

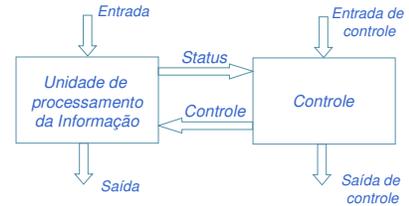
Sistemas Digitais

Aula 11

1

Projeto de sistemas digitais

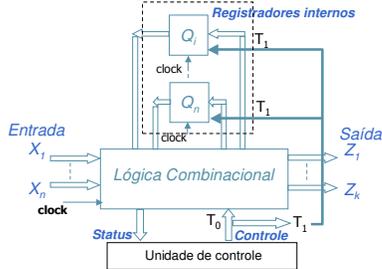
- Unidade de Processamento da Informação
 - Transferência de Informação
 - Operação sobre a informação
- Unidade de Controle
 - Determina a seqüência de operação a ser realizada



2

Unidade de Procesamento da Informação

- Componentes
 - Lógica Combinacional
 - Registradores internos
 - Circuito de temporização



3

Expressões de próximo estado na unidade de Processamento

- Registradores - > FLIP-FLOP Master/Slave
 - valor corrente ou estado corrente do registrador é o valor armazenado no registrador quando o pulso de transferência é aplicado.
 - Enquanto o pulso de transferência estiver "ON", a saída do registrador mantém o valor corrente.
 - Quando o pulso de transferência for removido (carga na descida do relógio), o registrador passará a conter o próximo estado definido pelas entradas.
- Equação de próximo estado

$$Q_i <- F_i(X_1, \dots, X_n, Q_1, \dots, Q_m, T)$$

X - Valor do sinal de entrada
Q - estado atual
T - sinal de controle
- Equação de saída
 - $Z_j := G_j(X_1, \dots, X_n, Q_1, \dots, Q_m, T)$

4

Unidade de Controle

STATUS

- Permite alterar a seqüência de computação
 - $S_i := X_i(X_1, \dots, X_n, Q_1, \dots, Q_m)$
- O sinal de status não é função de T
- O sinal de status S só alcança um estado permanente depois que a entrada X alcançar um estado permanente.
- A unidade de controle pode usar o valor corrente de S na determinação do valor corrente do sinal de controle T.
- O sinal de transferência, descida ou subida do relógio, é aplicado apenas quando a unidade de controle teve tempo suficiente para gerar o sinal de controle T, após a recepção do valor corrente de S.
- Exemplo de sinais de Status

O sinal de status pertence ao conjunto {XGT, XEQ, XNEQ, XGEO, XLT}

 - Onde
 - XGT X maior que [0]
 - XEQ X igual a [0]
 - XNEQ X diferente de [0]
 - XLT X menor que [0]

5

Unidade de Controle

Sinal de controle

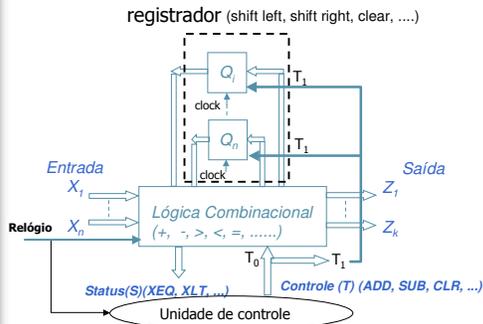
- O sinal de controle (T) é usado para definir:
 - Transferência de informação realizada pela unidade de processamento.
 - Valor do sinal de saída gera T
 - T pode assumir valores diferentes de acordo com a seqüência de operações do sistema
- Exemplo de T (instruções)

{CLR, ADD, SUB, INC, DEC}

 - CLR - faz A e B iguais a zero (clear)
 - ADD - Adiciona o conteúdo de A ao de B e coloca o resultado em A
 - SUB - Subtrai o conteúdo de B de A e coloca o resultado em A
 - INC - Adiciona [1] ao conteúdo de A
 - DEC - Subtrai [1] do conteúdo de A

6

Sinais de controle/status - exemplo



7

Considerações de tempo

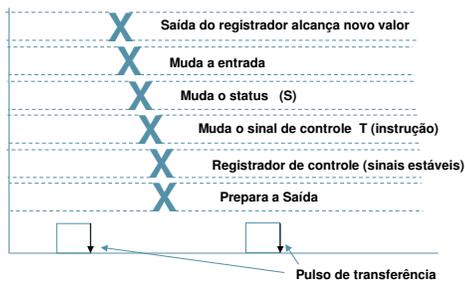
- **Clocked systems** - Existe um relógio geral que gera a seqüência padrão de pulsos de transferência.



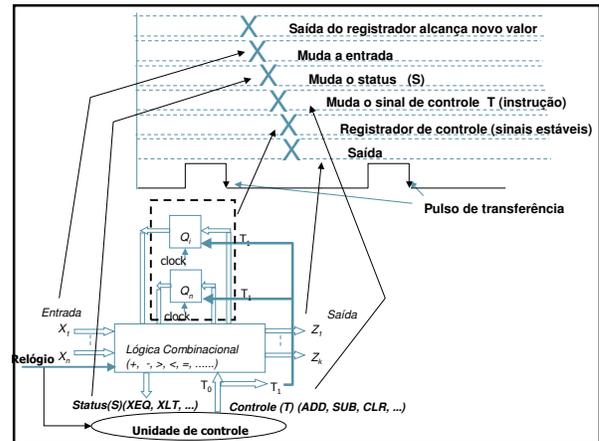
- Quando o pulso de transferência for aplicado é suposto que todos os sinais envolvidos na operação de transferência associado com esse evento tenham alcançado o estado permanente.
- Os valores das entradas e das saídas dos registradores devem permanecer constantes durante os t segundos em que o pulso de transferência estiver presente.
- Quando o pulso de transferência terminar (ou começar), a saída dos registradores envolvidos na transferência assume os novos valores.

8

Relação de tempo entre os diferentes sinais na unidade de processamento da informação



9



Unidade de controle

Algoritmo para computar $F(x) = Y$

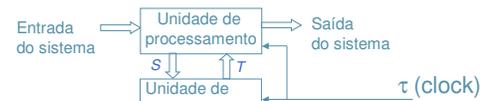
- Calcular $F(x)$ executando operações e testes na ordem prescritas
- Indicar que não existe Y que satisfaça as condições da computação

Condições de implementação do algoritmo

- Ter dispositivos capazes de implementar as operações previstas
- Descrever a computação como uma seqüência das operações especificadas (programa de máquina)
- Ter um dispositivo capaz de levar adiante os passos da computação (Unidade de controle)
 - Modelo da Unidade de Controle
 - Linguagem de descrição de hardware

11

Programa de hardware



- **Mecanismo que implementa a funcionalidade do sistema de computação**

- A unidade de processamento foi definida e as operações e testes completamente descritas por uma tabela de especificações.
 - Operações T
 - Testes S
- O sistema é síncrono. O pulso τ faz com que a operação indicada por T seja executada e a unidade de controle passe para a próxima instrução.
- Todos os sinais da unidade de informação devem ter alcançado o estado permanente antes do próximo τ ser aplicado.

12

Modelo da unidade de controle

A unidade de controle determina que evento ocorrerá no próximo pulso de relógio

Equação de T

$$T := F(S, Q)$$

Equação de próximo estado

$$Q \leftarrow G(S, Q)$$

Status

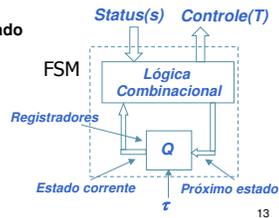
{ [s₁], ..., [s_n] }

Controle

{ [t₁], ..., [t_m] }

Estado

{ [q₁], ..., [q_n] }



13

Tabela de transição de estados

Q \ S	S _A	S _i	S _j
Q _A	Q ₀ /-				-/F(S,Q)
.....					
Q _j			G(S _i ,Q _i)/F(S _i ,Q _i)		-/-
.....					
Q _w					

G(S_i,Q_i) - próximo estado

F(S_i,Q_i) - Valor corrente de T

Nem todas as entradas da tabela de transição precisam estar completas. Existem situações onde se sabe que um certo status nunca ocorrerá para um dado estado. (Don't Care)

14

Processo de programação

1. Informação de entrada
 2. Realizar uma seqüência de operações
 3. Testar e decidir o que fazer nos próximos passos
 4. Repetir uma seqüência de passos até que certas tarefas computacionais sejam realizadas
 5. Informações de saída
 - Programa de hardware
- Tabela de especificação -> Linguagem
Operações -> Controle T
Testes -> Status S
- Projeto
 - Definir a tarefa computacional a ser realizada
 - Definir a unidade de processamento da informação
 - Definir o algoritmo para implementar a computação
 - Criar a tabela de estado da unidade de controle
 - Implementar e testar o hardware

15

Linguagem de hardware - exemplo

<LABEL> Qualquer elemento da classe
Exemplo: A1, Start, Fim

- <STATUS >
Exemplo:
a. Forma mnemônica GT, LT, EQU
b. Forma de vetor [1,0,1], [1,1,1]

- <CONTROLE >
Exemplo: NOP, CLR, ADD, SUB

- <COMENTÁRIOS>
/* Isto é um comentário

<STATEMENTS>
- Simple statements
- Compound statements

16

Eventos

τ_i : <expressão de transferência>

O evento descrito pela expressão de transferência ocorre no instante τ_i (período de clock)

Muitas vezes o processamento de uma informação exige uma seqüência de transferência até que a tarefa seja terminada.

Exemplo: Adição de três valores (X e Y são dois registradores)

tempo do evento 1 - Faça X igual a zero e leia o primeiro

τ_1 : X < [0] valor de Y. Acontece então a primeira soma parcial.

Y < [A]

tempo do evento 2 - Carregue valor em X, leia o segundo valor de Y e então execute a segunda soma parcial.

τ_2 : X < X+Y

Y < [A]

tempo do evento 3 - Carregue o valor em X, leia o terceiro valor de Y e então execute a terceira soma.

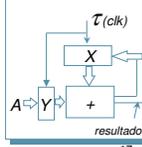
τ_3 : X < X+Y

Y < [A]

tempo do evento 4 - Carregue o resultado da soma em X e Y mantém-se inalterado.

τ_4 : X < X+Y

Y < [A]



17

Exemplo - Simple statements

Exemplo:

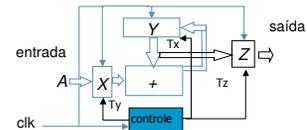
/* Programa que repete uma seqüência de adições de três sinais de entrada (a+b+c) e mostra o resultado

```
Sigad: CLRLD /* Clear acumulador Y, Z e carrega valor de A em X
ADD /* ADD 1o. valor, guarda resultado e lê novo valor
ADD /* ADD 2o. valor, guarda resultado e lê novo valor
ADD /* ADD 3o. valor, guarda resultado
DISP Sigad /* mostra resultado e repete cálculo em Z
```

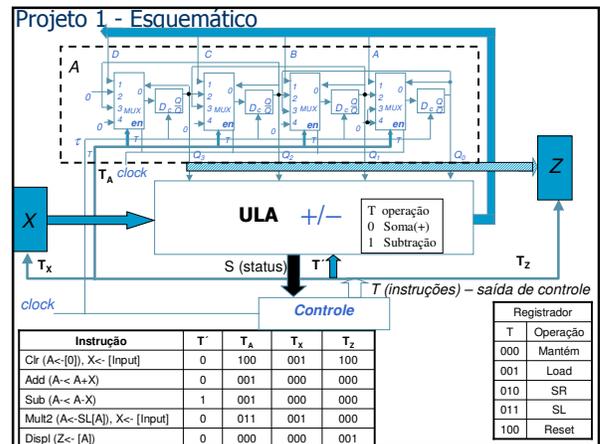
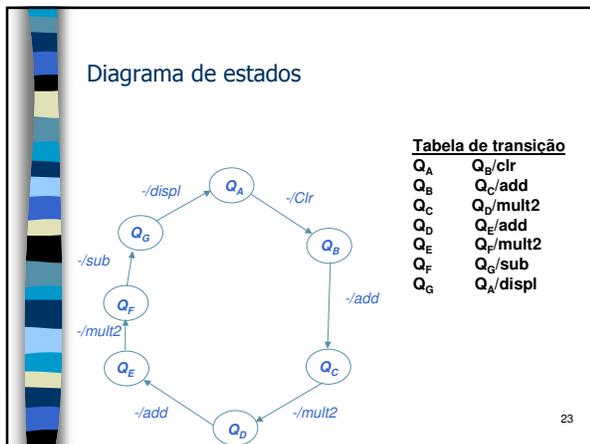
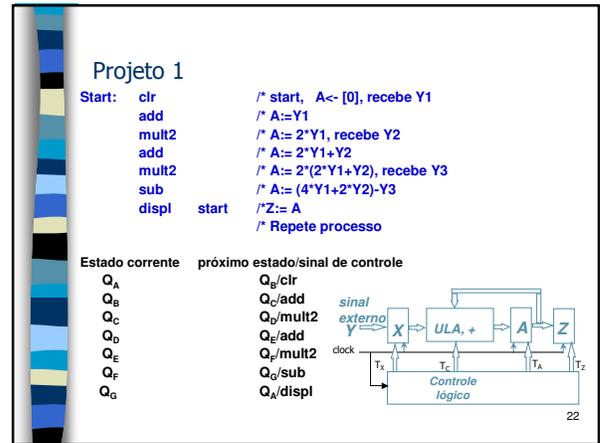
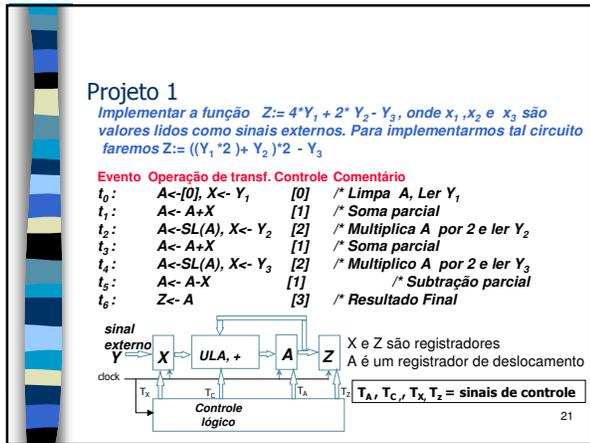
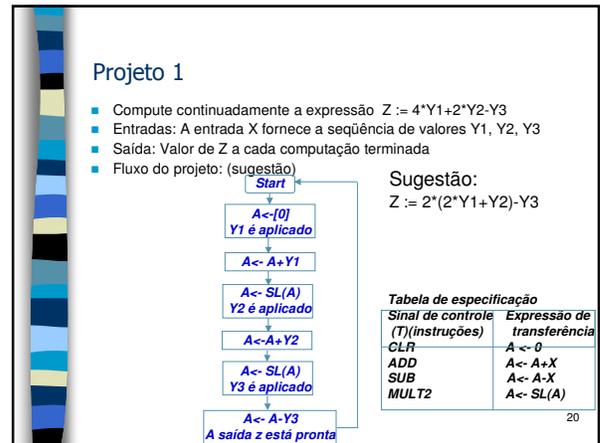
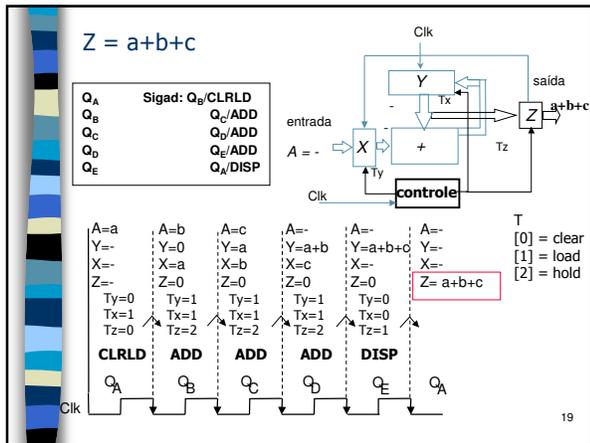
Como construir uma unidade de controle que implemente este programa?

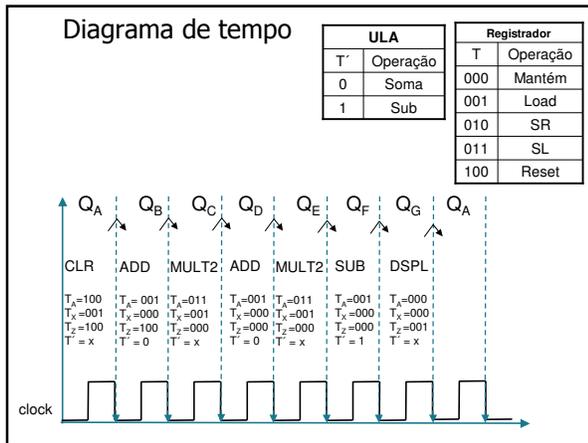
Estado Estado/saída

Q_A Sigad: Q₀/CLRLD
Q_B Q₀/ADD
Q_C Q₀/ADD
Q_D Q₀/ADD
Q_E Q_A/DISP



18





Implementação do controle

instruções

Estado presente	Próximo estado	FF 1	FF 2	FF 3	T _A	T _X	T _Z	T'	Inst
Y ₂ Y ₁ Y ₀	Y ₂₊₁ Y ₁₊₁ Y ₀₊₁	J ₂ K ₂	J ₁ K ₁	J ₀ K ₀	T ₂ T ₁ T ₀	T ₂ T ₁ T ₀	T ₂ T ₁ T ₀	T'	
0 0 0	0 0 1	0 X	0 X	1 X	1 0 0	0 0 1	1 0 0	X	CLR
0 0 1	0 1 0	0 X	1 X	X 1	0 0 1	0 0 0	0 0 0	0	ADD
0 1 0	0 1 1	0 X	X 0	1 X	0 1 1	0 0 1	0 0 0	X	MULT2
0 1 1	1 0 0	1 X	X 1	X 1	0 0 1	0 0 0	0 0 0	0	ADD
1 0 0	1 0 1	0 X	0 X	1 X	0 1 1	0 0 1	0 0 0	X	MULT2
1 0 1	1 1 0	X 0	1 X	X 1	0 0 1	0 0 0	0 0 0	1	SUB
1 1 0	0 0 0	X 1	X 1	0 X	0 0 0	0 0 0	0 0 1	X	DSPL
X X X	X X X	X X	X X	X X	X X X	X X X	X X X	x	x

* Implementação da máquina de estados usando Flip-Flop tipo JK

Implementação do circuito

Equações booleanas

$J_2 = y_1 y_0$

y ₂ \y ₁ y ₀	00	01	11	10
0	0	0	1	0
1	X	X	X	X

$K_2 = \bar{y}_1 y_0$

y ₂ \y ₁ y ₀	00	01	11	10
0	X	X	X	X
1	0	1	X	X

$J_1 = \bar{y}_2 y_0$

y ₂ \y ₁ y ₀	00	01	11	10
0	0	1	X	X
1	0	0	X	X

$K_1 = y_1 y_0$

y ₂ \y ₁ y ₀	00	01	11	10
0	X	X	1	0
1	X	X	X	X

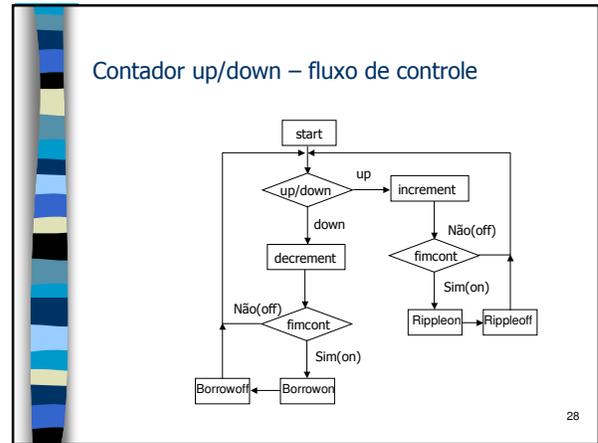
$J_0 = 1$

y ₂ \y ₁ y ₀	00	01	11	10
0	1	X	X	1
1	1	X	X	1

$K_0 = 1$

y ₂ \y ₁ y ₀	00	01	11	10
0	X	1	1	X
1	X	1	X	X

Implementar os T's?

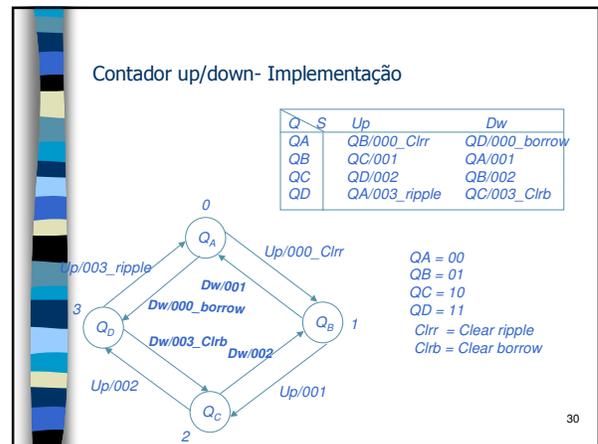


Projeto 2- Implementação de um contador síncrono UP/DOWN Módulo 4

- Conta em ordem crescente (up) 0, 1, 2, 3, 0, 1, ..., e decrescente 3, 2, 1, 0, 3, 2, ...
- Contador mostra ripple carry. Este sinal informa fim de contagem tanto na contagem crescente como na decrescente.
- Start: Clr case dir /* limpa contador
- Conte:


```

      {
      [up] increm /* incremente contador
      case fimcont /* chegou a três
      {
      [on] rippleon carryoff /* liga ripple carry
      [off] Conte
      }
      [down] decrem /* chegou a zero
      case fimcontd
      {
      [on] borrowon borrowoff /* liga ripple carry
      [off] conte
      }
      }
      
```
- carryoff: rippleoff Conte /* desligue ripple
- borrowoff: borrowoff Conte /* desliga borrow



Contador up/down - implementação

$up = \text{conta para cima}$
 $down = \text{conta para baixo}$
 $QA = 00$
 $QB = 01$
 $QC = 10$
 $QD = 11$

Saída presente		Próximo estado										
up/down	Qb	Qa	Qb'	Qa'	J2	K2	J1	K1	outb	outa	ripple borrow	
0	0	0	1	1	1	X	1	X	0	0	1	0
0	0	1	0	0	0	X	X	1	0	1	1	1
0	1	0	0	1	X	1	1	X	1	0	1	1
0	1	1	1	0	X	0	X	1	1	1	1	1
1	0	0	0	1	0	X	1	X	0	0	1	1
1	0	1	1	0	1	X	X	1	0	1	1	1
1	1	0	1	1	1	X	0	1	X	1	0	1
1	1	1	0	0	X	1	X	1	1	1	0	1

31

Contador up/down - implementação

$J_2 = ud \odot Qa$
 $K_2 = ud \odot Qa$
 $J_1 = 1$
 $K_1 = 1$

QbQa		ud			
Qb	Qa	00	01	11	10
0	0	1	0	X	X
1	0	0	1	X	X

QbQa		ud			
Qb	Qa	00	01	11	10
0	0	X	X	0	1
1	0	X	X	1	0

QbQa		ud			
Qb	Qa	00	01	11	10
0	0	1	X	X	1
1	0	1	X	X	1

QbQa		ud			
Qb	Qa	00	01	11	10
0	0	X	1	1	X
1	0	X	1	1	X

32

Contador up/down - implementação

33

Contador up/down - Diagrama de tempo

34

Contador up e down em VHDL

```

library ieee;
use ieee.std_logic_1164.all;

entity counter is
port (clk, clear, up_down : in std_logic;
      out : qc : out integer range 0 to 15);
architecture arch of conuter is
begin
process (clk)
variable cnt : integer 0 to 15;
end process;
end arch;
end entity;

begin
if (clear = '1') then
qc <= 0;
elsif (clk' event and clk = '1') then
if (up_down = '1') then
cnt := cnt+1; -- conta up
else
cnt := cnt -1; -- conta down
end if;
end if;
qc <= cnt;
end process;
end begin;

```

35

Aplicações - Interface de Entrada - Teclado

36

Aplicações - Interface de Saída - Impressora

