Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística
Ciências da Computação & Engenharia Eletrônica

Sistemas Digitais

INE 5406

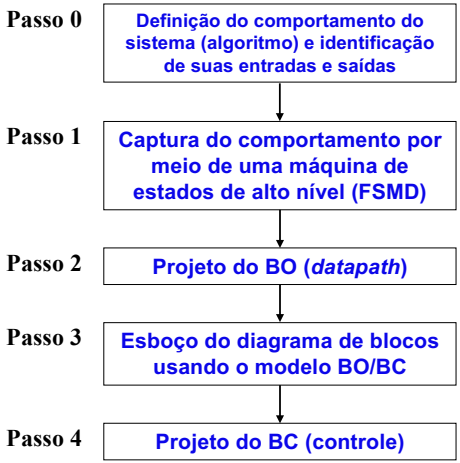
Aula 3-T (adendo)

2. Processadores Dedicados (Blocos Aceleradores). Método de Projeto no Nível RT. Exemplo 2 de somadores sequenciais.

Profs. José Luís Güntzel e Cristina Meinhardt
{j.guntzel, cristina.meinhardt}@ufsc.br

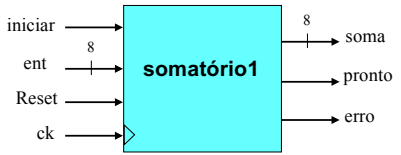
Processadores Dedicados

Método de Projeto de Sistemas Digitais no Nível RT



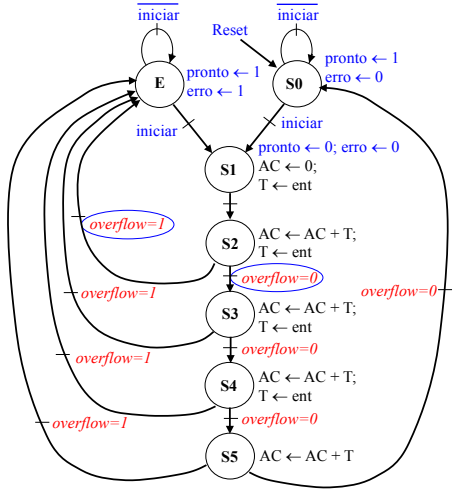
Processadores Dedicados

Aula Passada: Exemplo 1



1. AC ← 0; T← ent; // A está estável em ent
2. AC ← AC + T; T← ent; // B está estável em ent
3. AC ← AC + T; T← ent; // C está estável em ent
4. AC ← AC + T; T← ent; // D está estável em ent
5. AC ← AC + T; // O resultado final S estará em AC

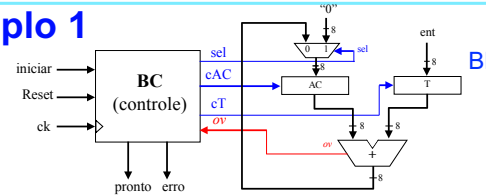
O 1º teste de overflow seria dispensável, uma vez que na primeira soma um dos operandos é zero. Porém, **iremos mantê-lo para permitir uma futura generalização do algoritmo.**



Processadores Dedicados

Aula Passada: Exemplo 1

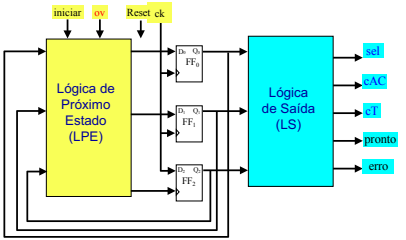
Sistema Digital somatório1



Bloco Operativo (BO)

Lógica de Próximo Estado (LPE)

Q2	Q1	Q0	iniciar	*ov*	$Q2^+$	$Q1^+$	$Q0^+$
1	**1**	**1**	0	X	**1**	**1**	**1**
1	**1**	**1**	1	X	0	0	1
0	0	0	0	X	0	0	0
0	0	0	1	X	0	0	1
0	0	1	X	X	0	1	0
0	1	0	X	0	0	1	1
0	1	0	X	1	**1**	**1**	**1**
0	1	1	X	0	1	0	0
0	1	1	X	1	**1**	**1**	**1**
1	0	0	X	0	1	0	1
1	0	0	X	1	**1**	**1**	**1**
1	0	1	X	0	0	0	0
1	0	1	X	1	**1**	**1**	**1**



Lógica de Saída (LS)

Q2	Q1	Q0	Sinais de comando			Saídas de controle	
			sel	cAC	cT	pronto	erro
1	1	1	X	0	0	1	1
0	0	0	X	0	0	1	0
0	0	1	1	1	1	0	0
0	1	0	0	1	1	0	0
0	1	1	0	1	1	0	0
1	0	0	0	1	1	0	0
1	0	1	0	1	1	0	0

T_{exec} = nº de ciclos x T = 5 ciclos x 5ns = 25 ns

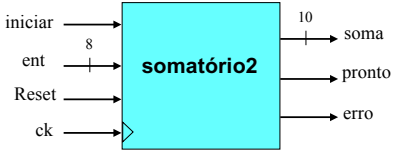
	Estimativa de Custo
B.O.	576 transistores
B.C.	7 estados
	5 sinais de controle

Processadores Dedicados

Exemplo 2: Enunciado

SD (bloco acelerador) para cálculo de um somatório de 4 números

Especificação das interfaces: Necessita-se de um sistema digital (SD) dedicado (i.e., um bloco acelerador) capaz de realizar o cálculo **A+B+C+D**, onde **A**, **B**, **C** e **D** são números* inteiros sem sinal, representados em binário com 8 bits. Este sistema digital, doravante denominado de "somatório2", possui uma entrada de relógio ("ck"), uma entrada de reset assíncrono ("Reset"), **uma** entrada de dados com 8 bits ("ent"), uma entrada de controle denominada "iniciar", **uma saída de controle ("pronto")** e uma saída de dados de **10** bits ("soma").



* Os números fornecidos como entrada do sistema são comumente chamados de "operandos (de entrada)"

O Enunciado do Exemplo 2 é quase igual ao enunciado do Exemplo 1. As diferenças estão grifadas em vermelho.

Processadores Dedicados

Exemplo 2: Enunciado

SD (bloco acelerador) para cálculo de um somatório de 4 números

Especificação do comportamento:

Há somente um estado inicial, S0. Enquanto um novo cálculo não inicia, "somatório2" permanece em S0;

O sinal externo "iniciar" dá o comando para iniciar um cálculo **A+B+C+D**.

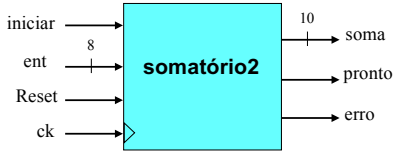
À medida que o cálculo é realizado, os valores dos operandos **A**, **B**, **C** e **D** vão sendo fornecidos pela entrada "ent", em bordas de relógio consecutivas.

Uma vez iniciado, o cálculo é realizado de maneira sequencial e cumulativa (i.e., cada novo operando de entrada que chega é somado ao valor acumulado até então).

Em nenhuma das adições ocorre *overflow*. Para garantir isso, os registradores e o somador a serem utilizados devem ser dimensionados convenientemente.

Processadores Dedicados

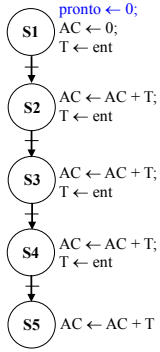
Exemplo 2: Passo 1 (Captura do comportamento por meio de uma FSMD)



1. AC ← 0; T← ent; // A está estável em ent
2. AC ← AC + T; T← ent; // B está estável em ent
3. AC ← AC + T; T← ent; // C está estável em ent
4. AC ← AC + T; T← ent; // D está estável em ent
5. AC ← AC + T; // O resultado final S estará em AC

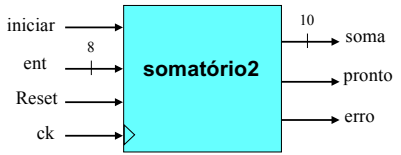
Criando um estado para cada passo do algoritmo

Igual ao Exemplo 1.



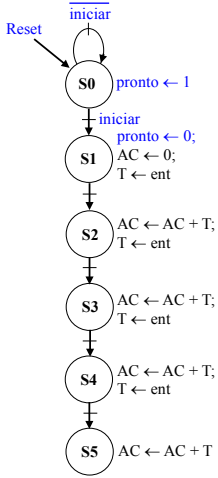
Processadores Dedicados

Exemplo 2: Passo 1 (Captura do comportamento por meio de uma FSMD)



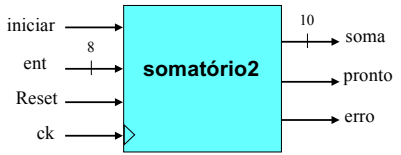
1. AC ← 0; T← ent; // A está estável em ent
2. AC ← AC + T; T← ent; // B está estável em ent
3. AC ← AC + T; T← ent; // C está estável em ent
4. AC ← AC + T; T← ent; // D está estável em ent
5. AC ← AC + T; // O resultado final S estará em AC

Acrescentando o estado inicial "S0"



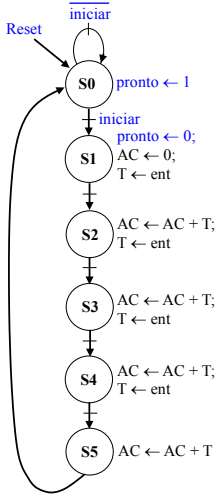
Processadores Dedicados

Exemplo 2: Passo 1 (Captura do comportamento por meio de uma FSMD)



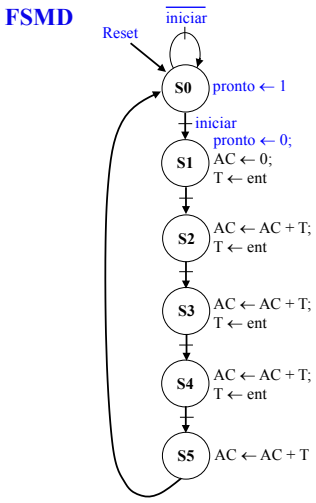
1. AC ← 0; T← ent; // A está estável em ent
2. AC ← AC + T; T← ent; // B está estável em ent
3. AC ← AC + T; T← ent; // C está estável em ent
4. AC ← AC + T; T← ent; // D está estável em ent
5. AC ← AC + T; // O resultado final S estará em AC

Como nunca vai ocorrer *overflow* quando chegar a S5, a FSM volta a S0 (e a execução termina com o resultado correto).



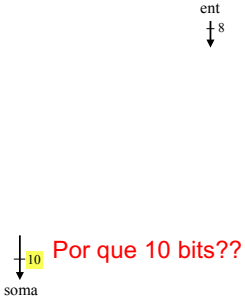
Processadores Dedicados

Exemplo 2: Passo 2 (Projeto do BO)



1ª questão para guiar o projeto do BO:
Quais são os sinais de interface do BO?

- "ent" e "soma"



Processadores Dedicados

Exemplo 2: Passo 2 (Projeto do BO)

Analisando o número mínimo de bits de todas as adições que o algoritmo realiza, a fim de evitar *overflow* ...

2ª adição

		binário	decimal	Nro de bits
AC		1111 1111	255	8
T	+	1111 1111	255	8
		11111 1110	510	9

3ª adição

		binário	decimal	Nro de bits
AC		11111 1110	510	9
T	+	1111 1111	255	8
		10 1111 1101	**765**	**10**

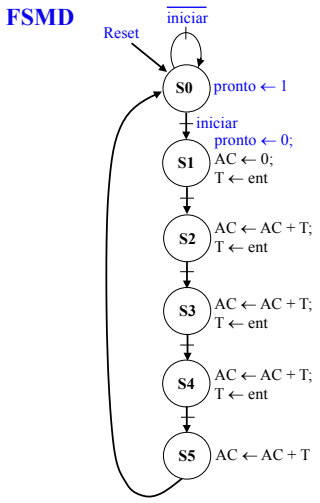
4ª adição

		binário	decimal	Nro de bits
AC		**10 1111 1101**	**765**	**10**
T	+	1111 1111	255	8
		11 1111 1100	**1020**	**10**

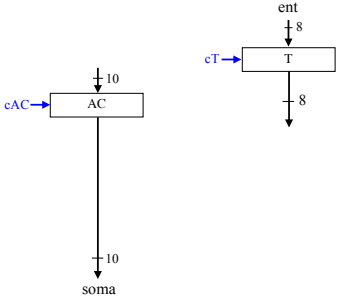
Conclusão: na 4ª soma há um operando de entrada e a própria saída do somador com 10 bits

Processadores Dedicados

Exemplo 2: Passo 2 (Projeto do BO)

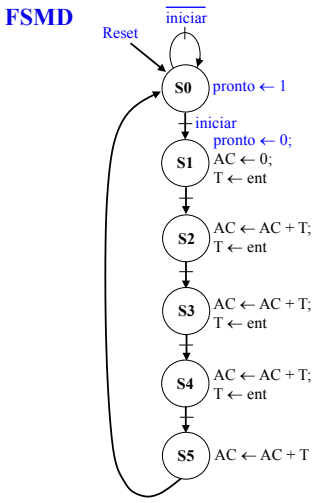


2ª questão para guiar o projeto do BO:
Quais variáveis são usadas para armazenar dados?
• "AC" e "T"
Logo, o BO precisa ter dois registradores. Chamemo-los de "AC" e "T".



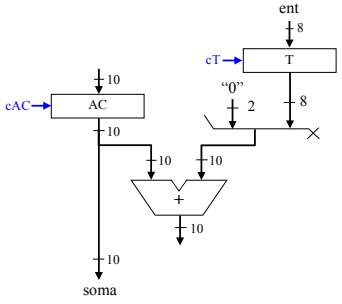
Processadores Dedicados

Exemplo 2: Passo 2 (Projeto do BO)



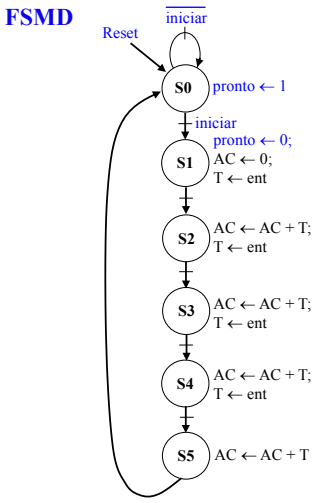
3ª questão para guiar o projeto do BO:

- Quais operações são realizadas?
- Adição para números de 10 bits e o somador não precisa testar *overflow*



Processadores Dedicados

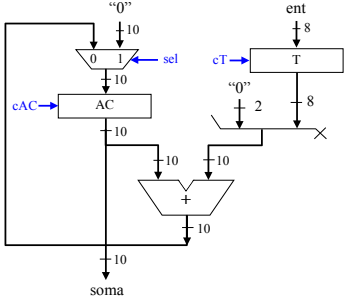
Exemplo 2: Passo 2 (Projeto do BO)



4ª questão para guiar o projeto do BO:
Quais operações são realizadas sobre quais dados (incluindo-se as condições)?

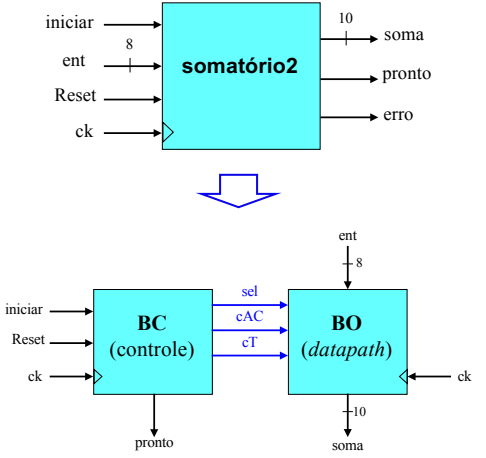
- T← ent, AC← 0; AC← AC+T

Logo, deve haver um mux2:1 na entrada de AC e conexões entre AC e +, T e + e + e AC.



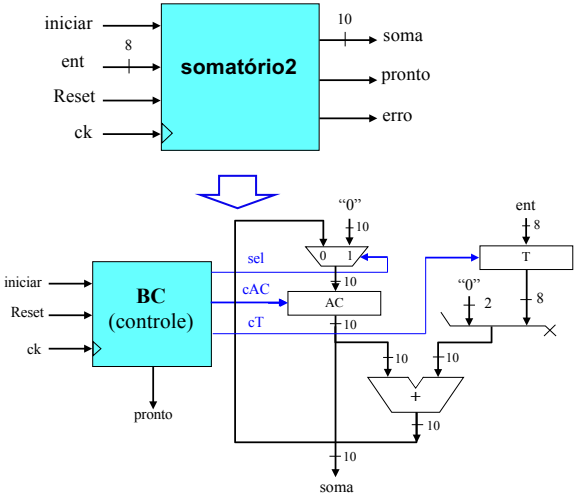
Processadores Dedicados

Exemplo 2: Passo 3 (Esboçando o Diagrama BO/BC)



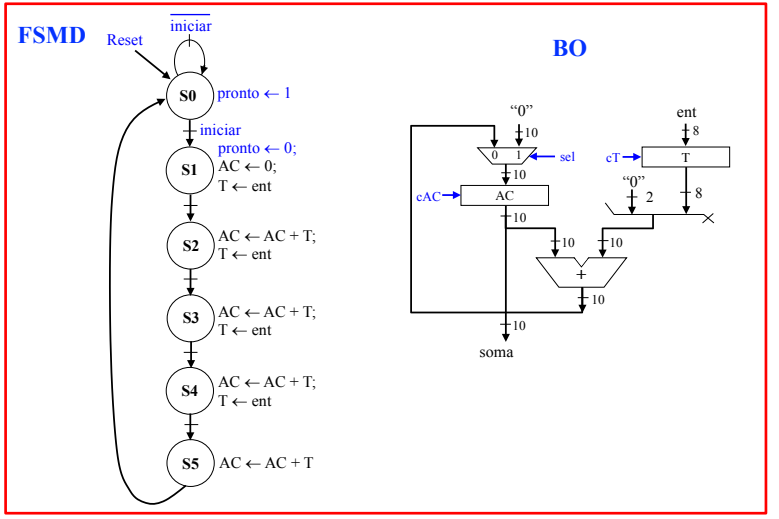
Processadores Dedicados

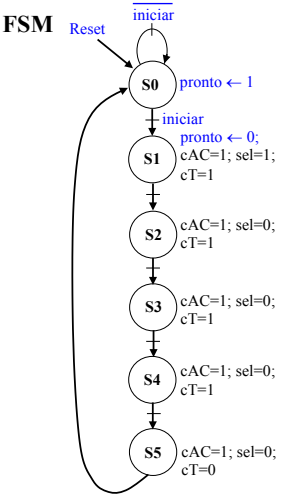
Exemplo 2: Passo 3 (Esboçando o Diagrama BO/BC)



Processadores Dedicados

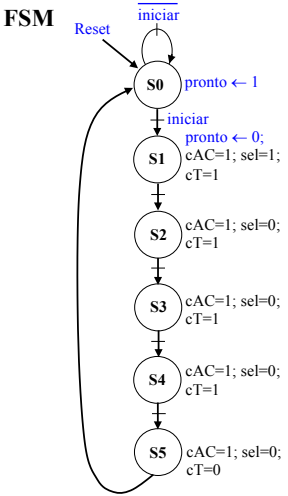
Exemplo 2: Passo 4 (Projeto do BC): Criando uma FSM a partir da FSMD e do BO





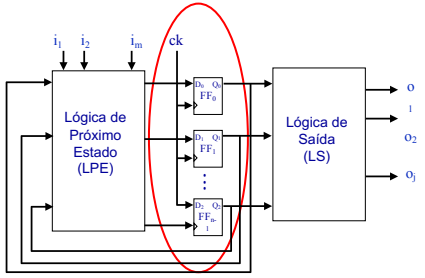
Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): Definindo o número de flip-flops



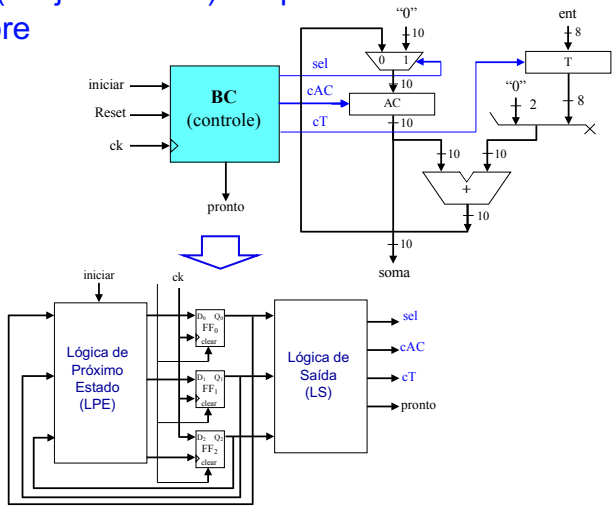
Quantos flip-flops são necessários para implementar a FSM?

Resp.: como são **6** estados (=**6** combinações), o cálculo do número de flip-flops é $\log_2 \mathbf{6} = 3$



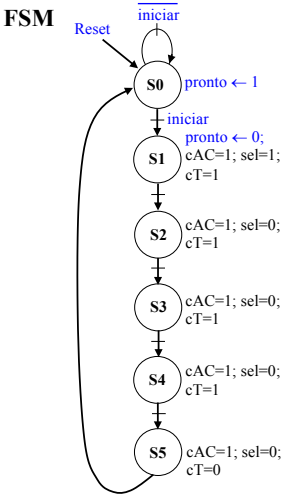
Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): Mapeando as interfaces do BC para o modelo de FSM de Moore



Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): Assinalamento de Estados



Corresponde a:

•Escolher uma codificação binária para representar cada estado...

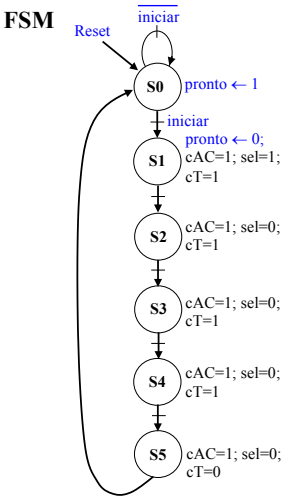
•Se for para minimizar o número de bits, usar $\log_2 N$, onde N é o número de estados. Exemplo de assinalamento:

Estado	Código binário dos estados		
	Q2	Q1	Q0
S0	**0**	**0**	**0**
S1	0	0	1
S2	0	1	0
S3	0	1	1
S4	1	0	0
S5	1	0	1

← Estado inicial: este é o estado para o qual a FSM vai caso o Reset seja acionado!!

Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): projetando a LPE



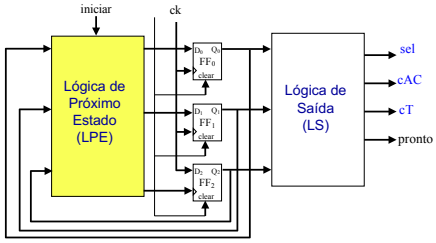


Tabela de Transição de Estados (LPE)

Estado atual	iniciar	Próximo estado
S0	0	S0
S0	1	S1
S1	X	S2
S2	X	S3
S3	X	S4
S4	X	S5
S5	X	S0

Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): projetando a LPE

Estado	Código binário dos estados		
	Q2	Q1	Q0
S0	**0**	**0**	**0**
S1	0	0	1
S2	0	1	0
S3	0	1	1
S4	1	0	0
S5	1	0	1

Substituindo os nomes dos estados pelos respectivos códigos binários...

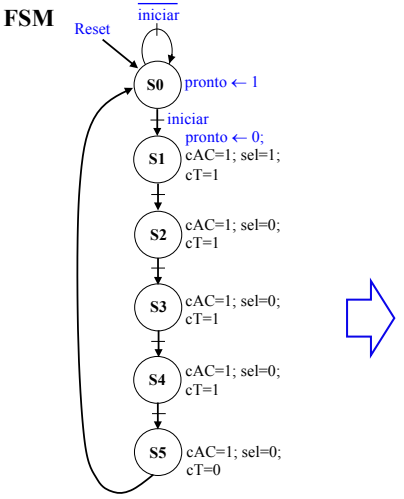
Tabela de Transição de Estados (LPE)

Tarefa: completar

Estado atual	iniciar	Próximo estado
S0	0	S0
S0	1	S1
S1	X	S2
S2	X	S3
S3	X	S4
S4	X	S5
S5	X	S0

Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): projetando a LS



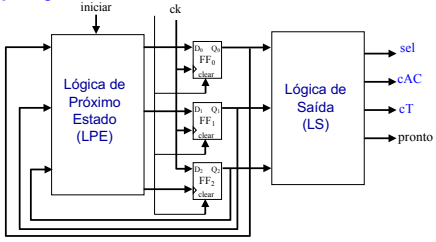


Tabela-Verdade da LS

Estado atual	Sinais de comando			
	sel	cAC	cT	pronto
S0	X	0	0	1
S1	1	1	1	0
S2	0	1	1	0
S3	0	1	1	0
S4	0	1	0	0
S5	0	1	0	0

Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): projetando a LS

Estado	Código binário dos estados		
	Q2	Q1	Q0
S0	**0**	**0**	**0**
S1	0	0	1
S2	0	1	0
S3	0	1	1
S4	1	0	0
S5	1	0	1

Tabela-Verdade da LS

Substituindo os nomes dos estados pelos respectivos códigos binários...

Tarefa: completar

Estado atual	Sinais de comando			
	sel	cAC	cT	pronto
S0	X	0	0	1
S1	1	1	1	0
S2	0	1	1	0
S3	0	1	1	0
S4	0	1	0	0
S5	0	1	0	0

Processadores Dedicados

Exemplo 2: Passo 4 (Projeto do BC): projetando a LS

Os passos faltantes são similares ao Exemplo 1

Tarefa:
1. Completar os passos faltantes do Exemplo 2
2. Comparar (de maneira qualitativa) esta FSM com a do Exemplo 1