMC, desta vez com o endereço da  $\mu I$  seguinte correto.

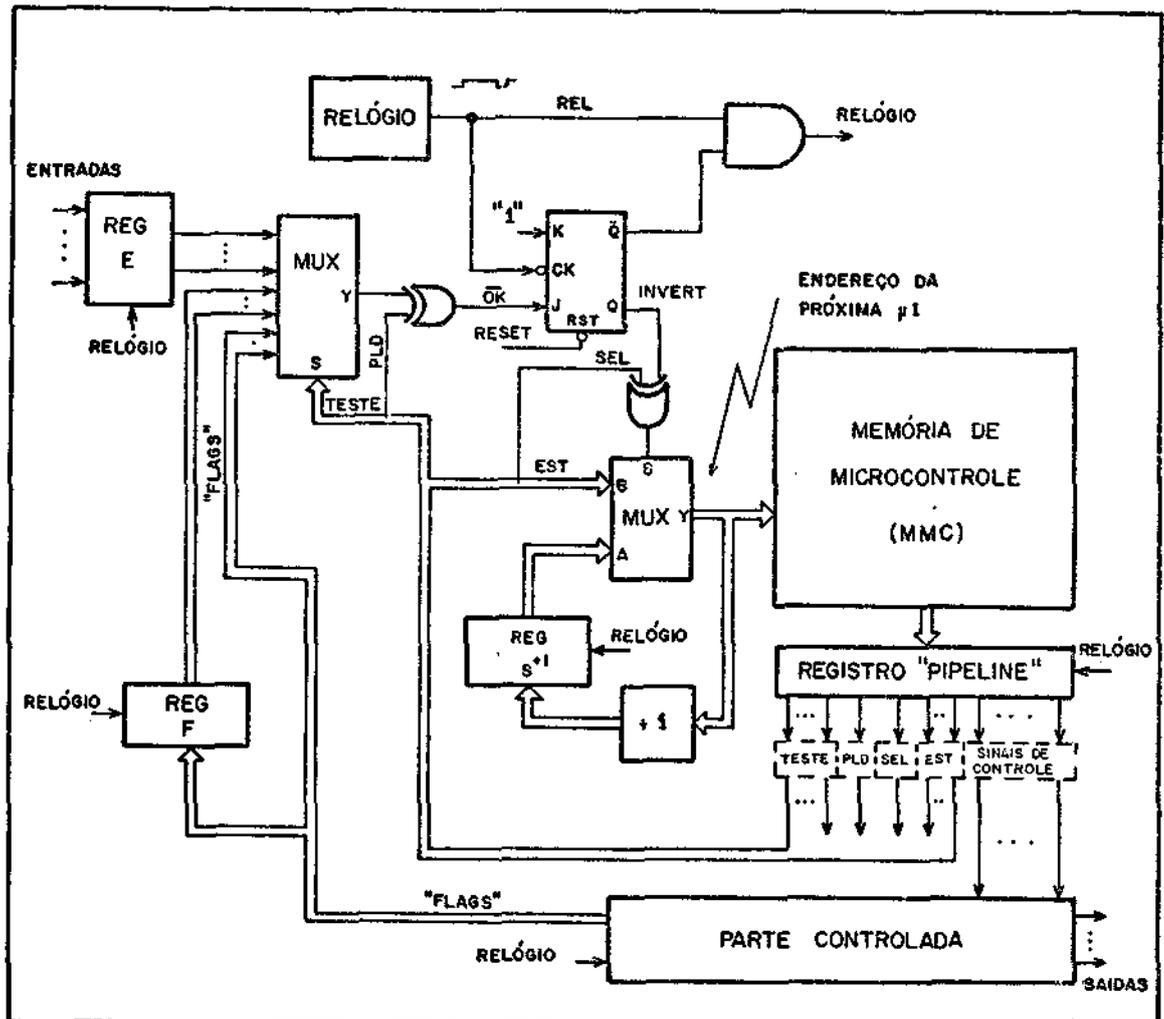


Fig. 4.10 - Estrutura de um controle microprogramado com registro "pipeline" e previsão do endereço da próxima  $\mu I$ .

Veja as formas de onda representadas na Figura 4.11 .

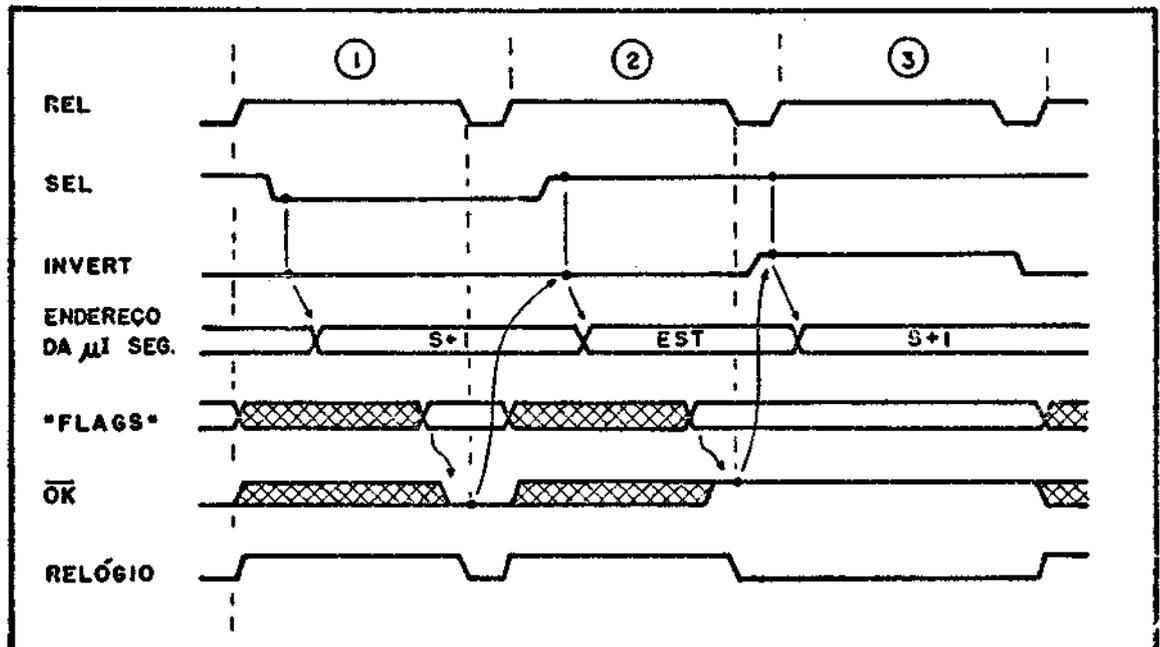


Fig. 4.11 - Formas de onda do Controle da Figura 4.10 .

- 1 - Endereço da  $\mu I$  seguinte foi confirmado.
- 2 - Endereço da  $\mu I$  seguinte não foi confirmado.
- 3 - Período do relógio é estendido.

O período mínimo do relógio para a estrutura apresentada na Figura 4.10 é maior que o da estrutura da Figura 4.9 ( $\overline{OK}$  tem que estar estável antes da *descida* do relógio). Mas é possível determinar a próxima  $\mu I$  sob a influência de um "flag" gerado pela atuação dos sinais de controle da  $\mu I$  atualmente sendo executada. Se a probabilidade de acertos na previsão de endereço for alta, um algoritmo pode ser executado de forma mais rápida do que com a estrutura da Figura 4.9 .

Um exemplo da eficiência da estrutura da Figura 4.10 é a implementação de "loops", já que o retorno para reexecutar o trecho do "loop" é uma opção mais provável do que sua saída.

#### 4.4 - ESTRUTURAS DE CONTROLE CLASSE 3 - AUMENTO DA EFICIÊNCIA NA DETERMINAÇÃO DA PRÓXIMA MICROINSTRUÇÃO

##### 4.4.1 - CONTROLE COM REGISTRO "PIPELINE" E REGISTRO AUXILIAR

Uma vez que o algoritmo a ser implementado no controle apresenta vários trechos idênticos (Figura 4.12.a), uma redução considerável da MMC é obtida se apenas *uma cópia* desses trechos for armazenada na MMC. Nesse caso, a estrutura do controle deve prover meios de o algoritmo sair de um fluxo principal, entrar no início do trecho em questão, e *retornar* ao ponto de onde havia saído no fluxo principal (Figura 4.12.b); o que é um comportamento idêntico ao tratamento de sub-rotinas em linguagens de programação.

A estrutura de controle da Figura 4.13 incorpora a característica mencionada nesta seção. Nesta estrutura, para determinação da  $\mu I$  seguinte, são necessários dois sinais:  $S_0$  e  $S_1$ , uma vez que é possível a escolha entre três opções: o campo EST do registro "pipeline", e os conteúdos do registro  $S^{+1}$  e do registro R. O sinal  $\overline{ENR}$  habilita o registro R a armazenar uma cópia do registro  $S^{+1}$  para ser usada posteriormente. Uma vez que os sinais  $S_0$ ,  $S_1$  e  $\overline{ENR}$  controlam o sequenciador, eles definem a *microoperação* desta unidade do controle, que é selecionada entre os campos IA e IB pela entrada que, por sua vez, é definida pelos campos TESTE e PLD.

O registro R, na forma como foi colocado, guarda o conteúdo do registro  $S^{+1}$ , em qualquer  $\mu I$  onde  $\overline{ENR}$  esteja ativo, independente de qual seja o endereço da próxima  $\mu I$  a ser executada. Isso é imprescindível para o retorno do algoritmo aos pontos do seu fluxo principal, de onde eventualmente tenha saído.

Por exemplo, considere-se a implementação do trecho de algoritmo da Figura 2.12.b com a estrutura da Figura 4.13. Na  $\mu I P_i$ , se  $S_1 S_0 = "10"$  com o campo EST =  $T_1$ , e  $\overline{ENR} = "0"$ , então a próxima  $\mu I$  a ser executada será  $T_1$  e no registro R estará guardado o endereço da  $\mu I P_j$ .

Na  $\mu I$   $T_n$  basta forçar  $S_1 S_0 = "01"$  que a  $\mu I$  seguinte ser $\bar{a}$   $P_j$ , desde que n $\bar{a}$  se tenha destru $\bar{i}$ do o cont $\bar{e}$ u $\bar{d}$ o original do registro R. Procedimento an $\bar{a}$ logo deve ser feito nas microinstru $\bar{c}$ o $\bar{e}$ s  $P_k$  e  $P_l$ , ou em quaisquer ou tros pontos do fluxo principal, onde se deseje utilizar o trecho  $T_1$  a  $T_n$ .

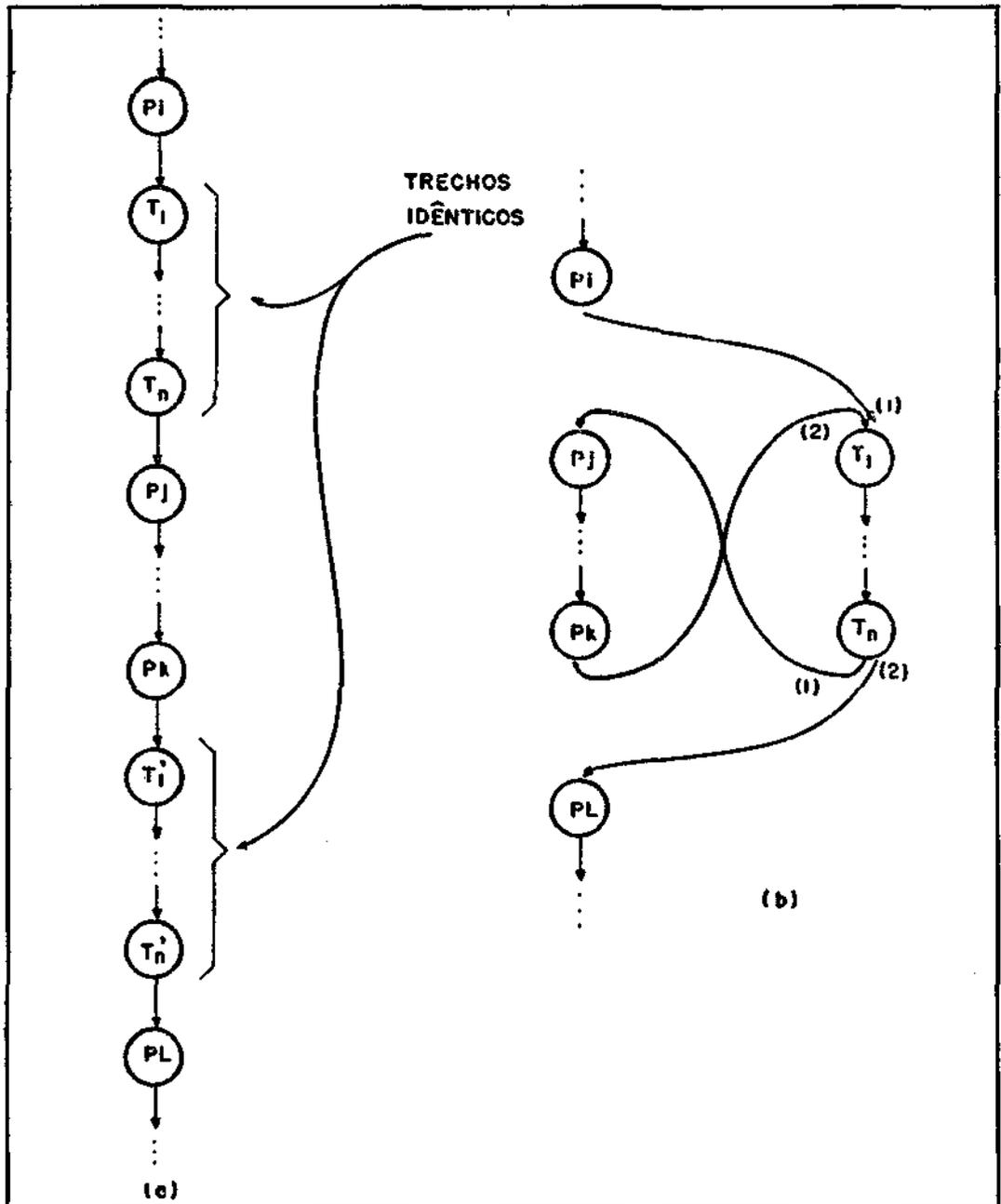


Fig. 4.12 - Algoritmo com trechos id $\bar{e}$ nticos repetidos(a) e com o trecho repetido implementado uma s $\bar{o}$  vez(b)



Uma outra aplicação para o registro R é a implementação de "loops", já que a volta ao ponto inicial do "loop" pode ser feita com o auxílio do registro R e não com o campo EST.

Em estruturas mais sofisticadas, o registro R é substituído por uma *pilha*, e no lugar do sinal  $\overline{ENR}$  passam a existir os controles de "push" e "pop" desta pilha.

#### 4.4.2 - CONTROLE COM PROM DE MAPEAMENTO

Para reduzir ainda mais a MMC requerida na implementação dos algoritmos e aumentar sua velocidade de execução, convém melhorar a eficiência da determinação do endereço das  $\mu I$ . Nas estruturas anteriores, se a próxima operação a ser executada na parte controlada dependesse de mais de 2 entradas e/ou "flags", perder-se-ia tempo e gastar-se-ia memória de microcontrole com  $\mu I$  que não executasse nada de útil na parte controlada, uma vez que elas apenas testavam entradas e/ou "flags".

Uma maneira de decidir rapidamente qual o endereço da próxima palavra de microcontrole útil para a parte controlada, em função de várias entradas, é a apresentada na Figura 4.14. Nessa estrutura é possível que combinações de algumas entradas especiais mapeiem diretamente os endereços, que podem ser os iniciais de trechos do algoritmo implementado no controle para tratar cada combinação dessas entradas.

Esta técnica é muito utilizada em duas situações. A primeira é quando o algoritmo implementado no controle está constantemente preocupado com a ocorrência de situações especiais, que devem ser prontamente reconhecidas durante a realização de outras tarefas. A segunda, mais comum, é a realização de interpretadores que implementam as instruções em linguagem de máquina de um processador. Após trazer uma instrução da memória principal para o processador ("fetch"), ao invés de o algoritmo fazer a decodificação da instrução, testando o OPCODE bit a bit, esse OPCODE é colocado no registro E1 (Figura 4.14), que pode ser o próximo registro de instrução do processador. Na saída da PROM de mapeamen

to, obtêm-se o endereço inicial correspondente à execução da instrução em questão.

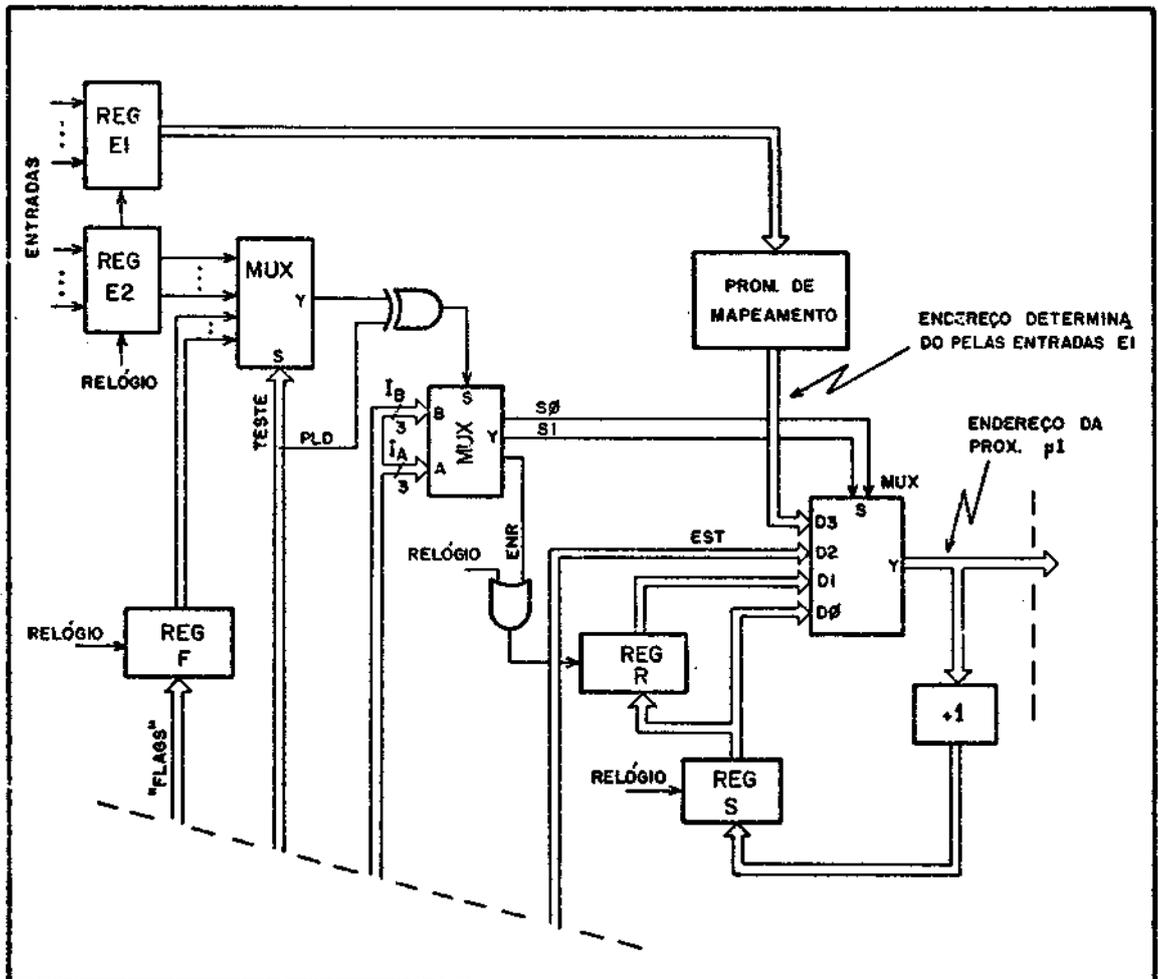


Fig. 4.14 - Estrutura de um controle microprogramado com PROM de mapeamento.



## CAPÍTULO 5

### CONCLUSÕES FINAIS

A introdução, no mercado, de memórias PROMs cada vez mais densas e rápidas, inclusive as já orientadas para o uso como memória de microcontrole que incluem o registro "pipeline", por exemplo a Am27S05 (Advanced Micro Devices, 1981), e de componentes "bit-slice", torna a implementação de controles microprogramados cada vez mais viável e atraente.

A rapidez de projeto e testes, a geração de um circuito organizado e de tamanho reduzido e, principalmente, a flexibilidade são características que tornam a microprogramação fundamental para implementar grandes sistemas digitais, como computadores.

Se esta metodologia de implementação gera circuitos com características tão desejáveis, já é tempo de ela ser efetivamente considerada também no projeto de controles de sistemas digitais mais simples. Utilizando as idéias básicas apresentadas neste trabalho, pode-se gerar estruturas de controles simples, mas poderosas, e adequadas à solução de cada problema em particular. Estruturas mais complexas resultam apenas da sofisticação daquelas que aqui foram discutidas, em função da aplicação específica a que se destinam.



## REFERÊNCIAS BIBLIOGRÁFICAS

- ADVANCED MICRO DEVICES. *Condensed catalog*. Sunnyvale, Cal.; c1981.
- AGRAWALA, A.K.; RAUSCHER, T.G. *Foundation of microprogramming: architecture, software and applications*. New York, Academic Press, 1976.
- AMARAL, P.F.S. *Emulador de memórias de microcontrole auxiliado por computador*. Tese de Mestrado em Eletrônica e Telecomunicações. São José dos Campos, INPE, 1979. 161 p. (INPE-1489-TDL/009).
- BLAKESLEE, T.R. *Digital design with standard MSI and LSI*. 2. ed. New York, John Wiley, 1979.
- CAVLAN, N.; DURHAM, S.J. Field-programmable arrays: powerful alternative to random logic. *Electronics*, 52(14):109-114, July 1979 a.
- . Sequencers and arrays transform truth tables into working systems. *Electronics*, 52(15):132-139, July 1979 b.
- CLARE, R.C. *Design logic systems using state machines*. New York, McGraw-Hill, 1973.
- KOHAVI, Z. *Switching and finite automata theory*. New Delhi, India, Tata McGraw-Hill, 1970.
- RAUSCHER, T.G.; ADAMS, P.M. Microprogramming: a tutorial and survey of recent developments. *IEEE Transactions on Computers*, C-29(1): 1-30, Jan. 1980.
- TEXAS INSTRUMENTS. *The TTL data book for design engineers*. 2. ed. Dallas, Tex., c1976.
- WILKES, M.V. The best way to design an automatic calculating machine. *In: Report of the Manchester University Computer Inaugural Conference*, Manchester, England, July 1951, p 16-18.
- YAMAGUTI, W. *LMP, uma linguagem de microprogramação*. Tese de mestrado em Eletrônica e Telecomunicações. São José dos Campos, INPE, 1981, 105p. (INPE-2031-TDL/049).