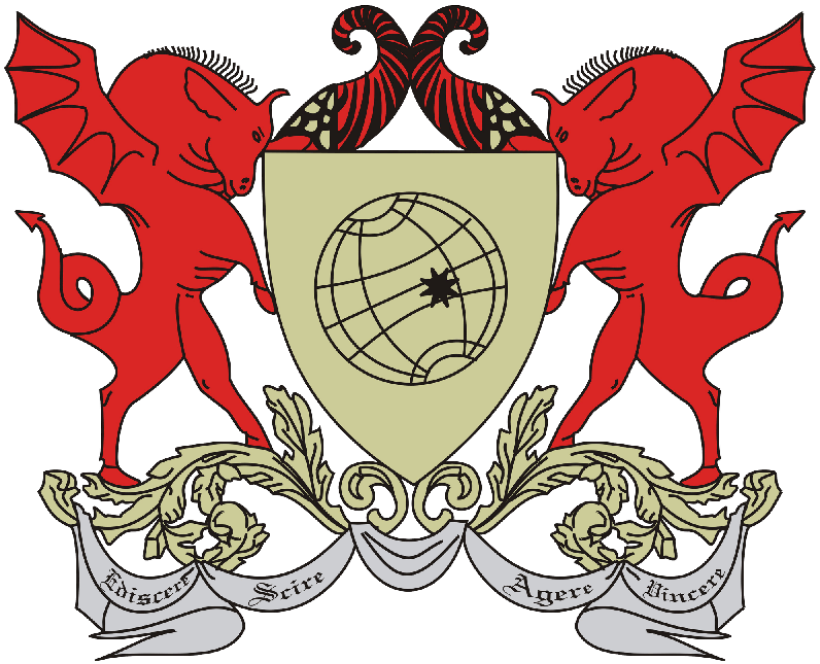


CCF 251 – Introdução aos Sistemas Lógicos

Aula 07 – Lógica sequencial

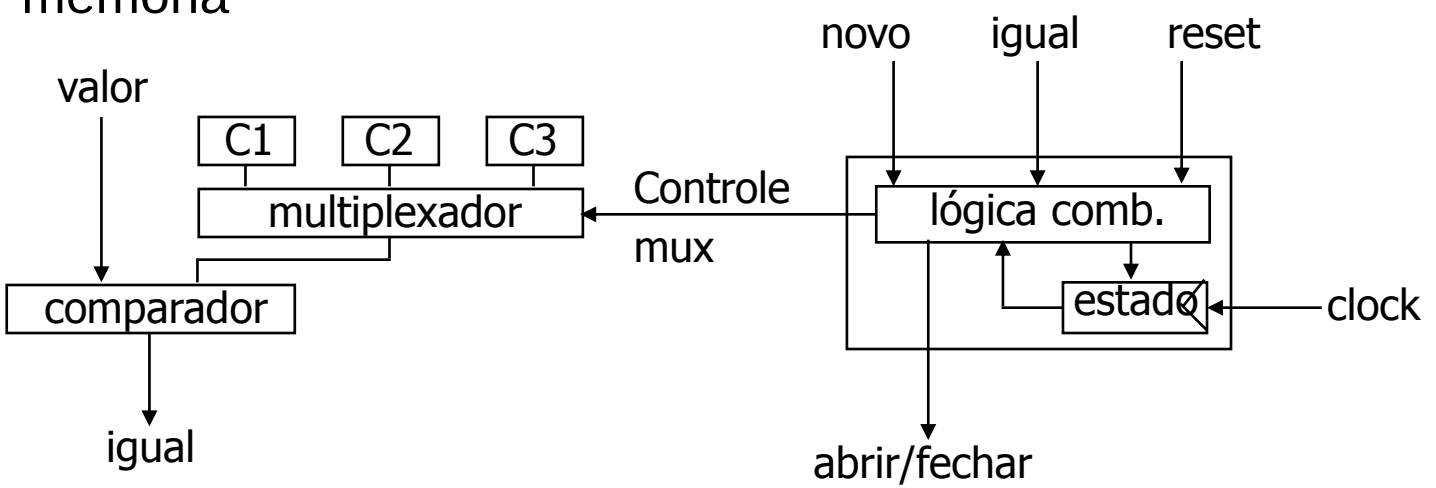
Prof. José Augusto Nacif – jnacif@ufv.br

Lógica Sequencial

- Circuitos Sequencial
  - Circuito simples com realimentação
  - Latches
  - Flip-flops disparados por borda
- Metodologias de temporização
  - Flip-flops cascateados para funcionamento adequado
  - Clock skew
- Entradas Assíncronas
  - Metaestabilidade e sincronização
- Registradores básicos
  - Registradores de deslocamento
  - Contadores simples
- Linguagem de descrição de hardware e lógica sequencial

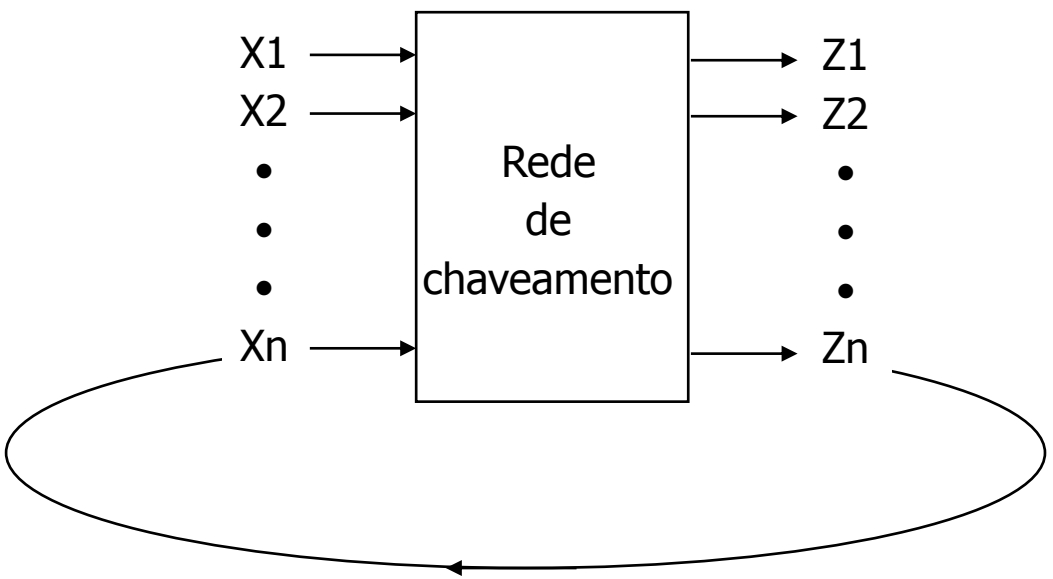
Circuitos Sequenciais

- Circuitos com realimentação
  - saidas = f(entrada, entradas passadas, saidas passadas)
  - Base para a construção dos circuitos lógico dentro da “memória”
  - door combination lock é um exemplo de circuito sequencial
    - Estado é uma memória
    - Estado é uma “saída” e uma “entrada” para lógica combinacional
    - Elementos de armazenamento de combinação são também memória



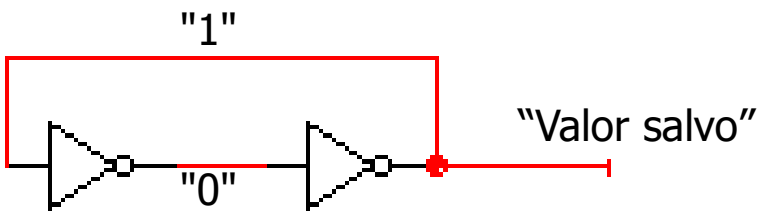
Circuitos com realimentação

- Como controlar a realimentação?
  - O que impede os valores de ciclos ao redor interminavelmente

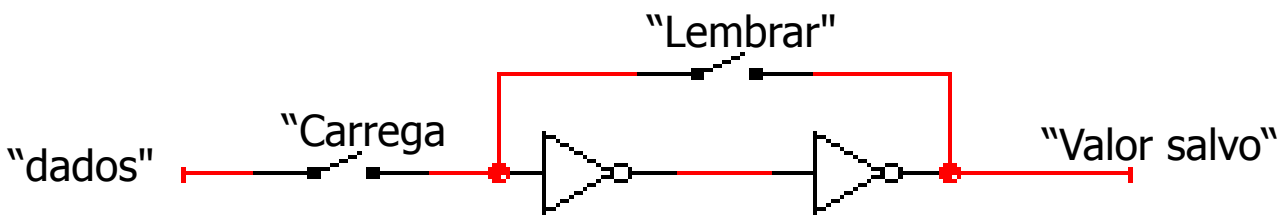


Circuitos mais simples com feedback

- Dois inversores formam uma célula de memória estática
  - Irá segurar valor, desde que tenha potência aplicada

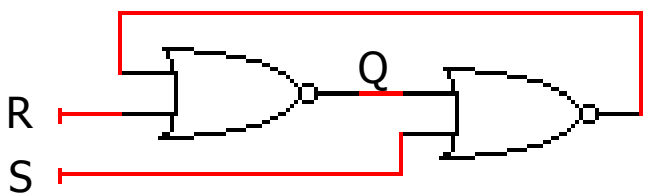


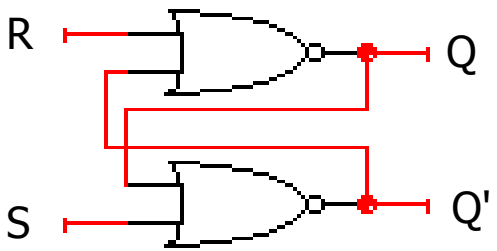
- Como obter um novo valor para a célula de memória?
  - Quebrar seletivamente o caminho feedback
  - Carregar um novo valor na célula



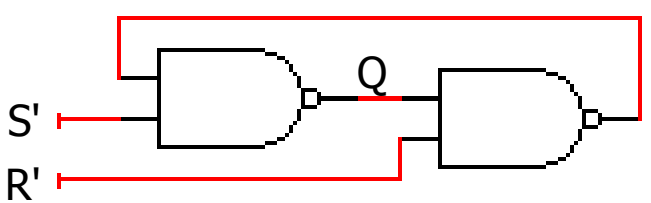
Memória com portas cross-coupled

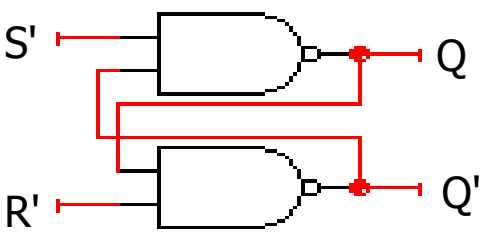
- Portas NOR cross-coupled
  - Similar para pares invertidos, com capacidade para forçar saída para 0 (reset = 1) ou 1 (set = 1).



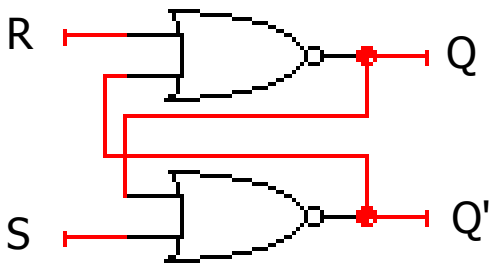


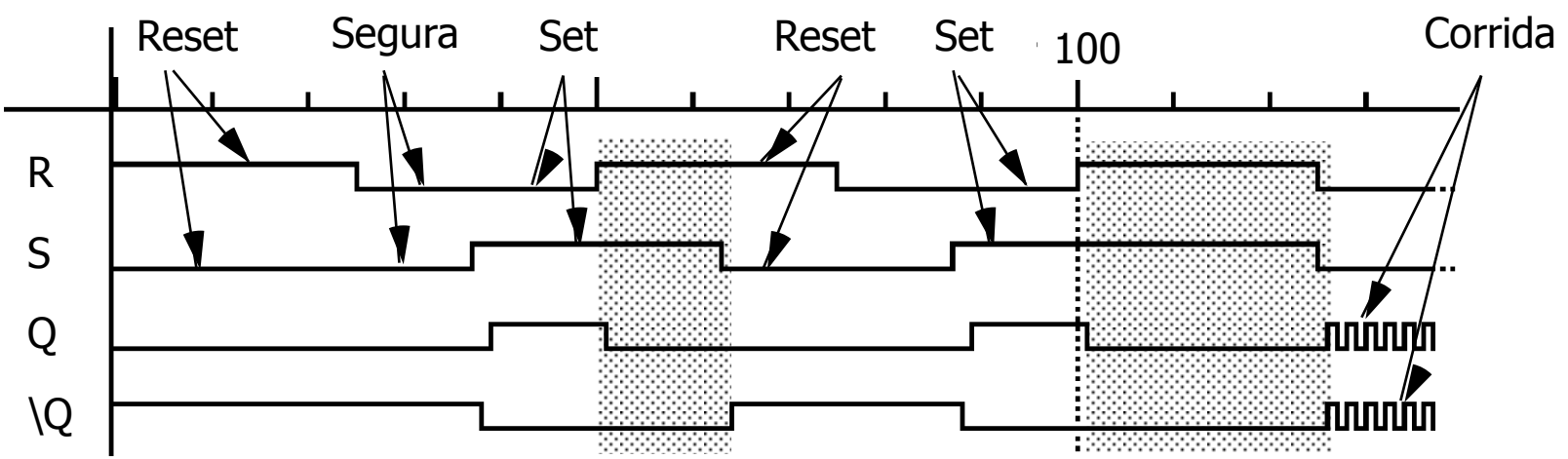
- Portas NAND cross-coupled
  - Similar para pares invertidos, com capacidade para forçar saída para 0 (reset = 0) ou 1 (set = 0).



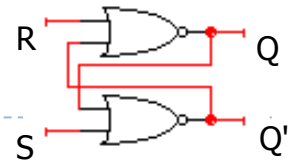


Comportamento temporal



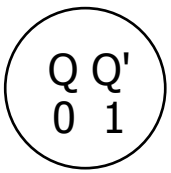


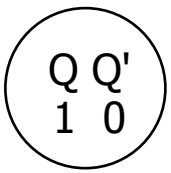
Comportamento estado latch R-S

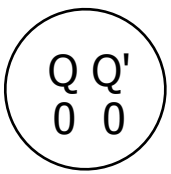


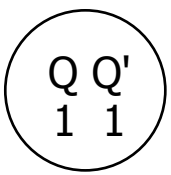
- Tabela verdade do comportamento latch R-S

S	R	Q
0	0	Segura
0	1	0
1	0	1
1	1	Instável

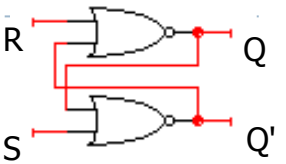




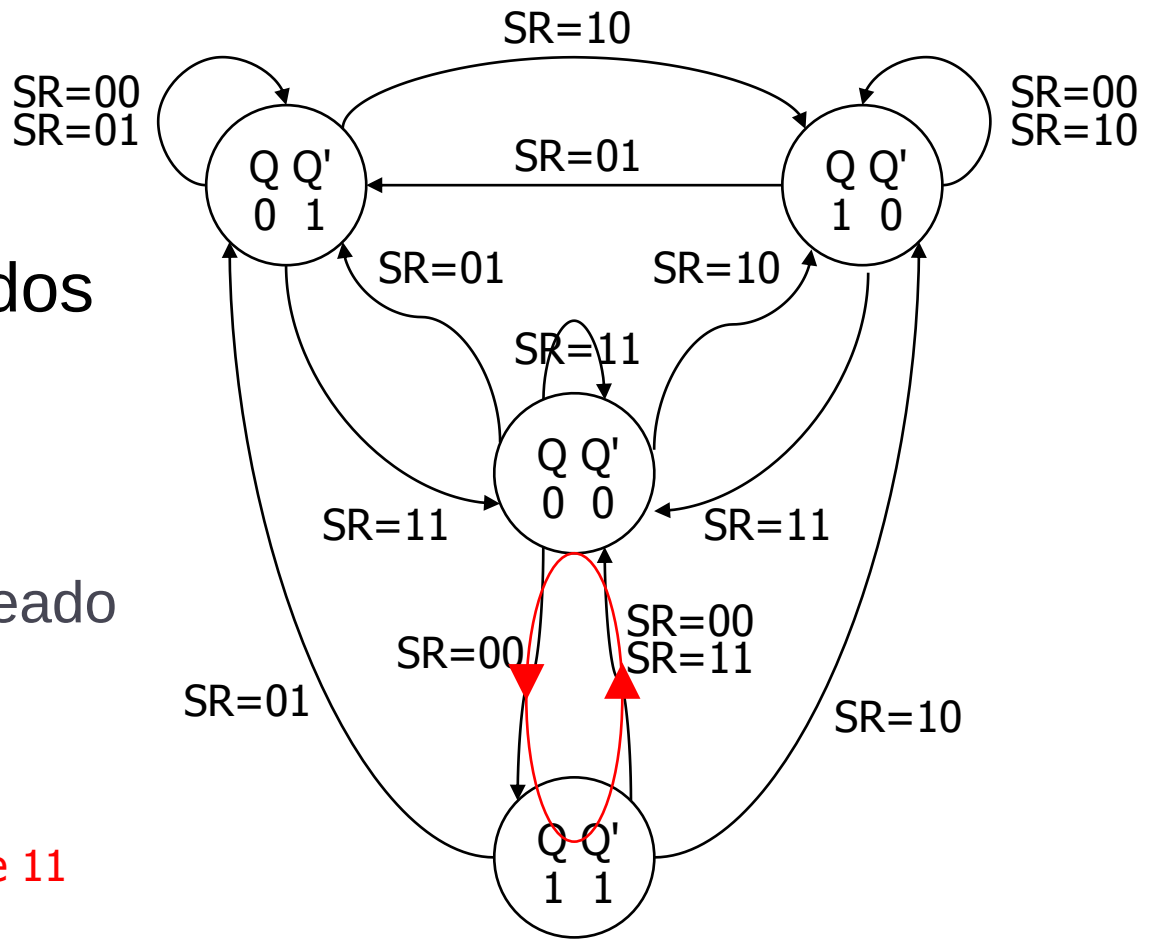




Teoria comportamento estado latch R-S



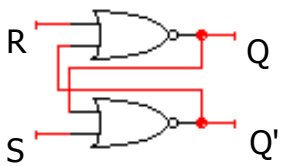
- Diagrama de estados
  - Estados: Valores possíveis
  - Transações: Modificações baseado nas entradas

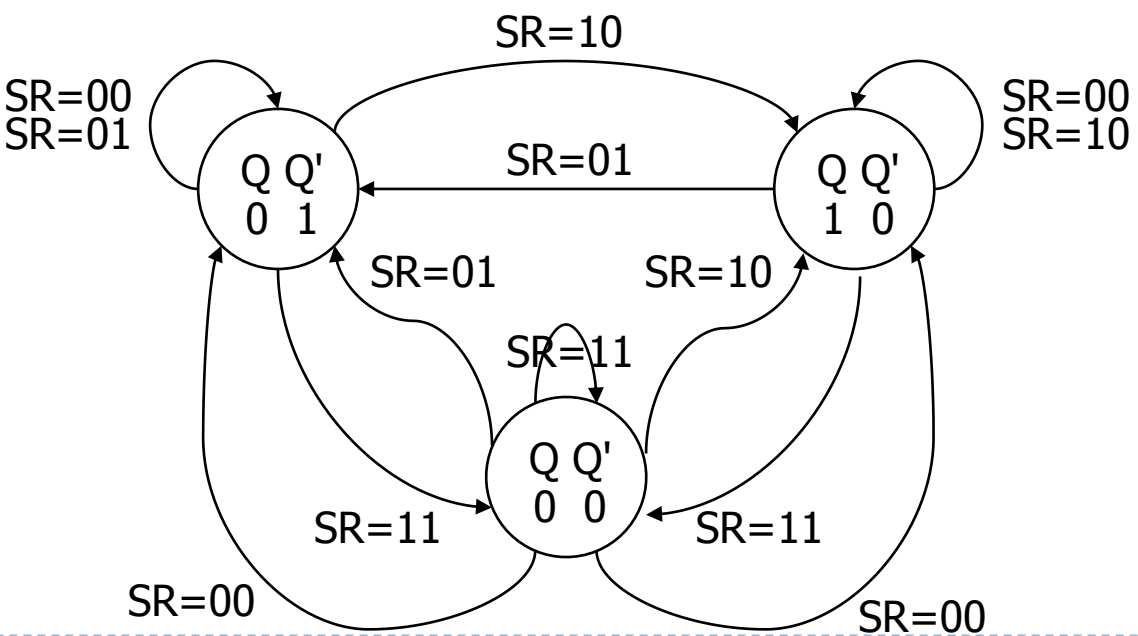


Possível oscilação entre os estados 00 e 11

Observando comportamento latch R-S

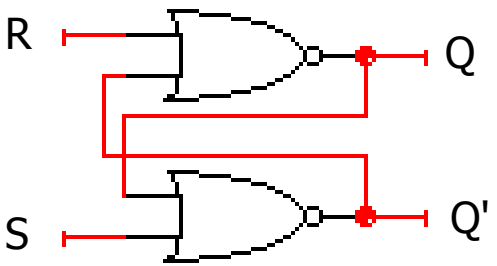
- Muito difícil de observar latch R-S no estado 1-1
  - R ou S geralmente muda primeiro
- Ambiguamente retorna para o estado 0-1 or 1-0
  - Chamado "condição de corrida"
  - Ou transição não determinística

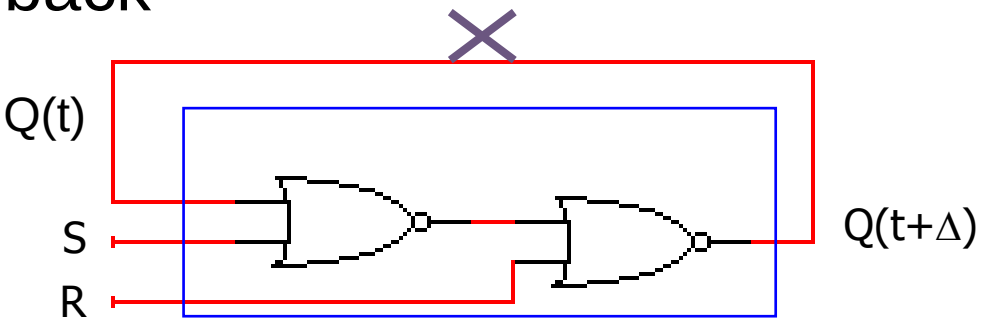




Análise latch R-S

- Quebrar caminho feedback





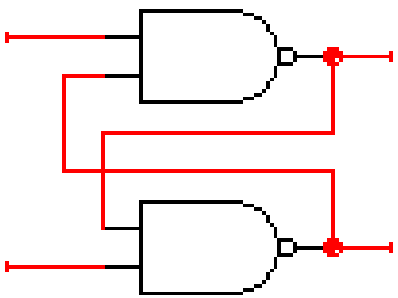
S	R	Q(t)	Q(t+Δ)	
0	0	0	0	Segura
0	0	1	1	
0	1	0	0	reset
0	1	1	0	
1	0	0	1	set
1	0	1	1	
1	1	0	X	Não permitido
1	1	1	X	

Q(t) \ S,R	0 0	0 1	1 1	1 0
0	0	0	X	1
1	1	0	X	1

Equação característica

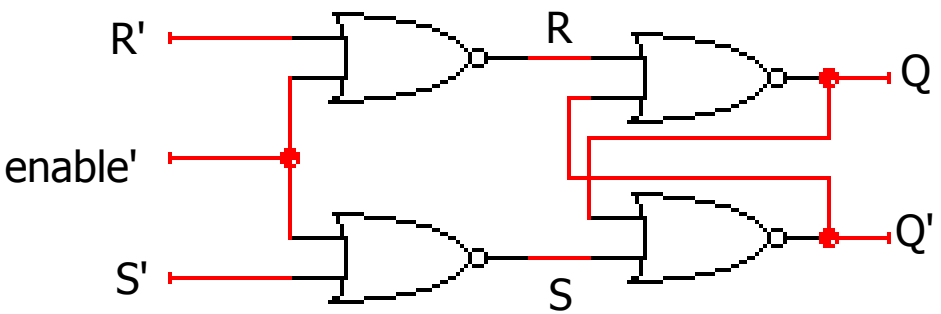
$Q(t+\Delta) = S + R' Q(t)$

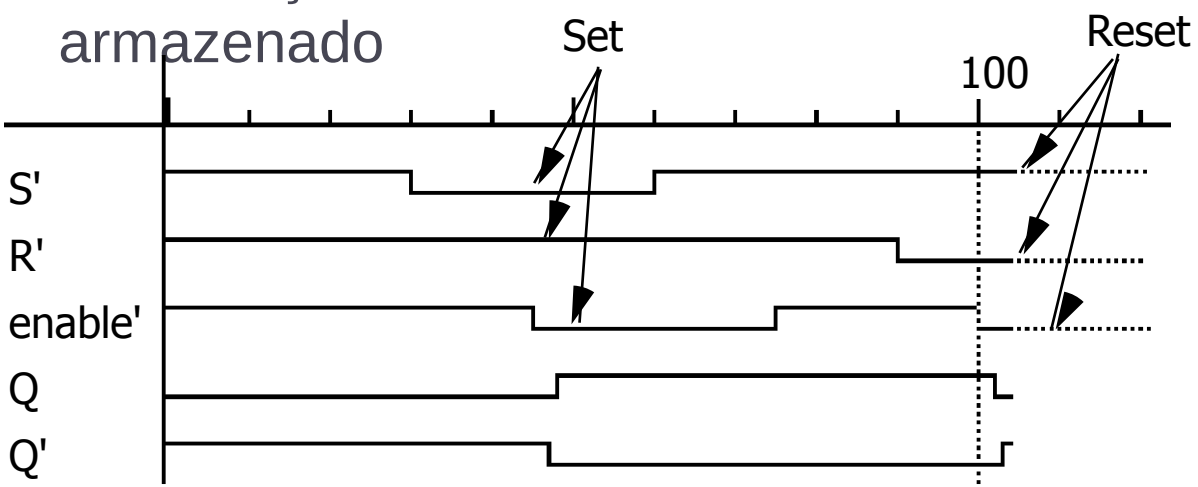
Atividade: latch R-S usando portas NAND



Latch R-S fechado

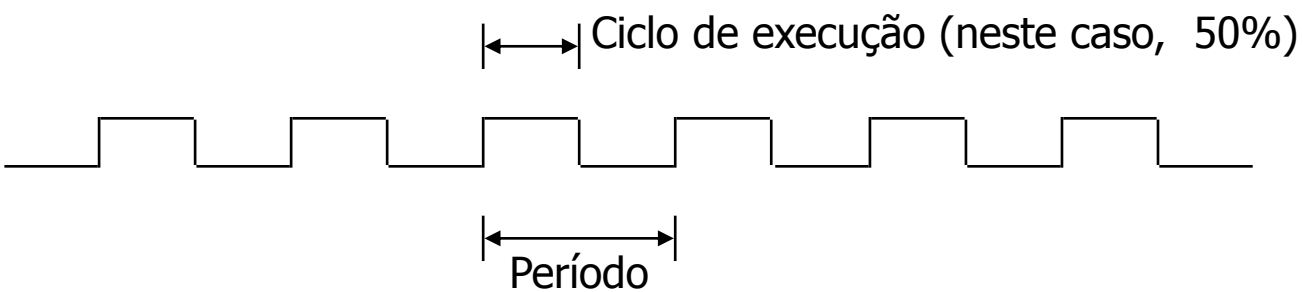
- Quando o controle de entradas R e S importa
  - Caso contrário, a menor falha em R ou S. Enquanto habilitada está baixa podendo causar modificações no valor armazenado





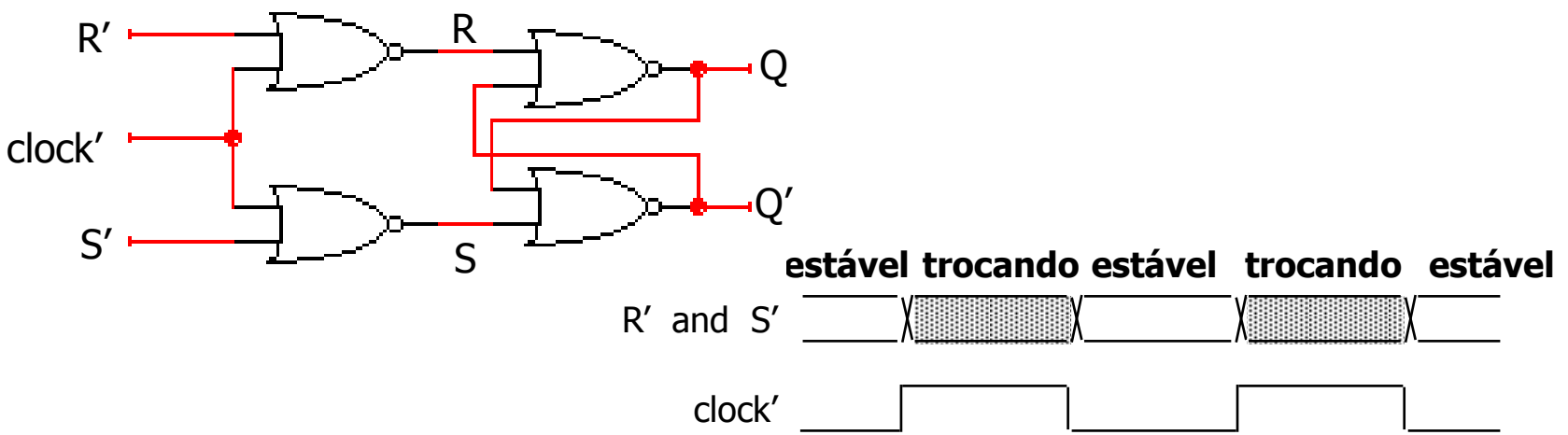
Clocks

- Usado para manter o tempo
  - Espera o tempo suficiente para entradas (R 'e S') para resolver
  - em seguida, permite efeito sobre o valor armazenado
- Clocks são sinais de período regular
  - Período (tempo entre instantes)
  - Ciclo de execução (tempo de clock é alto entre instantes – expressado como % do período)



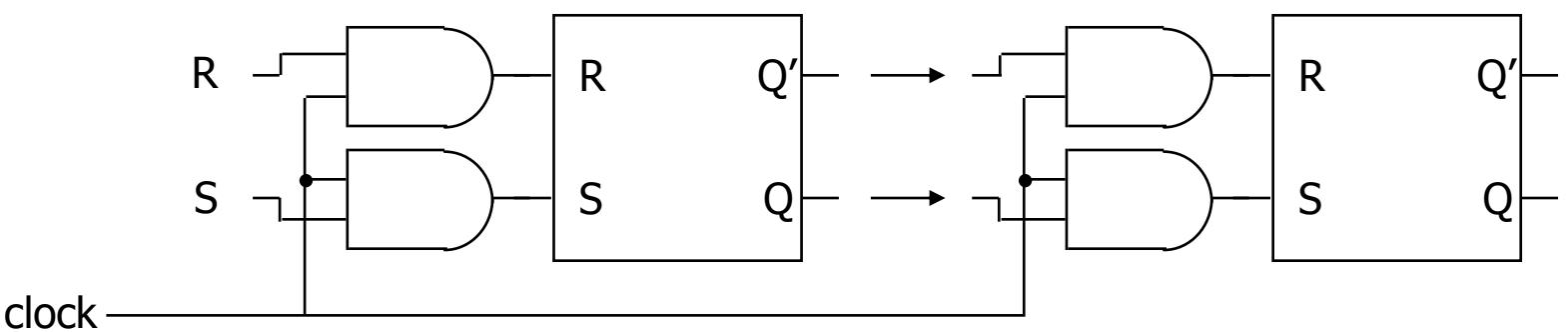
Clocks

- Controlando um latch R-S com um clock
  - Não podemos deixar que R e S se altere enquanto o clock está ativo (permitindo R e S passar)
  - Só tem metade do período de clock para troca de sinais se propagar
  - Sinais precisam ser estáveis para a outra metade do período de clock.



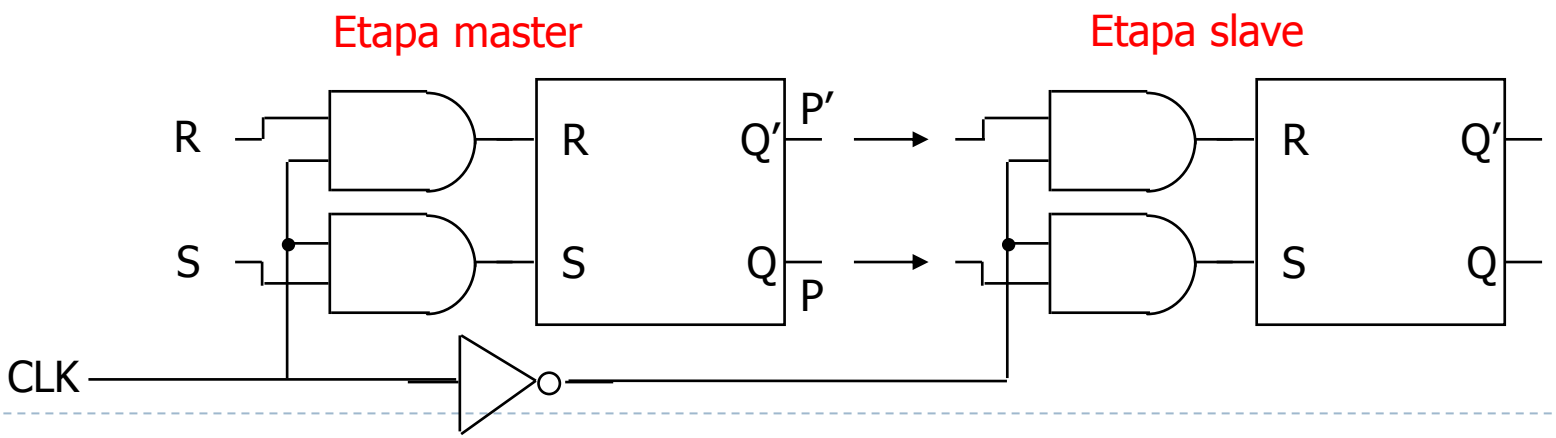
Cascateando latches

- Conecta a saída de um latch para entrada de outro
- Como parar as alterações das corridas através da cadeia (chain)?
  - Precisa ser capaz de controlar o fluxo de dados a partir de latch para o próximo
  - Move um latch por período de clock
  - Precisa se preocupar com a lógica entre latches (setas) que é muito rápido



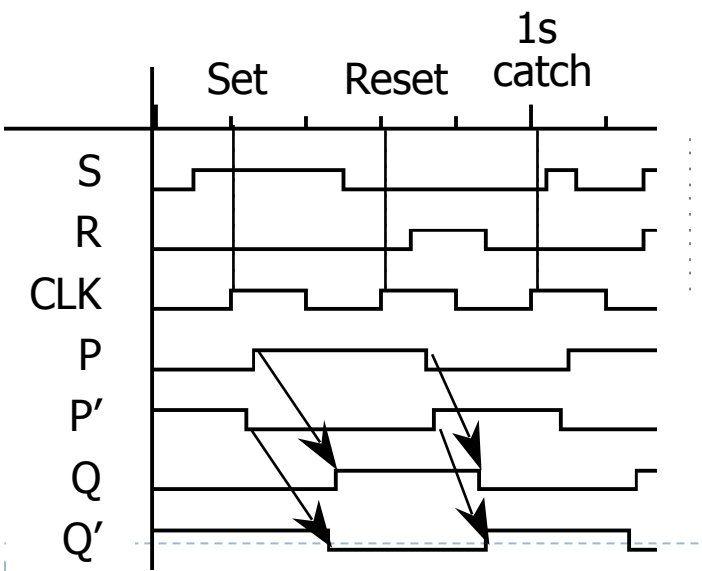
Estrutura Master-slave

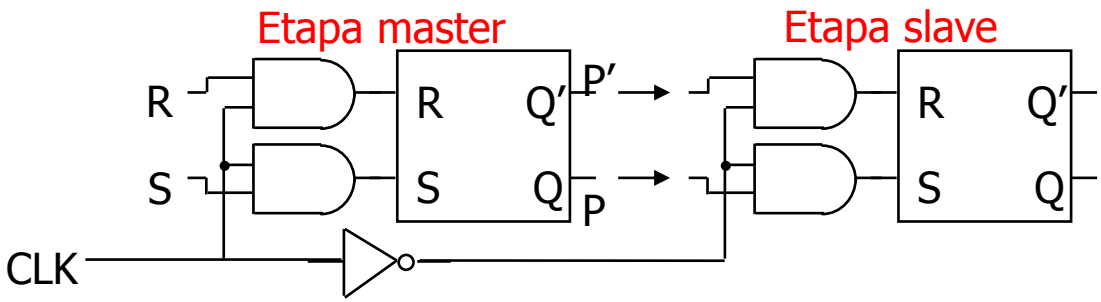
- Quebra o fluxo por modificação de clock (como um bloqueio de ar)
  - Usa clock positivo para entradas de latch dentro de um latch R-S
  - Usa clock negativo para alterar saídas com outro latch R-S
- Visualiza pares como uma unidade básica
  - Flip-flop master-slave
  - Duas vezes mais lógica
  - Saída alterar os atrasos na porta após a borda de descida do clock, mas não afeta os flip-flops em cascata



O problema catching 1s

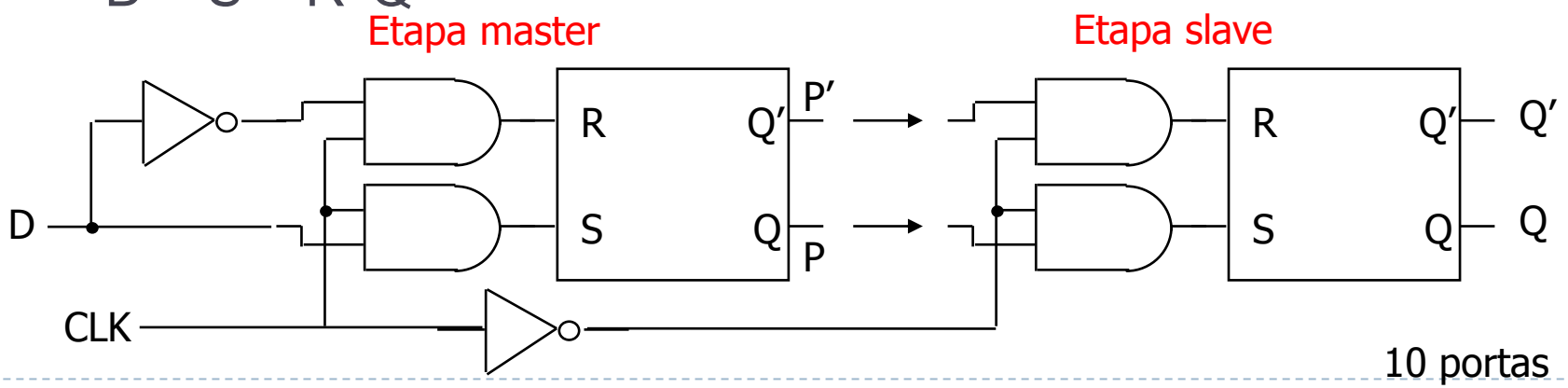
- Na primeira etapa R-S do FF master-slave
  - 0-1-0 falha em R ou S, enquanto o clock está alto e é “capturado" pelo etapa master
  - Conduz restrições na lógica para se livrar do hazard





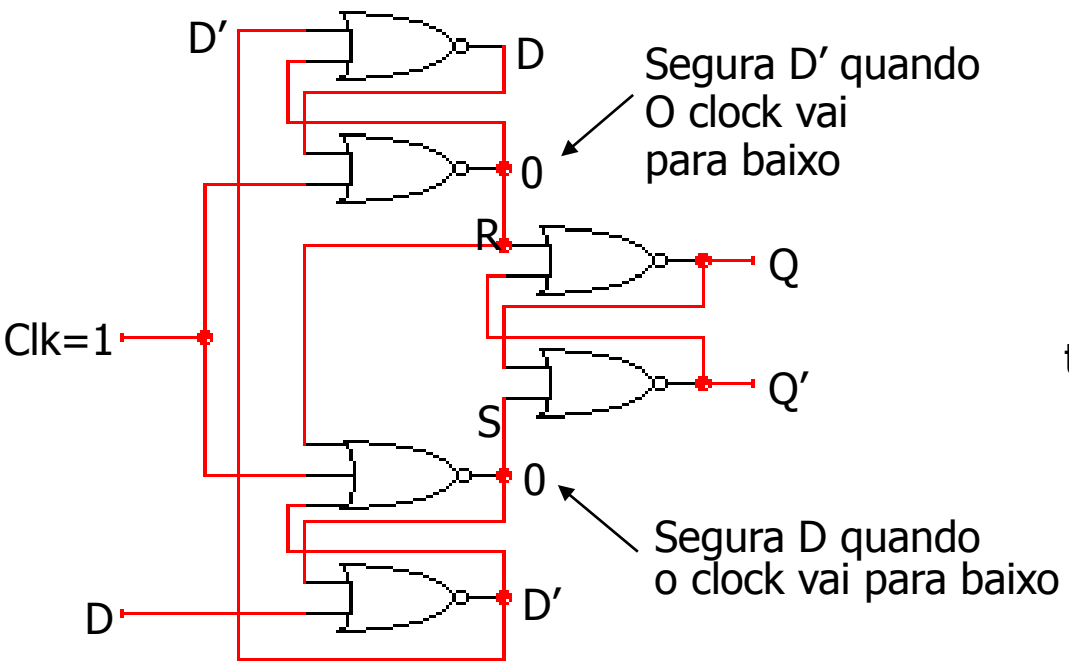
Flip-flop D

- Adiciona o complemento S e R uns dos outros
  - Elimina o problema catching 1s
  - Não pode simplesmente manter o valor anterior (precisa ler um novo valor a cada período de tempo)
  - Valor de somente D antes do clock vai para baixo e é armazenado no flip flop
  - Pode fazer um flip-flop R-S adicionando lógica para se fazer D = S + R' Q



Flip-flops disparados por borda

- Mais eficiente solução: Somente 6 portas
  - Sensível a fatores de produção só próximo da borda do sinal de clock (não enquanto alto)

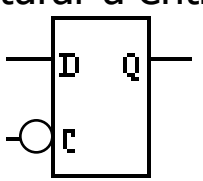


D disparado por borda negativa

flip-flop (D-FF)

Atrasos porta 4-5

Precisa de uma respectiva configuração e segura a restrição de tempo para capturar a entra com sucesso

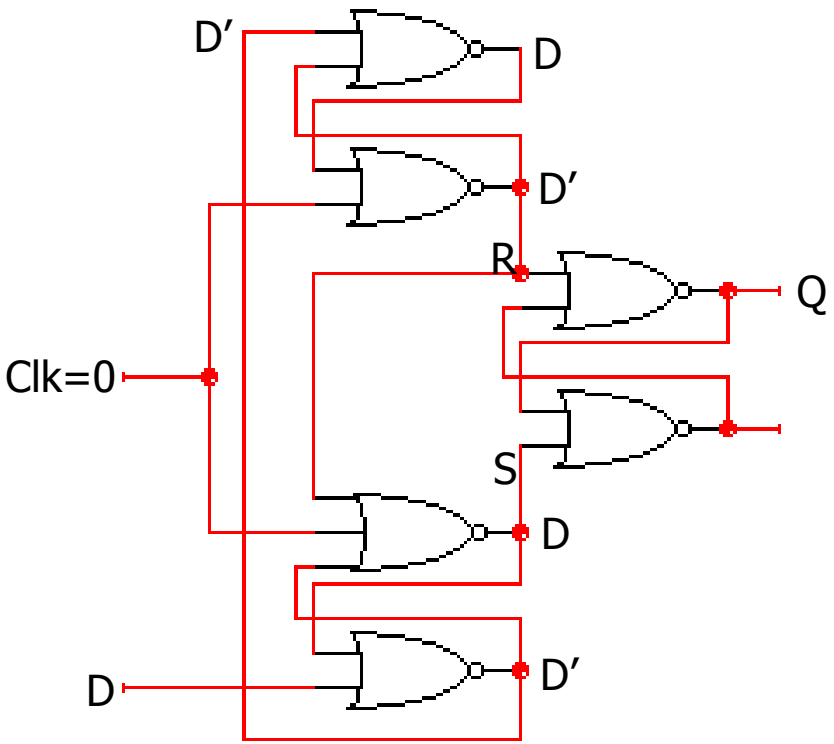


Equação característica

$Q(t+1) = D$

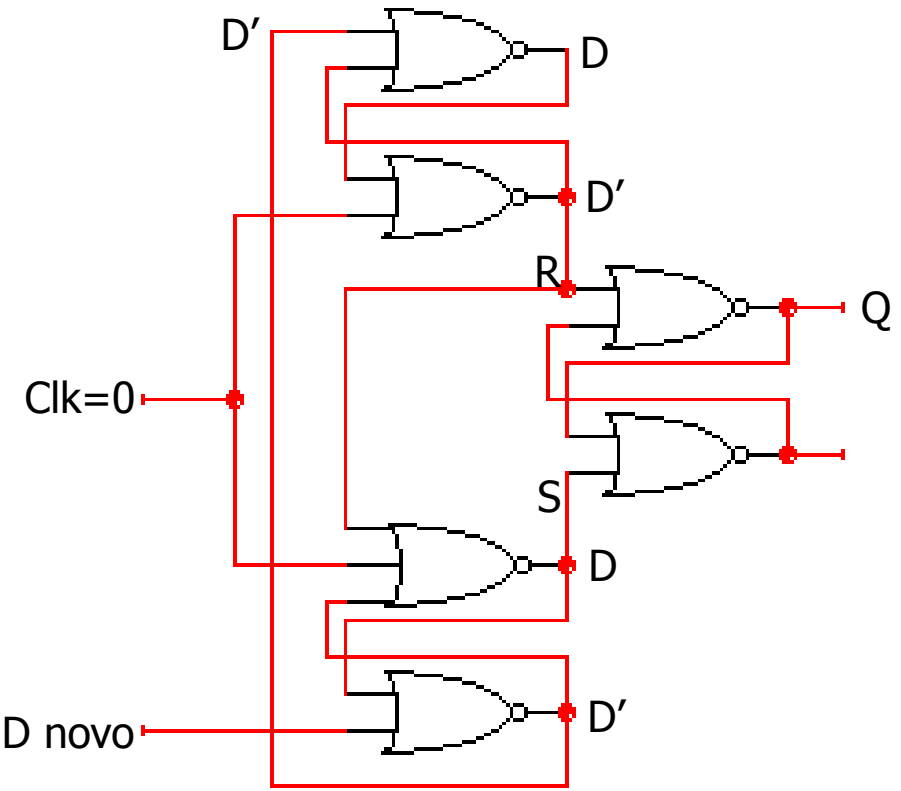
Flip-flops disparados por borda

- Análise passo a passo



Quando o clock vai de alto para baixo
o dado é travado

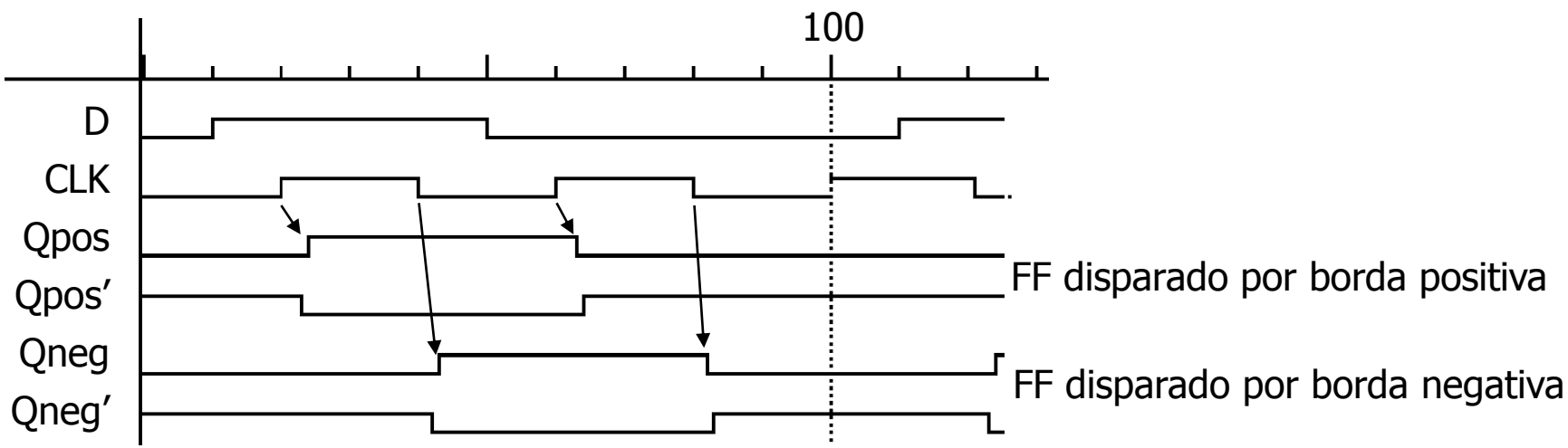
D novo ≠ D antigo



Quando o clock é baixo
o dado é retido

Flip-flops disparados por borda

- Disparado por borda de subida
  - Entradas mostradas na borda de subida; Saídas alteradas depois da borda de subida
- Flip-flops disparados por borda negativa
  - Entradas mostradas na borda de descida; Saídas alteradas depois da borda de descida

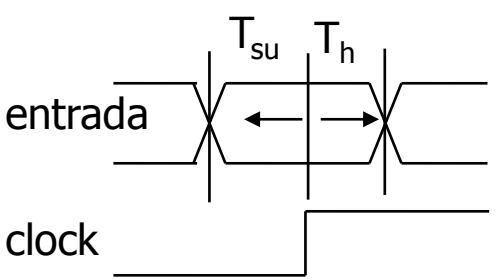


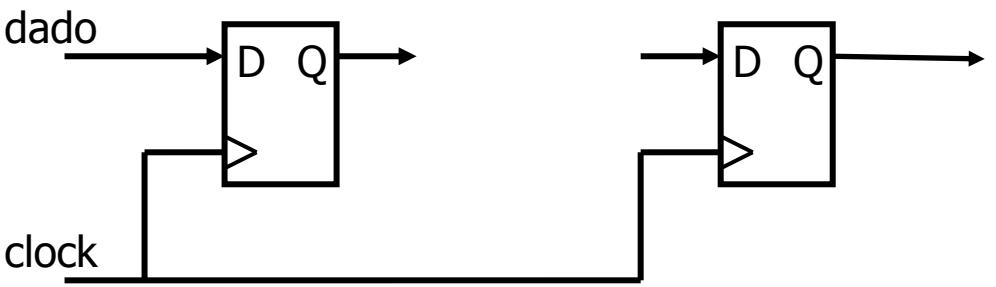
Metodologias de temporização

- Regras para componentes interconectados e clocks
  - Garante o funcionamento adequado do sistema quando estritamente seguido
- Abordagem depende da construção de blocos utilizados pelos elementos da memória
  - Vamos nos concentrar em sistemas com flip-flops disparados por borda
    - Encontrado em dispositivos lógicos programáveis
  - Muitos circuitos integrados costumam se concentrar em latches sensíveis de nível
- Regras básicas para temporização correta:
  - (1) Entradas corretas, com o respectivo tempo são fornecidas para os flip-flops
  - (2) Nenhum flip-flop altera o estado mais que uma vez por evento de ciclo de clock

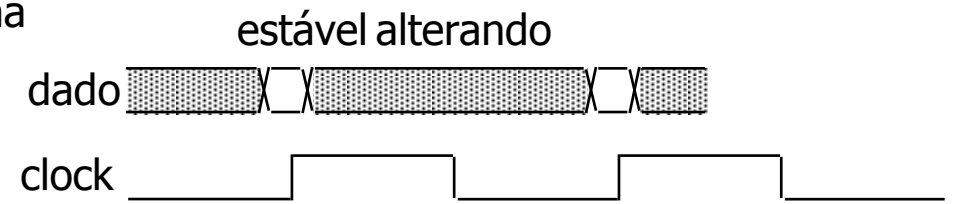
Metodologias de temporização

- Definição de termos
  - clock: Evento periódico, causa a troca do estado do elemento de memória. Pode ser borda de subida ou descida ou nível alto ou nível baixo;
  - Tempo de preparação: Tempo mínimo antes do evento de clock. A entrada precisa ser estável (Tsu).
  - Tempo de espera: Tempo mínimo depois do evento de clock até que a entrada se permaneça estável (Th).

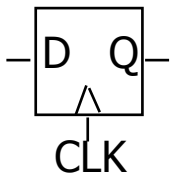




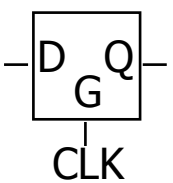
Existe uma "janela" de tempo próxima ao evento de clock. Durante este evento a entrada precisa ser estável e não se alterar de ordem, para se reorganizar.



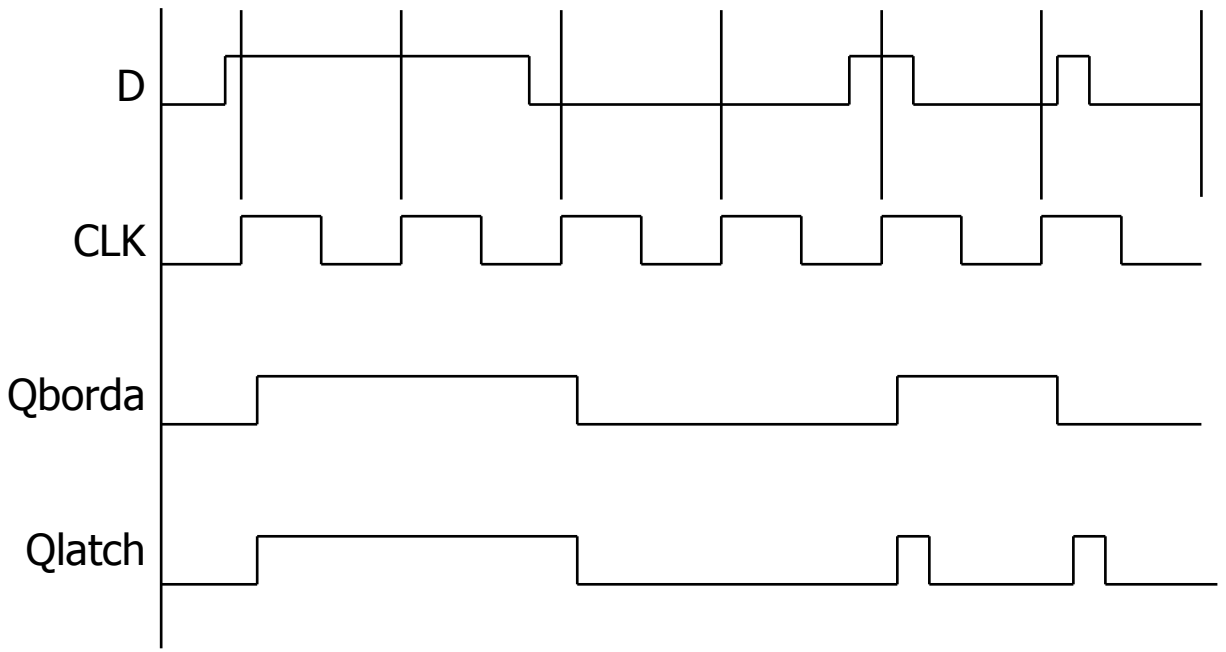
Comparação de latches e flip-flops



Flip-flop
disparado por borda



Transparente
latch
(Sensível por nível)



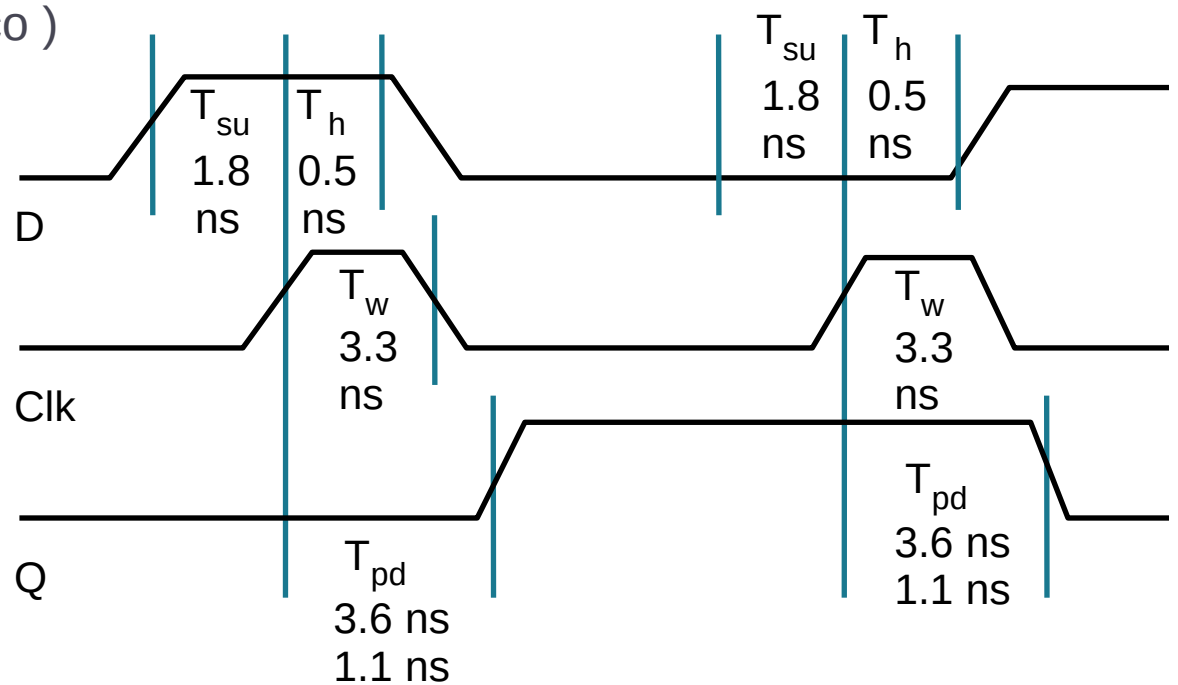
Comportamento é o mesmo, a menos se houver mudanças na entrada quando o clock for alto

Comparação de latches e flip-flops (cont'd)

Tipo	**Quando as entradas são mostradas**	**Quando a saída é válida**
Latch desbloqueado	Sempre	Atraso de propagação a partir da mudança da entrada
Latch Sensível por nível	Clock alto (Tsu/Th próximo a borda de descida do clock)	Atraso de propagação a partir da mudança da entrada ou borda de clock (O que ocorrer mais tarde)
Flip-flop Master – Slave	Clock alto (Tsu/Th próximo a borda de descida do clock)	Atraso de propagação a partir da borda de descida do clock
Flip-flop disparado por borda negativa	Transição do clock hi-para-lo (Tsu/Th próximo a borda de descida do clock)	Atraso de propagação a partir da borda de descida do clock

Especificações temporização típica

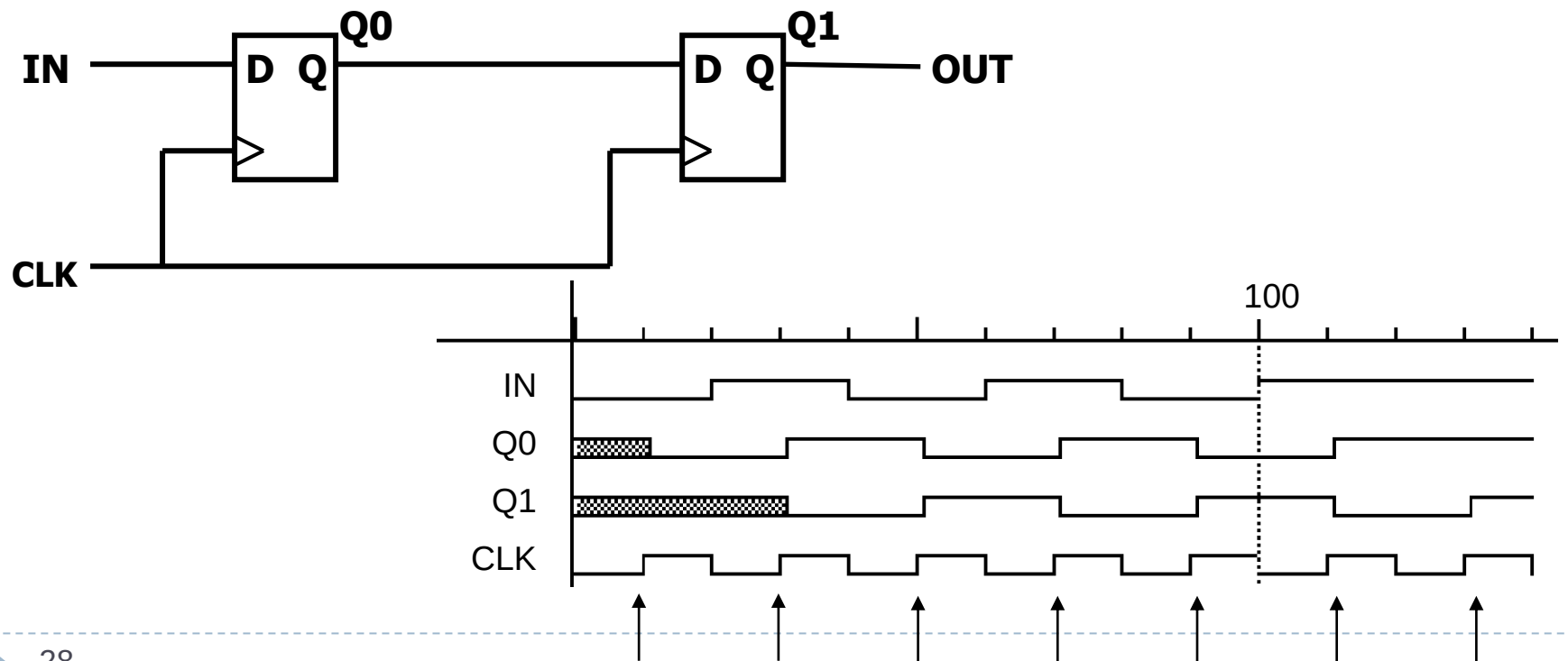
- Flip-flop D disparado por borda positiva
  - Tempos de preparação e espera
  - Largura mínima do clock
  - Atrasos de propagação (Baixo para alto, alto para baixo, máximo e típico)



Todas as medições são feitas a partir do evento de clock (borda de subida do clock)

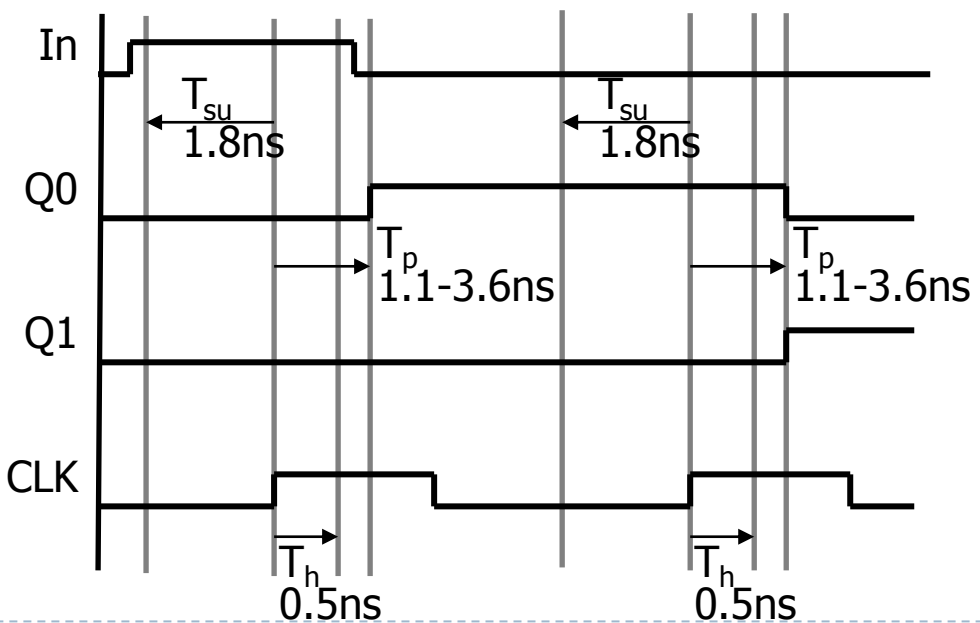
Flip-flops disparados por borda cascateados

- Registrador de deslocamento
  - Novo valor entra no primeiro estágio
  - Enquanto o valor anterior da primeira fase entrar na segunda fase
  - Considerar o atraso de propagação/espera/preparação (propação precisa ser maior > espera)



Flip-flops disparados por borda cascateados

- Por que isso funciona?
  - Atraso de propagação excede o tempo de espera
  - Restrição da largura do clock excede tempo de preparação
  - Isto garante que a etapa seguinte travará o valor atual antes de mudar para o novo valor

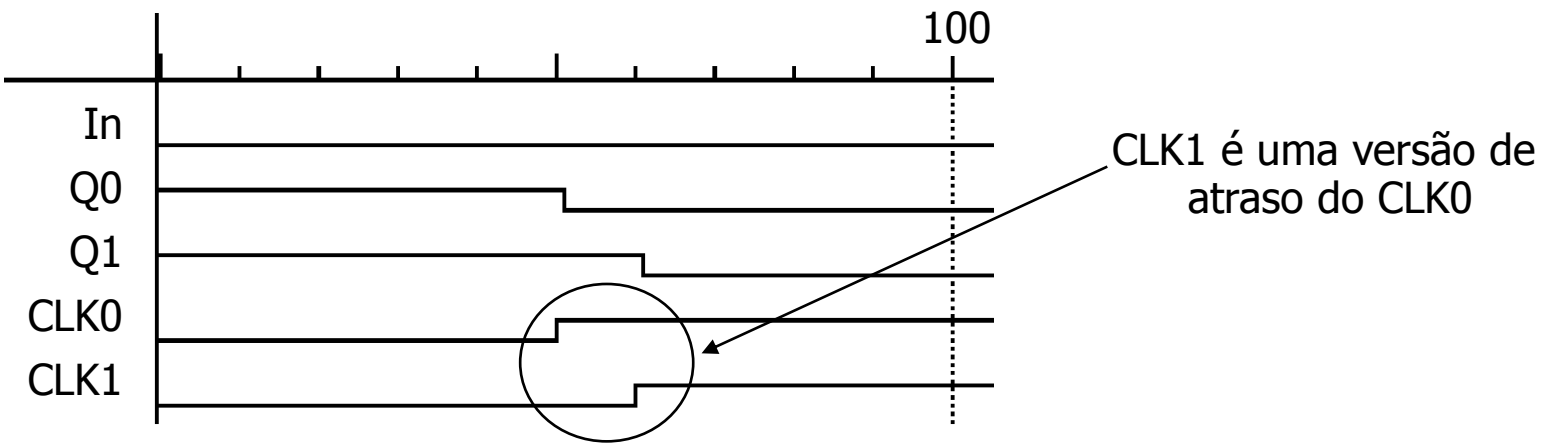


Restrições temporais garantem a devida operação dos componentes em cascata

Assume infinitamente a distribuição rápida do clock

Clock skew

- O problema
  - Comportamento correto assume o próximo estado de todos os elementos de armazenamento determinado por todos os elementos de armazenamento ao mesmo tempo
  - Isto é difícil em sistemas de alto desempenho, porque o tempo para o clock chegar ao flip-flop é comparável aos atrasos através da lógica
  - Efeito do skew em flip-flops cascateados:



Estado original: IN = 0, Q0 = 1, Q1 = 1
Devido ao skew, o próximo estado torna-se: Q0 = 0, Q1 = 0, and not Q0 = 0, Q1 = 1

Resumo de latches e flip-flops

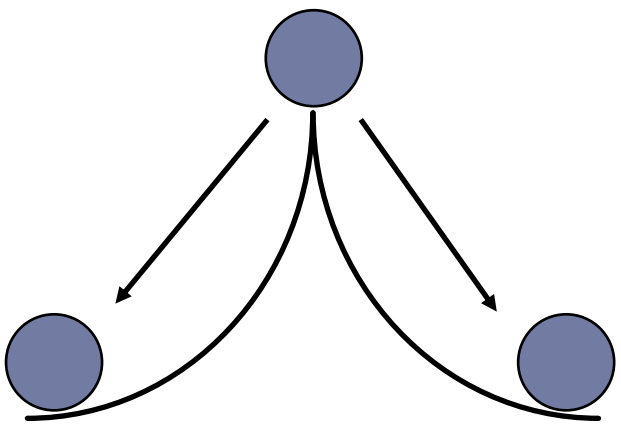
- Desenvolvimento do FF-D
  - Sensível ao nível usado em circuitos integrados personalizados
    - Pode ser feita com 4 interruptores
  - Usado em dispositivos lógicos programáveis disparado por borda
  - Boa escolha para registrador de armazenamento de dados
- Historicamente FF J-K foi mais popular mas nunca utilizado
  - Similar ao R-S mas com 1-1 sendo usado para alternar a saída (estado complementar)
  - Bom em dias de TTL / SSI (mais com função de entrada complexo: D = Q JQ '+ K'
  - Não é uma boa escolha para PALs / PLAs pois requer 2 entradas
  - Sempre pode ser implementado usando FF-D
- Predefinido e contribuições claras são altamente desejáveis em flip-flops
  - Utilizado no início ou para redefinir osistema para um estado conhecido

Entradas metastability e assíncrona

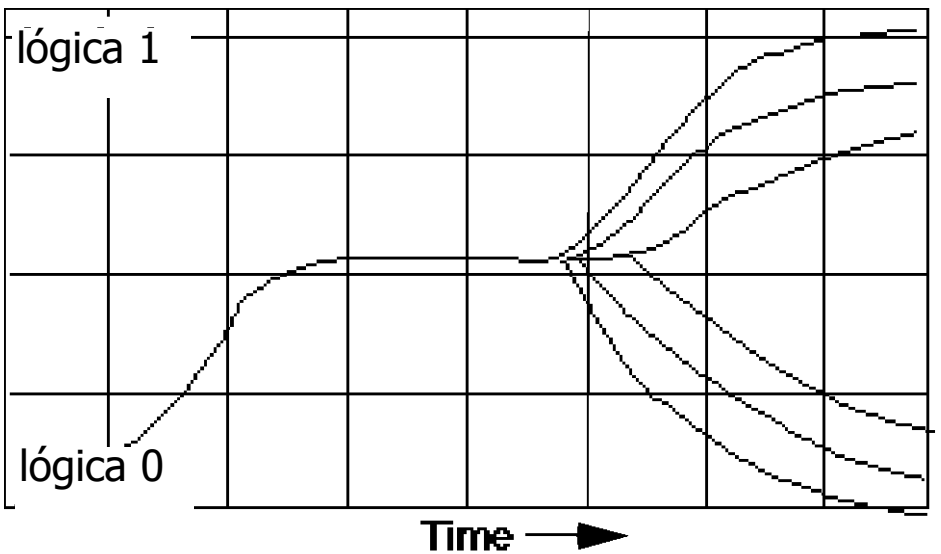
- Circuitos síncronos cronometrados
  - Entradas, estado, e saídas apresentadas ou alteradas em relação ao sinal de referência comum (chamado clock)
  - Ex.: master/slave, disparada por borda
- Circuitos assíncronos
  - Entradas, estado, e saídas apresentadas ou alteradas independentemente do sinal de referência comum (glitches/hazards são uma grande preocupação)
  - Ex.: Latch R-S
- Entrada assíncrona para circuitos síncrono
  - Entradas podem alterar qualquer tempo, não encontrará tempos preparação/espera
  - Perigosas, entradas síncronas são muito preferidas
  - Não pode ser evitado (Ex., sinal reset, espera memória, entrada usuário)

Falha de sincronização

- Ocorre quando a entrada FF muda próximo a borda de clock
  - O FF pode entrar em um estado metastable – nem uma lógica 0 ou 1
  - Ele pode permanecer neste estado uma quantidade indefinida de tempo
  - Não é provável na prática, mas tem alguma probabilidade



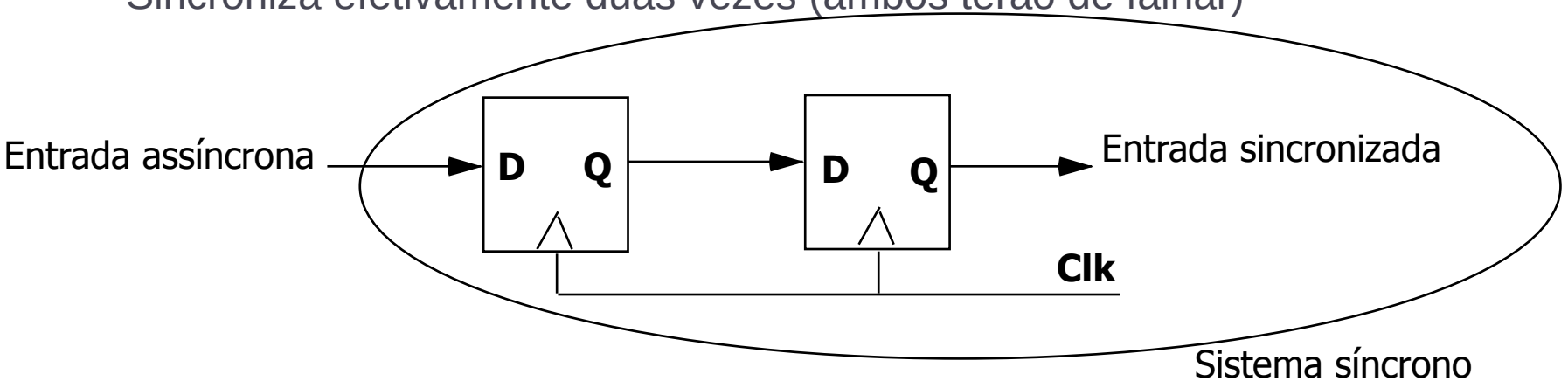
Pequeno, mas a probabilidade diferente de zero é que a saída FF vai ficar preso em um estado intermediário



Traços do osciloscópio demonstram falha do sincronizador e eventual deterioração do estado estacionário

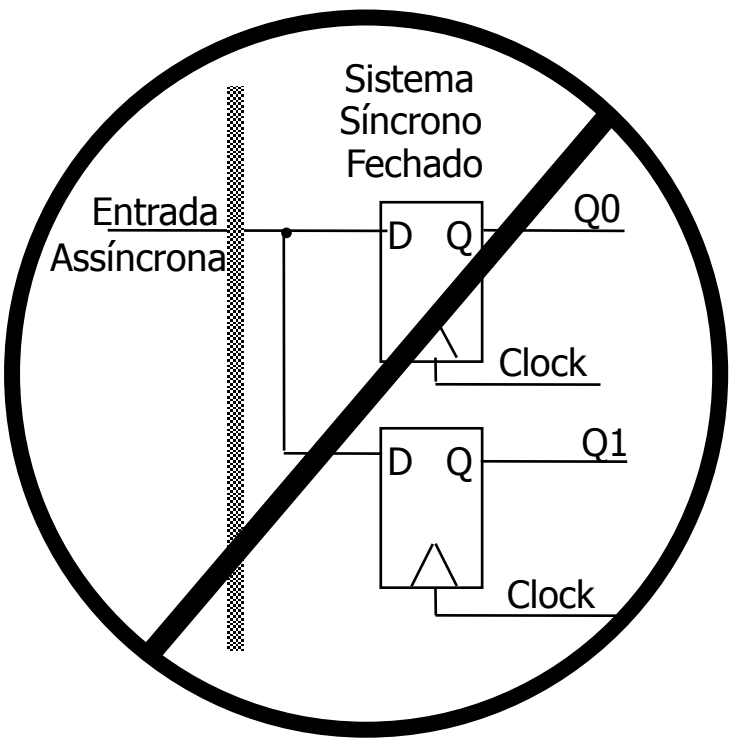
Lidar com a falha de sincronização

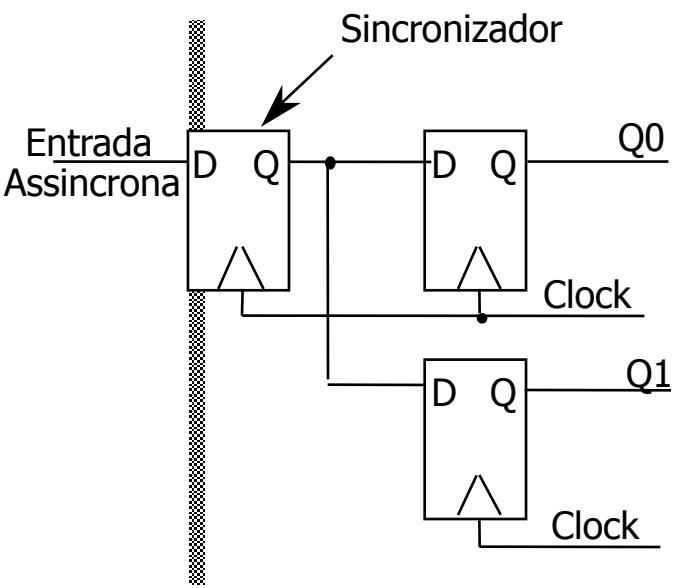
- Probabilidade de falha nunca pode ser reduzido para 0, mas pode ser reduzido
  - (1) Atrasar o sistema de clock
    Dá ao sincronizador mais tempo para decair em um estado estacionário;
    Falha do sincronizador torna-se um grande problema para os sistemas de altíssima velocidade
  - (2) Use a tecnologia de lógica o mais rápido que possível no sincronizador, this makes for a very sharp "peak" upon which to balance
  - (3) Em cascata dois sincronizadores
    Sincroniza efetivamente duas vezes (ambos terão de falhar)



Manipulação de entradas assíncronas

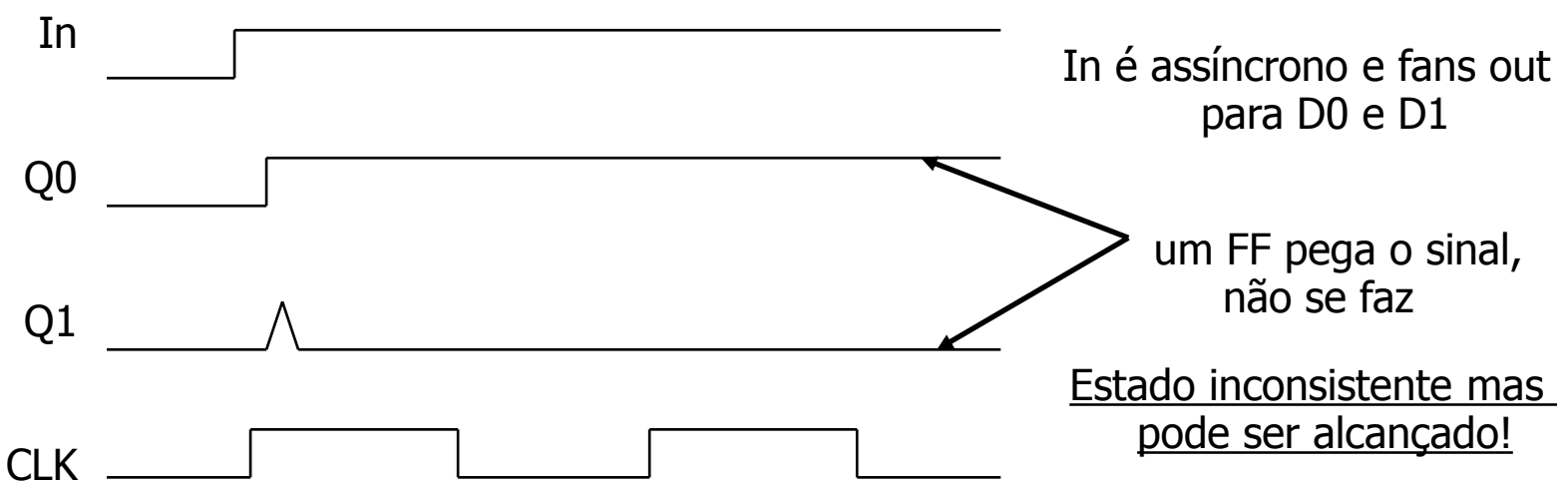
- Nunca permite entradas assíncronas para fan-out para mais que um flip-flop
  - Sincronizar o mais rápido que possível e, em seguida, tratar o sinal síncrono





Manuseio de entradas assíncronas (cont 'd)

- O que pode dar errado?
  - Entrada muda muito próxima do clock (violando a restrição de tempo de preparação)

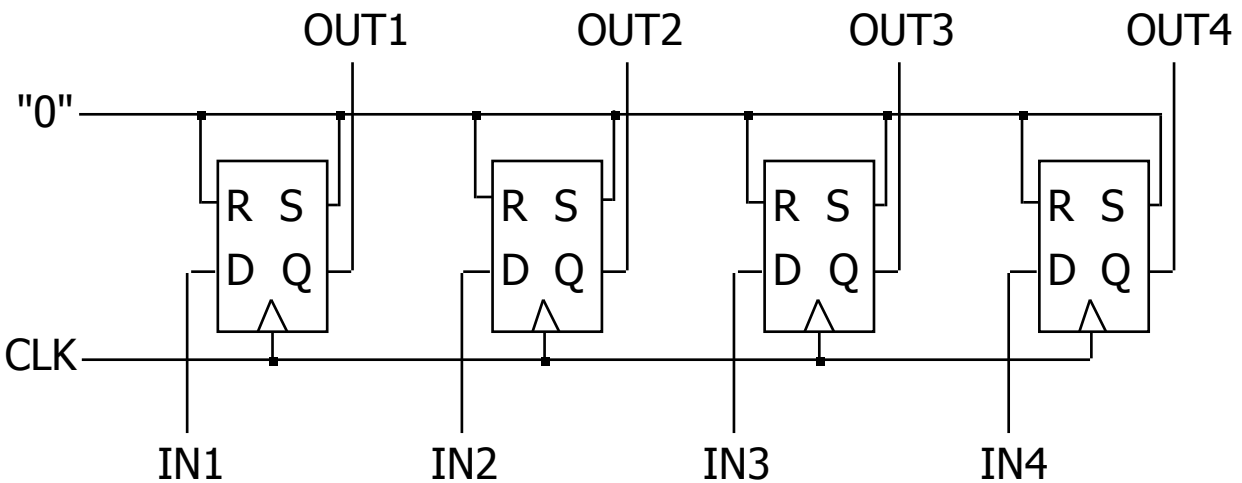


Características Flip-flop

- Reset (estado set para 0) – R
  - Síncrono: Dnew = R' • Dold (quando chega próximo da borda de clock)
  - Assíncrono: não espera pelo clock, rápido, mas perigoso
- Preset or set (estado set parar 1) – S (ou P as vezes)
  - Síncrono: Dnew = Dold + S (quando chega próximo da borda de clock)
  - Assíncrono : não espera pelo clock, rápido, mas perigoso
- Ambos reset e preset
  - Dnew = R' • Dold + S (set-dominante)
  - Dnew = R' • Dold + R'S (reset-dominantet)
- Capacidade de entrada seletiva (entrada enable ou load) – LD or EN
  - Multiplexador na entrada: Dnew = LD' • Q + LD • Dold
  - Pode carregar ou não pode sobrepor reset/set (usualmente R/S tem prioridade)
- Saidas complementares – Q e Q'

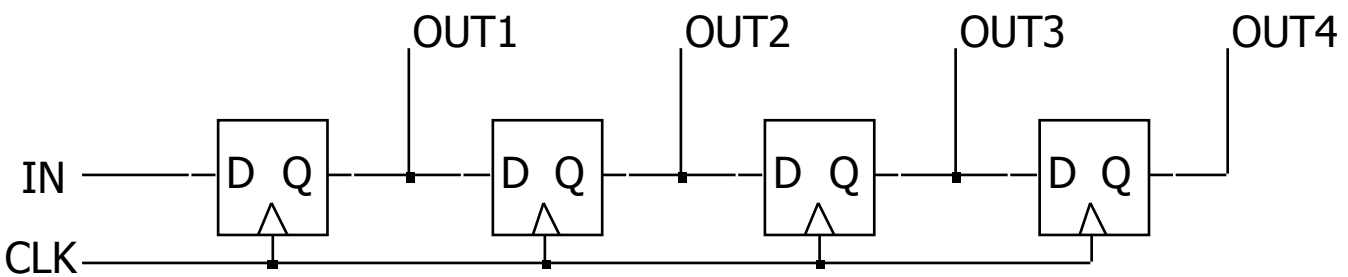
Registradores

- Coleção de flip-flops com controles similar e lógica
  - Valores armazenados de alguma forma relacionada (por exemplo, formam valor binário)
  - Clock compartilhado, reset, e linhas set
  - Lógica similar para cada etapa
- Exemplos
  - Registradores de deslocamento
  - Contadores



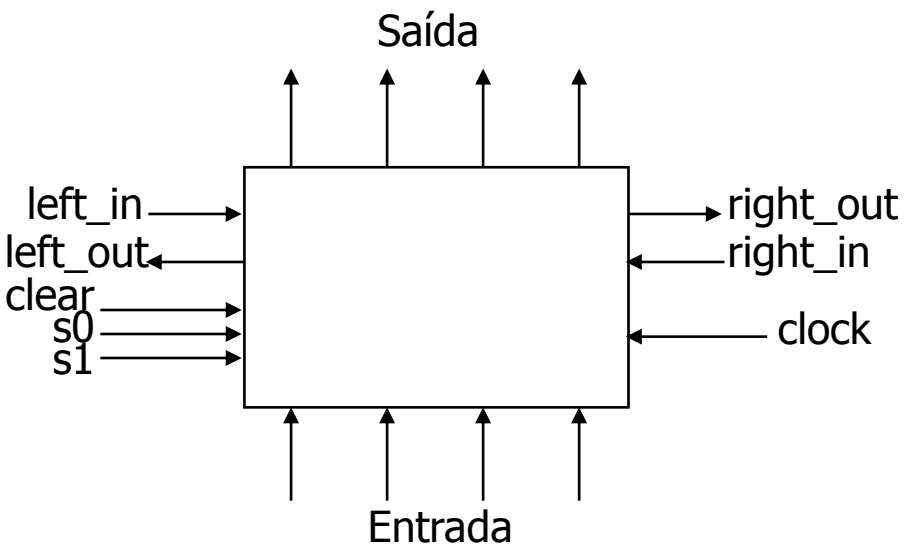
Registradores de deslocamento

- Armazena amostras de entrada
  - Armazenar os últimos 4 valores de entrada em sequência
  - Registrador de deslocamento 4-bit:



Registrador deslocamento universal

- Retêm 4 valores
  - Entradas serial e paralelas
  - Saídas serial e paralelas
  - Permite deslocamento para esquerda e direita
  - Mudança de novos valores a partir da esquerda ou direita



Limpa o conjunto do conteúdo dos registradores e saída para 0

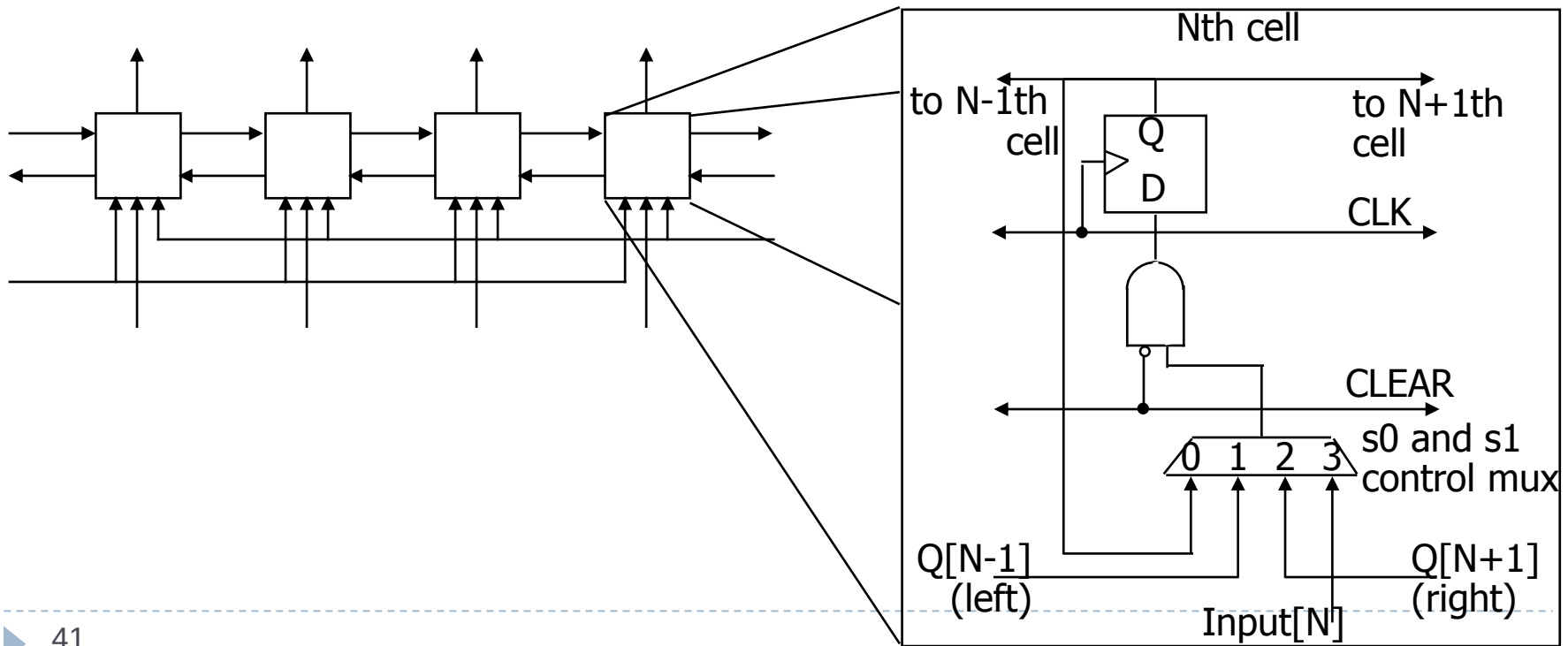
s1 e s0 determinam a função de deslocamento

s0	s1	função
0	0	estado de espera
0	1	deslocamento à direita
1	0	deslocamento à esquerda
1	1	carrega novo valor

Projeto registrador deslocamento universal

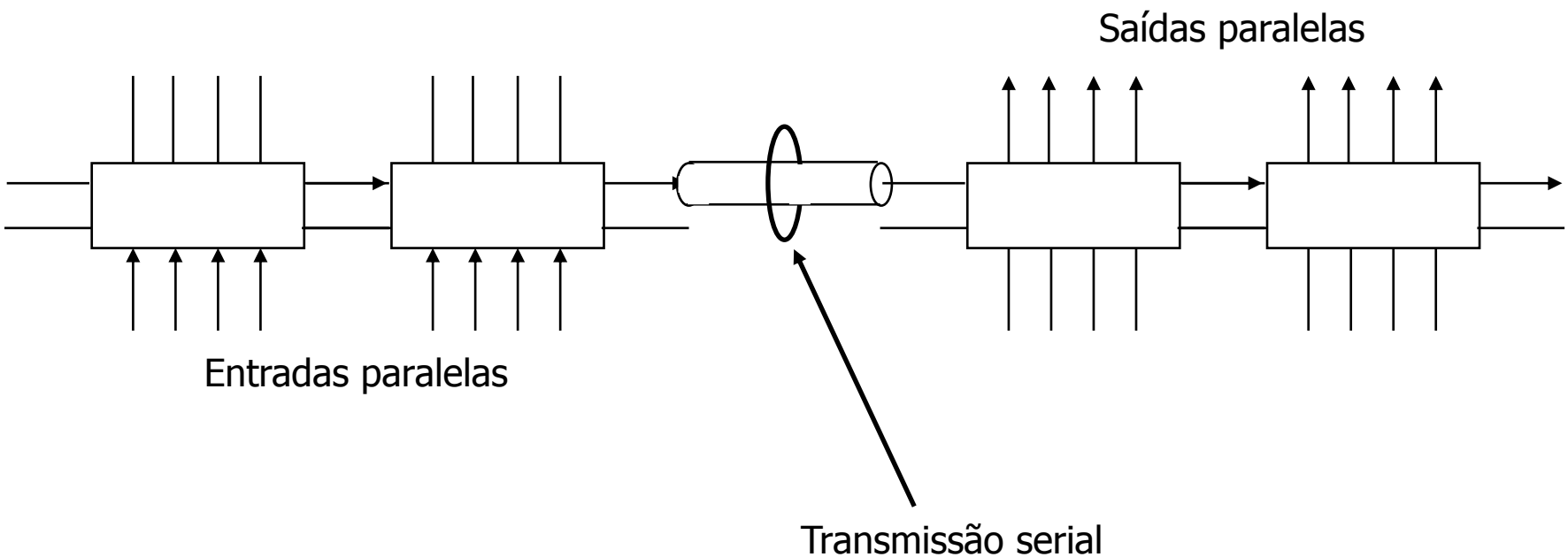
- Considera um dos quatro flip-flops
  - Novo valor no próximo ciclo de clock:

limpar	s0	s1	novo valor
1	–	–	0
0	0	0	saída
0	0	1	valor da saída do FF para esquerda (shift right)
0	1	0	valor da saída do FF para direita (shift left)
0	1	1	entrada



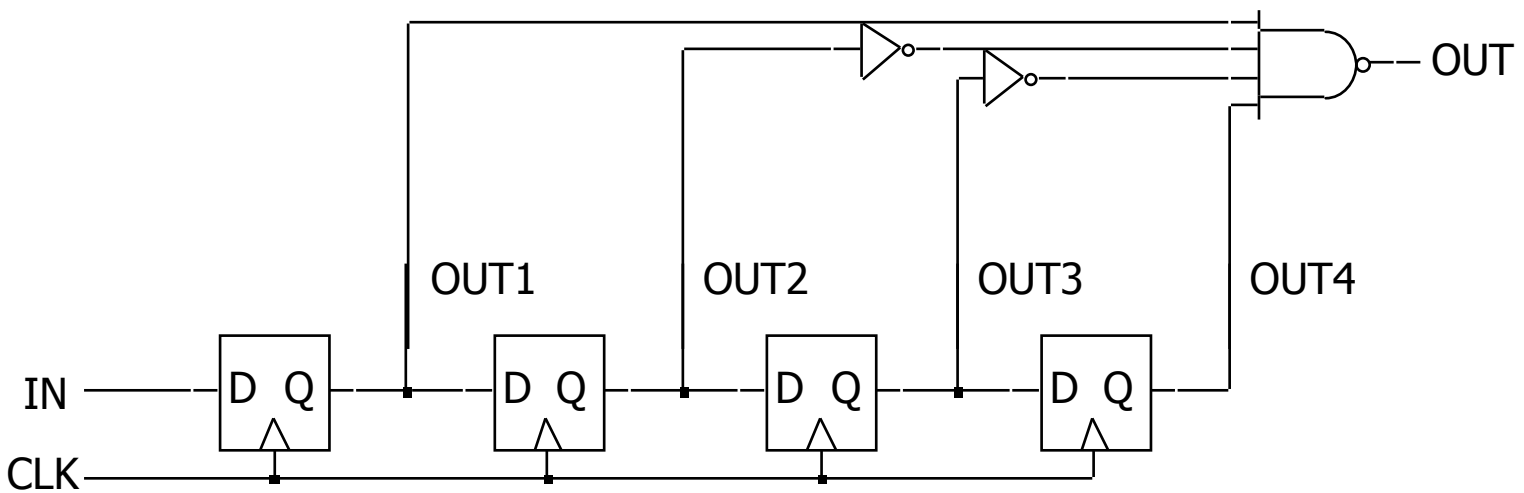
Aplicação registrador deslocamento

- Conversão paralelo em serial para transmissão serial



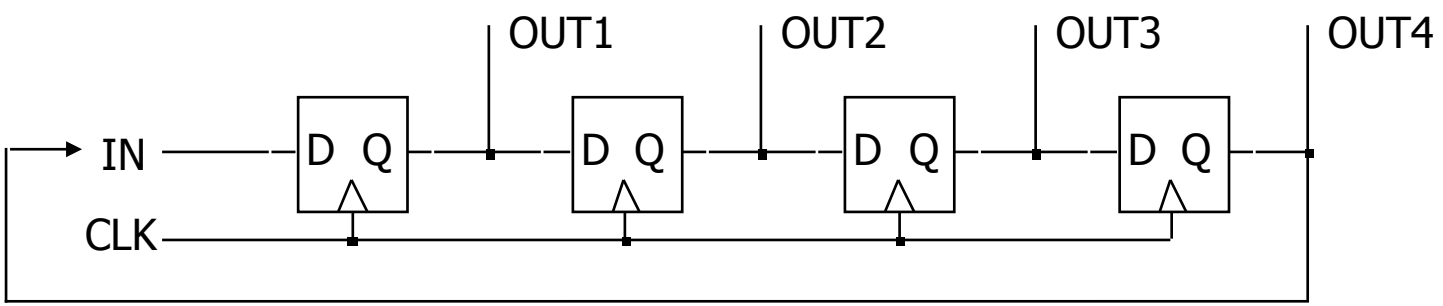
Padrão reconhecedor

- Função combinacional das amostras da entrada
  - Neste caso, reorganizando o padrão 1001 em um sinal de entrada simples



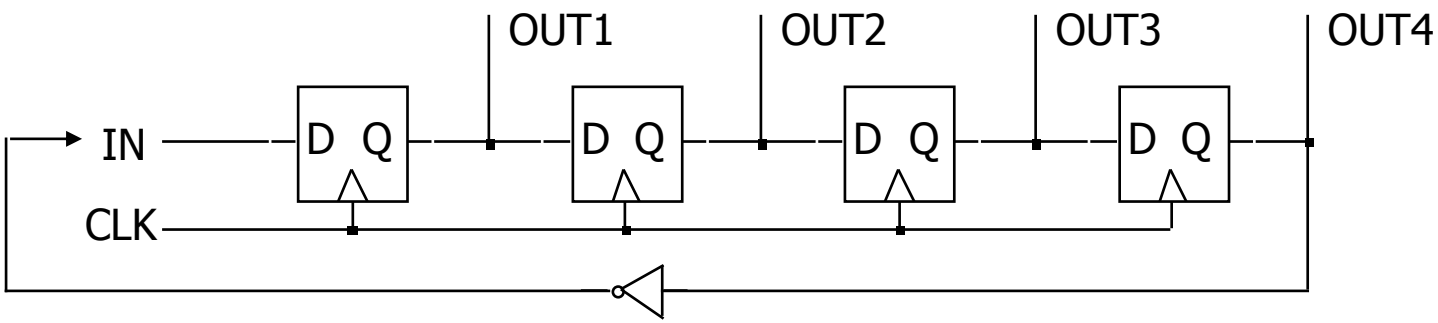
Contadores

- Sequência através de um conjunto fixado de padrões
  - Neste caso, 1000, 0100, 0010, 0001
  - Se um dos padrões é o estado inicial (loading ou set/reset)



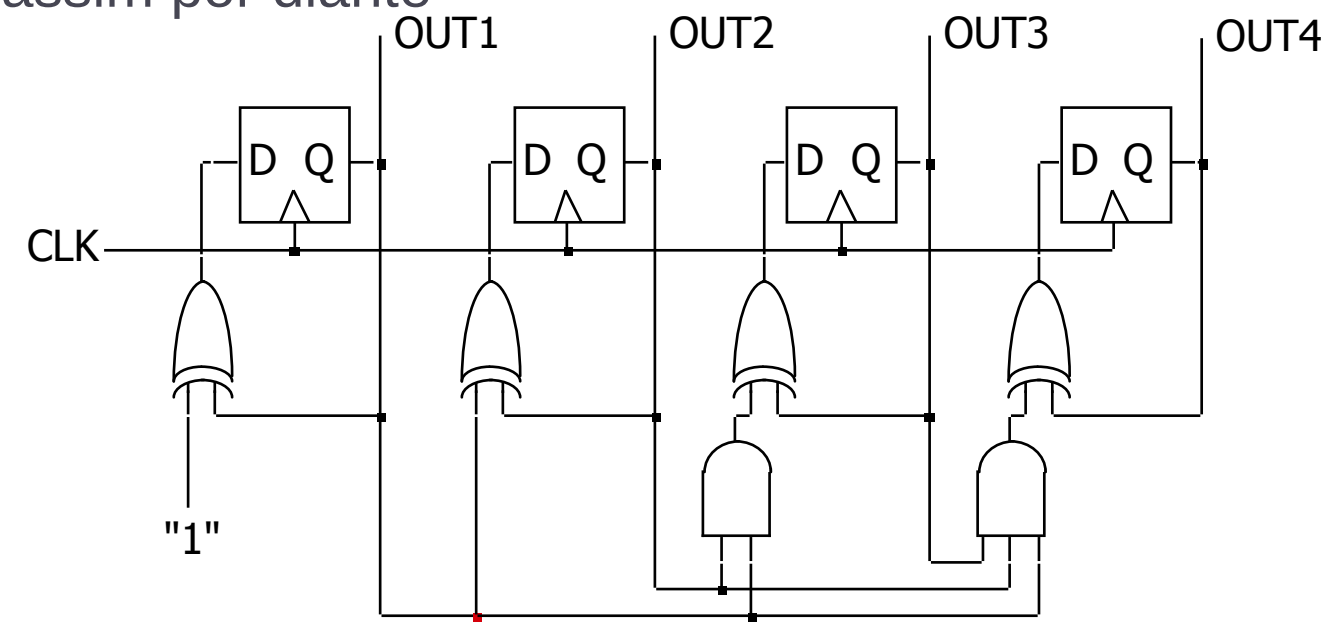
Atividade

- Como este contador funciona?



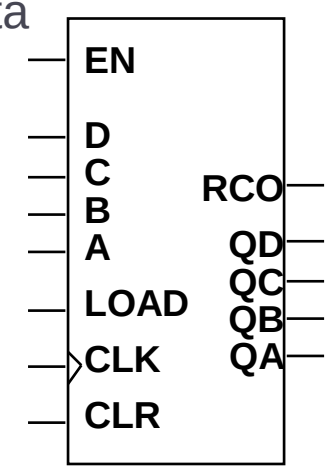
Contador Binário

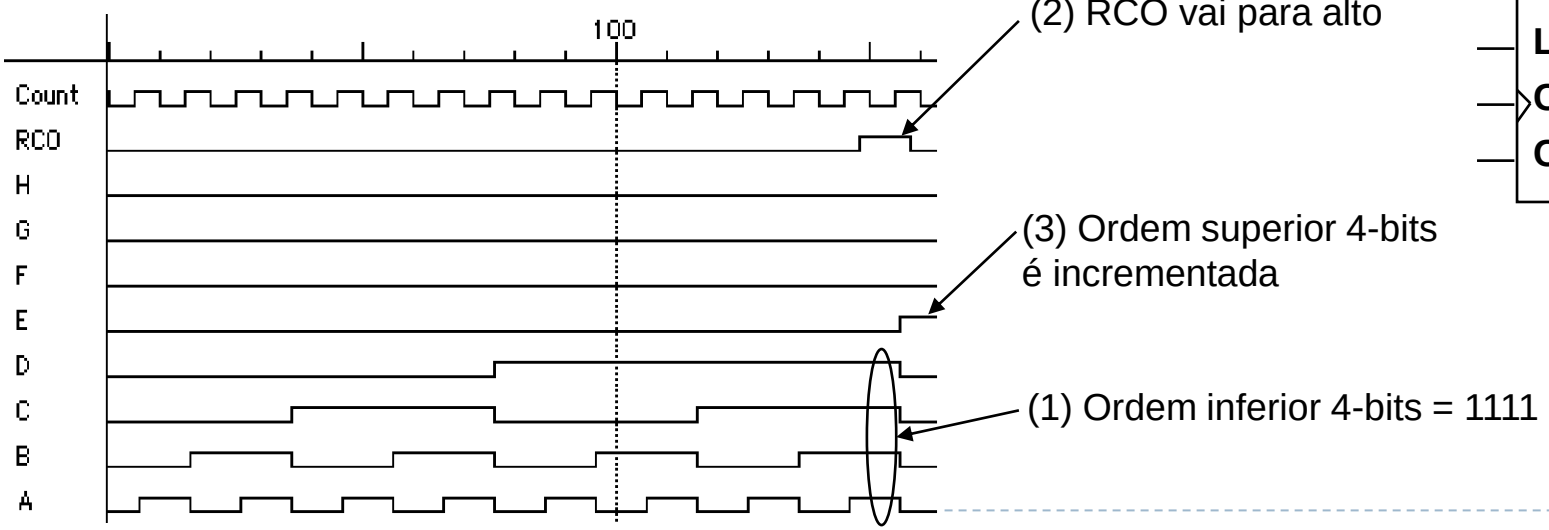
- Lógica entre registradores (não somente multiplexador)
  - XOR decide quando o bit deve ser alternado
  - Sempre para o bit de ordem inferior, somente quando o primeiro bit é verdade para o segundo bit, e assim por diante



Counter up síncrono binário de 4-bit

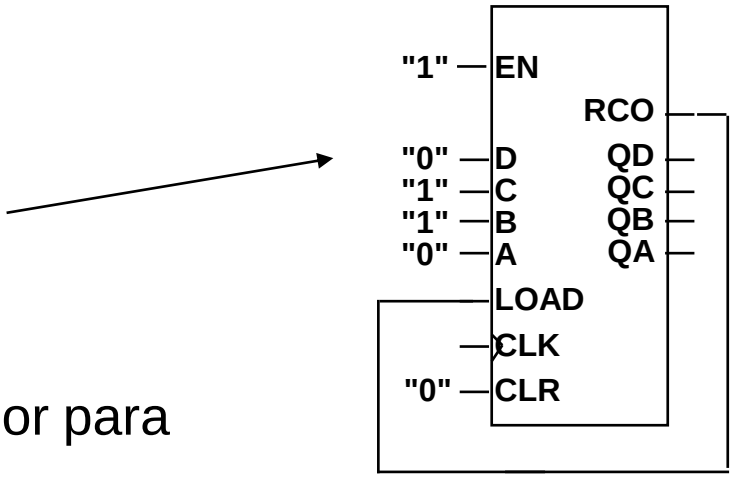
- Componente padrão com muitas aplicações
  - FFs disparados por borda positiva w/ synchronous load and clear inputs
  - Carregamento paralelo de dados a partir D, C, B, A
  - Entradas enable: precisa ser afirmado para habiitar a contagem
  - RCO: Saída ripple-carry utilizado para contadores em cascata
    - Alto quando o contador está no maior estado (1111)
    - Implementado usando uma porta AND



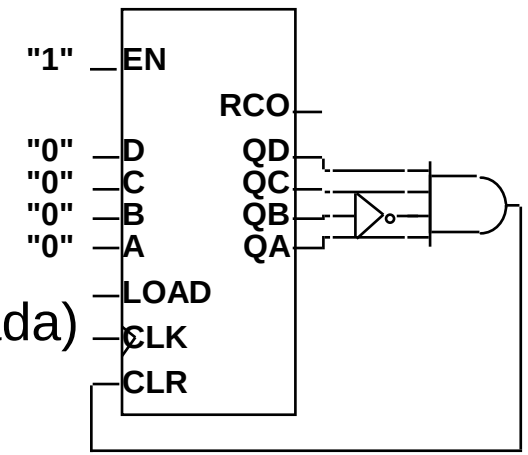


Contadores Offset

- Iniciando contadores offset – utiliza de load síncrono
  - Ex: 0110, 0111, 1000, 1001,
    1010, 1011, 1100, 1101, 1111, 0110, . . .



- Finalizando contador offset – comparador para finalização do valor
  - Ex: 0000, 0001, 0010, ..., 1100, 1101, 0000



- Combinação dos de cima (valor de começo e parada)

Linguagem de descrição de hardware e lógica sequencial

- Flip-flops
  - Representação de clocks – temporização das mudanças do estado
  - Assíncrono vs. Síncrono
- Registrador de deslocamento
- Contadores simples

Flip-flop em Verilog

- Usa sempre lista de sensibilidade do bloco para esperar pela borda de clock.

```
module dff (clk, d, q);

    input  clk, d;
    output q;
    reg    q;

    always @(posedge clk)
       q = d;

endmodule
```

Mais Flip-flops

- Síncrono/assíncrono reset/set
  - Único segmento que aguarda o clock
  - Três segmentos paralelos - das quais apenas um espera pelo clock

Síncrono

```
module dff (clk, s, r, d, q);
    input  clk, s, r, d;
    output q;
    reg    q;

    always @(posedge clk)
        if (r)      q = 1'b0;
        else if (s) q = 1'b1;
        else        q = d;

endmodule
```

Assíncrono

```
module dff (clk, s, r, d, q);
    input  clk, s, r, d;
    output q;
    reg    q;

    always @(posedge r)
        q = 1'b0;
    always @(posedge s)
        q = 1'b1;
    always @(posedge clk)
        q = d;

endmodule
```

Flip-flop incorreto em Verilog

- Usa sempre lista de sensibilidade do bloco para esperar o clock mudar

```
module dff (clk, d, q);

    input  clk, d;
    output q;
    reg    q;

    always @(clk)
        q = d;

endmodule
```

Incorreto! Q sempre será alterado pela mudança do clock, não apenas em uma borda.

Atribuição bloqueante e não bloqueante

- Atribuições bloqueantes (X=A)
  - Completa a atribuição antes de continuar a próxima instrução
- Atribuições não bloqueantes (X<=A)
  - Concluída em tempo zero e não altera o valor do alvo até um ponto de bloqueio (delay / espera) é encontrado
- Exemplo: swap

```
always @(posedge CLK)
    begin
        temp = B;
        B = A;
        A = temp;
    end
```

```
always @(posedge CLK)
    begin
        A <= B;
        B <= A;
    end
```

Atribuição em RTL (Register-transfer-level)

- Atribuição não bloqueante também é conhecida como uma atribuição RTL
  - Se usado em um bloco always disparado por uma borda de clock
  - Todos os flip-flops se modificarão juntos

```
// B,C,D todos irão obter o
// valor de A
always @(posedge clk)
   begin
      B = A;
      C = B;
      D = C;
   end
```

```
// implementado em um registrador
de deslocamento também
always @(posedge clk)
   begin
      B <= A;
      C <= B;
      D <= C;
   end
```

Contador Mobius em Verilog

```
initial
   begin
      A = 1'b0;
      B = 1'b0;
      C = 1'b0;
      D = 1'b0;
   end

always @(posedge clk)
   begin
      A <= ~D;
      B <= A;
      C <= B;
      D <= C;
   end
```

Contador binário em Verilog

```
module binary_counter (clk, c8, c4, c2, c1);

   input  clk;
   output c8, c4, c2, c1;

   reg [3:0] count;

   initial begin
      count = 0;
   end

   always @(posedge clk) begin
      count = count + 4'b0001;
   end

   assign c8 = count[3];
   assign c4 = count[2];
   assign c2 = count[1];
   assign c1 = count[0];

endmodule
```

```
module binary_counter (clk, c8, c4, c2, c1, rco);

   input  clk;
   output c8, c4, c2, c1, rco;

   reg [3:0] count;
   reg rco;

   initial begin . . . end

   always @(posedge clk) begin . . . end

   assign c8 = count[3];
   assign c4 = count[2];
   assign c2 = count[1];
   assign c1 = count[0];
   assign rco = (count == 4b'1111);

endmodule
```

Resumo lógica sequencial

- Construindo bloco de circuitos fundamental com estado
  - latch e flip-flop
  - Latch R-S, mestre/escravo R-S, mestre/escravo D, edge-triggered D flip-flop
- Metodologias de temporização
  - Utiliza de clocks
  - FFs cascateados trabalham porque os atrasos de propagação excedem o tempo de espera
  - Cuidado com clock skew
- Entrada assíncrona e seus perigos
  - Falha sincronizador: o que é e como minimizar o seu impacto
- Registradores básicos
  - Registradores de deslocamento
  - Contadores
- Linguagem de descrição de hardware e lógica sequencial