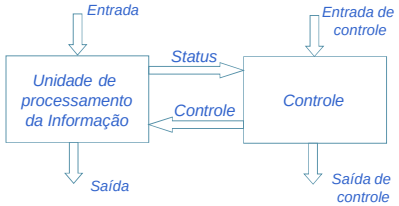
Sistemas Digitais

Aula 20

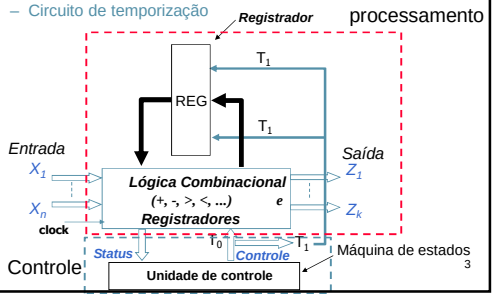
Projeto de sistemas digitais

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



Unidade de Procesamento da Informação

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



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, ..., X_k, Q_1,.....,Q_m, T)

  X - Valor do sinal de entrada

  Q - estado atual

  T - sinal de controle
- Equação de saída
  - **$Z_i := G_j$ (X_1, ..., X_k , Q_1, ..., Q_m , T)**

Unidade de Controle - status

- É um sinal de entrada que informa a FSM alguma condição de teste
- O Status permite alterar a seqüência de computação
  - $S_i := X_i$ (X_1 ,....., X_k , Q_1 , ... 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, XGEQ, XLT}

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

Unidade de Controle

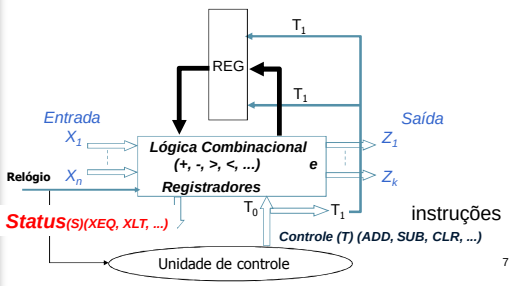
Sinal de controle ("Instruções")

- O sinal de controle (T) é usado para definir:
  - Transferência de informação realizada pela unidade de processamento.
  - Valor do sinal de saída gera T (instrução)
  - 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

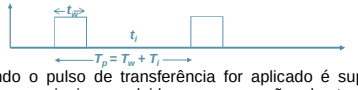
Sinais de controle/status - exemplo

registrador (shift left, shift right, clear, ...)



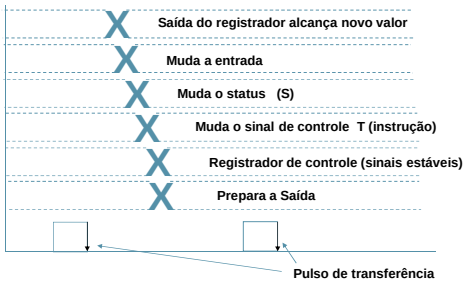
Considerações de tempo

- **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.

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



Unidade de controle

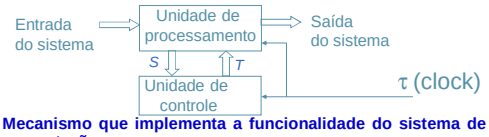
Algoritmo para computar F(x) = Y

- 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**

Programa de hardware



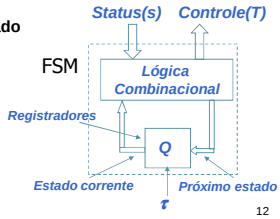
- **Mecanismo que implementa a funcionalidade do sistema de computação**
  - **A unidade de processamento foi definida e as operações e testes completamente descritas por uma tabela de especificações.**
    - **Operações T (instruções)**
    - **Testes S**
  - **O sistema é síncrono. O pulso τ faz com 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.**

Modelo da unidade de controle

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

- **Equação de T (Instruções)**
  T := F(S,Q)
- **Equação de próximo estado**
  Q <- G(S,Q)

Status
{ [s1, ..., sd]}
Controle
{ [t1, ..., tm]}
Estado
{ [q1, ..., qn]}



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 (instrução)**
  - **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**

Linguagem de hardware - exemplo

- <STATUS >
  - Indica o resultado de uma comparação entre dois vetores

Exemplo:

a. Forma mneumônica
- GT - maior que
- LT - menor que
- EQU - igual

b. Forma de vetor
- GT = [1,0,1]
- LT = [1,1,1]
- EQU = [0,1,1]

- <CONTROLE > ou <Instruções = T>

Exemplo:
- NOP - No Operation [0,0,0]
- CLR – Clear [0,0,1]
- ADD - Soma [0,1,0]
- SUB – Subtração [0,1,1]

Exemplo – Declarações simples

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

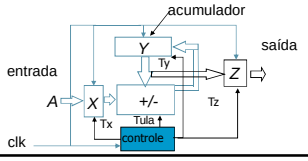
Exemplo 01:

```
/* Programa que a função (a+b+c) e mostra o resultado (z = a=b-c)
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
         SUB            /*SUB 3o .valor,  guarda resultado
         DISP    Sigad  /*mostra resultado e repete cálculo em Z
```

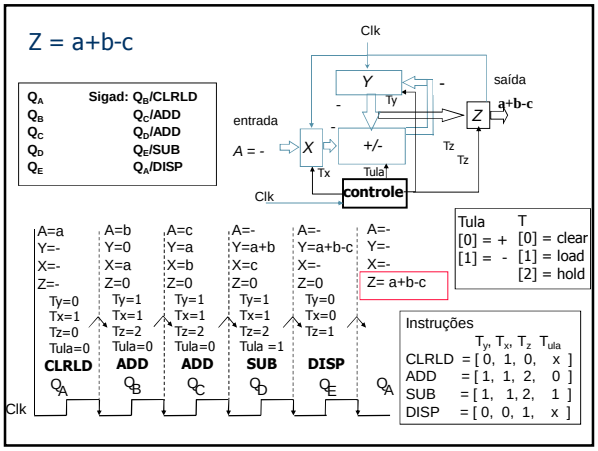
Como construir uma *unidade de controle* que implemente este programa?

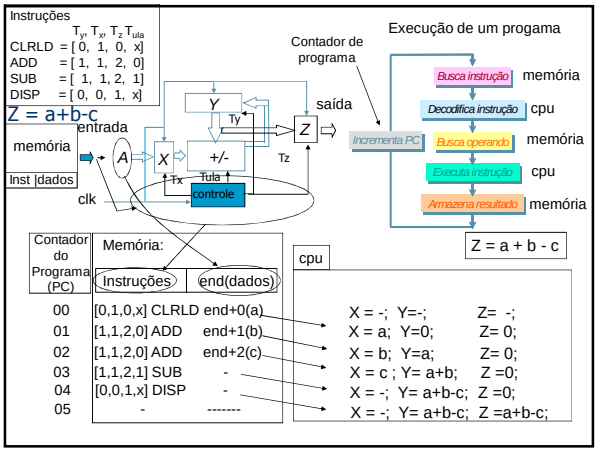
Estado Estado/saída

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



Z = a+b-c



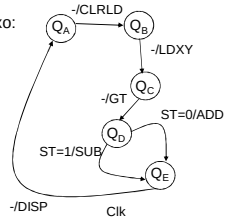


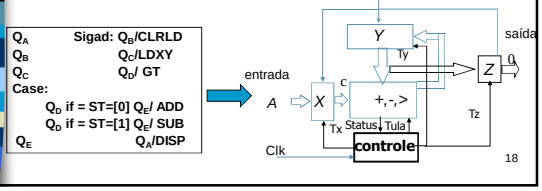
Exemplo 02:

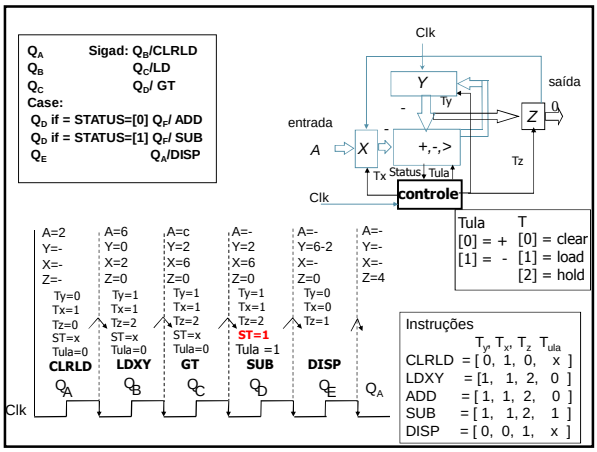
- Implementar o algortimo abaixo:

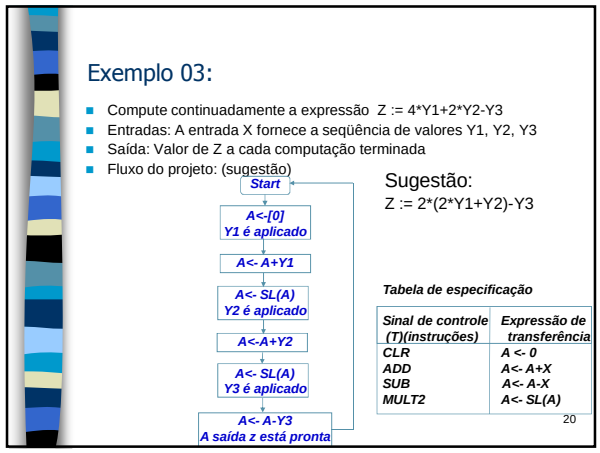
```
y := 2
x := 6
If(x>y) then
   z := x+y
else
   e:= x-y
```

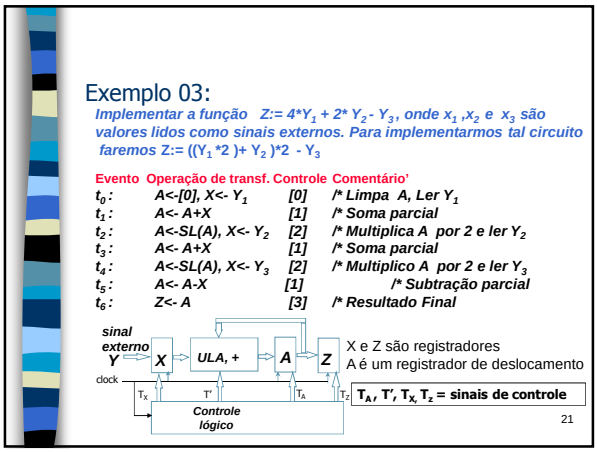
* STATUS = ST

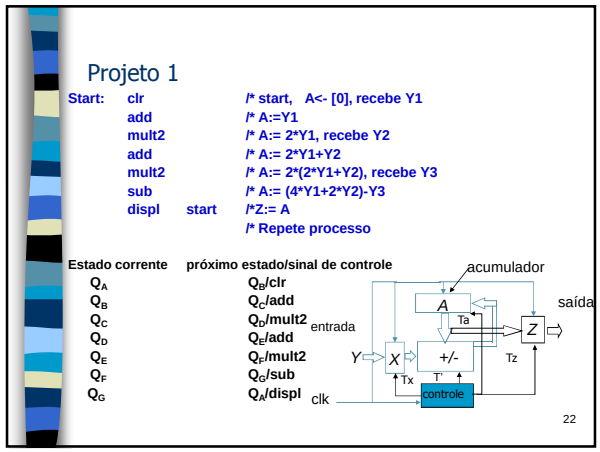





Q_A Sigad: Q_B/CLRLD
Q_B Q_C/LD
Q_C Q_D/ GT
Case:
Q_D if = STATUS=[0] Q_E/ ADD
Q_D if = STATUS=[1] Q_F/ SUB
Q_E Q_A/DISP
Clk
Y
Ty
saída
Z
entrada
A
X
+,-,>
Tz
Tx
Status
Tula
controle
Clk
Tula T
[0] = + [0] = clear
[1] = - [1] = load
[2] = hold
A=2 Y=- X=- Z=-
Ty=0 Tx=1 Tz=0 ST=x Tula=0
CLRLD
A=6 Y=0 X=2 Z=0
Ty=1 Tx=1 Tz=2 ST=x Tula=0
LDXY
A=c Y=2 X=6 Z=0
Ty=1 Tx=1 Tz=2 ST=x Tula=0
GT
A=- Y=2 X=6 Z=0
Ty=1 Tx=1 Tz=2 ST=1 Tula =1
SUB
A=- Y=6-2 X=- Z=0
Ty=0 Tx=0 Tz=1
DISP
A=- Y=- X=- Z=4
Instruções
T_y, T_x, T_z T_{ula}
CLRLD = [0, 1, 0, x]
LDXY = [1, 1, 2, 0]
ADD = [1, 1, 2, 0]
SUB = [1, 1, 2, 1]
DISP = [0, 0, 1, x]
Clk


Exemplo 03:
Compute continuadamente a expressão Z := 4*Y1+2*Y2-Y3
Entradas: A entrada X fornece a seqüência de valores Y1, Y2, Y3
Saída: Valor de Z a cada computação terminada
Fluxo do projeto: (sugestão)
Start
A<-[0]
Y1 é aplicado
A<- A+Y1
A<- SL(A)
Y2 é aplicado
A<-A+Y2
A<- SL(A)
Y3 é aplicado
A<- A-Y3
A saída z está pronta
Sugestão:
Z := 2*(2*Y1+Y2)-Y3
Tabela de especificação
Sinal de controle (T)(instruções)
Expressão de transferência
CLR
A <- 0
ADD
A<- A+X
SUB
A<- A-X
MULT2
A<- SL(A)
20


Exemplo 03:
Implementar a função Z:= 4*Y1 + 2* Y2 - Y3, onde x1 ,x2 e x3 são valores lidos como sinais externos. Para implementarmos tal circuito faremos Z:= ((Y1 *2)+ Y2)*2 - Y3
Evento Operação de transf. Controle Comentário'
t0 : A<-[0], X<- Y1 [0] /* Limpa A, Ler Y1
t1 : A<- A+X [1] /* Soma parcial
t2 : A<-SL(A), X<- Y2 [2] /* Multiplica A por 2 e ler Y2
t3 : A<- A+X [1] /* Soma parcial
t4 : A<-SL(A), X<- Y3 [2] /* Multiplico A por 2 e ler Y3
t5 : A<- A-X [1] /* Subtração parcial
t6 : Z<- A [3] /* Resultado Final
sinal externo Y
X
ULA, +
A
Z
clock
Controle lógico
X e Z são registradores
A é um registrador de deslocamento
TA , T', Tx, Tz = sinais de controle
21


Projeto 1
Start: clr /* start, A<- [0], recebe Y1
add /* A:=Y1
mult2 /* A:= 2*Y1, recebe Y2
add /* A:= 2*Y1+Y2
mult2 /* A:= 2*(2*Y1+Y2), recebe Y3
sub /* A:= (4*Y1+2*Y2)-Y3
displ start /*Z:= A
/* Repete processo
Estado corrente
próximo estado/sinal de controle
Q_A
Q_B
Q_C
Q_D
Q_E
Q_F
Q_G
Q_B/clr
Q_C/add
Q_D/mult2
Q_E/add
Q_F/mult2
Q_G/sub
Q_A/displ
acumulador
saída
entrada
clk
controle
22

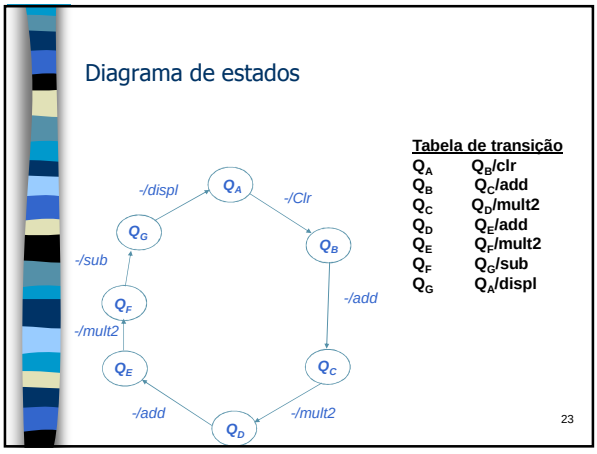


Diagrama de estados
-/displ
-/Clr
-/add
-/sub
-/mult2
-/add
-/mult2
Tabela de transição
Q_A Q_B/clr
Q_B Q_C/add
Q_C Q_D/mult2
Q_D Q_E/add
Q_E Q_F/mult2
Q_F Q_G/sub
Q_G Q_A/displ
23

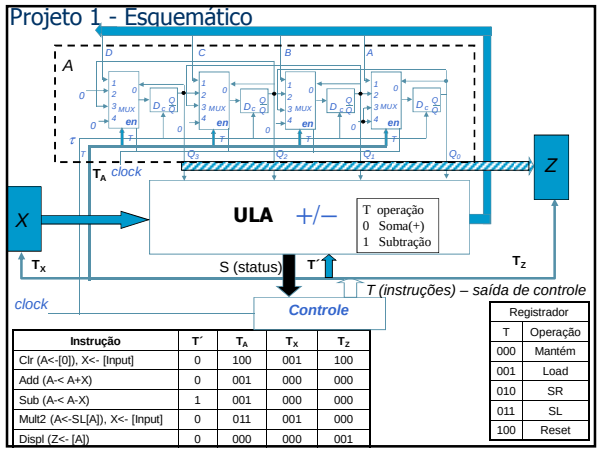
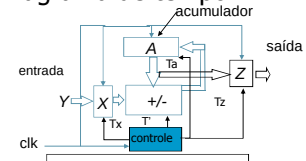

Projeto 1 - Esquemático
A
clock
ULA +/−
T operação
0 Soma(+)
1 Subtração
X
Z
S (status)
T´
T (instruções) – saída de controle
clock
Controle
Registrador
T Operação
000 Mantém
001 Load
010 SR
011 SL
100 Reset
Instrução T´ TA TX TZ
Clr (A<-[0]), X<- [Input] 0 100 001 100
Add (A-< A+X) 0 001 000 000
Sub (A-< A-X) 1 001 000 000
Mult2 (A<-SL[A]), X<- [Input] 0 011 001 000
Displ (Z<- [A]) 0 000 000 001

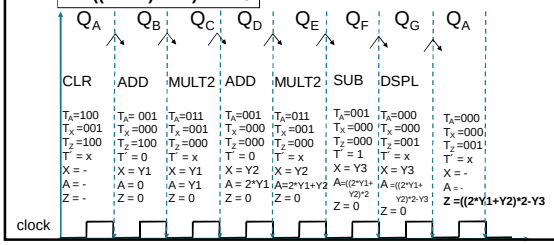
Diagrama de tempo



ULA	
T´	Operação
0	Soma
1	Sub

Registrador	
T	Operação
000	Mantém
001	Load
010	SR
011	SL
100	Reset

Z:= ((Y1 *2)+ Y2)*2 - Y3



Implementação do controle

instruções

Estado presente	Próximo estado	FF 1	FF 2	FF 3	T_A	T_X	T_Z	T´	
$y_2\ y_1\ y_0$	$y_{2+1}\ y_{1+1}\ y_{0+1}$	$J_2\ K_2$	$J_1\ K_1$	$J_0\ K_0$	$T_2\ T_1\ T_0$	$T_2\ T_1\ T_0$	$T_2\ T_1\ T_0$	T´	Inst
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_1y_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_1y_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_1y_0	00	01	11	10
0	0	1	X	X
1	0	0	X	X

$K_1 = y_1 y_0$

y_2 \ y_1y_0	00	01	11	10
0	X	X	1	0
1	X	X	X	X

$J_0 = 1$

y_2 \ y_1y_0	00	01	11	10
0	1	X	X	1
1	1	X	X	1

$K_0 = 1$

y_2 \ y_1y_0	00	01	11	10
0	X	1	1	X
1	X	1	X	X

Implementar os

T´s ?