Universidade Federal de Santa Catarina
Centro Tecnológico
Departamento de Informática e Estatística
Curso de Graduação em Ciências da Computação

Sistemas Digitais

INE 5406

Aula 3-T

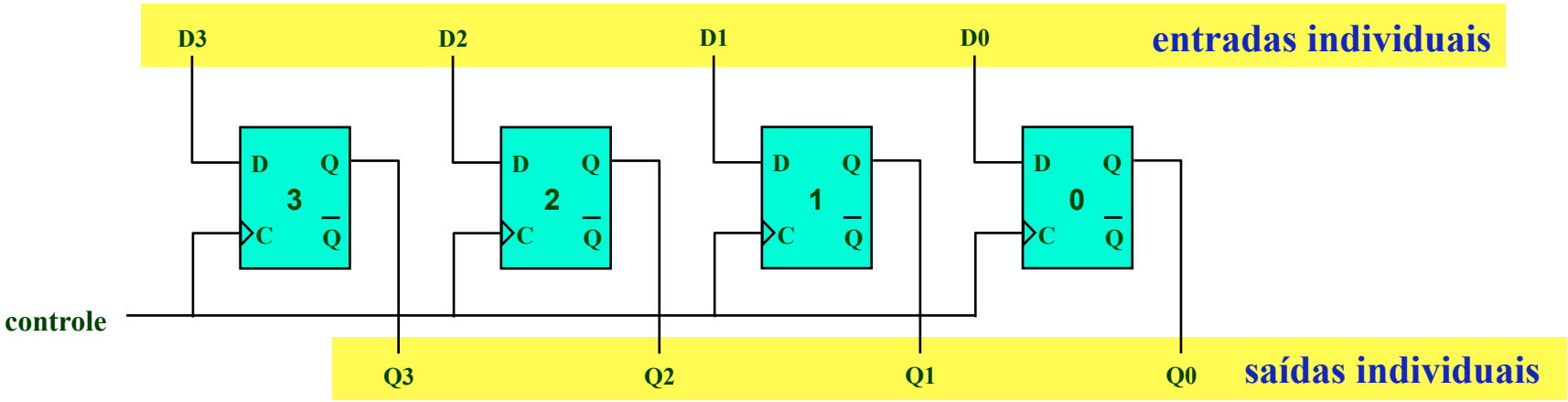
1. **Projeto de Unidade Lógico-Aritmética (ULA). Funcionamento e características temporais de registradores. Cadenciamento com sinal de relógio (*clock*). Registradores de uso específico. Exploração de paralelismo na ULA.**

Prof. José Luís Güntzel
guntzel@inf.ufsc.br

www.inf.ufsc.br/~guntzel/ine5406/ine5406.html

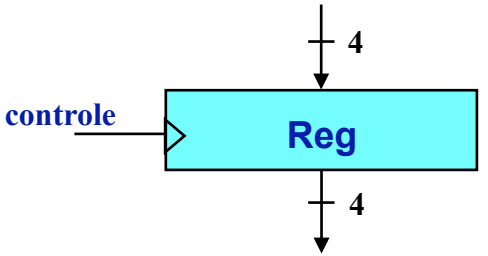
1. Projeto de Unidade Lógico-Aritmética

Registrador com Carga Paralela



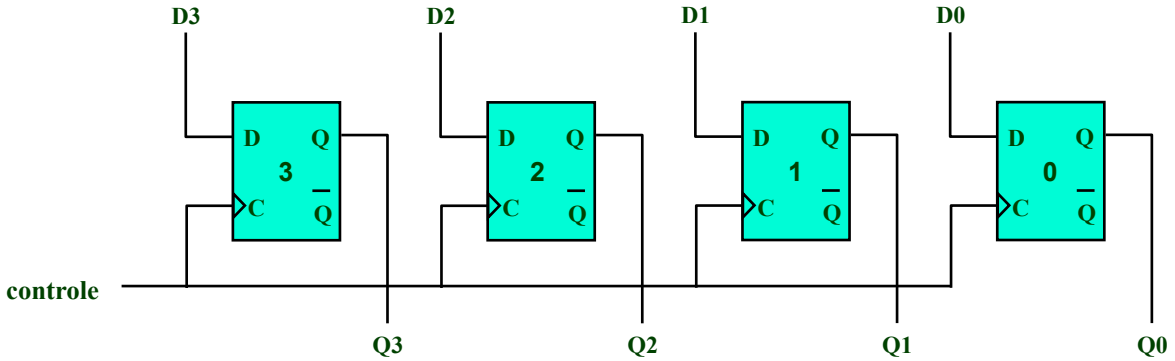
- Existe um FF para cada bit a ser armazenado
- Todos os FFs "obedecem" a um mesmo sinal de controle
- O termo "carga paralela" refere-se ao fato de existir uma entrada para cada bit, de modo que é possível carregar simultaneamente todos os bits do dado

Símbolo no nível RT

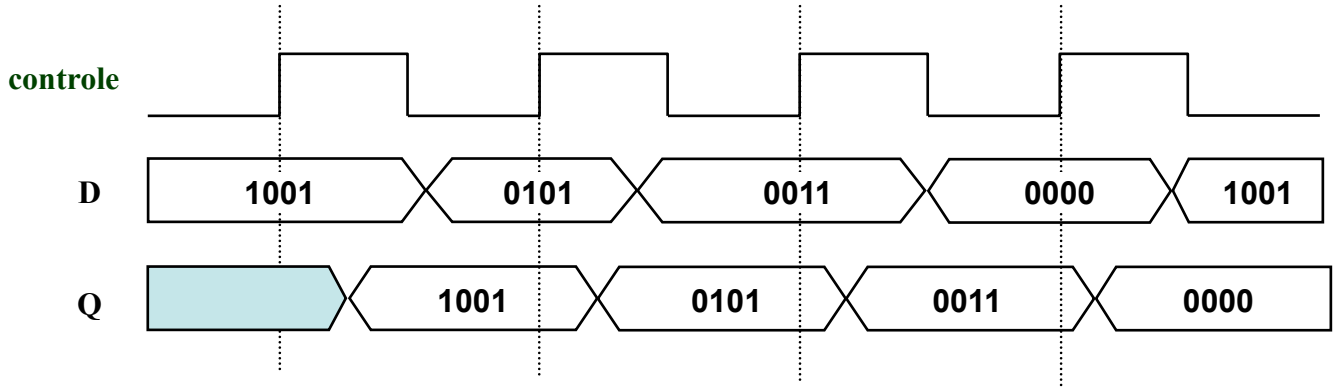


1. Projeto de Unidade Lógico-Aritmética

Registrador com Carga Paralela



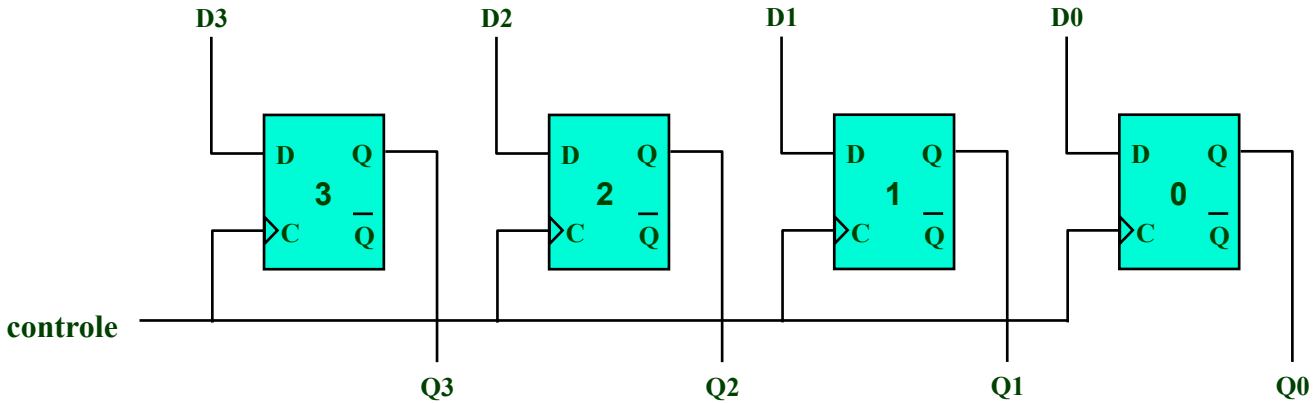
Exemplo de funcionamento (assumindo que tsu, th e td são satisfeitos)



A cada borda de subida de "controle" um novo dado é amostrado e fica armazenado no registrador (até a próxima borda de subida de "controle")

1. Projeto de Unidade Lógico-Aritmética

Registrador com Carga Paralela



- Peculiaridade: a cada borda de subida de "controle" um novo dado é amostrado, **mesmo que não se queira amostrar um dado novo…**
- Porém, às vezes pode ser necessário que o registrador obedeça a um sinal de controle sincronizado com o sinal de relógio. (Solução na próxima transparência…)

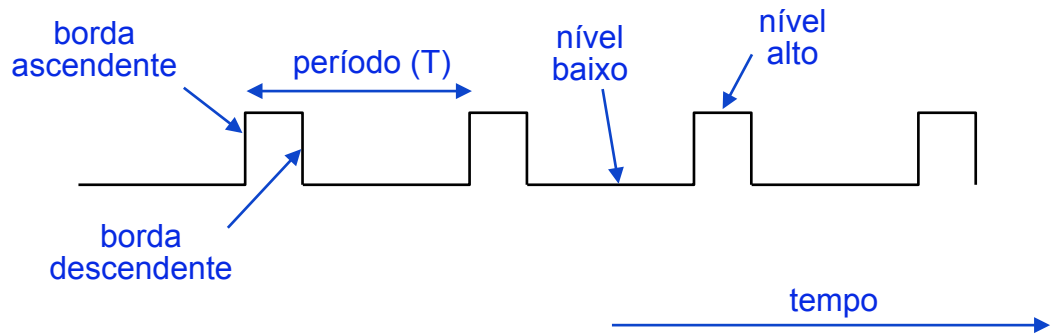
1. Projeto de Unidade Lógico-Aritmética

Cadenciamento de Sistemas Digitais

- A maior parte dos sistemas digitais são sincronizados por um sinal monótono denominado **relógio** (ou ***clock***, em inglês).
- Sistemas digitais cadenciados por sinal de relógio são denominados **síncronos.**
- No projeto de sistemas digitais síncronos, registradores são utilizados.

1. Projeto de Unidade Lógico-Aritmética

Sinal de Relógio (ou *Clock*)



Nomenclatura

borda ascendente =
borda de subida =
borda positiva =
flanco de subida etc

borda descendente =
borda de descida =
borda negativa =
flanco de descida etc

Características:

- **Período** (T): tempo entre duas bordas consecutivas de mesmo tipo. Medido em submúltiplos do segundo (ms, µs, **ns**)
- **Freqüência**: **f = 1/T**, medida em múltiplos do hertz (kHz, MHz, **GHz**)
- ***Duty cycle***: T1/T x 100 %, onde T1 é o tempo entre uma borda ascendente e a borda descendente que a segue.

1. Projeto de Unidade Lógico-Aritmética

Estimativa do Período do Relógio

Exemplo 1: Estime o período do relógio para um circuito combinacional cujo atraso crítico é 10 ns.

Preliminares:

1 ns (1 nanossegundo) = 1×10^{-9} s

$T = 1/f \Rightarrow f = 1/T$

1/1s = 1 Hz

Cálculo:

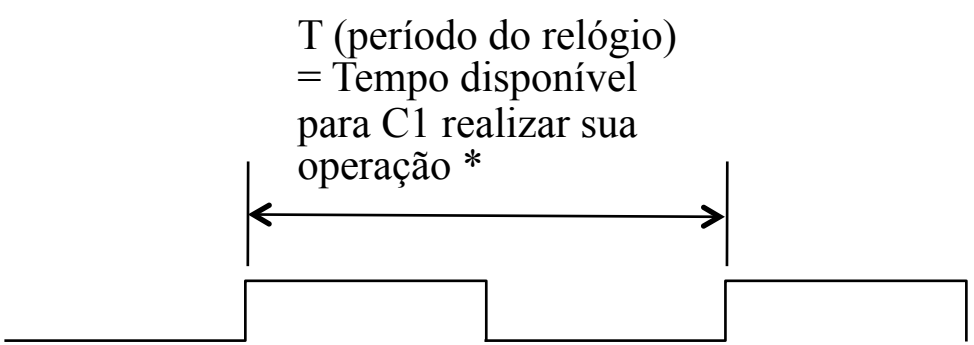
$f = 1/ (10 \times 10^{-9})$ Hz = $0{,}1 \times 10^{+9}$ Hz = $100 \times 10^{+6}$ Hz =

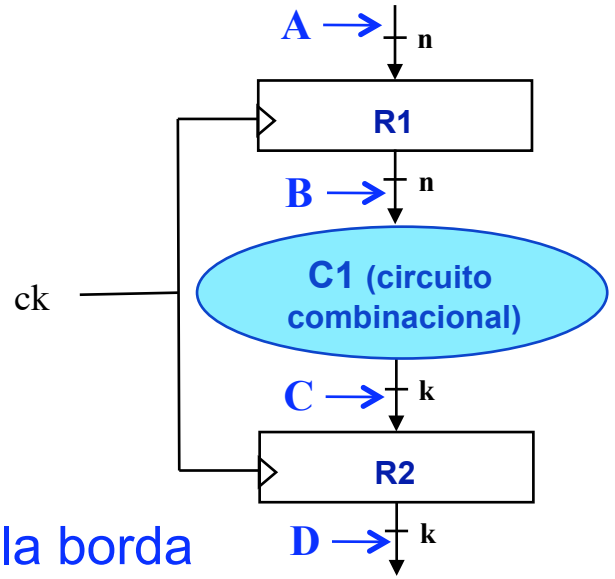
= 100 MHz

1. Projeto de Unidade Lógico-Aritmética

Cadenciamento de Sistemas Digitais

Registradores são usados para criar "barreiras temporais" que isolam os circuitos combinacionais





* (obs) :

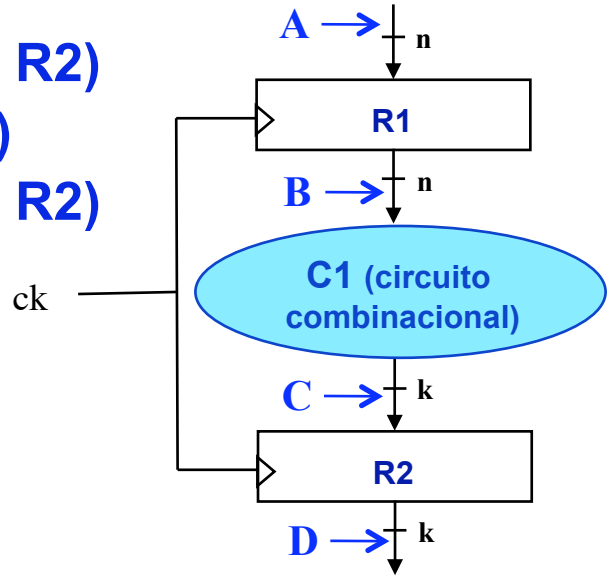
1. Supondo que R1 e R2 sejam disparados pela borda de subida de ck.
2. Aproximação grotesca; falta considerar as características temporais dos registradores R1 e R2.

1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{c1} = 2ns$ (atraso crítico (máximo) de C1)

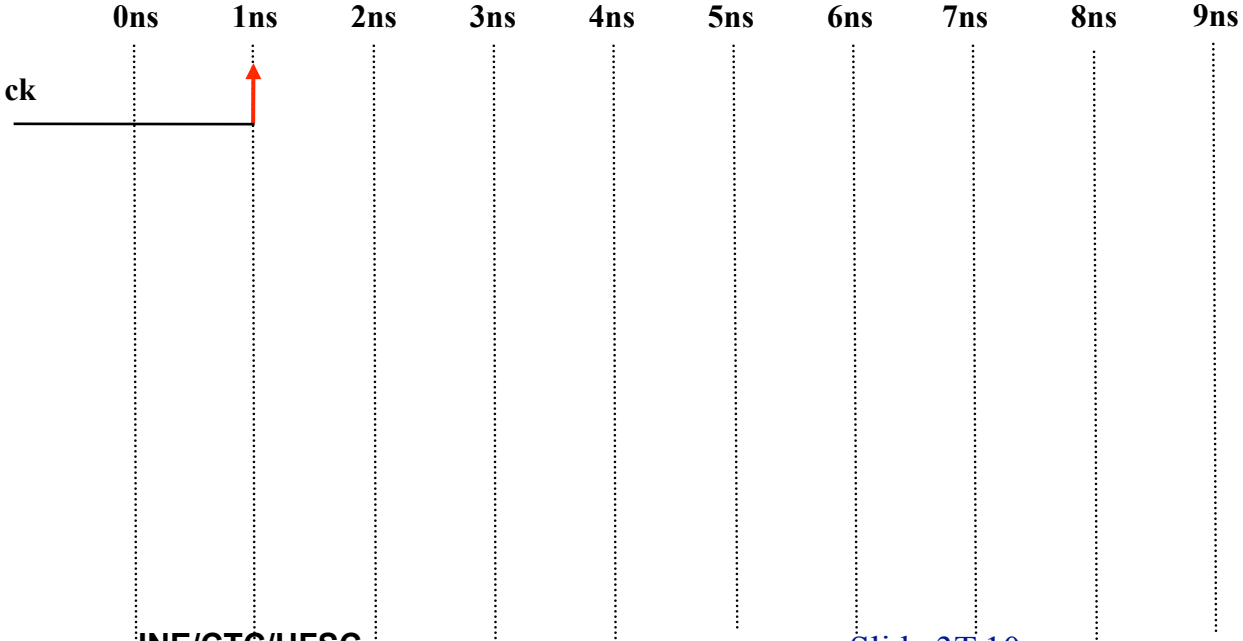


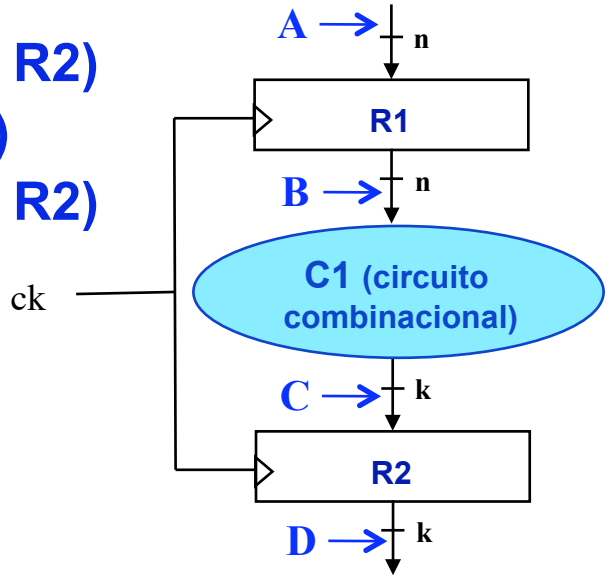
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{c1} = 2ns$ (atraso crítico (máximo) de C1)



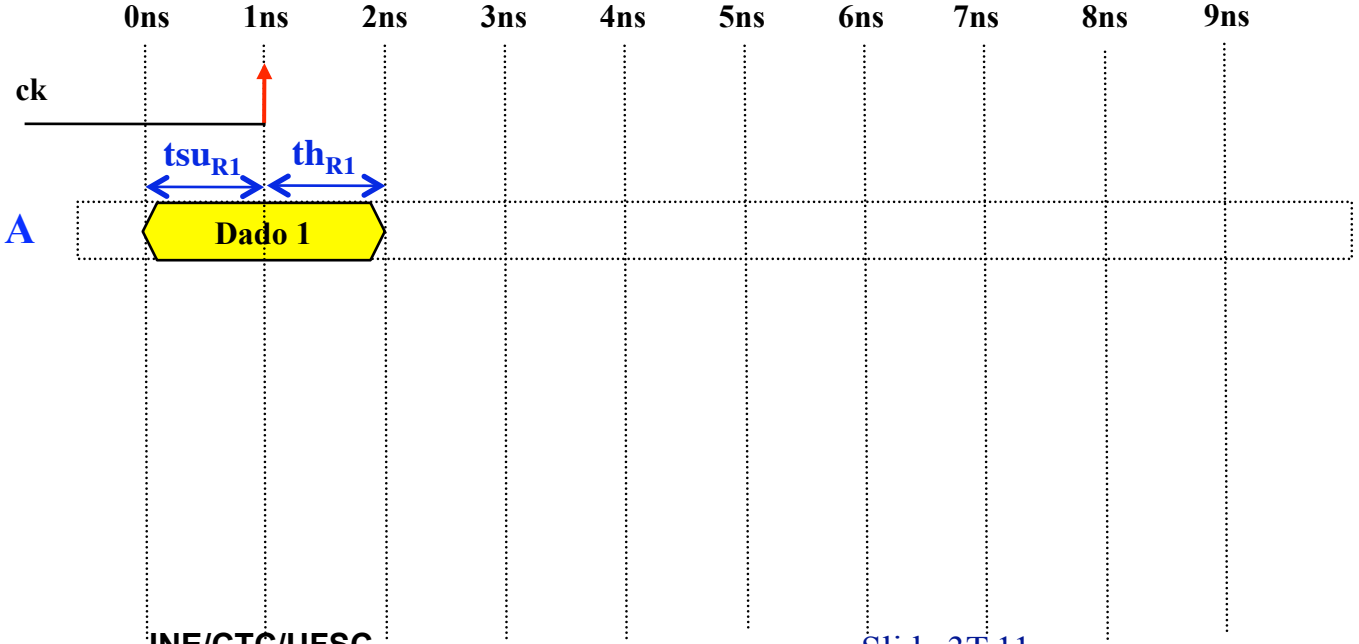


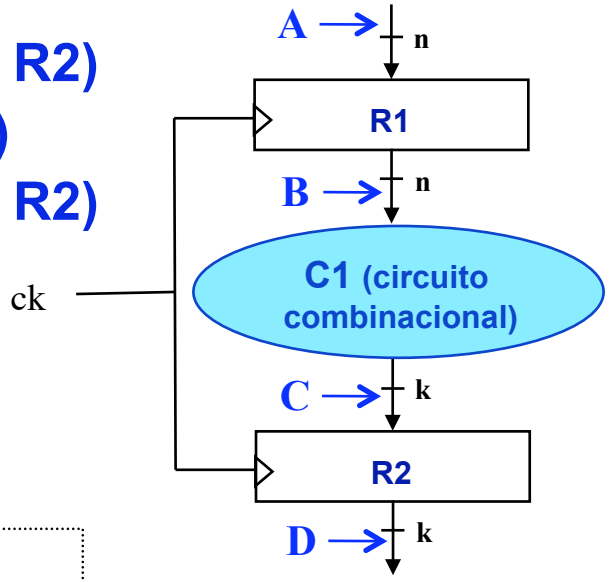
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{c1} = 2ns$ (atraso crítico (máximo) de C1)



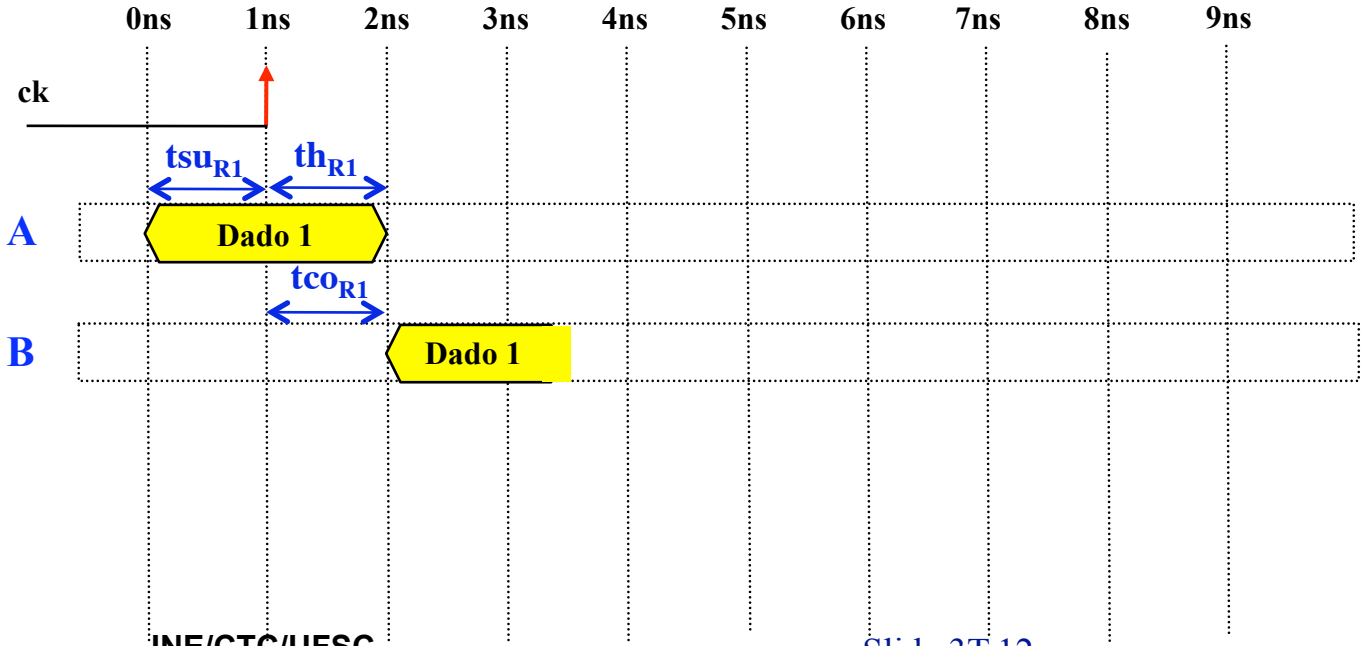


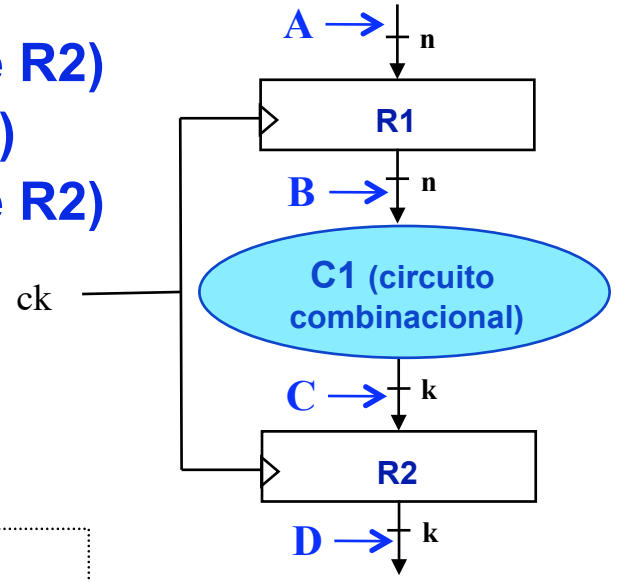
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{C1} = 2ns$ (atraso crítico (máximo) de C1)



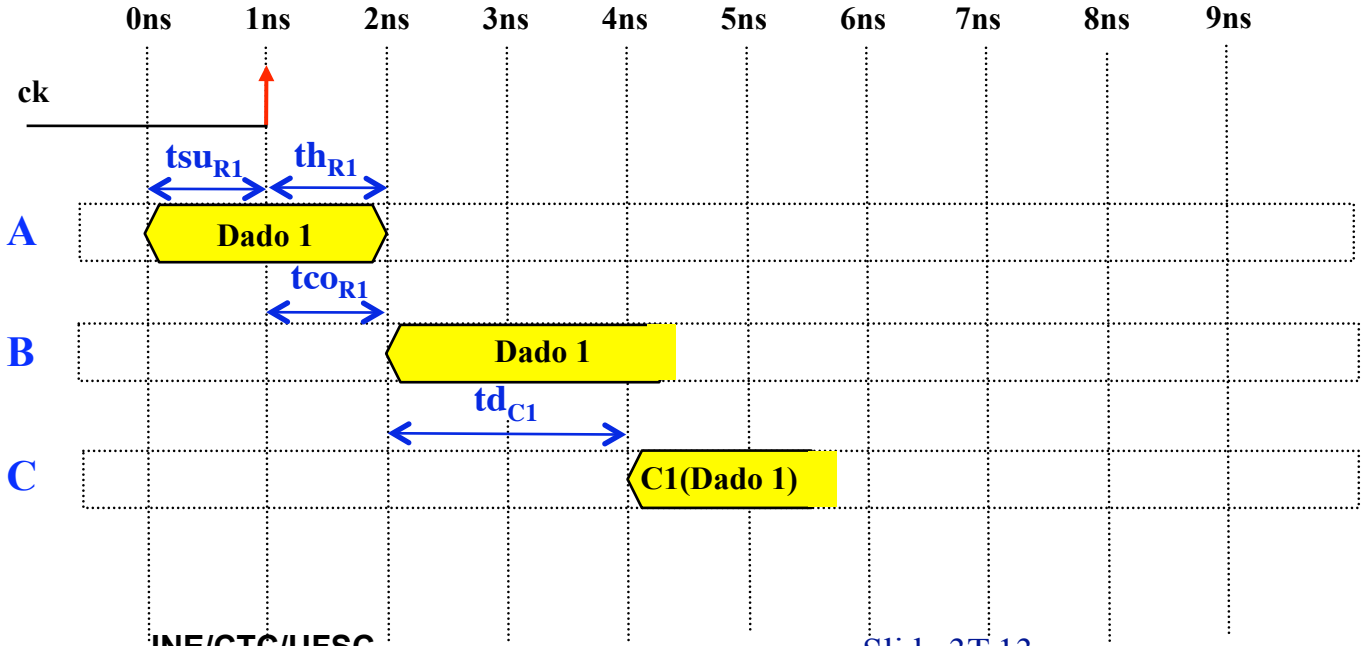


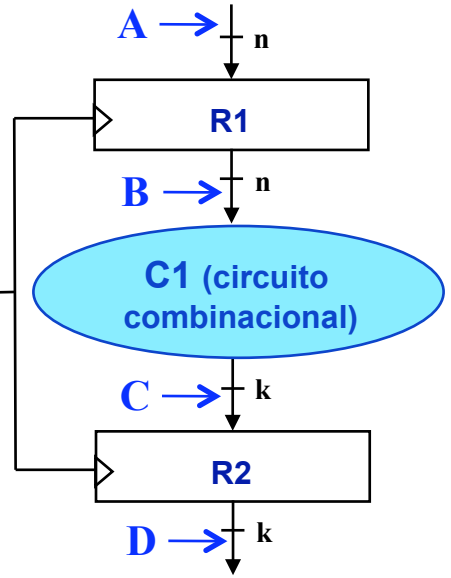
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{C1} = 2ns$ (atraso crítico (máximo) de C1)



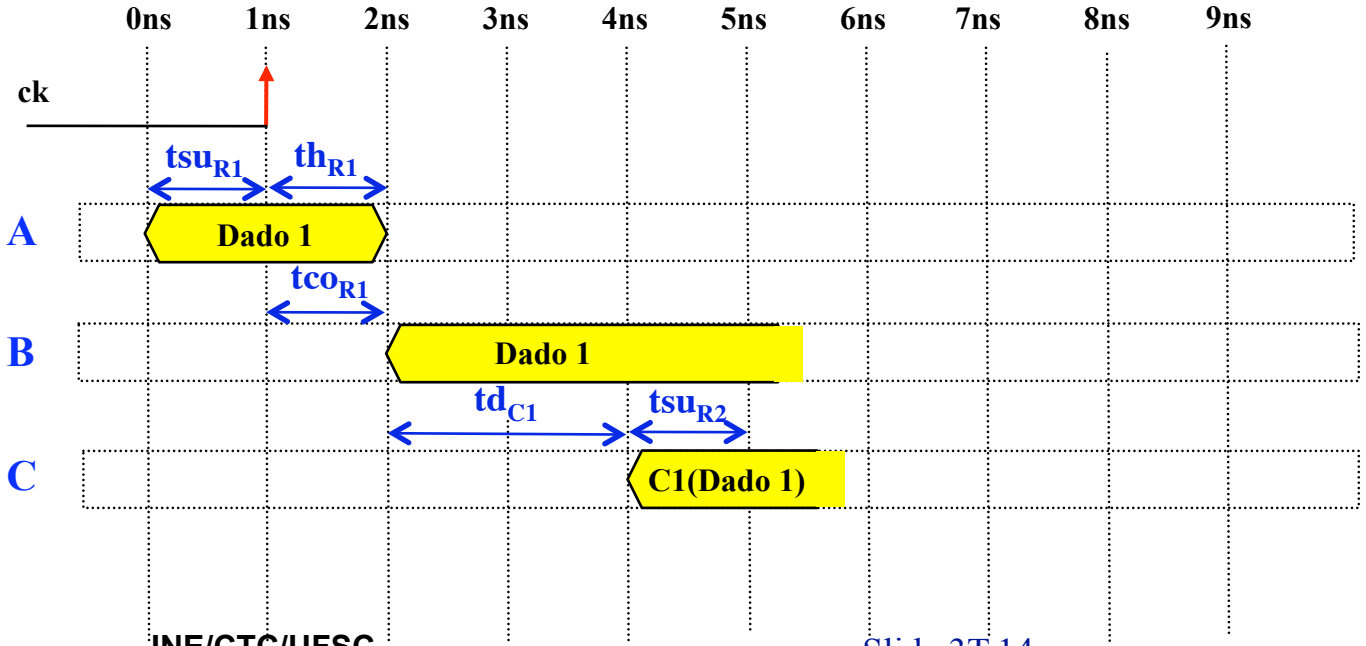


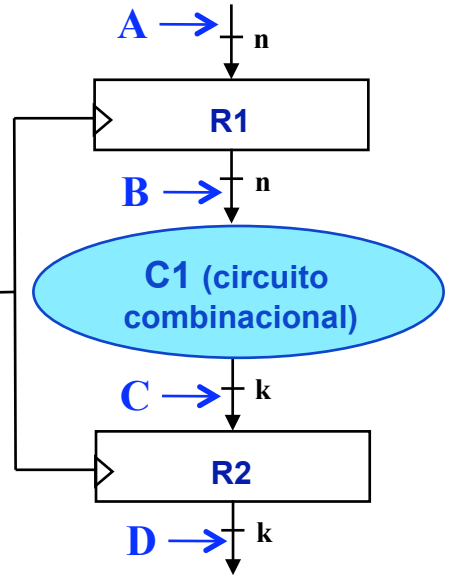
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{C1} = 2ns$ (atraso crítico (máximo) de C1)



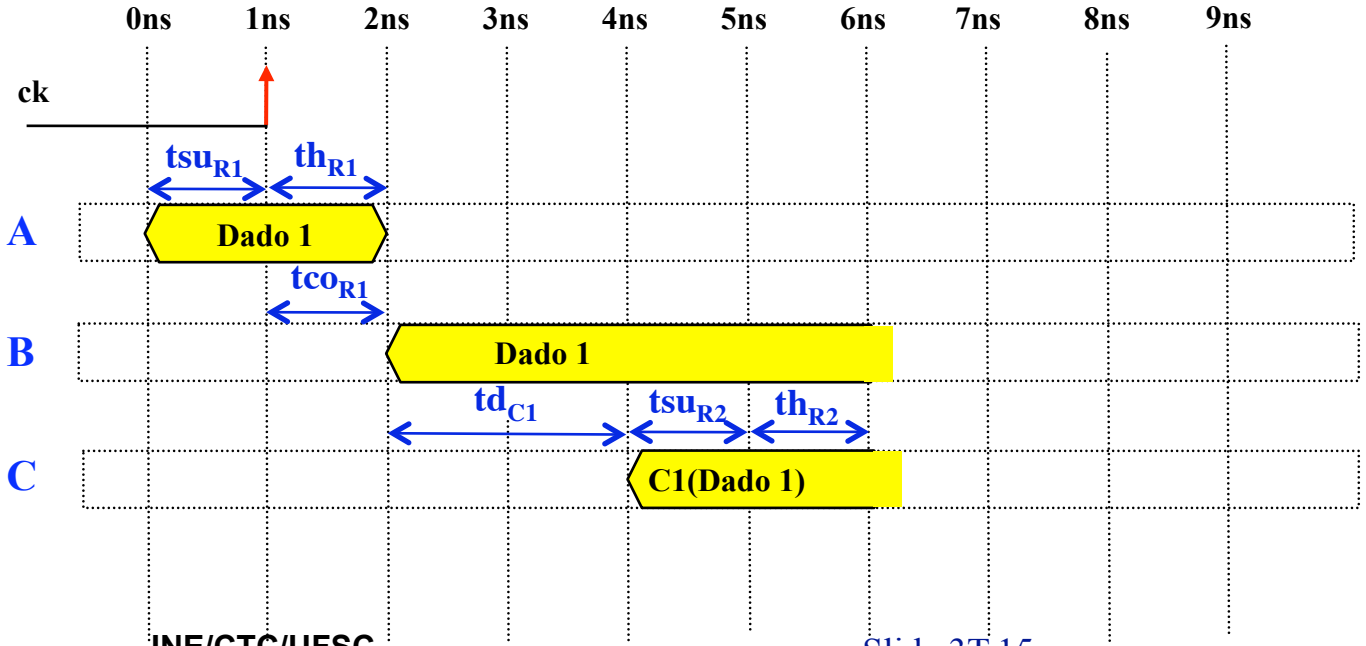


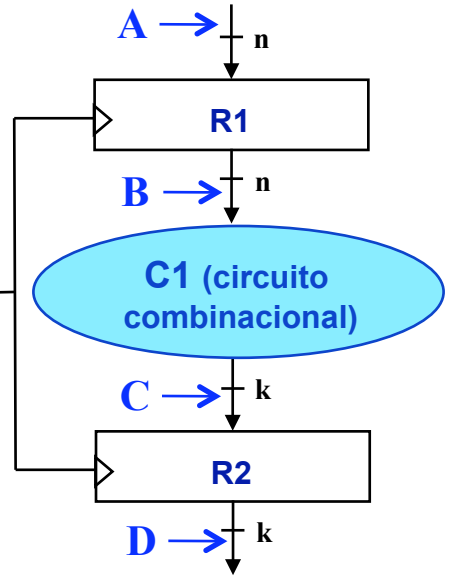
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{C1} = 2ns$ (atraso crítico (máximo) de C1)



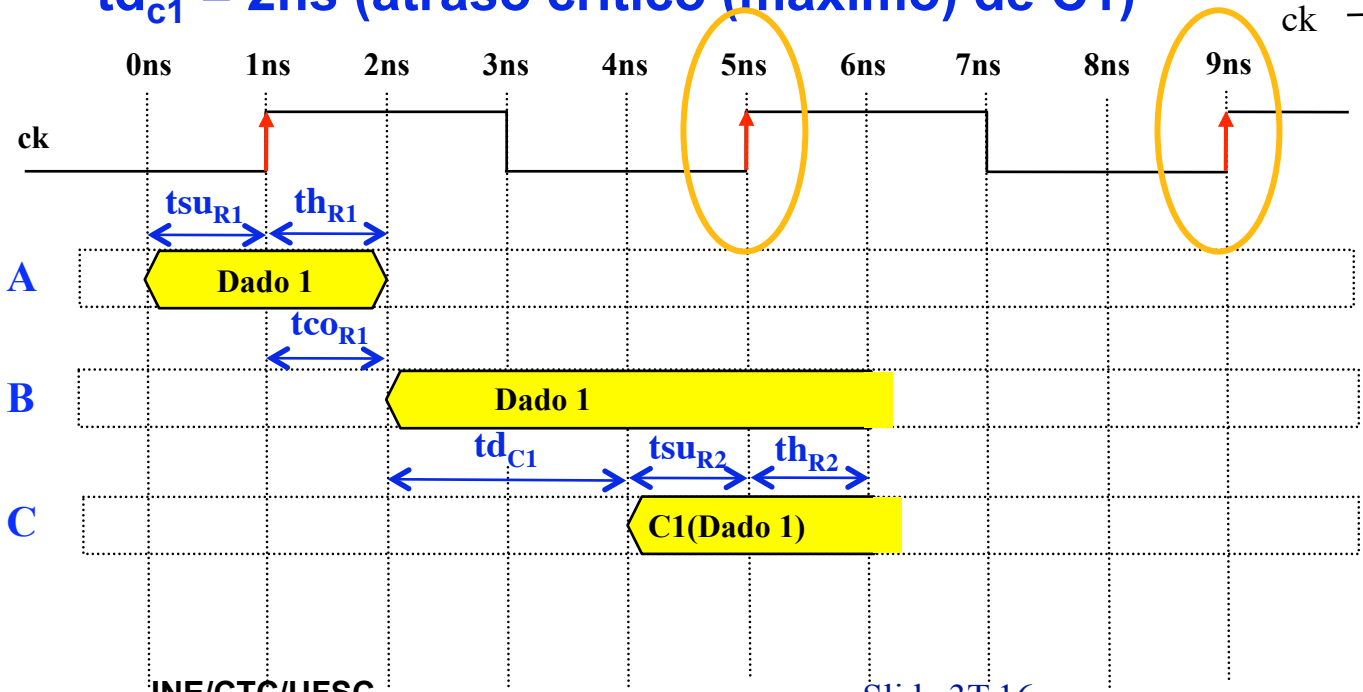


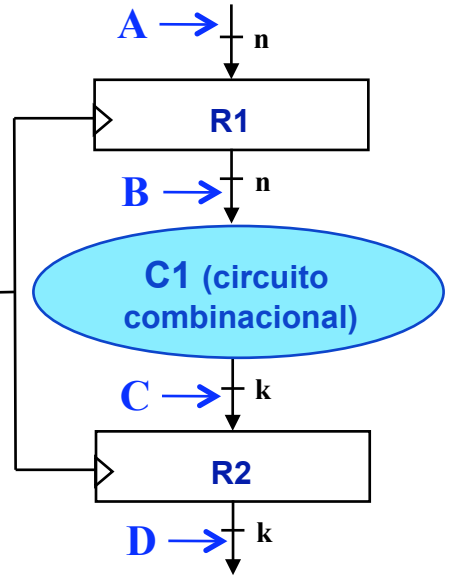
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{C1} = 2ns$ (atraso crítico (máximo) de C1)





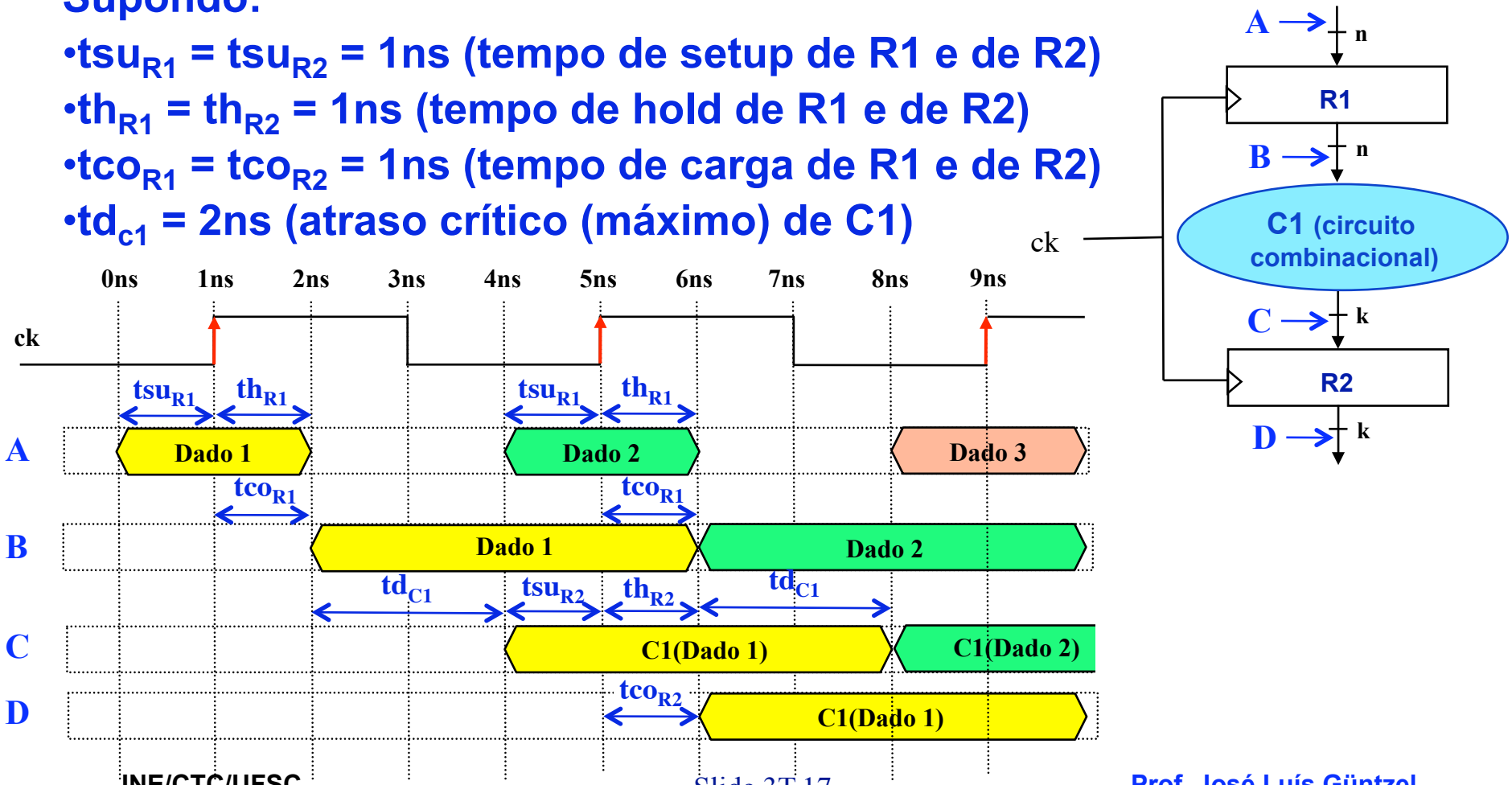
Período do relógio = 4ns

1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{c1} = 2ns$ (atraso crítico (máximo) de C1)

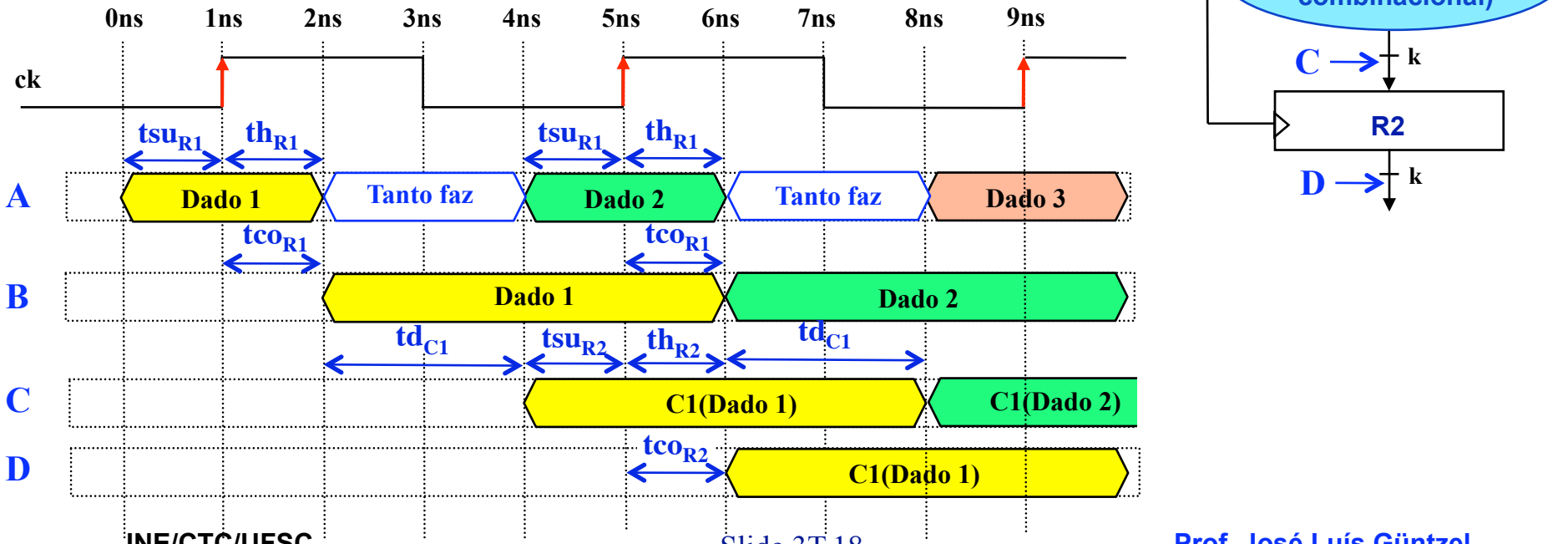


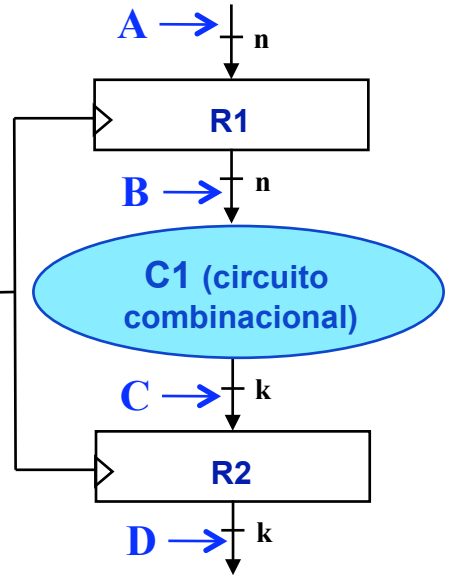
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Supondo:

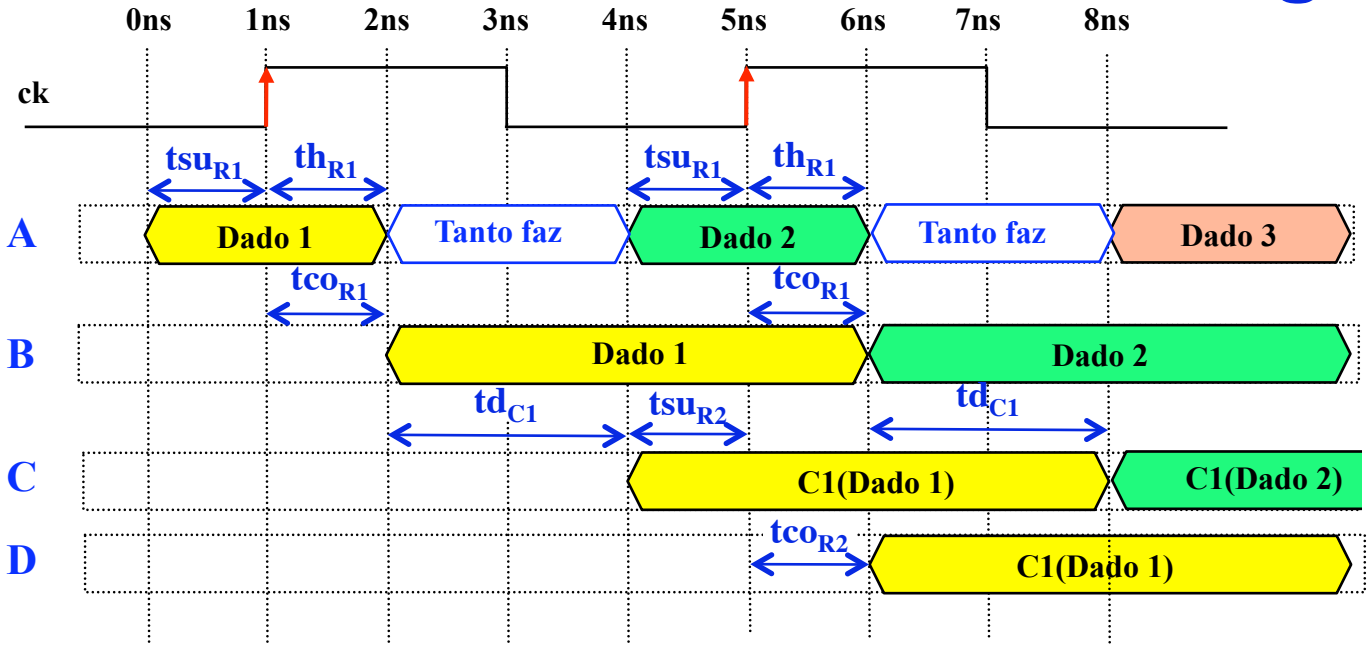
- $tsu_{R1} = tsu_{R2} = 1ns$ (tempo de setup de R1 e de R2)
- $th_{R1} = th_{R2} = 1ns$ (tempo de hold de R1 e de R2)
- $tco_{R1} = tco_{R2} = 1ns$ (tempo de carga de R1 e de R2)
- $td_{c1} = 2ns$ (atraso crítico (máximo) de C1)

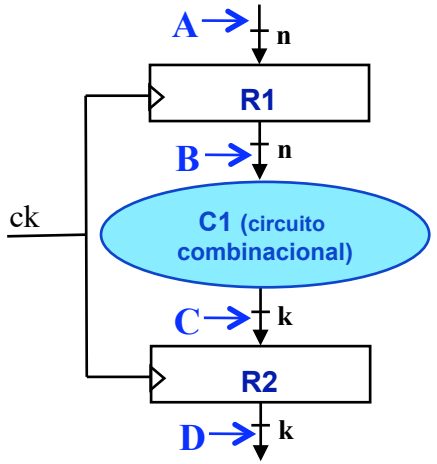




1. Projeto de Unidade Lógico-Aritmética

Cálculo do Período do Relógio





Conclusão. O período do relógio pode ser calculado por:

$$T = \max\{tco_{R1}, th_{R1}\} + td_{C1} + tsu_{R2} =$$
$$= \max\{1, 1\} + 2 + 1 = 4 \text{ ns}$$

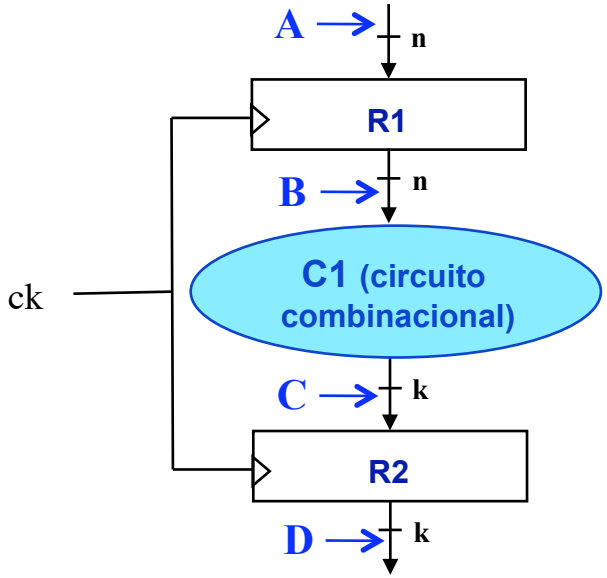
1. Projeto de Unidade Lógico-Aritmética

Diagramas de Tempo

Pergunta:

Como estimar td_{C1}?

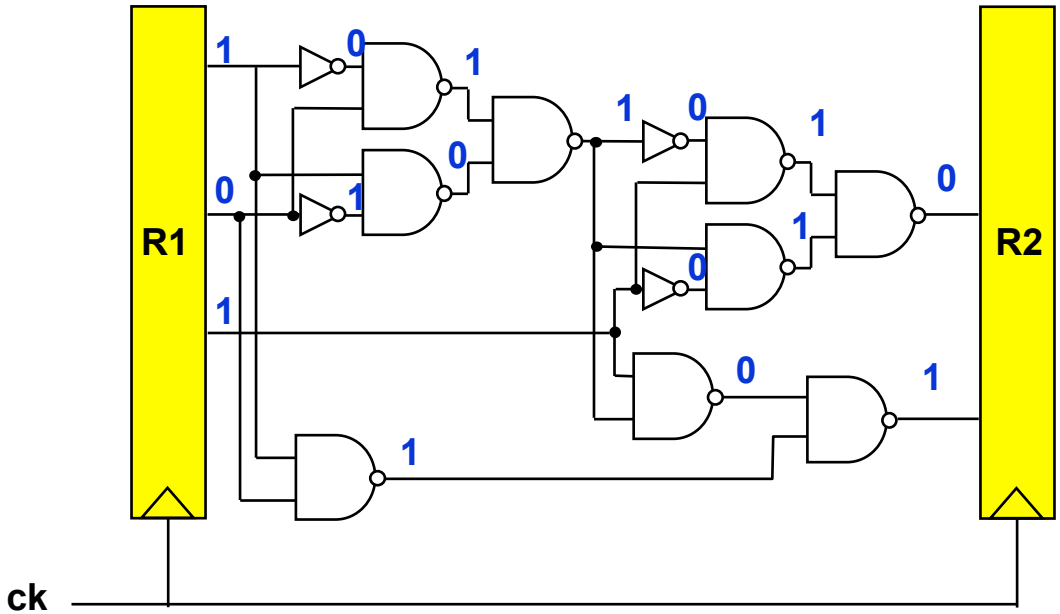
- **Notar que C1 é um circuito combinacional**



1. Projeto de Unidade Lógico-Aritmética

Estimando o Atraso Crítico de Circuito Combinacional

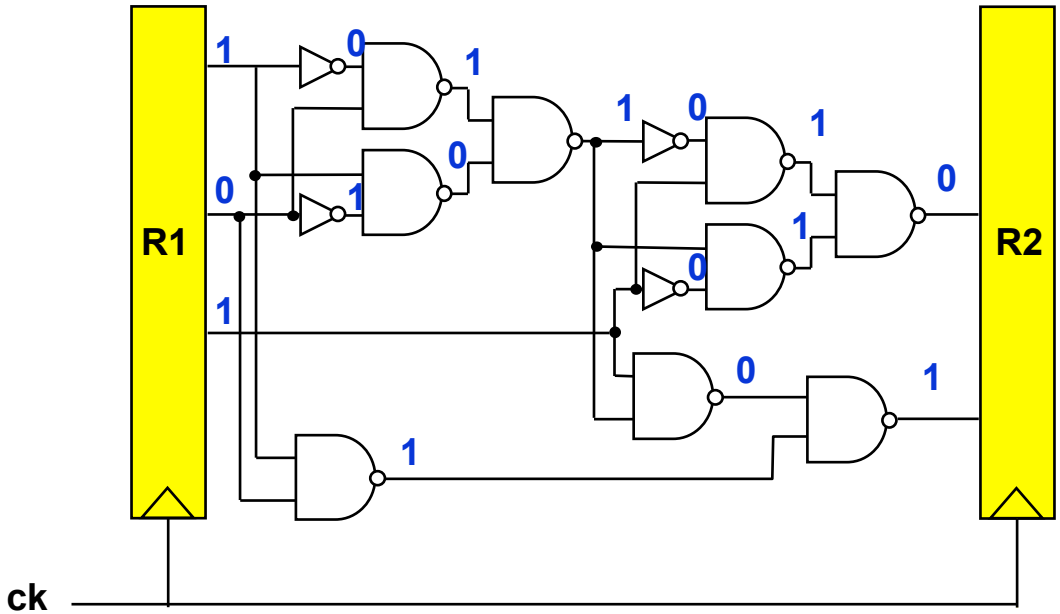
Supondo que td=1 unid. para qualquer porta lógica



1. Projeto de Unidade Lógico-Aritmética

Estimando o Atraso Crítico de Circuito Combinacional

Supondo que td=1 unid. para qualquer porta lógica



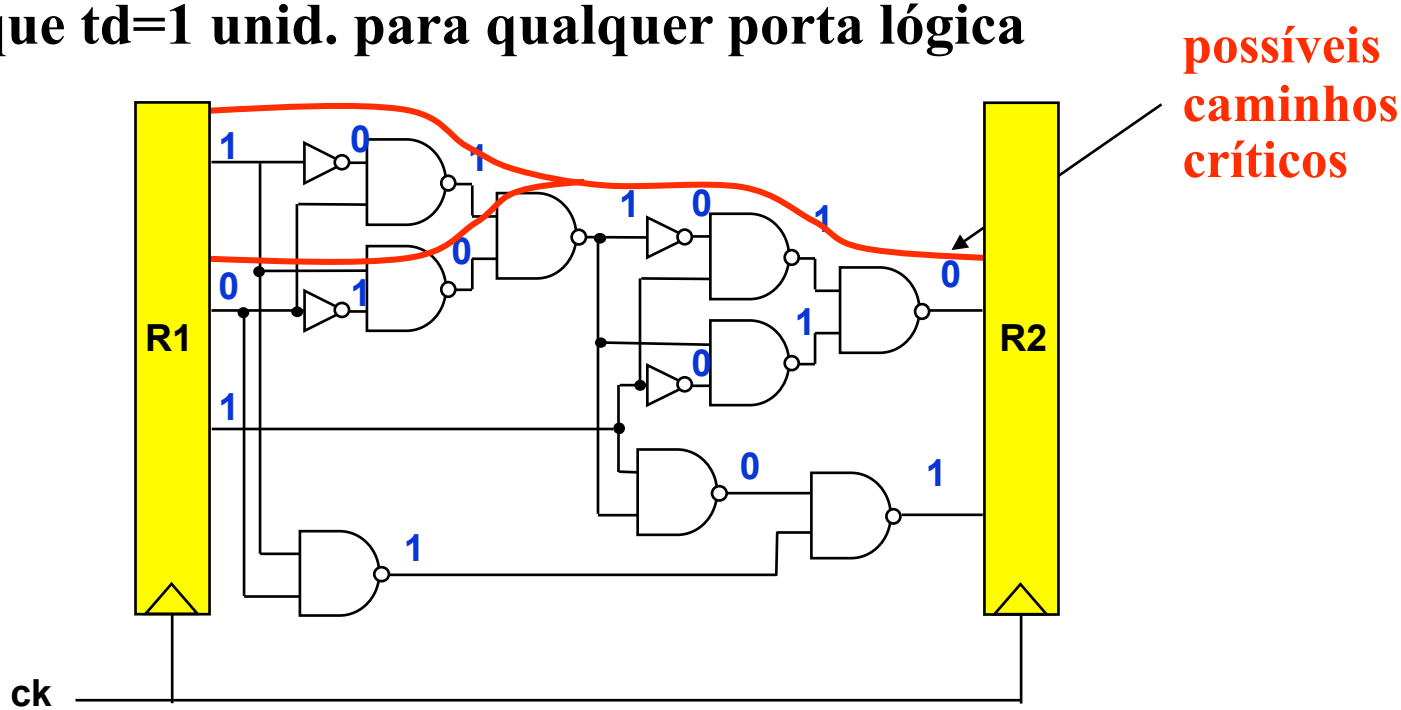
Possíveis Técnicas:

1. **Simulação: repetir a análise para cada um dos 2^n vetores de entrada.**

1. Projeto de Unidade Lógico-Aritmética

Estimando o Atraso Crítico de Circuito Combinacional

Supondo que td=1 unid. para qualquer porta lógica



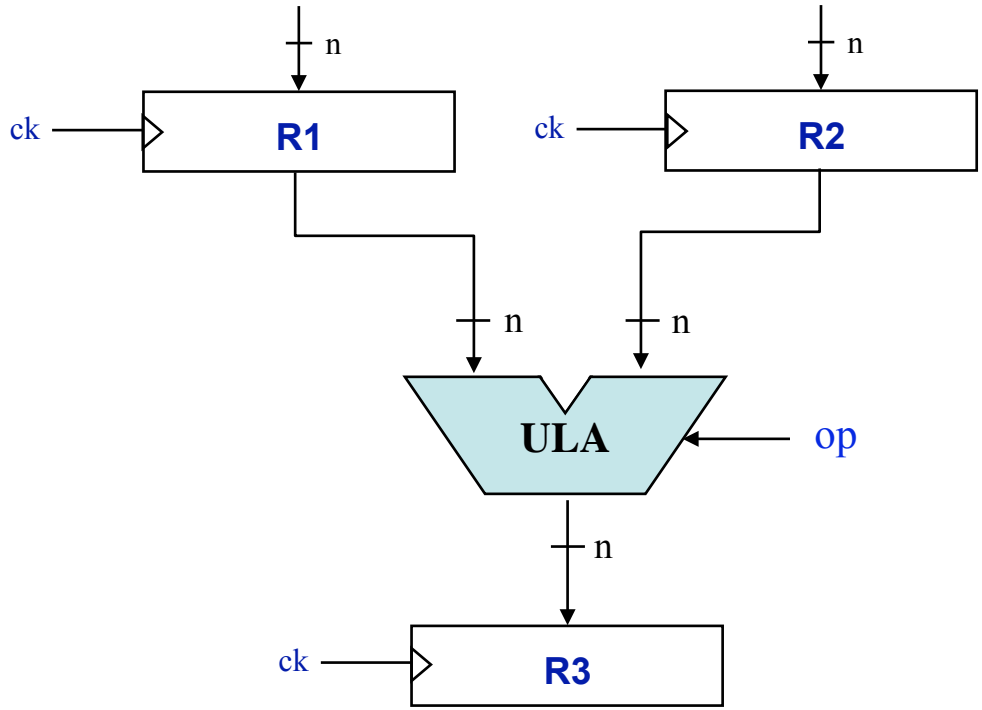
Possíveis Técnicas:

1. **Simulação: repetir a análise para cada um dos 2^n vetores de entrada.**
2. **Análise de Timing (STA): analisar os atrasos dos caminhos entre R1 e R2.**

1. Projeto de Unidade Lógico-Aritmética

Topologia Básica de uma ULA

Caso Genérico



A cada borda de relógio, este circuito faz:

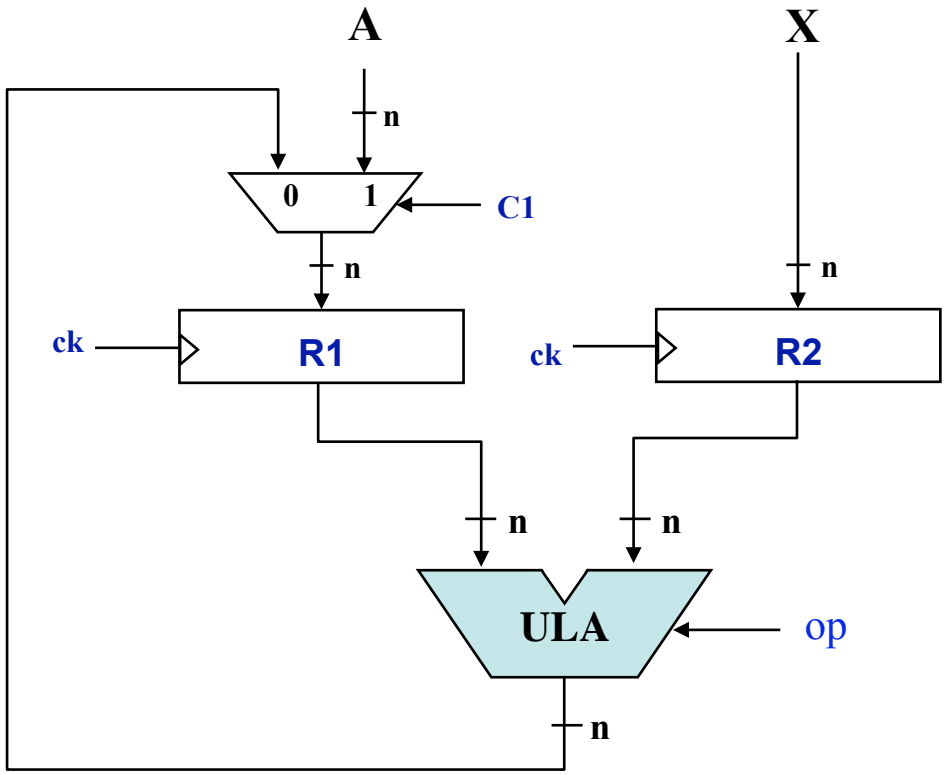
R3 ← R1 op R2

onde **op** é uma das operações disponíveis na ULA

1. Projeto de Unidade Lógico-Aritmética

Topologia Básica de uma ULA

Uma Possível Variação do Caso Genérico



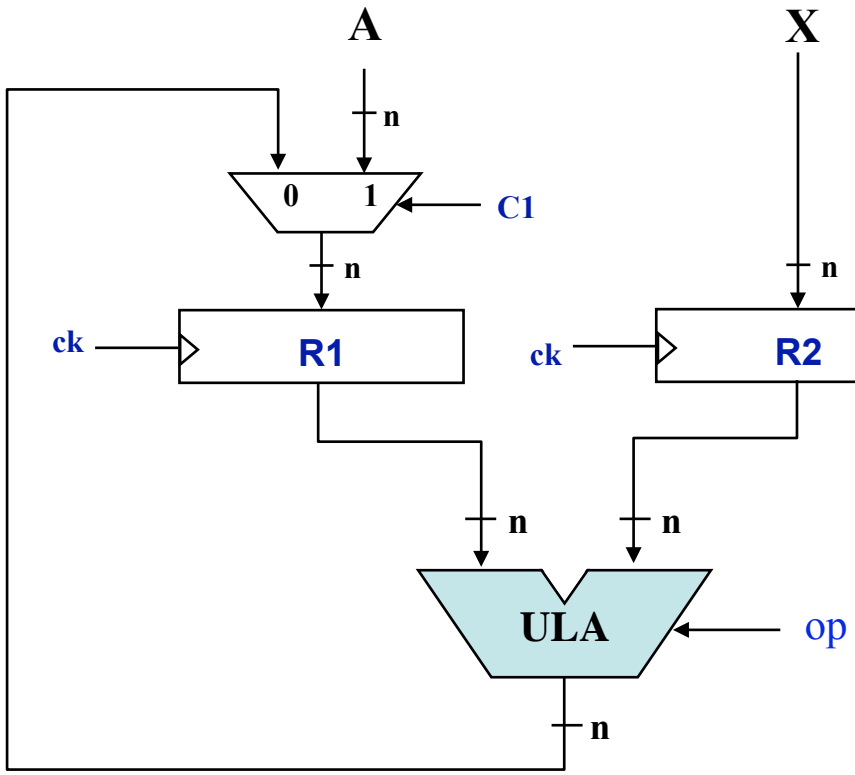
Adição de n parcelas em n passos:

R1 ← A; R2 ← X; // em paralelo
R1 ← R1 op R2; R2 ←X;
...

1. Projeto de Unidade Lógico-Aritmética

Topologia Básica de uma ULA

Uma Possível Variação do Caso Genérico



Adição de n parcelas em n passos:

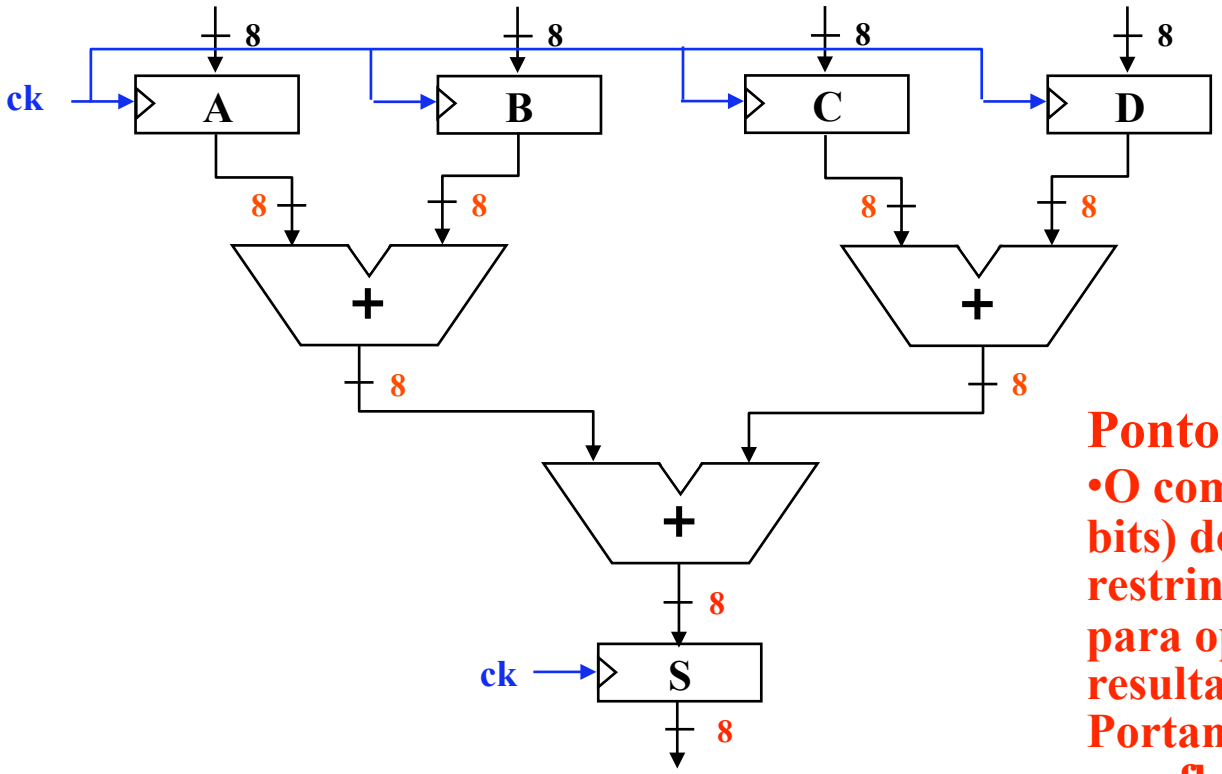
R1 ← A; R2 ← X; // em paralelo
Enquanto (condição)
{
R1 ← R1 op R2; R2 ←X;
}

Obs:
- op = combinação que seleciona adição
- Os operandos B, C e D devem ser fornecidos pela entrada X, nos tempos corretos. Consequência: deve haver um circuito de controle (não mostrado no desenho...)

1. Projeto de Unidade Lógico-Aritmética

Uma ULA Dedicada

Adição em Paralelo de 4 Operandos Inteiros Sem Sinal



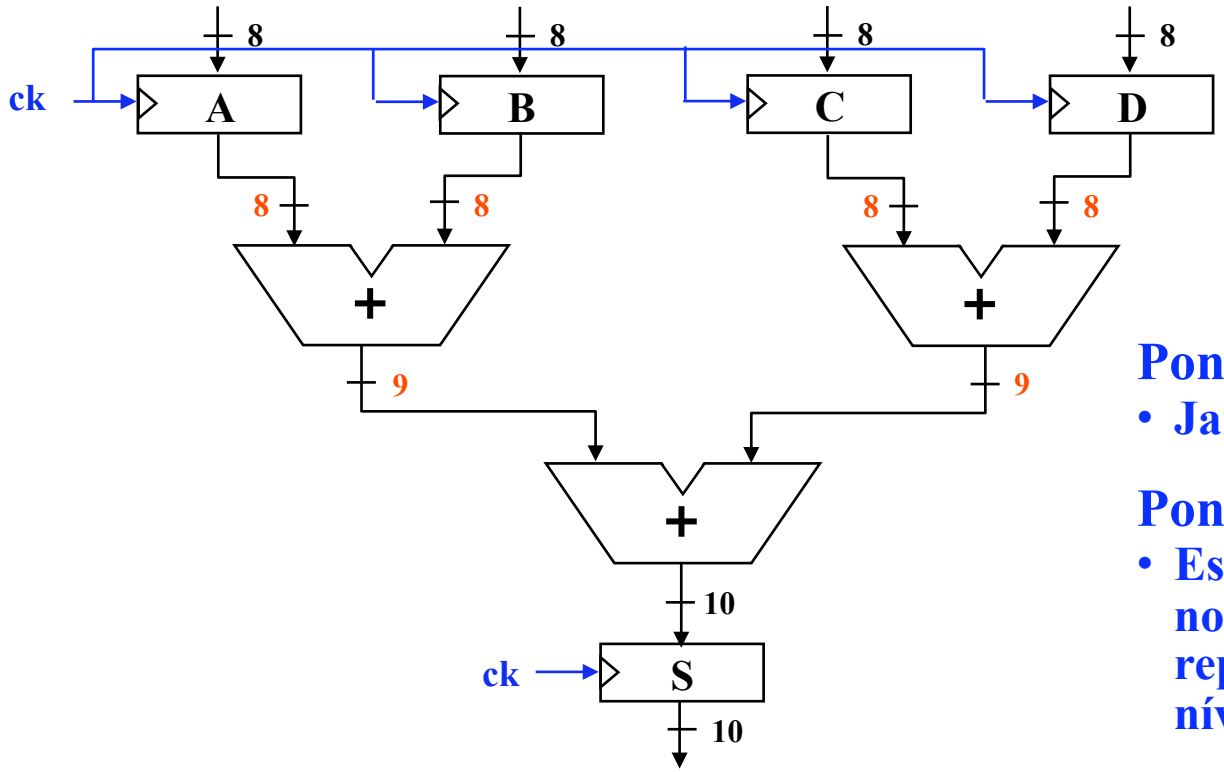
Ponto Negativo:

- **O comprimento (i.e., o nº de bits) do registrador "S" restringe o uso desta ULA para operações nas quais o resultado não ultrapasse 255. Portanto, pode ocorrer overflow.**

1. Projeto de Unidade Lógico-Aritmética

Uma ULA Dedicada

Adição em Paralelo de 4 Operandos Inteiros Sem Sinal



Ponto Positivo:

- Jamais ocorre overflow.

Ponto Negativo:

- Este desenho não obedece nossa convenção para representação de dados no nível RT...

1. Projeto de Unidade Lógico-Aritmética

Uma ULA Dedicada

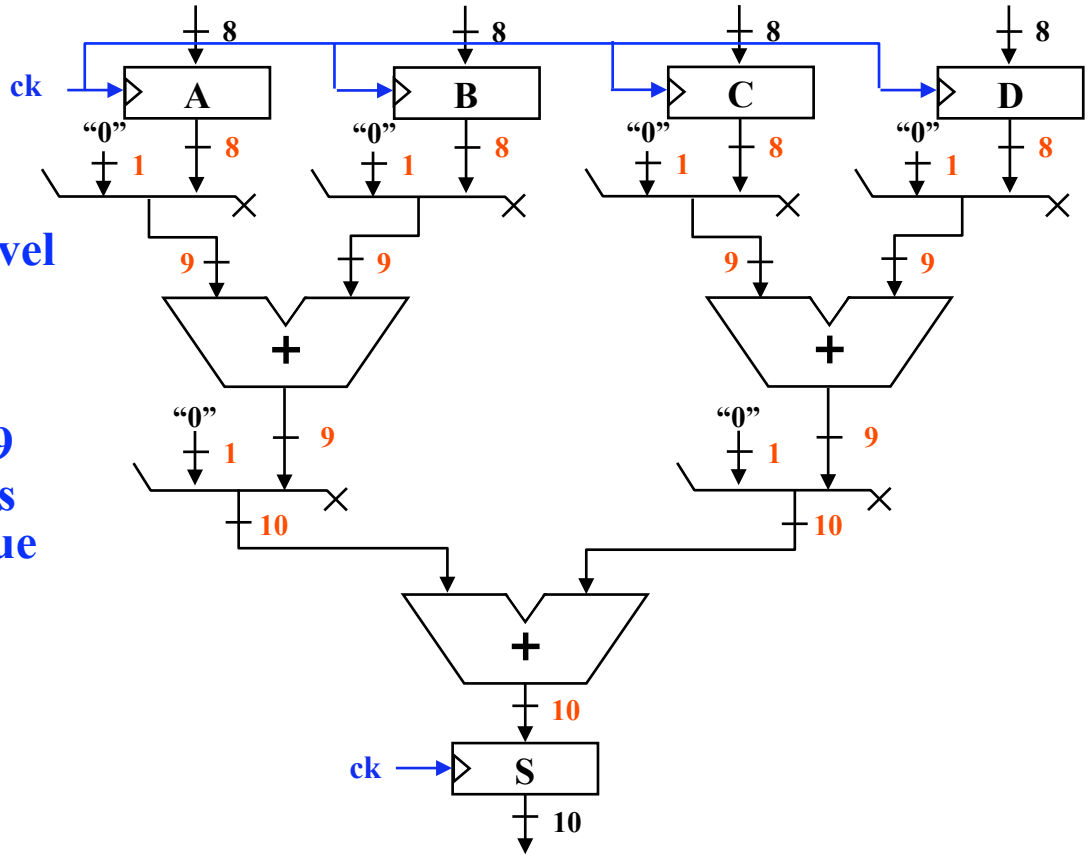
Adição em Paralelo de 4 Operandos Inteiros Sem Sinal

Pontos Positivos:

- Jamais ocorre overflow.
- A convenção para dados no nível RT é obedecida.

Ponto Negativo:

- 2 somadores p/ números com 9 bits e 1 somador para números de 10 bits (mais recursos do que o necessário...)



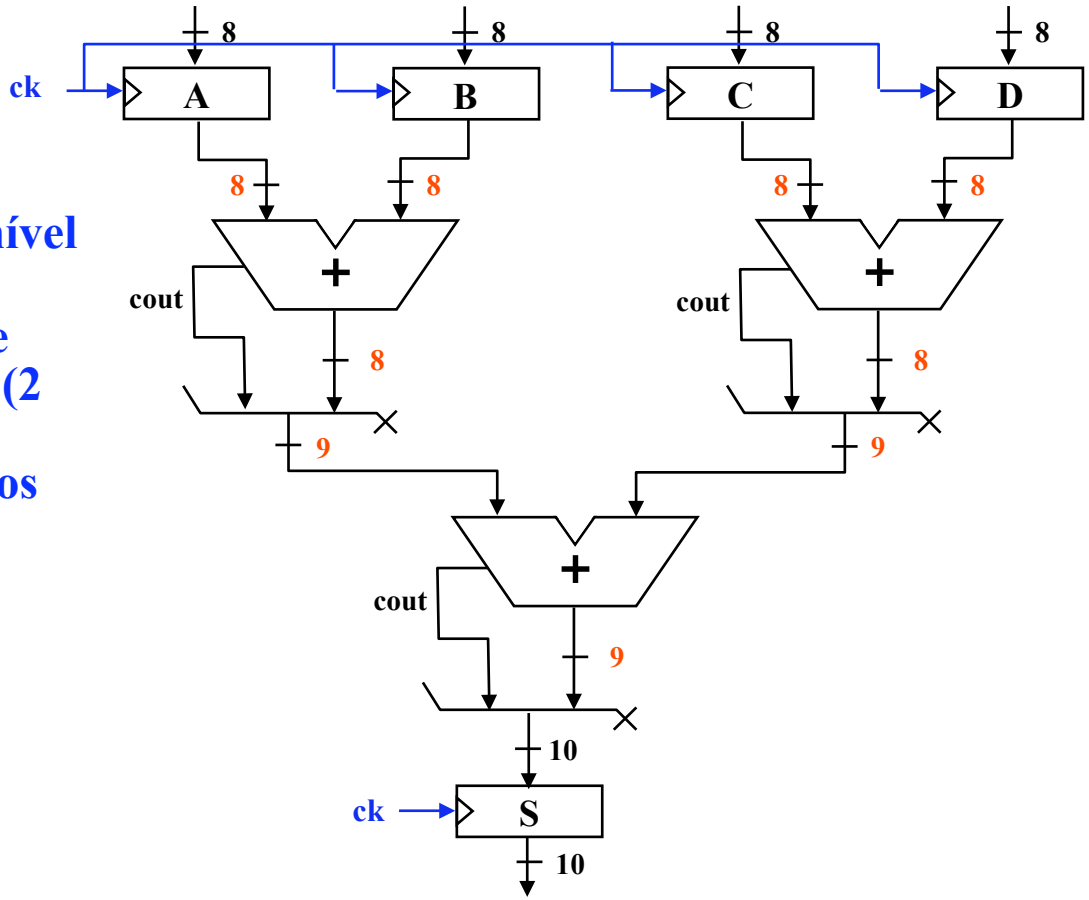
1. Projeto de Unidade Lógico-Aritmética

Uma ULA Dedicada

Adição em Paralelo de 4 Operandos Inteiros Sem Sinal

Pontos Positivos:

- Jamais ocorre overflow.
- A convenção para dados no nível RT é obedecida.
- Somadores dimensionados de modo a economizar recursos (2 somadores p/ números com 8 bits e 1 somador para números de 8 bits)



1. Projeto de Unidade Lógico-Aritmética

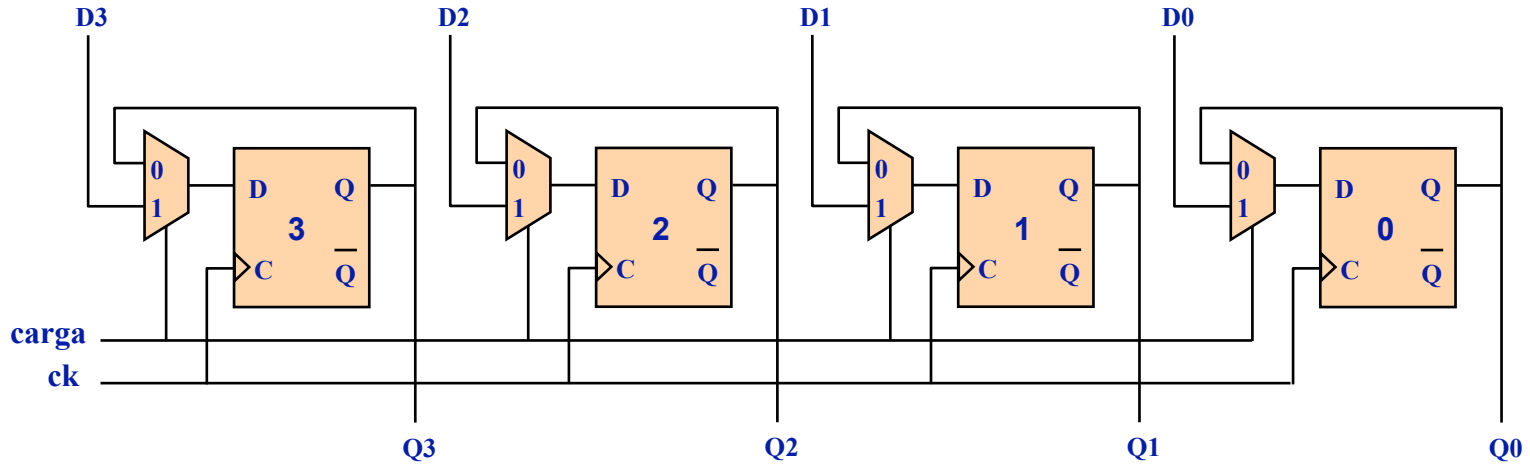
Registradores de Uso Específico

No Projeto de Sistemas Digitais no Nível RT, também é comum se utilizar os seguintes tipos de registradores:

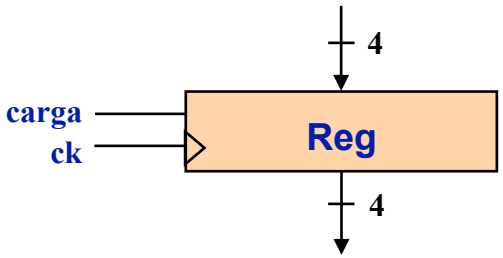
- **Registrador com carga paralela controlada (ck + sinal de carga)**
- **Registrador de Deslocamento.**
- **Registrador-Contador.**

1. Projeto de Unidade Lógico-Aritmética

Registrador com Carga Paralela Controlada



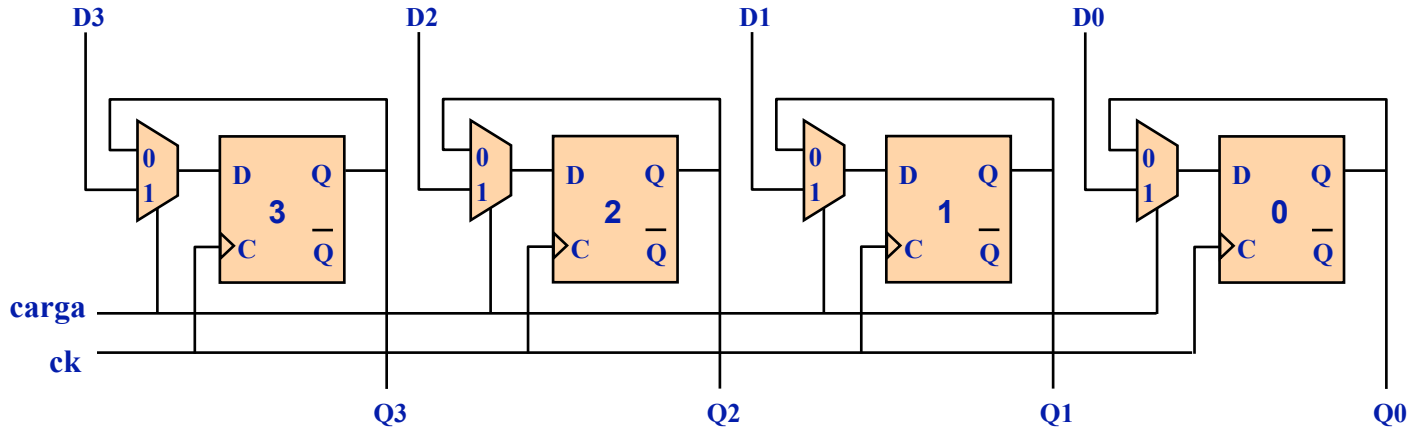
Símbolo no nível RT



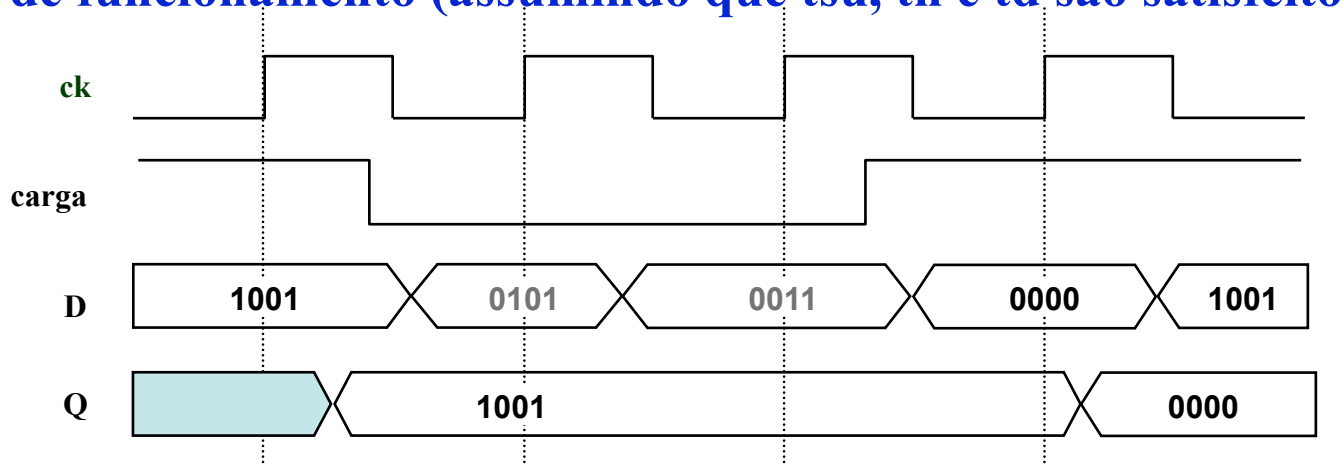
- **Existe um sinal de "carga" e um sinal de relógio (ck)**
- **A cada borda ativa de "ck" cada FF amostra um valor de sua respectiva entrada "D":**
  - **Se carga=0, cada FF amostra seu respectivo "Q"**
  - **Se carga=1, cada FF amostra um valor externo "Di"**

1. Projeto de Unidade Lógico-Aritmética

Registrador com Carga Paralela Controlada



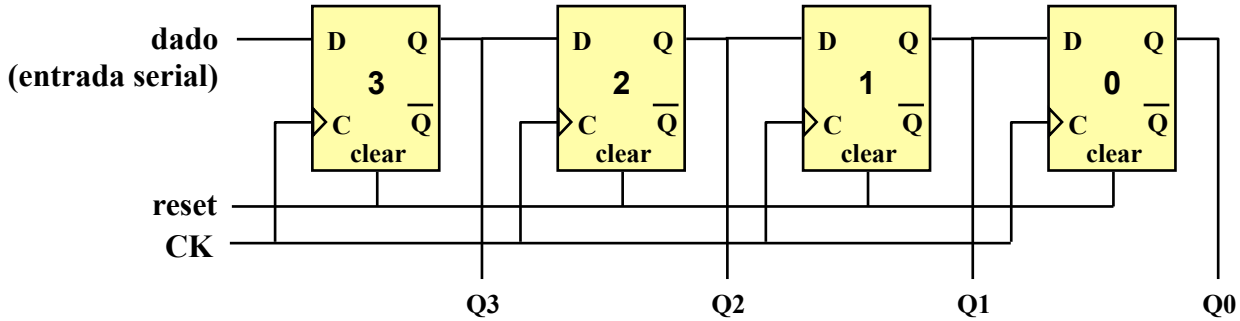
Exemplo de funcionamento (assumindo que tsu, th e td são satisfeitos)



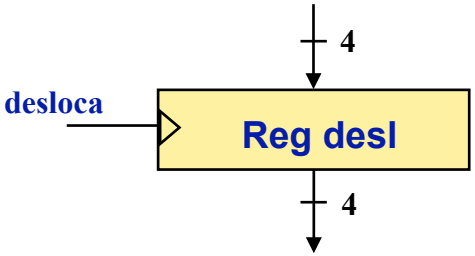
1. Projeto de Unidade Lógico-Aritmética

Registradores de Deslocamento à Direita

(*Shift Register*)



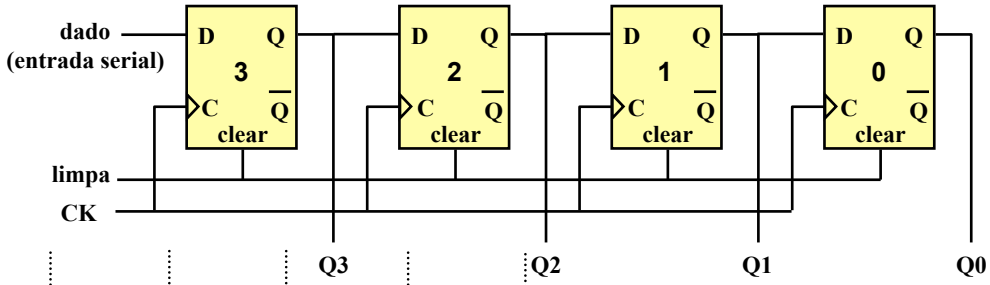
Símbolo no nível RT



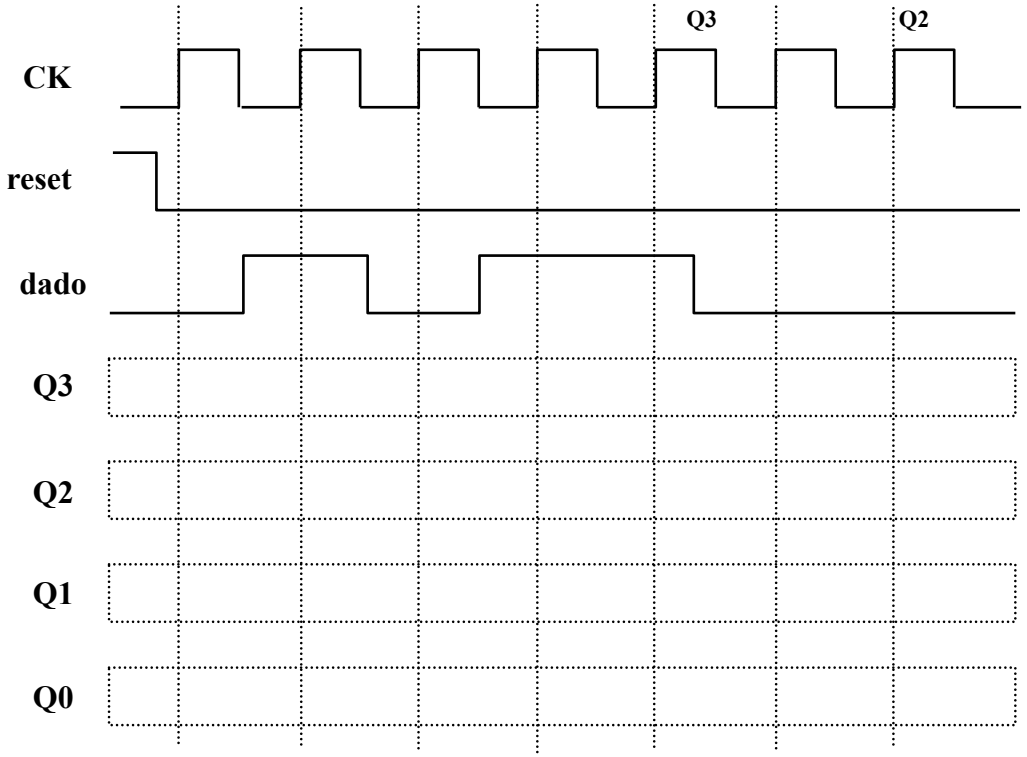
- **Existe um sinal de "desloca"**
- **A cada borda ativa do sinal "desloca", cada FF amostra o valor da saída "Q" do FF adjacente à esquerda**
- **O FF mais à esquerda lê um bit da entrada "dado", a qual corresponde a uma "entrada serial"**
- **Este registrador também pode ser visto como um registrador com carga serial…**

1. Projeto de Unidade Lógico-Aritmética

Registradores de Deslocamento à Direita



Exemplo de funcionamento

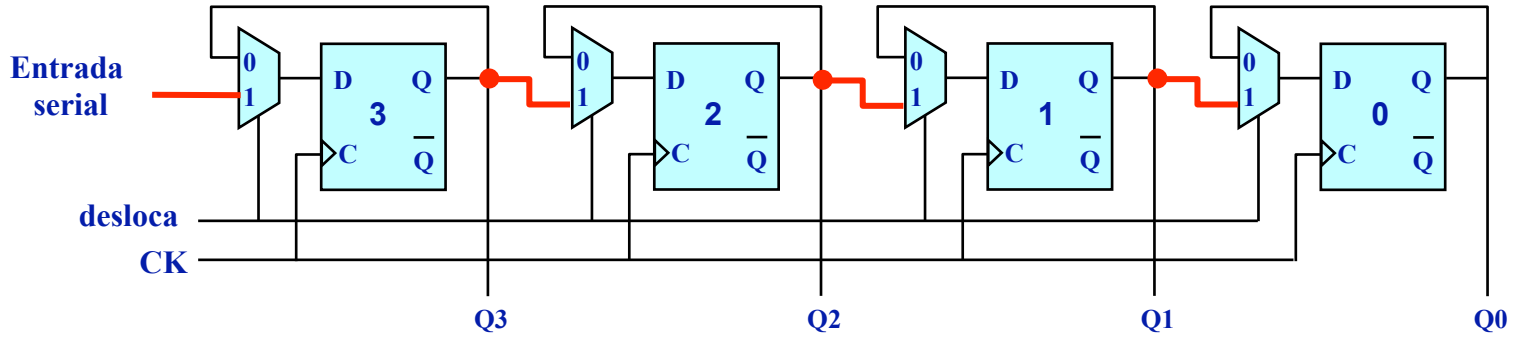


1. Projeto de Unidade Lógico-Aritmética

Registradores de Deslocamento à Direita

Registrador de Deslocamento com Controle

Complete as ligações faltantes

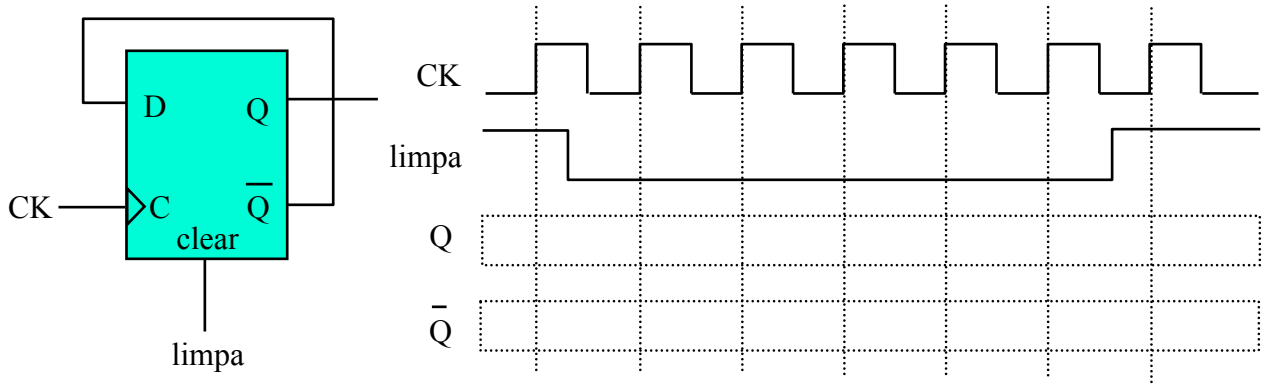


Quantos ciclos de relógio será preciso manter "desloca"=1 até preencher o conteúdo deste registrador?

1. Projeto de Unidade Lógico-Aritmética

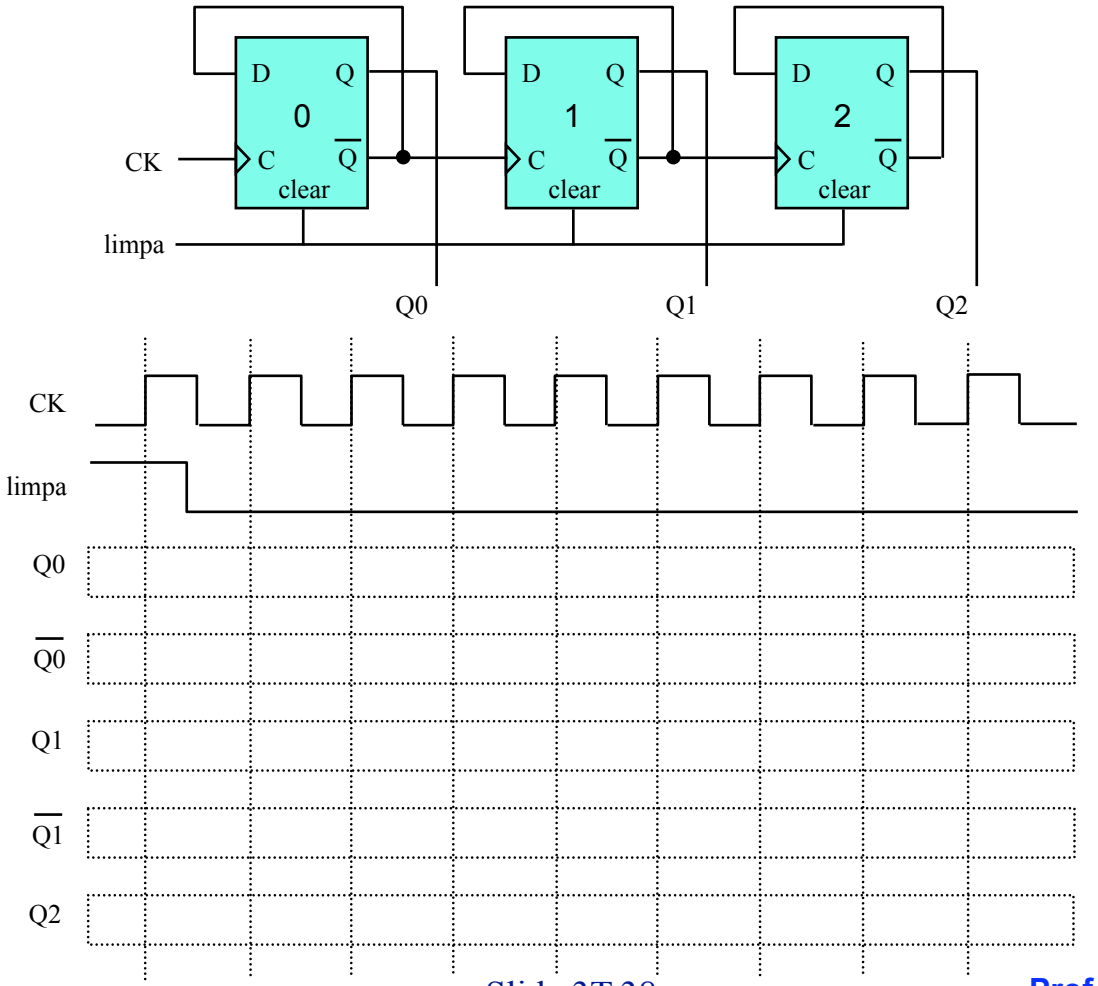
Registrador-Contador (*Ripple*)

Contador de 1 bit



1. Projeto de Unidade Lógico-Aritmética

Registrador-Contador (*Ripple*) de 3 Bits

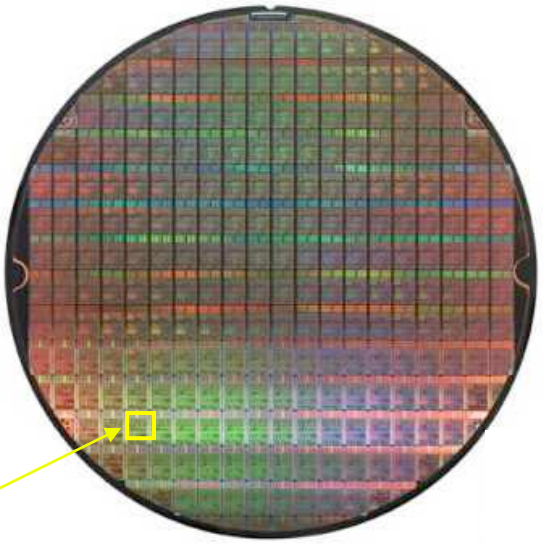


1. Projeto de Unidade Lógico-Aritmética

Mapeamento Tecnológico

Basicamente, há duas soluções para se materializar um sistema digital:

1. **Com FPGA** (usado na aula prática).
2. **Mandando fabricar um chip dedicado – também referido por ASIC** (não iremos utilizar devido o custo e o tempo).

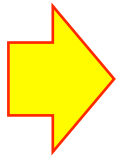


From http://www.amd.com

1. Projeto de Unidade Lógico-Aritmética

Mapeamento Tecnológico

2. **Mandando fabricar um chip dedicado – também referido por ASIC**



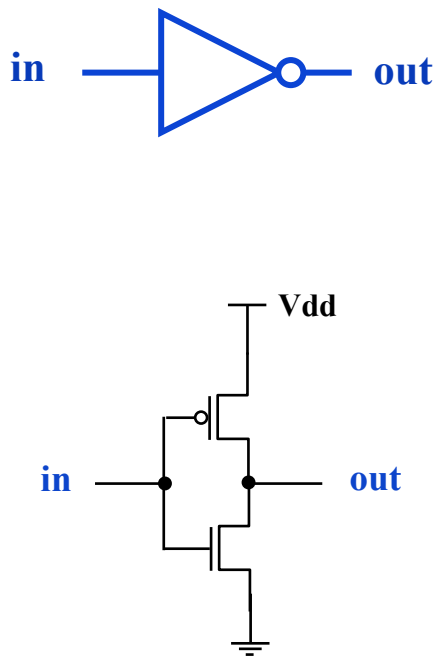
Exige que o circuito seja "mapeado" para a tecnologia de fabricação (CMOS)

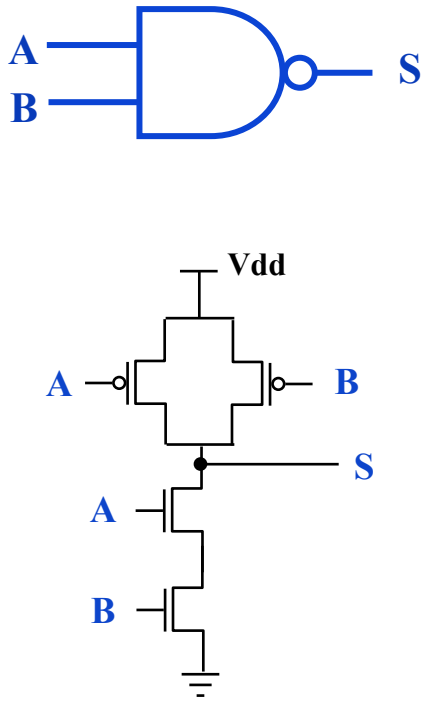
Exemplos de portas lógicas em CMOS:

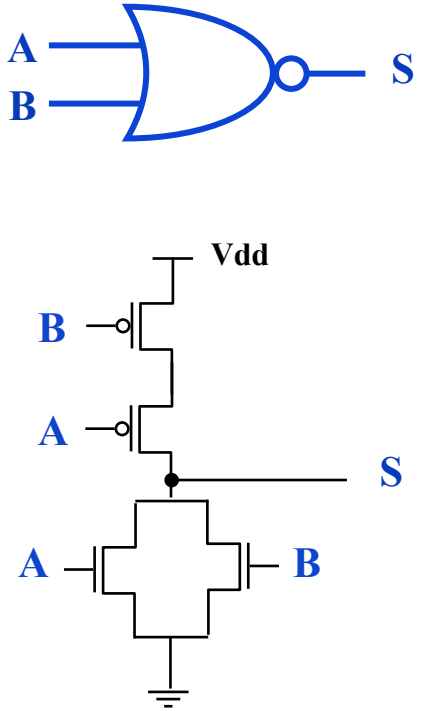
1. Projeto de Unidade Lógico-Aritmética

Mapeamento Tecnológico

Exemplos de portas lógicas em CMOS:



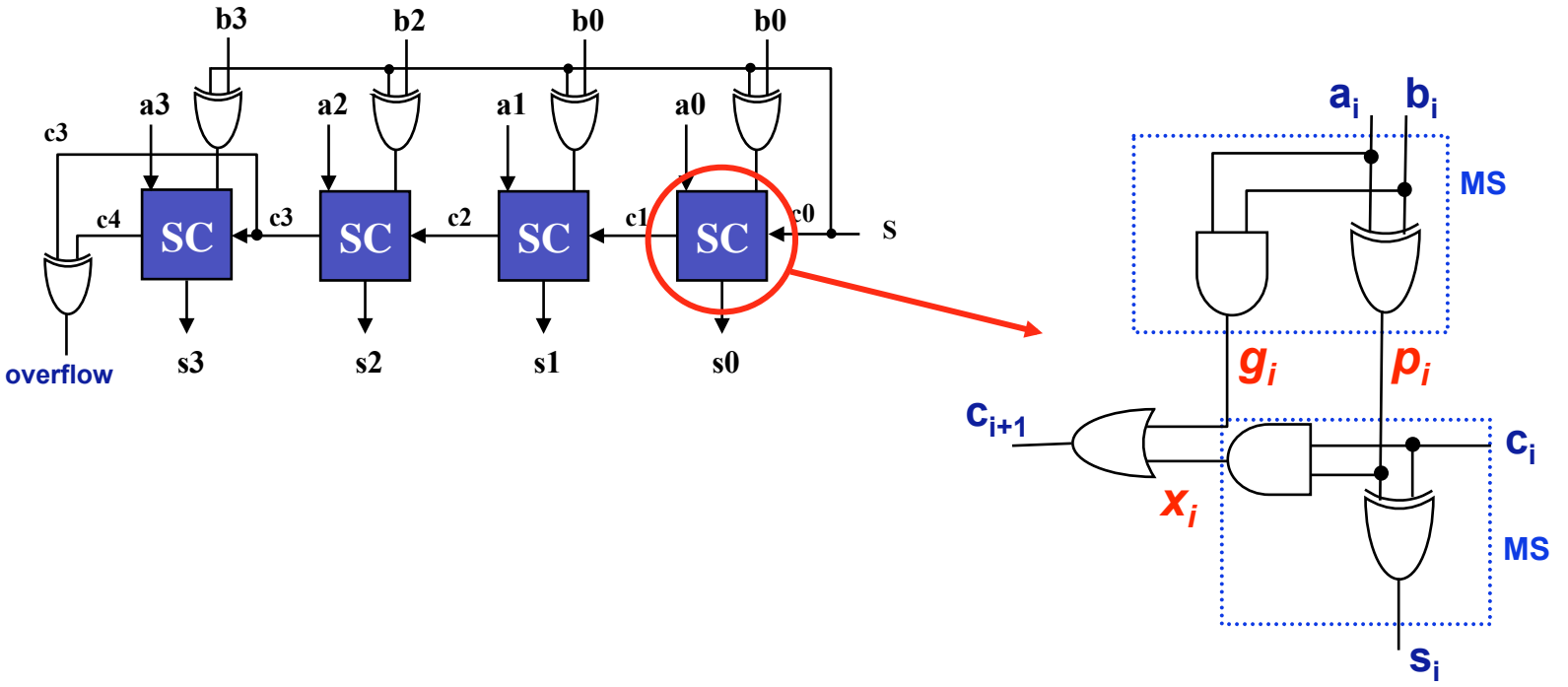




1. Projeto de Unidade Lógico-Aritmética

Custo dos Blocos Básicos do Nível RT

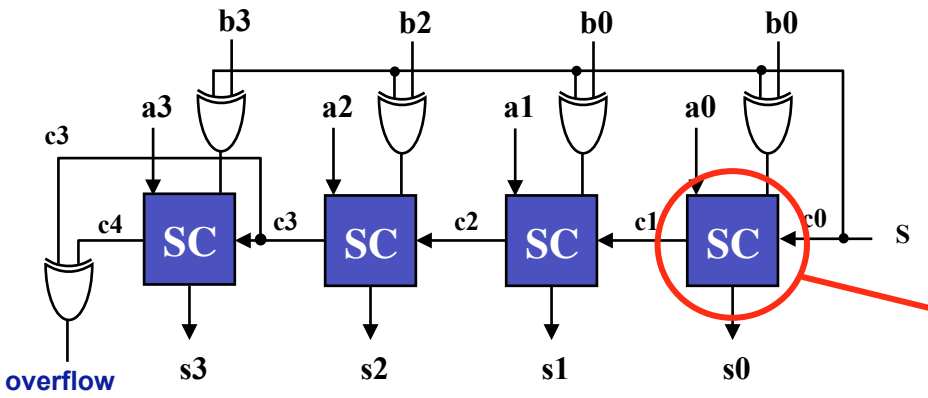
Somador/Subtrator



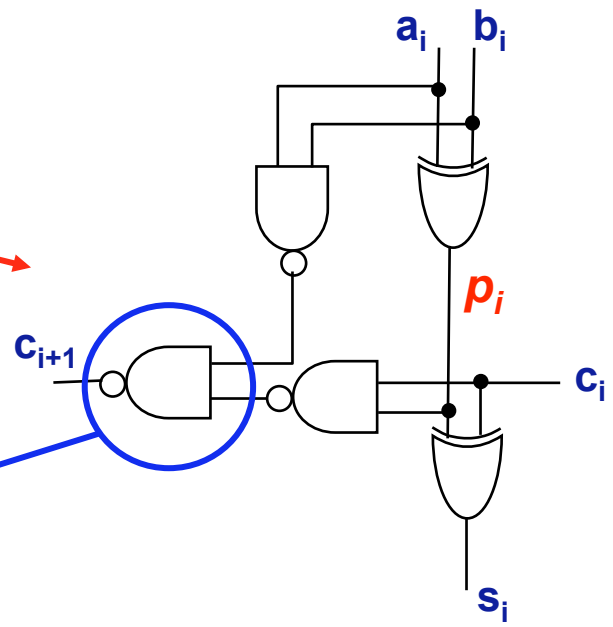
1. Projeto de Unidade Lógico-Aritmética

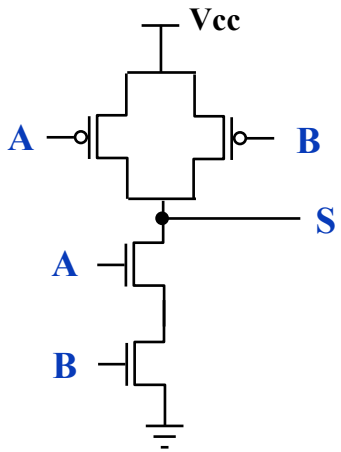
Custo dos Blocos Básicos do Nível RT

Somador/Subtrator



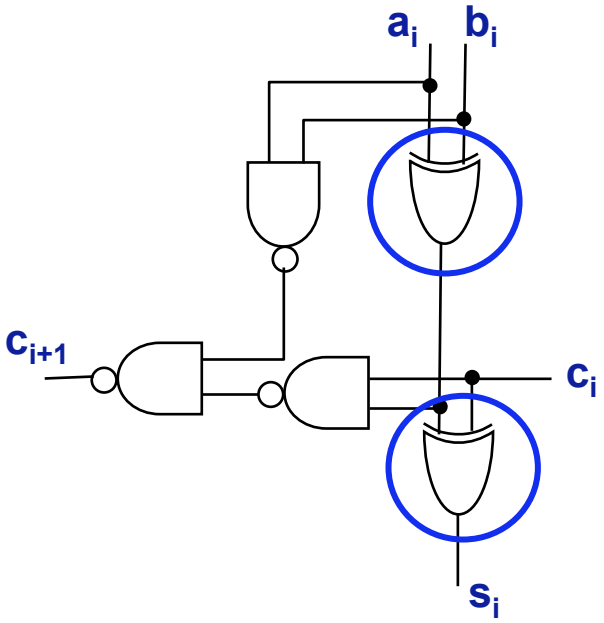
Mas em CMOS...



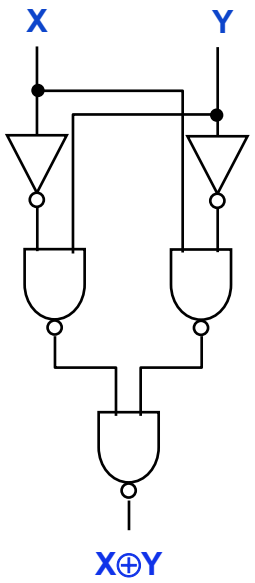


Custo dos Blocos Básicos do Nível RT

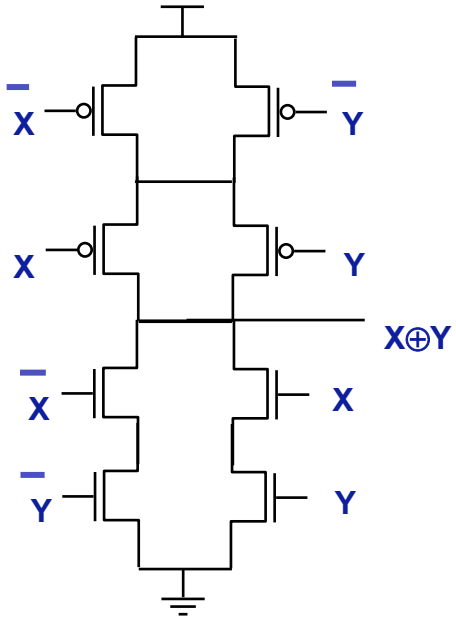
Somador/Subtrator



Algumas Implementações CMOS para a xor



16 transistores

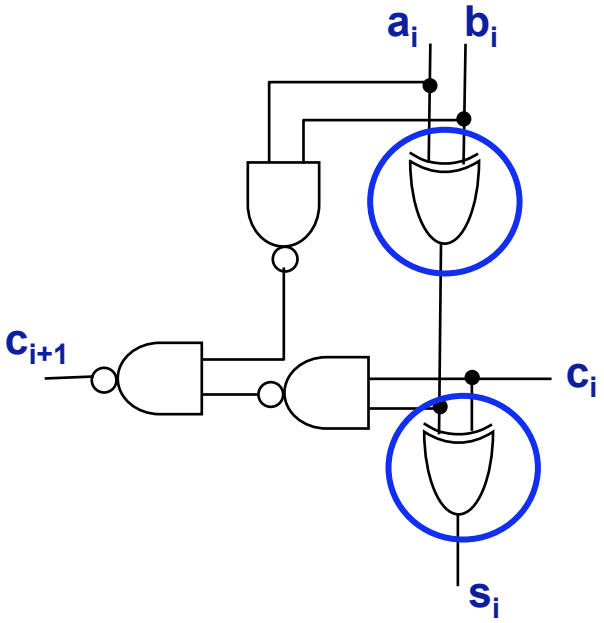


12 transistores
(necessita de 2 inversores)

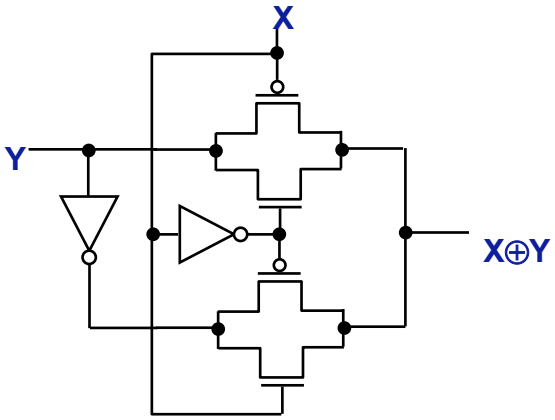
1. Projeto de Unidade Lógico-Aritmética

Custo dos Blocos Básicos do Nível RT

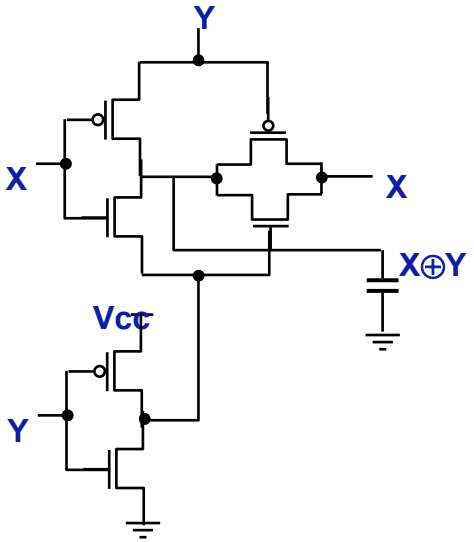
Somador/Subtrator



Algumas Implementações CMOS para a xor



8 transistores

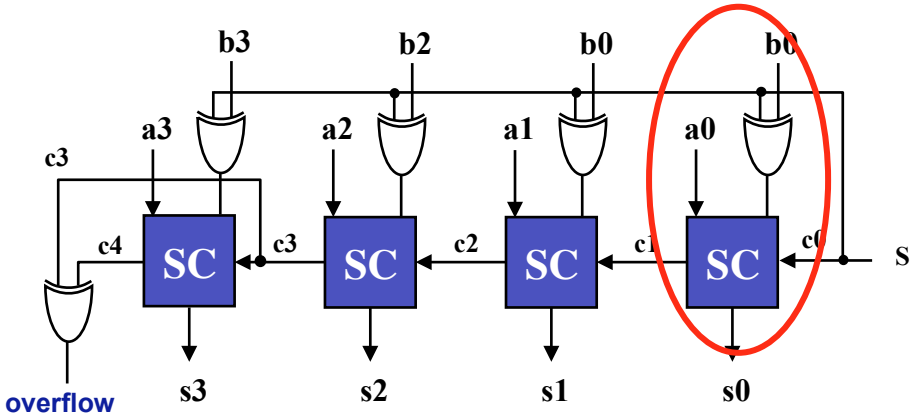


6 transistores
(é a mais usada)

1. Projeto de Unidade Lógico-Aritmética

Custo dos Blocos Básicos do Nível RT

Somador/Subtrator



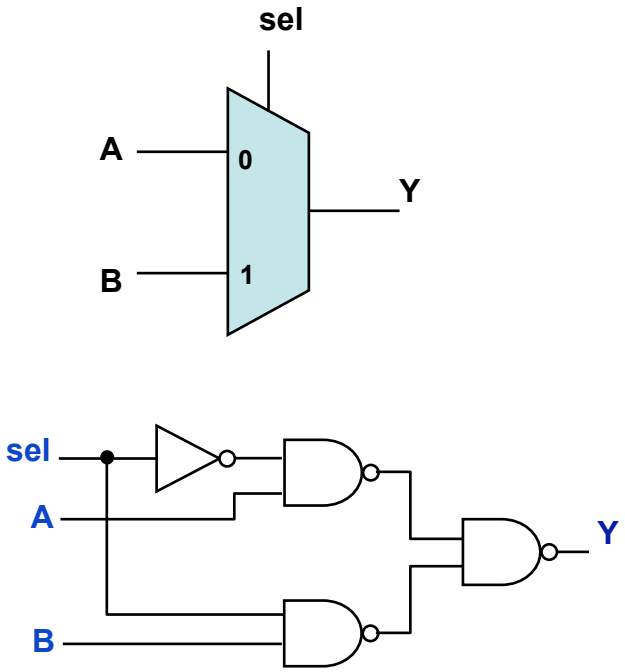
Custo do Somador/subtrator, por bit:

- **3 portas xor: 3 x 6 = 18 transistores**
- **3 portas nand de duas entradas: 3 x 4 = 12 transistores**
- **Logo, custo de um bit = 30 transistores (ignorando-se a xor que calcula o overflow)**

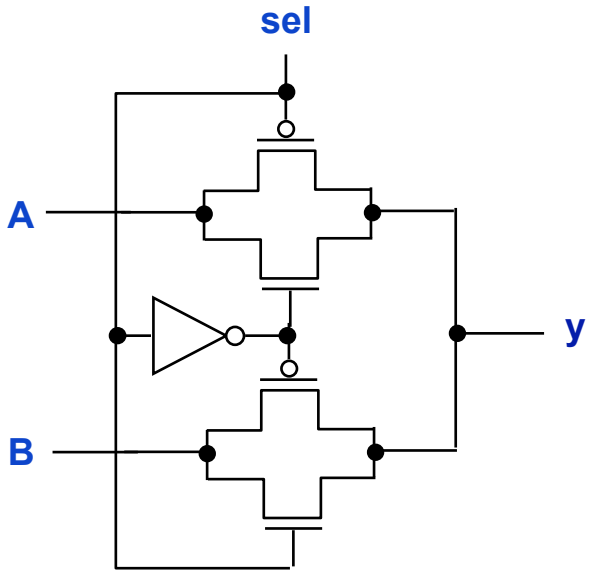
Custo de um somador/subtrator de n bits: 30n transistores

Custo dos Blocos Básicos do Nível RT

Multiplexador 2:1 (mux2:1)



14 ou 12 transistores

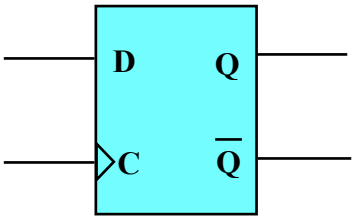


6 ou 4 transistores
(mais usado)

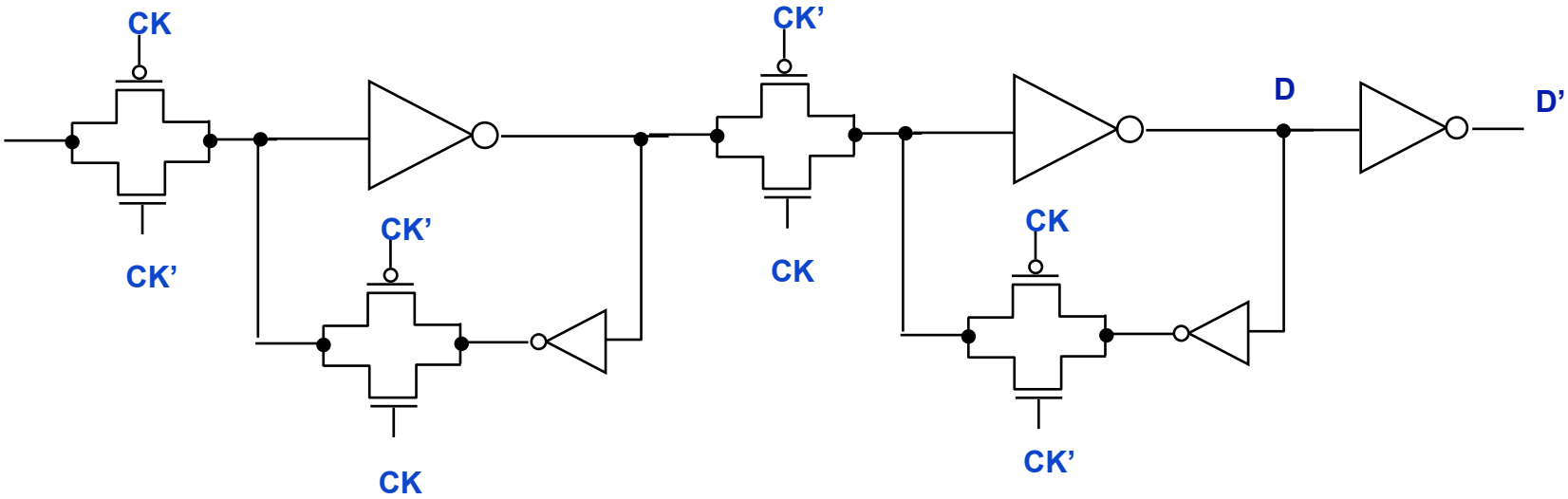
1. Projeto de Unidade Lógico-Aritmética

Custo dos Blocos Básicos do Nível RT

Flip-flop D (Mestre-Escravo)



18 ou 20 transistores (eventualmente, podemos considerar somente um inversor para o clock de todos os bits)

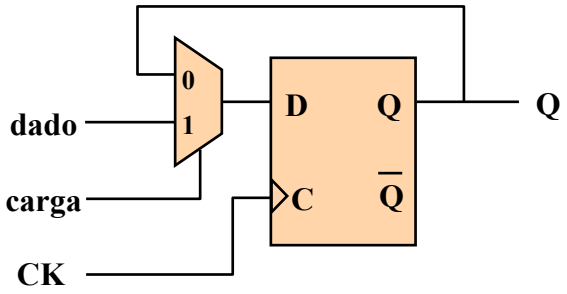


OBS: para set ou reset assíncrono, adicionar 2 transistores

1. Projeto de Unidade Lógico-Aritmética

Custo dos Blocos Básicos do Nível RT

Flip-flop D CMOS com habilitação de carga paralela



18+4= 22 transistores

OBS: para set ou reset assíncrono, adicionar 2 transistores

1. Projeto de Unidade Lógico-Aritmética

Custo dos Blocos Básicos RT

Componente RT	Custo (nº transistores)
Somador	24n
Subtrator	26n
Somador/subtrator	30n
Mux 2:1	4n
Registrador com carga paralela (+4 transistores para set ou reset assíncrono)	18n
Registrador com carga paralela controlada (+4 transistores para set ou reset assíncrono)	22n