Exemplo 1 de Projeto de Circuito Síncrono: Multiplicador Binário

Resumo elaborado por Edith Ranzini, a apartir do livro KIME, C R;
MANO, M.M. "Logic and Computer Design Fundamentals".
New Jersey Prentice Hall – 2000

1.Algoritmo Tradicional

O algoritmo tradicional de multiplicação binária de dois números binários sem sinal é feita com sucessivos deslocamentos do multiplicando à esquerda (que constituem as parcelas do cálculo dos produtos parciais) e uma soma. Vamos executar este algoritmo com os número binários `1101` e `1011`, como ilustrado abaixo.

```
 13                      1101  multiplicando
 11                      1011  multiplicador
                         1101
                        1101     produtos
                       0000      parciais
                      1101
143                  10001111  produto
```

A execução do algoritmo leva em consideração um bit do multiplicador de cada vez, com o bit menos significativo em primeiro lugar. Se o bit do multiplicador for 1, o multiplicando é copiado para ser somado posteriormente. Em caso contrário, o bit do multiplicador for 0, um valor nulo é copiado em seu lugar. Os números copiados em linhas sucessivas são deslocados à esquerda de uma posição em relação à linha anterior. Finalmente, os números são somados gerando o produto final.

Um circuito digital que implementa este algoritmo deve executar esta soma em etapas. Assim, quando um novo número for copiado, duas parcelas devem ser somadas gerando uma soma parcial. Ilustramos abaixo esta modificação no mesmo exemplo acima.

```
 13                      1101  multiplicando
 11                      1011  multiplicador
                         1101
                        1101      soma dos
                       100111     produtos
                       0000       parciais
                       100111     2 a 2
                      1101
143                  10001111  produto
```

Este algoritmo tradicional, inspirado no processo manual que utilizamos para multiplicar, apresenta, como grande desvantagem, a necessidade de se utilizar um somador de *2n* bits, quando os operandos possuirem *n* bits.

2. Algoritmo Melhorado

O somador de *2n* bits pode ser substituído por um de *n* bits com um algoritmo levemente diferente. O algoritmo melhorado, que apresentamos aqui, necessita de um somador de *n* bits e de operações de deslocamento para a direita do produto parcial.

O algoritmo segue os seguintes passos:

1. Inicialmente o produto parcial é ajustado para `0` (zero);
2. Um bit do multiplicador é processado de cada vez, começando pelo bit menos significativo;
   2.1. Se o bit sendo processado for `1` (um), o multiplicando é somado ao produto parcial e depois é realizado um deslocamento pela direita do produto parcial;
   2.2. Se o bit sendo processado for `0` (zero), o produto parcial é apenas deslocado para a direita;
3. O bit de vai-um ("*carry*") do somador é armazenado em um flip-flop que deve estar conectado ao registrador deslocador contendo o produto parcial;
4. A soma deve ser realizada apenas nos *n* bits mais significativos de produto parcial.

Um exemplo de aplicação deste algoritmo com *n*=4, é ilustrado abaixo:

```
 13            1101        multiplicando
 11            1011        multiplicador
               0000        valor inicial do produto parcial
               1101        soma multiplicando, bit do multiplicador é 1
               1101
               0110 1      desloca para a direita
               1101        soma multiplicando, bit do multiplicador é 1
             1 0011 1
               1001 11     desloca para a direita
               0100 111    só desloca para a direita, bit do multiplicador é 0
               1101        soma multiplicando, bit do multiplicador é 1
             1 0001 111
               1000 1111   desloca para a direita
              \_______/
143            10001111  produto
```

O diagrama de blocos de um circuito que executa tal algoritmo será apresentado na figura 3.

3. O projeto

3.1. Especificação do Multiplicador Binário

O Multiplicador Binário (**MB**) é responsável pela realização de uma multiplicação de dois números binários sem sinal de 4 bits, introduzidos separadamente no circuito através de uma única via de dados (chaves `CH0` a `CH3`). A operação é iniciada com o acionamento do sinal `INICIAR` (`botão B1`), e o resultado da operação com 8 bits (`OUT`) deve ser conectado a dois displays de saída. Os operandos da multiplicação são especificados para o MB pelos sinais `ENTRA_MULTIPLICANDO` (`CH6`) e `ENTRA_MULTIPLICADOR` (`CH7`).

Os sinais de entrada e saída do Multiplicador Binário são os seguintes:

- `IN` - via de dados de entrada, com quatro bits;
- `OUT` - via de dados de saída, com oito bits;
- `INICIAR` - sinal de controle utilizado iniciar a multiplicação;
- `ENTRA_MULTIPLICANDO` - sinal de controle utilizado para especificar o multiplicando da operação;
- `ENTRA_MULTIPLICADOR` - sinal de controle utilizado para especificar o multiplicador da operação.

A figura 1 abaixo mostra o MB com os sinais descritos anteriormente:

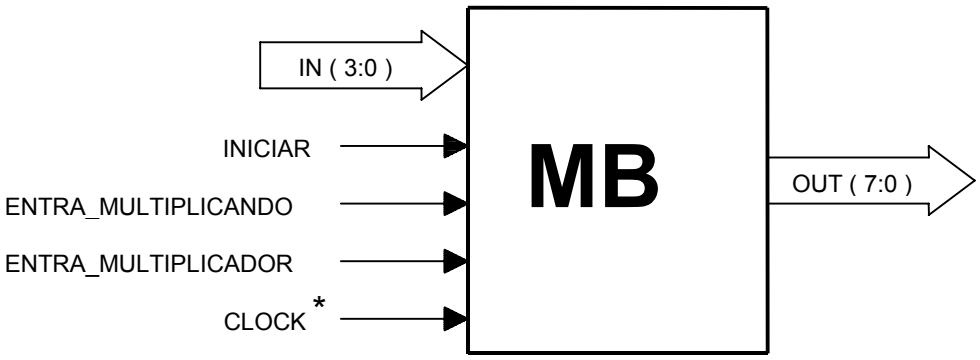


Figura 1 – Sinais de entrada e de saída do Multiplicador Binário.

A operação do circuito deve seguir os seguintes passos:

1. Acertar um valor binário na vida de dados de entrada (`IN`);
2. Ativar o sinal `ENTRA_MULTIPLICANDO`;
3. Colocar outro valor na via de dados de entrada;
4. Ativar o sinal `ENTRA_MULTIPLICADOR`;
5. Acionar o botão `INICIAR` para a execução da multiplicação binária;
6. Verificar resultado na via de dados de saída (`OUT`).

* Num projeto "profissional", o sinal CLOCK é gerado internamente ao Sistema Digital, através, por exemplo, de um cristal.

3.2. Projeto do MB com Circuitos Digitais

O SD especificado no item 3.1 pode ser projetado de diversas maneiras. Uma delas é baseada no particionamento em Fluxo de Dados (FD) e Unidade de Controle (UC).

Como o FD deve conter os elementos de transformação e/ou armazenamento dos dados do SD, os sinais IN e OUT devem ser conectados a ele. Já os sinais de ajuste de valores dos operandos e de início de operação devem ser ligados à UC, pois estes sinais estão relacionados com o controle da operação do circuito. Alguns sinais de controle e de estado devem ser conectados entre o FD e a UC. A figura 2 abaixo ilustra a estrutura interna do SD do MB.

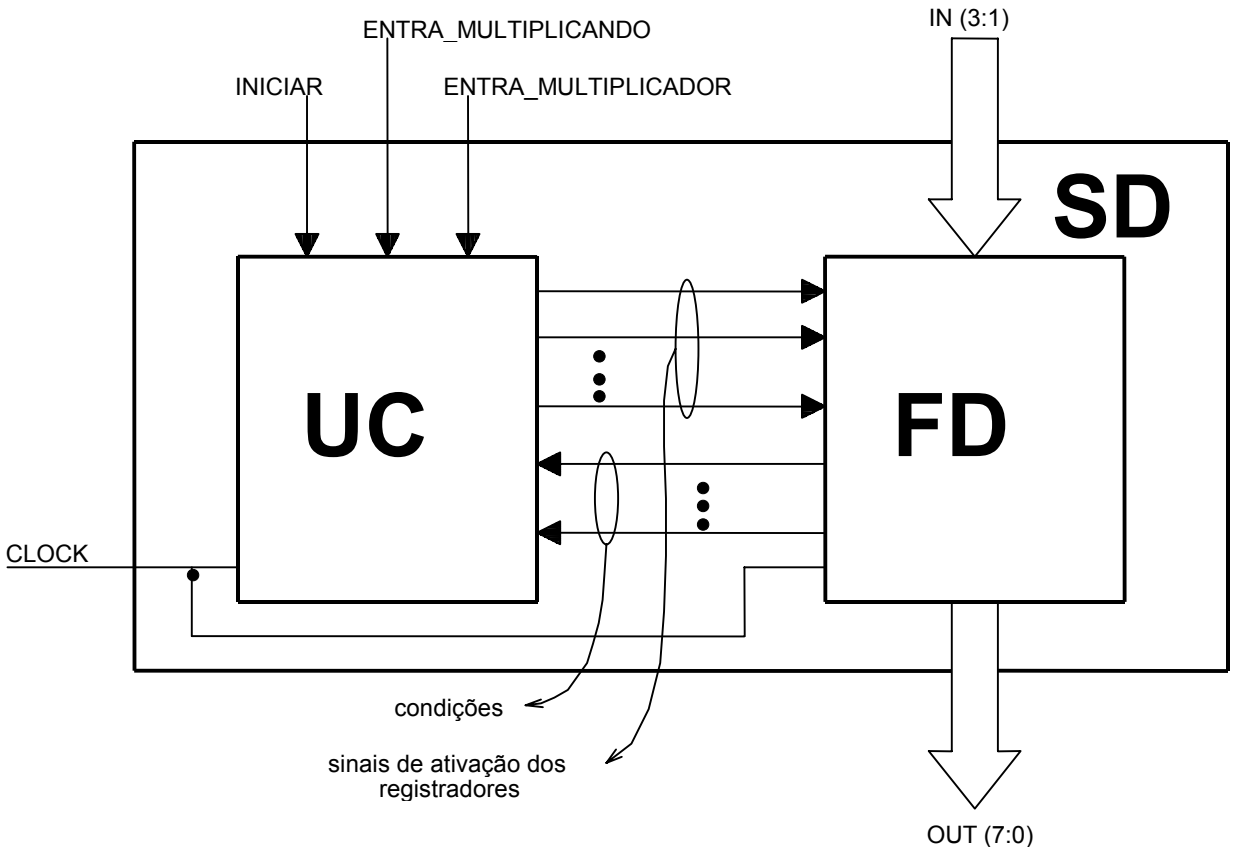


Figura 2 – Estruturação do SD do Projeto do MB.

Que sinais são gerados na UC?

Inicialmente, projeta-se o fluxo de dados (fig. 3) com os blocos necessários para a execução do algoritmo e com caminhos interligando-os, de forma a viabilizar a execução do mesmo. Pelo algoritmo descrito, é óbvio que necessitamos de um somador, de registradores de deslocamento, do flip-flop do Carry e do registrador B.

Para saber se a execução do algoritmo foi encerrada, optou-se por um contador decrescente, que é inicialmete carregado com (n-1) e, a cada passo, é decrementado, até atingir o valor zero.

Também analisando o algoritmo descrito, é importante conhecer, para executá-lo, o valor do bit menos significativo do multiplicador (Q_0).

Portanto, a UC receberá do FD a informação sobre Q_0 e sobre a saída do detetor de zero e deverá gerar os sinais de ativação das operações que serão executadas. Por exemplo, dependendo do valor de Q_0, o Deslocador A ou deverá ser carregado com a saída do Somador ($Q_0 = 1$)ou apenas deverá receber comando para deslocar à direita ($Q_0 = 0$).

3.2.1 Projeto do Fluxo de Dados

O fluxo de dados deve conter todos os elementos necessários para a execução do algoritmo. O diagrama lógico que representa o FD do MB é gerado a partir de um Diagrama de Blocos, como o da figura 3.

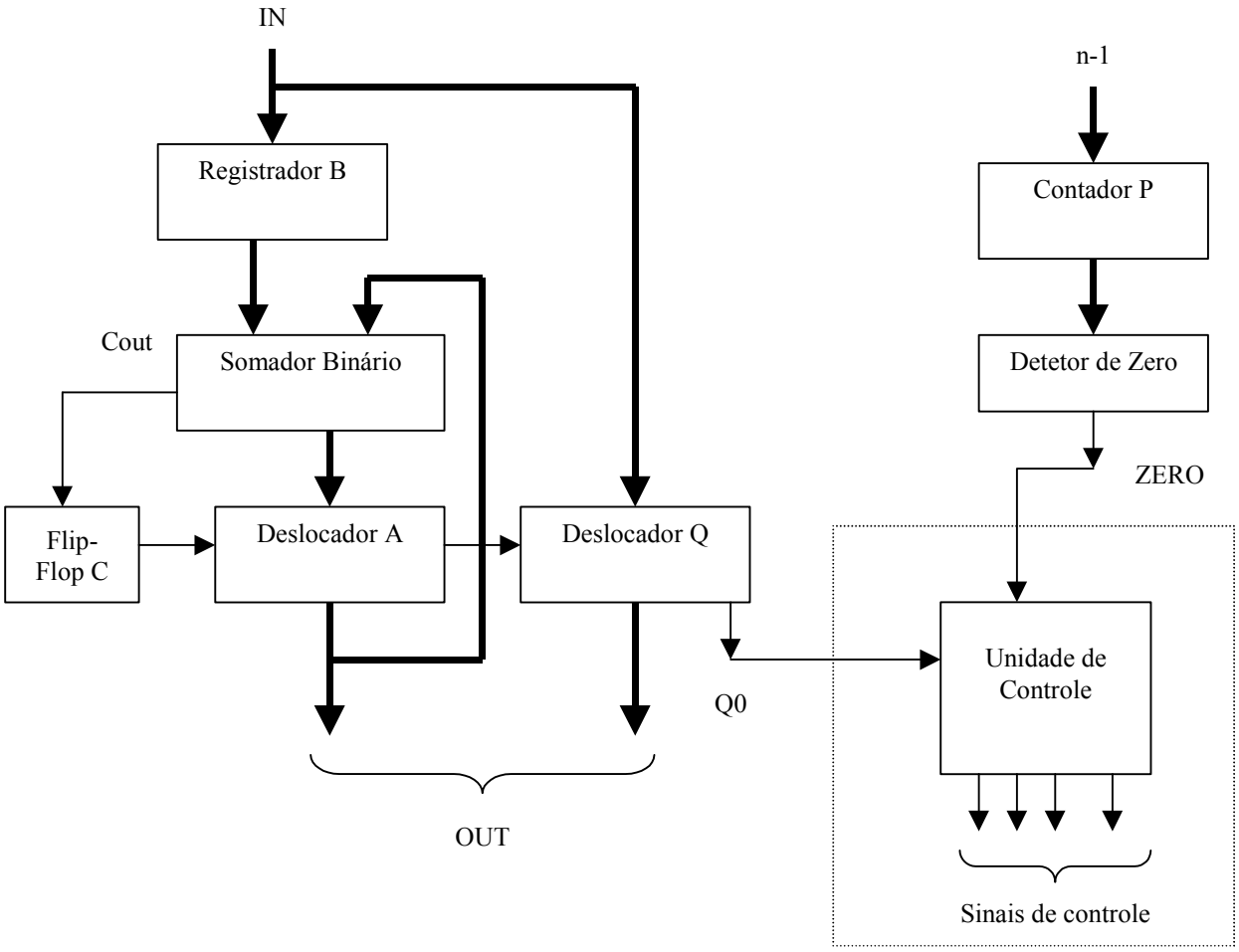


Figura 3 - Diagrama de Blocos do Multiplicador Binário.*

As etapas seguintes consistem da escolha dos componentes que realizam as funções dos blocos.

Para a implementação física do FD, podemos adotar os componentes da família 74XX, lembrando que todos os blocos seqüenciais devem efetuar as operações sincronamente e, de modo geral, devem possuir ENABLE:

- Contador P (74???)
- Flip-flop C (7474 + lógica)
- Registrador B 74173
- Registrador deslocador A 74194
- Registrador deslocador Q 74194
- Somador binário 7483/283
- Detetor de zero 7402 (NOR)

* Neste diagrama omitiram-se os sinais de comando dos registradores/contador (tipo LOAD, CLEAR, etc) e também o CLOCK.

3.2.2. Projeto da Unidade de Controle

A unidade de controle pode ser projetada de várias modos. Adotaremos aqui o Diagrama ASM. (Ler antes o resumo sobre diagramas ASM)

Para o MB, uma das soluções para a unidade de controle pode ser vista na figura 4. O diagrama ASM desta figura descreve o algoritmo de operação dos componentes do multiplicador. **Para a construção deste diagrama adotou-se a seguinte simplificação: o multiplicando e o multiplicador já foram previamente armazenados nos registradores B e Q.**

Neste diagrama é importante relembrar que a transição de um estado para outro só ocorre quando houver uma borda do CLOCK e que as ações contidas dentro dos blocos do estado ou das saídas condicionais somente serão realizadas na próxima borda do CLOCK.

Também é importante observar que o diagrama ASM foi construído supondo que os blocos seqüenciais do FD são genéricos (não são os mencionados no item 3.2.1) e que possuem sinais de controle (ativos em 1) para CLEAR, LOAD, SHIFT, CONTA, etc.

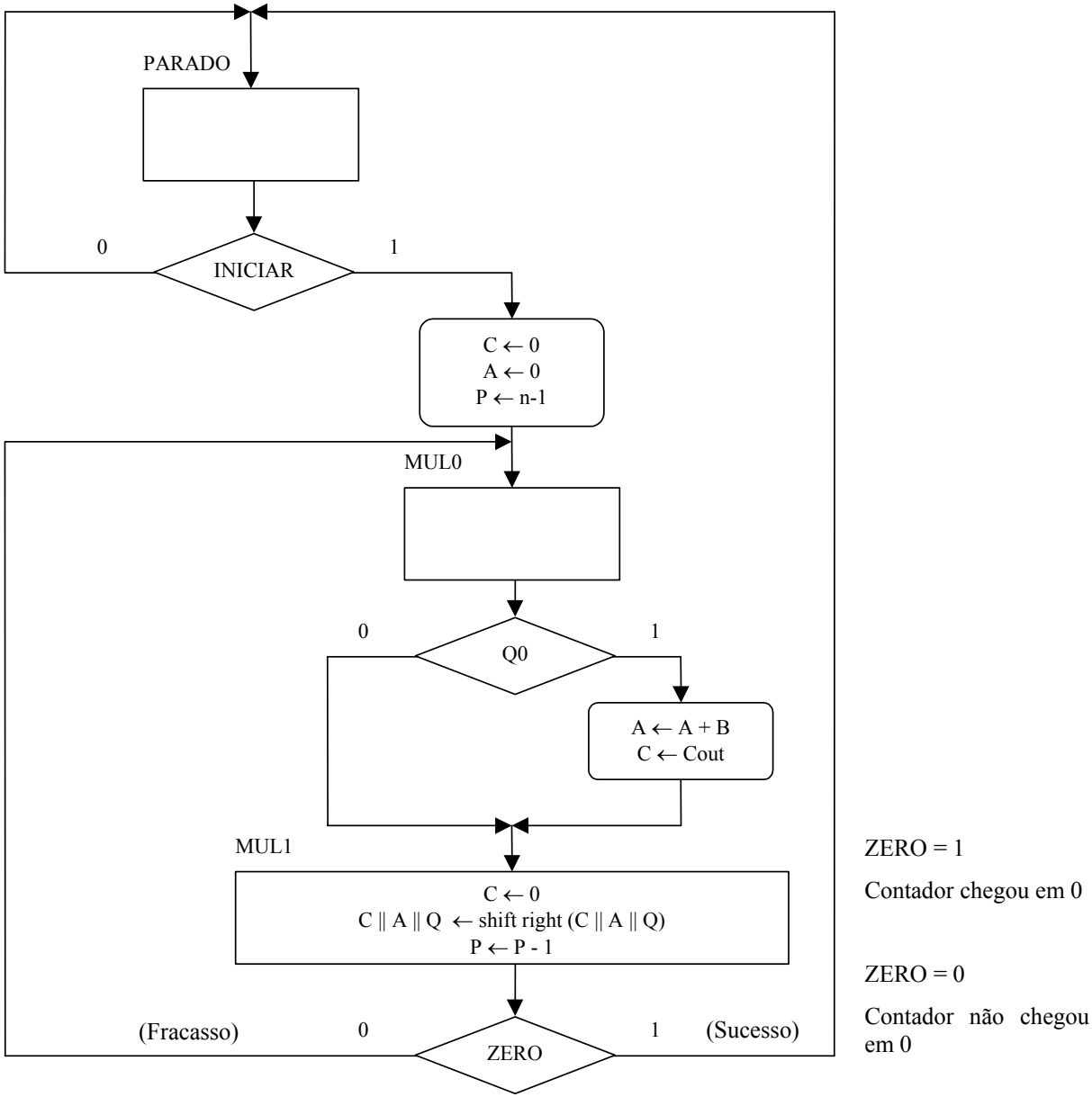


Figura 4 – Diagrama ASM da UC do Projeto do MB.

3.2.3 - Implementação da Unidade de Controle

Podemos adotar várias alternativas

a) Modelo clássico de circuito seqüencial síncrono

Construímos um circuito semelhante aos apresentados em PCS 2215, ou seja:

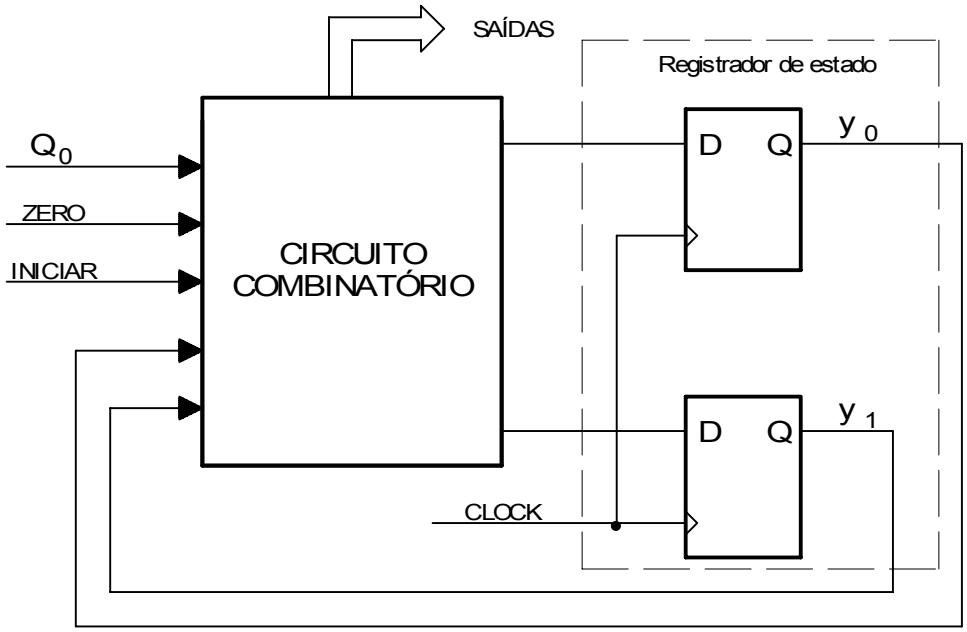


Figura 5 – Modelo Clássico.

Três estados → 2 Flip-flops

Saídas → 8 { CLEAR A, CLEAR C, LOAD P
LOAD A, LOAD C, SHIFT A, SHIFT Q,
CONTA

O circuito final pode ser montado com portas e flip-flops, ou com EPROM e flip-flops ou com PAL que tenha flip-flops na saída.

Com relação às saídas, podemos fazê-las totalmente independentes, gerando 8 sinais ou então, adotamos a estratégia resumida abaixo:

"Agrupamos saídas que sempre tem que estar ativas nas mesmas situações."

Observando o ASM, verificamos, por exemplo, que sempre que devemos acionar Shift A, também temos que acionar Shift Q. Portanto, uma única saída Shift, ligada aos dois registradores, é suficiente.

A tabela 1 mostrada a seguir resume essa estratégia.

Bloco	Ação	Condição	Grupo	Saída adotada
Reg A	A ← 0 (Clear A) A ← A+B (Load A) C || A || Q ← SR C || A || Q (Shift A)	PARADO . INICIAR MUL0 . Q_0 MUL1	1 2 3	Initialize Load Shift_dec
F.F.C.	C ← 0 (Clear C) C ← Cout (Load C)	PARADO . INICIAR+MUL1 MUL0 . Q_0	4 2	Clear C Load
Reg Q	C || A || Q ← SR C || A || Q (Shift Q)	MUL1	3	Shift_dec
Cont P	P ← n - 1 (Load P) P ← P - 1 (Conta)	PARADO . INICIAR MUL1	1 3	Initialize Shift_dec

∴ São apenas necessárias 4 saídas

Tabela 1 – Agrupamento das Saídas

b) Modelo de "UM Flip-Flops por ESTADO"

Os diagramas ASM podem ser diretamente mapeados em circuitos, de acordo com as equivalências mostradas na figura 6, extraída da referência 1

Como temos 1 flip-flop por estado, o "nome do estado" representa uma variável de chaveamento.

variável / estado	PARADO	MUL0	MUL1
PARADO	1	0	0
MUL0	0	1	0
MUL1	0	0	1

Adotando esta técnica e agrupando as saídas como detalhado anteriormente, o circuito resultante é mostrado na figura 7. Observe que as expressões das saídas já estão expressas na própria tabela 1, na coluna CONDIÇÃO.
Essa solução requer que a condição inicial, imposta ao ligar o circuito, seja 100 (ou seja, estado PARADO).

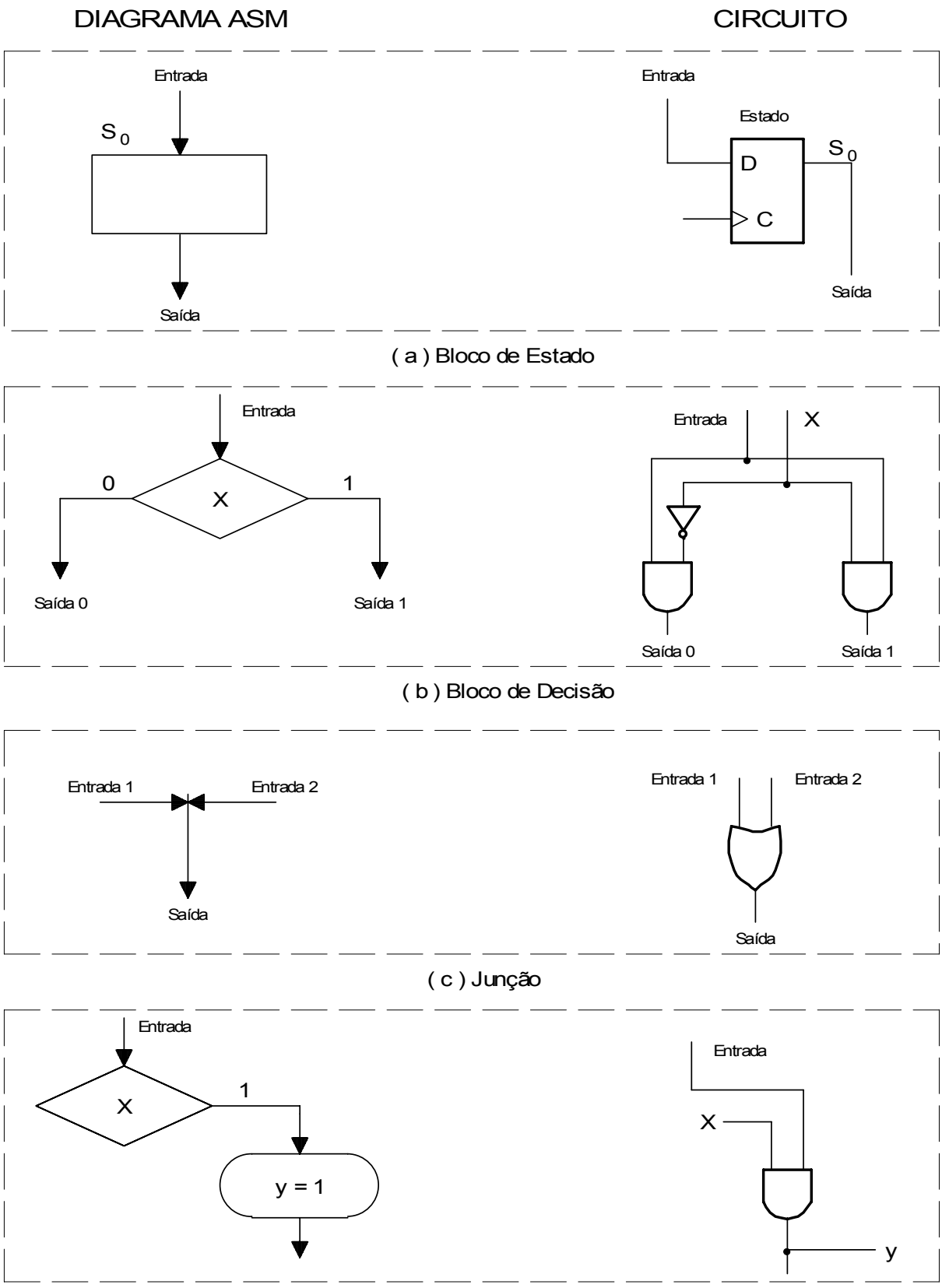
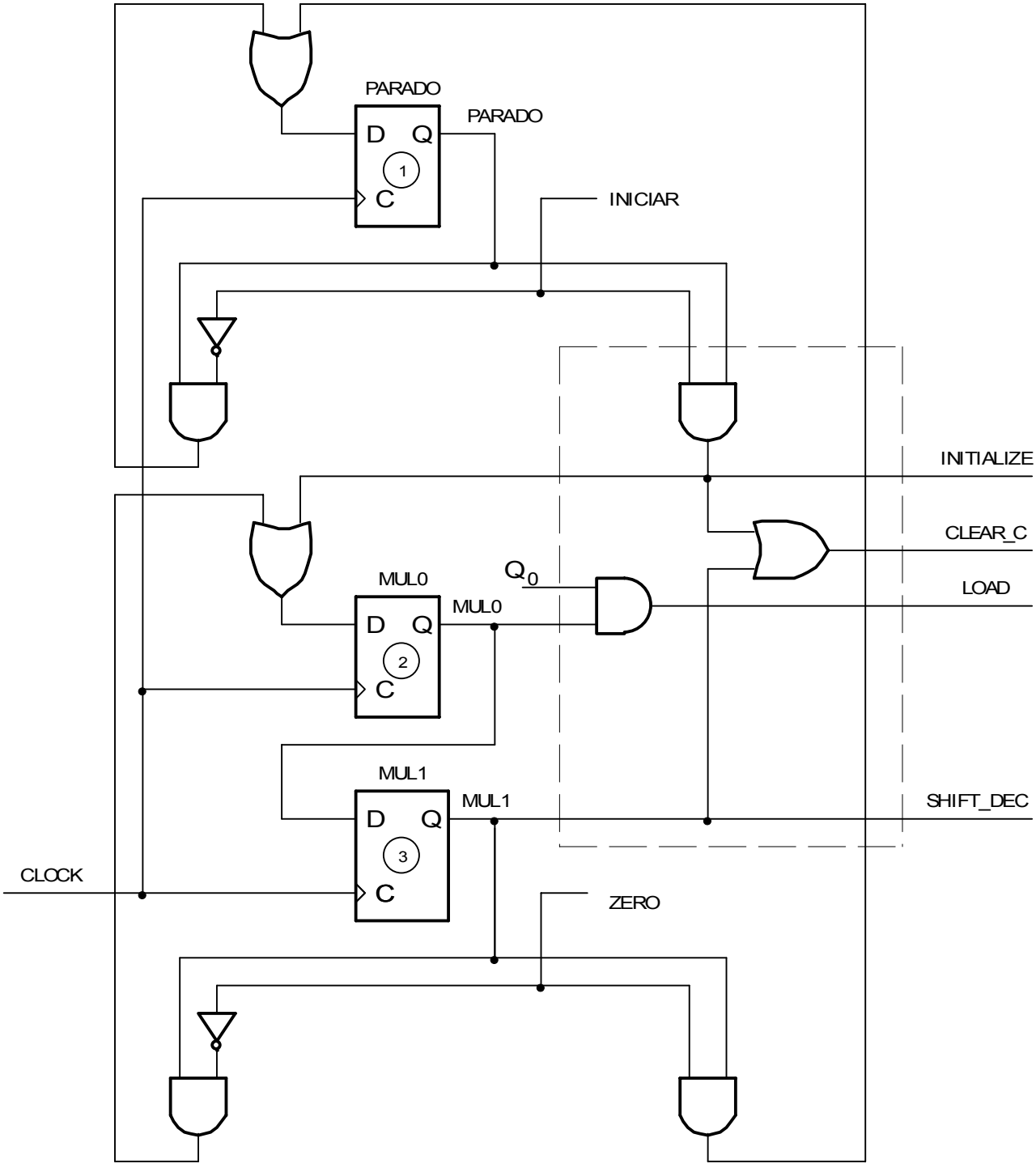


Figura 6 - Mapeamento ASM → CIRCUITO



Condição inicial : (ao ligar)

- **FF ① - 1**
- **FF ② - 0**
- **FF ③ - 0**

Figura 7 – Circuito da UC - Mapeamento direto

OBS: O bloco pontilhado será mencionado apenas no item c)

c) Modelo Registrador de Estado-Decodificador

A solução do item b apresenta a grande vantagem de facilitar a obtenção do circuito mas, quando o número de estados é muito grande, o número de flip-flops pode encarecer o produto.
Por outro lado, é muito conveniente ter a informação sobre os estados, de forma decodificada. Por exemplo, para saber se o circuito está num estado S_i é só olhar para a saída do Flip-flop do estado S_i.

O modelo apresentado na figura 8 é uma mistura dos modelos a) e b). Os estados serão representados de forma codificada (*n* flip-flops $\rightarrow 2^n$ estados), mas serão decodificados, de forma a se obter variáveis iguais às do item b, para representá-los (PARADO, MUL0 MUL1).

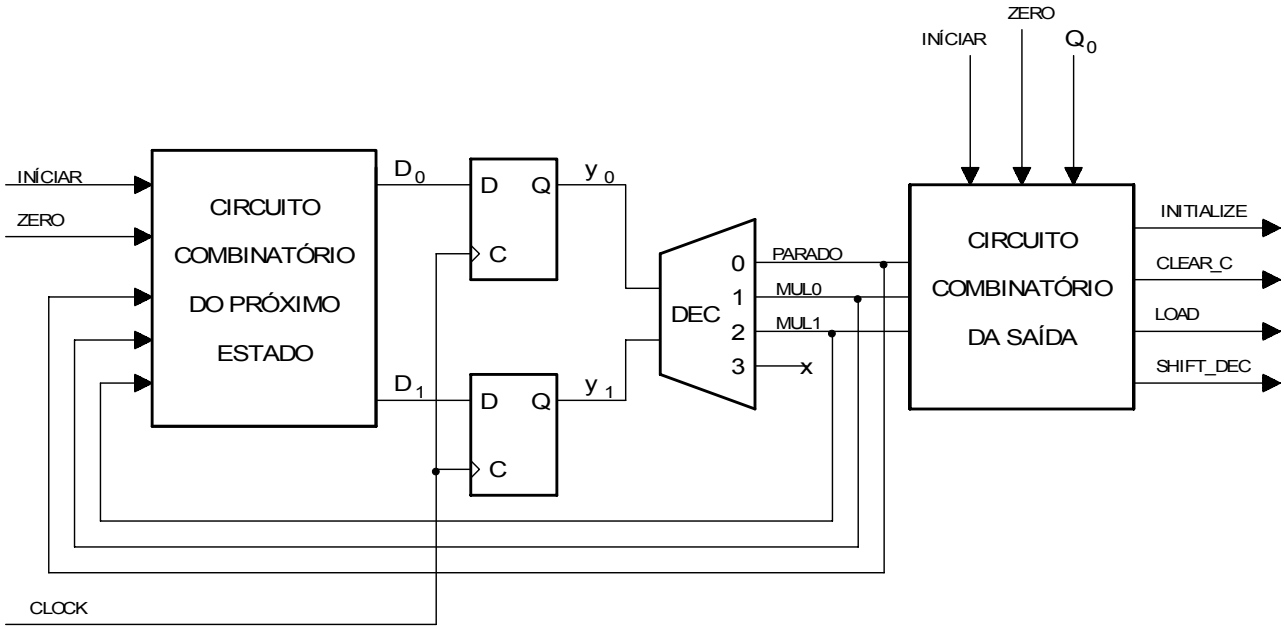


Figura 8 – Modelo Registrador de Estado - Decodificador

O circuito combinatório da saída é exatamente igual ao bloco pontilhado da figura 7.
O circuito combinatório do próximo estado pode ser extraído da tabela 2.

Estado atual			Condições		Próximo Estado		Decodificador		
	y_1	y_0	**Iniciar**	**Zero**	y_1	y_0	**PARADO**	**MUL0**	**MUL1**
Parado	0	0	0	X	0	0	1	0	0
	0	0	1	X	0	1	1	0	0
MUL0	0	1	X	X	1	0	0	1	0
MUL1	1	0	X	0	0	1	0	0	1
			X	1	0	0	0	0	1

Tabela 2 – Tabela de transição

Observando a tabela (e lembrando que $D_i(t) = y_i(t+1)$), temos

$D_0 = (PARADO) \cdot (INICIAR) + (MUL1) \cdot \overline{(ZERO)}$

$D_1 = MUL0$

Portanto, o circuito final está detalhado na figura 9, onde se observa que a porta ① é comum ao bloco combinatório do próximo estado e ao bloco combinatório da saída.

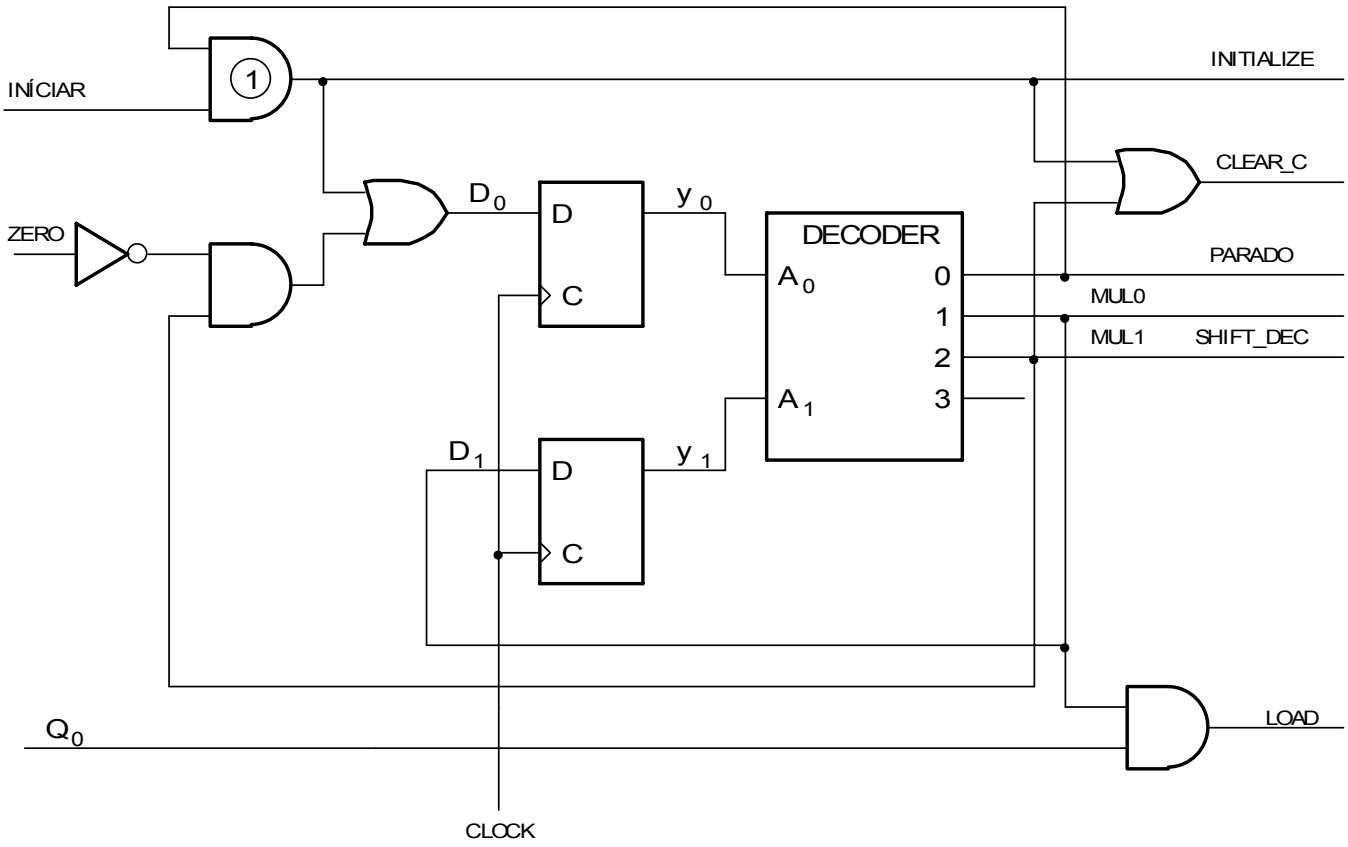


Figura 9 – Circuito detalhado

4.Bibliografias Importantes

Midorikawa, ET, multiplicador binário - Apostila de PCS – 2355 / 2308

Kime, CR; Mano, MM. “Logic and Computer Design Fundamentals”. New Jersey Prentice Hall - 2000