Lógica Seqüencial

Referências bibliográficas para esse assunto: capítulos 6 a 8 de [Nelson et al., 1995], capítulos 9 e 10 de [Hill and Peterson, 1993].

Circuito combinacional: são aqueles nos quais a saída depende apenas dos valores de entrada correntes.

Circuito seqüencial: são aqueles nos quais a saída depende não apenas dos valores de entrada correntes, mas também do "estado presente". O estado presente, por sua vez, depende das entradas passadas. Ou seja, um circuito seqüencial precisa "armazenar" o estado presente. Em sistemas digitais, os dispositivos que são capazes de armazenar dados são denominados **memória**. Esses dispositivos de memória podem ser de vários tipos: circuitos de semi-condutores, dispositivos magnéticos, dispositivos mecânicos, etc. Um tipo de circuito semi-condutor que tem a capacidade para "armazenar" dados são os chamados flip-flops. Os flip-flops, além de armazenar os dados, podem "mudar de estado" mediante estímulo com sinal de entrada adequado.

Exemplo de dispositivo seqüencial 1 (capítulo 9 de [Hill and Peterson, 1993]): Considere as fechaduras com segredo, daqueles que são comuns em malas ou então cofres. No caso de uma fechadura de mala, podemos pensar que a posição dos botões, ou mais precisamente, os valores visíveis de cada um dos botões, corresponde à entrada corrente e que a situação da fechadura, i.e., aberta/fechada, corresponde à saída. A fechadura encontra-se aberta se os botões giratórios que a compõem encontram-se numa determinada configuração; em todas as demais configurações, a fechadura está fechada. Ou seja, a saída do dispositivo depende apenas da posição corrente dos botões e portanto esse tipo de fechadura pode ser vista como um dispositivo combinacional. Por outro lado, em fechaduras de cofre, daqueles que possuem um botão giratório, a entrada corrente é a posição do botão e a saída é a situação aberta/fechada da fechadura. Essa situação aberta/fechada depende de uma seqüência de posições; apenas a posição corrente não basta. Portanto, essas fechaduras giratórias de cofre (que são chamadas de "fechaduras combinacionais") podem ser vistas como dispositivos seqüenciais.

Exemplo de dispositivo seqüencial 2 (capítulo 6 de [Nelson et al., 1995]): O elevador pode ser visto como um dispositivo seqüencial. As ações de um elevador dependem das operações no painel de controle (interno ao elevador ou externo, em cada andar de um prédio) e também do estado presente do elevador. O *estado presente* do elevador inclui a sua posição atual e as ações anteriores. Por exemplo, "elevador está no terceiro andar e subindo" é diferente de "elevador está no terceiro andar e descendo". O *próximo estado* do elevador é definido em função do estado presente e da entrada corrente. A *entrada corrente* neste caso corresponde à situação dos botões de controle nos painéis de controle. Por exemplo, se o estado presente é "elevador está no terceiro andar e descendo" e há uma requisição de descida solicitada no segundo andar, faz sentido tomar a ação de parar o elevador no segundo andar. Já se a requisição do segundo andar for para subir, ela deve ser ignorada no presente

momento. Uma vez decidida a ação a ser tomada e a mesma sendo executada, ocorre uma *transição de estado* que levará o sistema ao próximo estado.

Como podemos montar circuitos que funcionam como memória, no sentido de "armazenar" um determinado dado? Como fazer com que esse dado permaneça inalterado até decidirmos modificá-lo? Circuitos com capacidade de armazenamento podem ser obtidos considerando-se realimentação, ou seja, fazendo com que as saídas do circuito sejam também entradas.

Exemplo 1: Considere uma porta OU com ambas as entradas em 0 e conecte a saída em uma das entradas. Em seguida, mude o valor da outra entrada para 1. O que acontece se colocarmos o valor dessa mesma entrada de volta para 0?

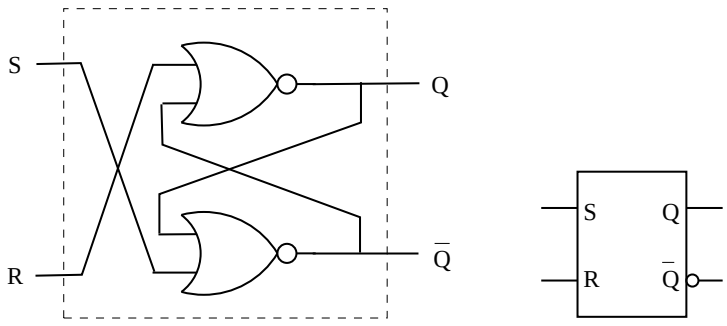
Exemplo 2: Considere uma porta NÃO-OU com ambas as entradas em 0 e conecte a negação da saída em uma das entradas. Em seguida, mude o valor da outra entrada para 1. O que acontece se colocarmos o valor dessa mesma entrada de volta para 0?

Os exemplos acima são chamados de *set latch* e *reset latch*, respectivamente. No primeiro, uma vez mudada para 1 (set), a saída não mais pode ser alterada. No segundo, uma vez mudada para 0 (reset), a saída não mais pode ser alterada. Pelo fato de não podermos mudar o estado desses dispositivos mais de uma vez, eles tem possibilidade de uso muito limitado. Podemos, porém, usar idéia similar para construir dispositivos que permitem alterar o estado repetidas vezes. Tais circuitos, descritos em seguida, são conhecidos por *set-reset latches* e *set-reset flip-flops*.

1.1 Flip-flops

1.1.1 Flip-flop SR

O nome SR vem de *set-reset*. Alguns autores denominam essa classe de flip-flops de RS em vez de SR. A figura a seguir mostra o circuito de um latch SR baseado em portas NÃO-OU e a respectiva representação simbólica por um diagrama de bloco.



A operação do latch SR pode ser vista no diagrama da figura 1.1. Inicialmente, ambas as entradas, S e R estão em 0, e a saída Q está também em 0 (conseqüentemente, a saída $\overline{Q}$ está em 1). Esse estado é consistente. No instante t_1, o sinal S vai a 1. Depois de um certo atraso, no instante t_2, $\overline{Q}$ vai a 0 e depois de outro atraso, em t_3, Q vai a 1. Quando, em t_4, S volta a 0, nenhuma mudança ocorre nos demais sinais. Em t_5, o sinal R vai a 1, e depois de um certo atraso, em t_6, Q vai a 0 e, em seguida, no instante t_7, $\overline{Q}$ vai a 1. No instante t_8, R volta a 0, porém nemhum dos outros sinais é modificado.

Para representar a relação entrada-saída desse circuito, considere as seguintes notações:

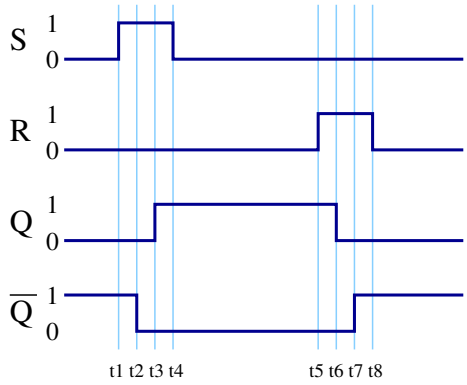


Figura 1.1: Operação de um latch SR (S=sinal set; R=sinal reset; Q=saída (estado)).

S_i denota o valor do sinal que alimenta a entrada S num certo instante de tempo t_i.

R_i denota o valor do sinal que alimenta a entrada R num certo instante de tempo t_i.

Q_i denota a saída ou estado do latch num certo instante de tempo t_i.

Q_{i+1} denota a saída (ou próximo estado) do latch em decorrência de termos, no instante de tempo t_i, o estado Q_i e os valores S_i e R_i nas entradas S e R, respectivamente.

O comportamento do latch SR pode ser descrito pela seguinte tabela-verdade:

S_i	R_i	Q_i	Q_{i+1}	
0	0	0	0	Nenhuma mudança
0	0	0	1	
0	1	0	0	reset
0	1	1	0	
1	0	0	1	set
1	0	1	1	
1	1	0	?	proibido
1	1	1	?	

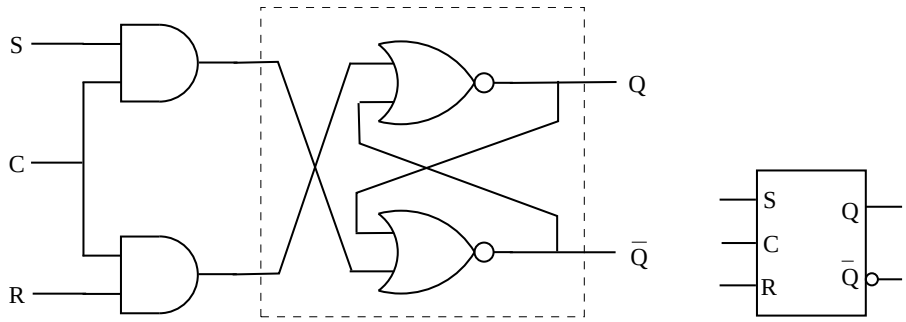
Em palavras, significa que quando a entrada S é ativada, realiza-se a operação *set* (ou seja, o estado Q do latch passa a 1) e quando a entrada R é ativada, realiza-se a operação *reset* (ou seja, o estado Q do latch passa a 0). Observe que o pulso nos sinais de entrada deve ter uma duração suficiente para que essas operações se completem e se estabilizem. Se a duração for muito curta, o comportamento poderá ser diferente. A situação $R = S = 1$ não é permitida por duas razões. Primeiro, se tivermos ambas as entradas em 1, teremos $Q = \overline{Q}$, violando a condição de que uma saída é o complemento da outra. Segundo, se supormos que $Q = \overline{Q} = 0$, então numa situação em que ambas as entradas estão em 1 e passam simultaneamente para 0, o estado Q passa a 1. Em seguida, se as duas portas NÃO-OU funcionarem de forma exatamente iguais, o estado voltará para 0, o que faz com que em seguida passe para 1 e depois novamente para 0 e assim por diante. Isso levaria o estado do circuito a oscilar (ou seja, a saída não se estabiliza). Se as duas portas não funcionarem de forma exatamente iguais, aquela que responde primeiro ditará o comportamento do circuito. Como na prática é razoável supormos que uma das portas responderá antes da outra e como numa realização física não se tem controle de qual porta responde primeiro, o comportamento será imprevisível.

Pelas razões descritas acima, consideramos que a entrada $R_i = S_i = 1$ é proibida e, portanto, na descrião do comportamento do circuito via uma expressão lógica, ela será tratada como don't care. Desta forma, a expressão resultante é:

$$Q_{i+1} = S_i + Q_i \overline{R}_i$$

que pode ser facilmente obtida desenhando-se o mapa de Karnaugh correspondente à tabela-verdade dada acima.

Em geral, um sistema é composto por várias unidades de flip-flops e a mudança de estado desses flip-flops precisa ser coordenado. Para isso, existem os flip-flops controlados como o da figura a seguir. A figura a seguir mostra um flip-flop SR controlado. Note que existe um terceiro sinal de entrada C. Quando $C = 0$, a saída de ambas as portas E é 0 e portanto mudanças no valor de R e S não tem efeito nenhum sobre o estado do flip-flop. Quando $C = 1$, temos o mesmo comportamento descrito na tabela acima.



Um sinal de controle usado para coordenar a mudança de estado dos flip-flops é o sinal *clock*.

No caso do flip-flop SR controlado, a expressão do próximo estado é dada por

$$Q_{i+1} = S_i C_i + Q_i \overline{R}_i + Q_i \overline{C}_i \,.$$

Se $C_i = 0$, temos $Q_{i+1} = Q_i$ e se $C_i = 1$ então temos $Q_{i+1} = S_i + Q_i \overline{R}_i$, a equação do latch SR sem a entrada de controle.

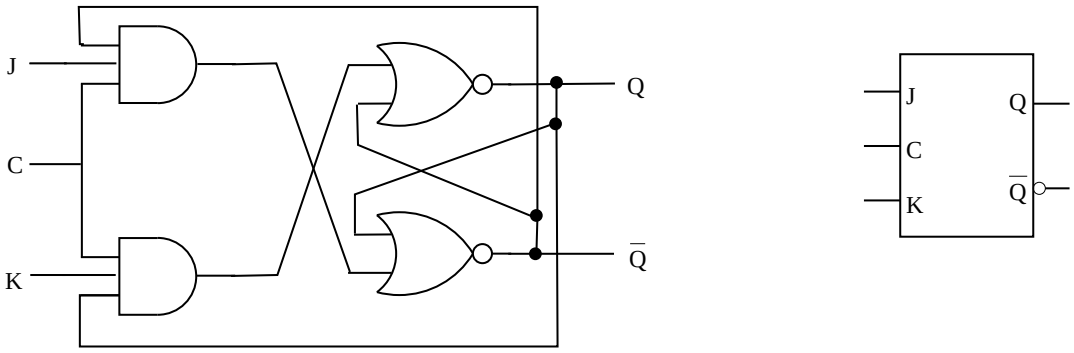
Observação: alguns autores denominam esses circuitos de *flip-flops*, independente de os mesmos possuírem ou não uma entrada de controle. Outros, denominam de *flip-flop* aqueles com entrada de controle e de *latches* aqueles sem entrada de controle.

FALTA O flip-flop SR pode também ser realizado usando-se portas NÃO-E, da seguinte forma:

1.1.2 Flip-flop J-K

Nos flip-flops SR, após ambas as entradas serem ativadas simultaneamente, o flip-flop SR pode ter comportamento imprevisível. Isso, do ponto de vista prático, é indesejável pois o projetista do circuito teria de garantir que as duas entradas nunca ficarão ativas simultaneamente.

Os flip-flops JK são uma evolução do SR e não possuem esse problema. No JK, quando ambas as entradas passam para um (1), o circuito tem o efeito de mudar de estado, isto é, se a saída era 1, então passa a ser 0 e vice-versa. A figura a seguir mostra uma possível realização do flip-flop JK e a respectiva representação diagramática.



Vejamos o que acontece quando $J = K = 1$. Suponha inicialmente que $Q_i = 0$ e J_i e K_i passam a 1. Com $C_i = 1$, a porta E de cima fica com saída 1 e em conseqüência temos $Q_{i+1} = 1$ e $\overline{Q}_{i+1} = 0$. Similarmente, se tivéssemos $Q_i = 1$, e J_i e K_i passassem para 1, com $C_i = 1$ teríamos $Q_{i+1} = 0$ e $\overline{Q}_{i+1} = 1$. Ou seja, $J = K = 1$ tem o efeito de inverter o estado do flip-flop. Para os demais valores de entrada, o comportamento do JK é igual ao do SR.

Tabela de estados do flip-flop JK:

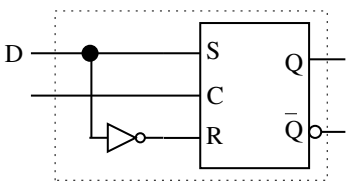
J_i	K_i	C_i	Q_{i+1}	
$\times$	$\times$	0	Q_i	não muda
0	0	1	Q_i	mantém
0	1	1	0	reset
1	0	1	1	set
1	1	1	$\overline{Q}_i$	inverte

Para $C_i = 1$ a equação do próximo estado é dada por

$$Q_{i+1} = J_i\overline{Q}_i + \overline{K}_i Q_i\,.$$

1.1.3 Flip-flop D

É um flip-flop que, quando o controle está ativo, copia a entrada D para a saída Q. Uma possível implementação é alimentar a entrada S de um flip-flop SR com o sinal D e R com o seu inverso, como na figura a segur. Outra possibilidade é utilizar o flip-flop JK em vez do flip-flop SR.



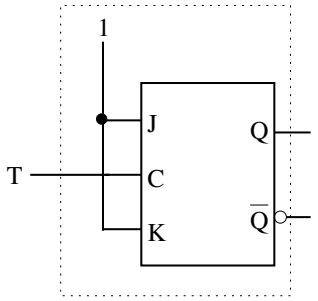
A expressão do próximo estado do flip-flop D é dada por

$$Q_{i+1} = D_i\,C_i + \overline{C}_i\,Q$$

ou seja, se $C_i = 0$, $Q_{i+1} = Q_i$ e, se $C_i = 1$ então $Q_{i+1} = D_i$.

1.1.4 Flip-flop T

É um flip-flop que, quando o sinal de entrada T passa a 1, inverte o estado. Uma possível implementação é alimentar J e K de um flip-flop JK com 1 e alimentar a entrada C dele com o sinal T.



A expressão do flip-flop T é dada por

$$Q_{i+1} = \overline{Q}_i \, .$$

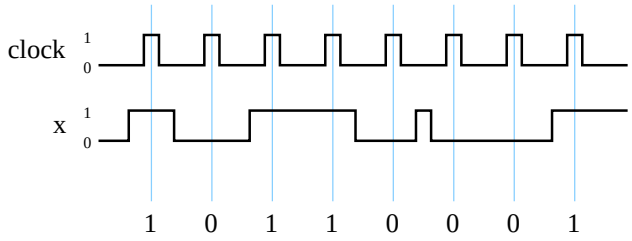
1.1.5 Flip-flops mestre-escravo

Os flip-flops controlados vistos acima podem mudar de estado várias vezes enquanto o sinal que alimenta o controle C estiver alto. Em geral, o sinal que alimenta a entrada C é o sinal de *clock*. Em condições ideais, poderíamos fazer com que a duração de um pulso de um sinal de clock seja menor que o tempo necessário para que ocorram mais de uma mudança de estado nos flip-flops. No entanto, na prática, este tipo de controle é difícil e pode nem ser possível.

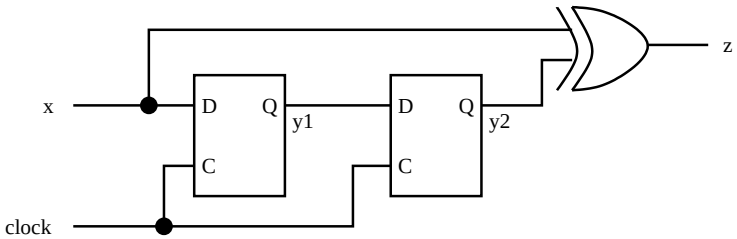
Para ilustrar o problema de múltiplas mudanças de estado enquanto o clock está alto, considere o seguinte

Problema: Seja x_i, $i = 1, 2, \ldots$ uma seqüência de bits. Deseja-se gerar uma outra seqüência de bits z_i, $i = 1, 2, \ldots$ tal que $z_i = 1 \iff x_i \neq x_{i-2}$. Vamos supor que $x_i = 0$ para $i < 1$. Por exemplo, se a seqüência x é dada por 10110001 então a seqüência z deverá ser 10011101.

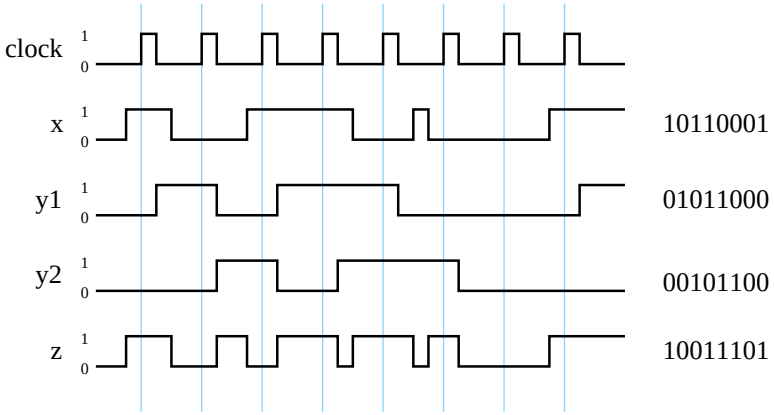
Na prática, sinais que alimentam um circuito são sinais contínuos no domínio do tempo. Tais sinais contínuos podem ser interpretados como uma seqüência de bits, tomando-se o valor do sinal (baixo=0 e alto=1) nos instantes de tempo nos quais o sinal de clock está alto. Ou seja, para produzir uma determinada seqüência de bits, basta sincronizarmos de forma adequada a subida e decida do sinal de entrada com o sinal de clock. Um exemplo é mostrado na figura a seguir. O sinal de clock é um sinal periódico, formado por pulsos (subida seguido de descida) de pequena duração. As linhas verticais no gráfico correspondem aos instantes de tempo nos quais o sinal de clock está alto. Logo, a seqüencia de bits é 10011101, que são os valores do sinal x nos instantes que o sinal de clock está alto. Observe que o sinal x apresenta um pulso entre os quinto e sexto pulsos do sinal de clock. Porém, como trata-se de um pulso num instante em que o sinal de clock está baixo, não deve ser encarado como um bit adicional na seqüência x_i.



Um circuito seqüencial para produzir a seqüência z é mostrado na figura a seguir:



O flip-flop D copia a entrada vigente quando o clock está alto para a respectiva saída Q. A simulação do comportamento dos sinais desse circuito ao longo do tempo é mostrada na figura a seguir:

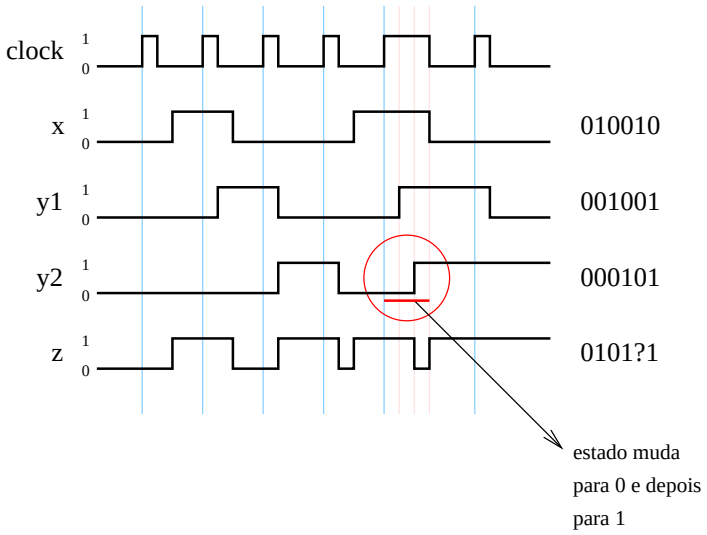


Observe que o primeiro flip-flop funciona como um deslocador, ou seja, sua saída y_1 é exatamente igual à entrada x, exceto pelo fato de que os bits estão defasados em uma posição. Similarmente, o segundo flip-flop funciona como um deslocador em relação ao sinal y_1, ou seja, y_2 é exatamente igual a x, exceto pelo dato de estar defasado de duas posições. Logo, ao se computar o XOR entre x e y_2, obtém-se z ($z_i = 1 \iff x_i \neq x_{i-2}$). Observe que z apresenta um pulso entre o segundo e terceiro pulsos do clock bem como descidas entre o quarto e quinto pulsos e entre o quinto e sexto pulsos do sinal de clock, que não são levados em consideração na seqüência z_i.

Observe também que cada flip-flop muda de estado apenas uma vez numa subida do sinal de clock e que ambos os flip-flops, quando mudam de estado, mudam simultanemante. Na simulação acima, estamos supondo que o atraso para o sinal de entrada do flip-flop se propagar até a saída é exatamente igual à duração de um pulso do sinal de clock. Quanto à porta XOR, estamos supondo que não há atraso nenhum.

Vamos agora supor que, por alguma razão, um pulso de clock pode durar mais do que o tempo normal em algumas situações. O diagrama da figura a seguir mostra a simulação do mesmo circuito acima,

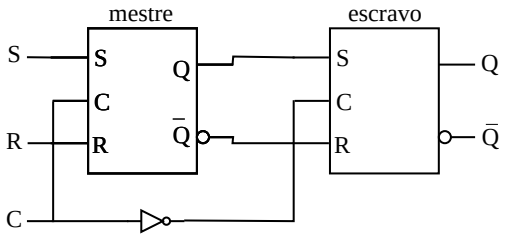
para um outro sinal de entrada x. Observe que o quinto pulso do sinal de clock tem uma duração três vezes maior que o normal.



Na evolução do sinal y_2, a dupla mudança de estado aparece em destaque em t_5 (no quinto pulso do clock). A subida do clock em t_5 faz com que o sinal y_1 também suba logo em seguida. Essa subida de y_1 só deveria afetar o sinal y_2 na próxima subida do clock, em t_6. Porém, como o pulso do clock em t_5 é demorado, acaba afetando y_2, ou seja, y_2 também acaba subindo em t_5. Isto significa que, o sinal y_2 cujo estado anterior em t_4 era 1 passou para 0 em t_5 e ainda em t_5 passou novamente para 1, ou seja, mudou de estado duas vezes em t_5. Isso compromete toda a evolução subseqüente do sinal de saída.

Para contornar esse tipo de problema, foram introduzidos os chamados *edge-triggered* flip-flops que são aqueles que mudam de estado somente na transição 1 para 0 (descida) ou na transição 0 para 1 (subida) do sinal de controle. Um exemplo desse tipo de flip-flop são os **flip-flops mestre-escravo**.

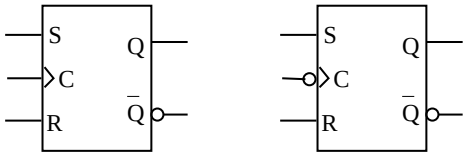
A figura a seguir mostra o esquema de um flip-flop SR mestre-escravo.



Quando o clock está baixo, mudanças nas entradas R e S não tem efeito no flip-flop mestre. O flip-flop escravo está habilitado para mudanças (sua entrada C é 1), mas nenhuma mudança ocorre nas suas entradas R e S (que vem das saídas do flip-flop mestre).

Quando o clock sobe, o mestre muda de estado de acordo com as entradas R e S e o escravo fica desabilitado (deve-se apenas garantir que o escravo fique desabilitado antes que ocorra qualquer mudança na saída do mestre). Quando o clock desce, o mestre fica desabilitado (e portanto "congela" a saída dele) e o escravo se habilita (ou seja, a saída do mestre é copiada para a saída do escravo). Desde que o clock subiu, a saída do mestre pode ter oscilado algumas vezes, porém a saída do flip-flop como um todo só muda quando o clock baixa. Este é, portanto, um exemplo de flip-flop que muda de estado na descida do sinal de clock.

Similarmente, pode-se construir flip-flops que mudam de estado apenas na subida do sinal de controle. Pode-se também construir tais flip-flops usando-se como base o flio-flop JK ou ainda outros tipos de circuitos com realimentação. A figura a seguir mostra a representação diagramática de flip-flops SR mestre-escravo com mudança de estado respectivamente na subida e na descida do sinal de controle. O triângulo na entrada C indica que o flip-flop é edge-triggered; a bolinha indica que a mudança de estado ocorre na descida do sinal de controle, enquanto a ausência de bolinha indica que a mudança de estado ocorre na subida do sinal de controle.



1.2 Exemplos de circuitos seqüenciais

1.2.1 Síncronos × Assíncronos

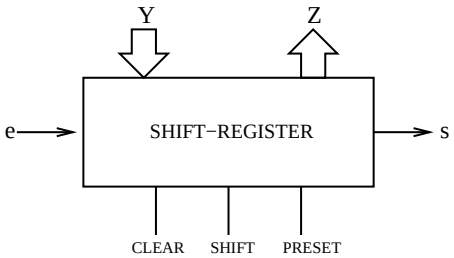
Em **circuitos síncronos**, a mudança de estado de todos os elementos de memória do circuito (flip-flops) ocorre em sincronia ("ao mesmo tempo") e é controlada pelo sinal de um *clock*.

Em **circuitos assíncronos**, a mudança de estado não é sincronizada, ou seja, não se utiliza o sinal de um *clock* para se promover a mudança de estado de todos os flip-flops. Há dois modos de promover a mudança de estado em circuitos assíncronos. No modo pulso, o controle do flip-flop pode ser alimentado por qualquer outro sinal. No modo fundamental, o retardo intrínseco ou propositadamente colocado no circuito é utilizado para funcionar como "memória". Em ambos os casos há restricões que devem ser satisfeitas para o circuito funcionar propriamente.

Nesta seção apresentamos alguns poucos exemplos de circuitos seqüenciais. A maior parte dos exemplos que veremos serão de circuitos síncronos. Quando não for o caso, isso será explicitamente mencionado.

1.2.2 Registradores

A figura a seguir mostra um esquema de um registrador-deslocador (*shift-register*) genérico.



Entrada paralela: $Y = y_n\, y_{n-1} \ldots y_2\, y_1$ são os n bits a serem armazenados no registrador, num pulso do sinal PRESET.

Saída paralela: $Z = z_n\, z_{n-1} \ldots z_2\, z_1$ são os n bits que estão armazenados no registrador.

Entrada serial: e é 1 bit de entrada que ocupará a posição mais à esquerda, num pulso do sinal SHIFT.

Saída serial: s é 1 bit de saída num pulso do sinal SHIFT (é o bit que ocupava a posição mais à direita no registrador quando o sinal SHIFT subiu).

Os seguintes são sinais de controle:

CLEAR (Limpa): zera o registrador

PRESET (Carrega): armazena Y no registrador

SHIFT (desloca): desloca, uma posição, todos os bits para a direita

Podemos pensar em diferentes modos de operação para este tipo de registrador:

- entrada e saída seriais

  A entrada serial (bit e) deve estar sincronizada com o sinal SHIFT.

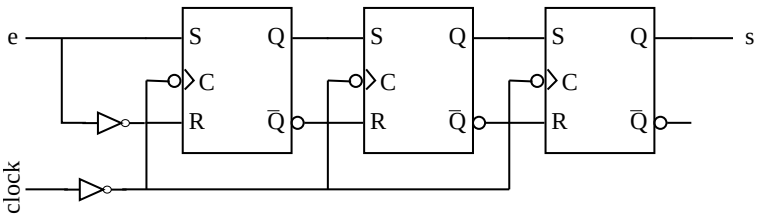
- entrada paralela e saída serial
  - CLEAR para zerar o registrador
  - alimentar Y (para que esteja com os valores a serem armazenados no registrador)
  - PRESET para armazenar Y (supondo que em Y estão os valores que se deseja armazenar)
  - n SHIFTS, para produzir n bits (saídas) em série

- entrada serial e saída paralela
  - CLEAR para zerar o registrador
  - entrada de n bits em série, juntamente com n pulsos no sinal SHIFT
  - os n bits ficam disponíveis em Z

- entrada e saída paralelas
  - alimentar Y com os bits que se deseja armazenar
  - PRESET para armazenar Y (supondo que em Y estão os valores que se deseja armazenar)
  - os n bits ficam disponíveis em Z

Um registrador como o esquematizado acima pode ser realizado por um conjunto de n flip-flops. Cada flip-flop armazenaria 1 bit.

Exemplo: Uma realização de um registrador com entrada e saída seriais é mostrada na figura a seguir. Os flip-flops utilizados são do tipo SR *edge-triggered* na descida (ou seja um SR cujo valor de saída muda na descida do sinal de controle).



Note que o sinal que alimenta a entrada R é o complemento do sinal que alimenta S em todos os flip-flops. Portanto, esses correspondem aos flip-flops D que armazenam o valor de entrada ao pulso do sinal de controle. Quando há um pulso do clock, o bit e é armazenado no flip-flop mais à esquerda, o valor de cada flip-flop é armazenado no flip-flop a sua direita e o valor do flip-flop mais à direita é o bit de saída s.

Note que a mudança de estado dos flip-flops ocorre na transição de 1 para 0 do sinal que alimenta a entrada C. Portanto, a mudança de estado ocorre na transição de 0 para 1 do clock (uma vez que o sinal do clock é negado antes de alimentar C).

Registradores em outros modos de operação podem ser implementados usando flip-flops que possuem sinais de controle adicionais. Tipicamente, os flip-flops comerciais possuem, além da entrada para o sinal do *clock*, entradas para os sinais de controle CLEAR e PRESET. O primeiro faz $Q = 0$ enquanto o segundo faz $Q = 1$, independente dos outros sinais (obviamente parece não ter sentido ativar CLEAR e PRESET simultaneamente ...).

1.2.3 Contadores

O objetivo de um circuito contador é, a cada pulso do sinal de *clock*, incrementar (ou decrementar) o valor armazenado em alguma memória.

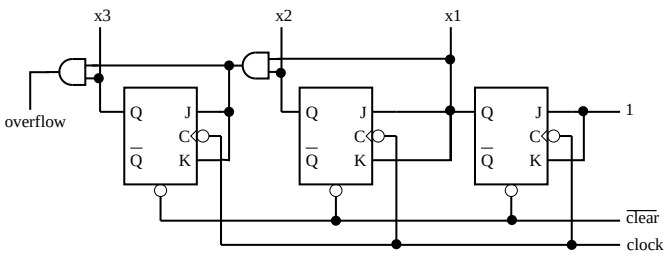
Um **contador incremental módulo** 2^n, com valor inicial 0, apresenta a seguinte seqüência de transição de valores:

$$0, 1, 2, 3, \ldots, 2^n - 1, 0, 1, 2, 3, \ldots$$

Exemplo: Um contador incremental módulo 2^3 é definido pela seguinte tabela:

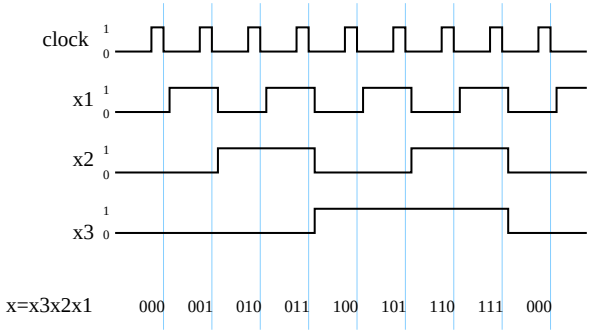
Estado atual $x_3\, x_2\, x_1$	Próximo estado $x_3^*\, x_2^*\, x_1^*$
000	001
001	010
010	011
011	100
100	101
101	110
110	111
111	000

Uma possível implementação de um contador incremental módulo 2^3, usando flip-flops JK edge-triggered, é mostrado na figura a seguir:

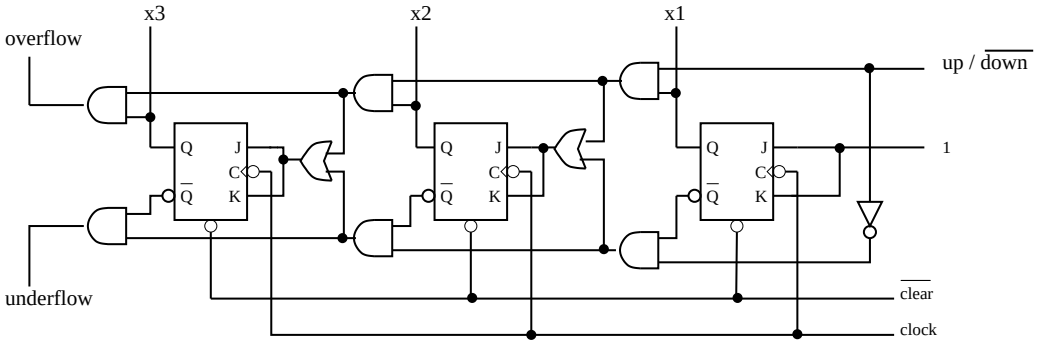


Observe que nos três flip-flops temos $J = K$. O fato dos flip-flops serem negative edge-triggered significa que a mudança de estado ocorre na descida do sinal de clock. O valor armazenado num cero instante é dado por $x = x_3\, x_2\, x_1$. O sinal $\overline{\text{clear}}$ é equivalente a colocar um inversor no sinal clear. Isso é feito uma vez que o os flip-flops são *reset* na descida do sinal que alimenta sua entrada CLEAR e, portanto, ao se inverter o sinal clear, tem-se que os flip-flops são zerados na subida do sinal clear.

Suponha que incialmente todos os flip-flops estão em 0. A cada pulso do clock, o estado do flip-flop mais à direita inverte (passa de 0 para 1, ou de 1 para 0). O estado do segundo flip-flop muda a cada dois pulsos do clock: mais precisamente, no início é 0 e no primeiro pulso do clock também permanece em 0 pois x_1 é O. No segundo pulso, como x_1 é 1, inverte o estado, com x_2 passando para 1. No terceiro pulso, como x_1 é 0, x_2 não muda e permenace em 1. No quarto pulso, x_1 é 1 e portanto inverte a saída do segundo flip-flop, ou seja, x_2 volta a 0, e assim por diante. A mudança de estado do terceiro flip-flop (e portanto de x_3) ocorre de forma similar ao do segundo flip-flop, porém a cada 4 pulsos do clock. O bit overflow é 1 quando todos os bits x_i são 1. Quando o circuito encontra-se no estado 111, o próximo estado será 000. A simulação do circuito pode ser vista na figura a seguir.

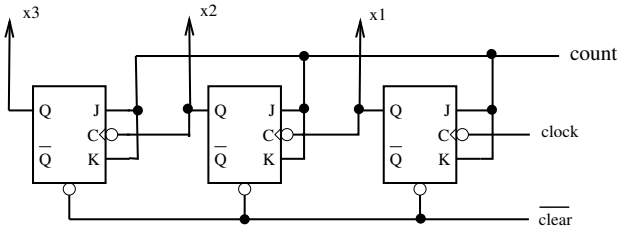


Contadores incremental/decremental (que ora operam incrementalmente e ora operam decrementalmente) podem ser obtidos modificando-se ligeiramente a estrutura do contador incremental acima. Haverá um sinal de entrada adicional para controlar o modo de operação incremento/decremento do contador. Veja a figura a seguir.



Observe que se o sinal up estiver alto, o comportamento do circuito será exatamente igual ao do ciucuito anterior. Se o sinal up estiver baixo, então o circuito realiza a operação de decremento, de forma muito similar à operação de incremento. Simule o circuito para ver a seqüência de bits gerados pelo circuito quando up=0.

Podemos também ter contadores incrementais módulo 2^n assíncronos, ou seja, aqueles cuja mudança de estado de todos os flip-flops não são controlados pelo sinal clock, como no circuito a seguir.



O circuito acima é baseado na observação de que, num contador incremental, ocorre inversão de um certo bit x_i sempre que há uma transição de 1 para 0 no bit x_{i-1}. Veja isso na tabela a seguir:

x	x_3	x_2	x_1
0	0	0	0
1	0	0	1
			↓
2	0	1	0
3	0	1	1
		↓	↓
4	1	0	0
5	1	0	1
			↓
6	1	1	0
7	1	1	1
		↓	↓
0	0	0	0

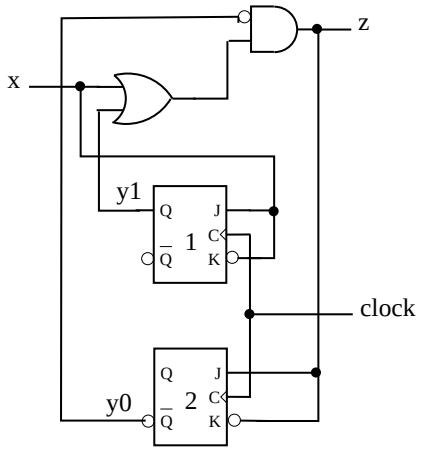
O sinal count, quando alto, habilita o circuito para contagem, enquanto que se baixo desabilita a contagem (fazendo com que o circuito funcione como uma memória – *data holder*). Supondo count=1 a valor inicial armazenado igual a 000, a cada descida do sinal de clock o estado do flip-flop mais à direita é invertido (como no circuito anterior). Apenas o primeiro flip-flop é alimentado pelo sinal do *clock*. Os demais flip-flops são alimentados pelas saídas dos flip-flops anteriores. Toda vez que há uma transição de 0 para 1 na saída de um flip-flop, a saída do próximo flip-flop é invertida.

Devido aos atrasos intínsecos na propagação de sinal em um flip-flop, podem ocorrer estados transitórios na passagem de um estado para outro do circuito. Por exemplo, do estado 011 o circuito passa transitoriamente pelos estados 010 e 000 até ficar em 100 (que seria o estado seguinte ao estado 011). Portanto, circuitos combinacionais que possam fazer uso dos valores $x_3x_2x_1$ devem ser projetados de forma a evitar esses estados transitórios. Tal efeito (estados transitórios) pode ser verificado fazendo-se uma simulação do circuito e considerando que há um certo atraso até a saída de um flip-flop afetar a saída do próximo flip-flop.

1.3 Análise de circuitos seqüenciais

A análise de circuitos seqüenciais consiste em, a partir do circuito, obter uma descrição funcional do mesmo. Tal descrição funcional pode ser obtida com o auxílio das equações de estado, as tabelas de estado e diagramas de estado. Estes conceitos serão apresentados a seguir através de um exemplo.

Considere o seguinte circuito:



a) Equação das entradas dos flip-flops (Note que y_0 é a saída $\overline{Q}$ do segundo flip-flop, ou seja, $y_0 = \overline{Q}$)

$J_1 = x$

$K_1 = \overline{x}$

$J_2 = z = (x + y_1)\,\overline{y}_0$

$K_2 = \overline{z} = \overline{(x + y_1)\,\overline{y}_0}$

b) Equação para os próximos estados:

(Lembre que $Q_{i+1} = J_i\,\overline{Q}_i + \overline{K}_i\,Q_i$)

$y_1^* = x\,\overline{y}_1 + \overline{\overline{x}}\,y_1 = x$

$y_0^* = [(x + y_1)\,\overline{y}_0]\,\overline{y}_0 + \overline{[\overline{(x + y_1)\,\overline{y}_0}]}y_0 = (x + y_1)\,\overline{y}_0$

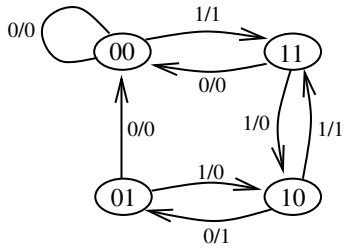
c) Tabela de estados

Cada célula da tabela representa o próximo estado ($y_1^*\,y_0^*$) e a saída (z). Ou seja, leia-se cada célula como $y_1^*\,y_0^*$ / z.

Estado atual ($y_1\,y_0$)	x	
	0	1
00	00/1	11/1
01	00/0	10/0
10	01/1	11/1
11	00/0	10/0

d) Diagrama de estados

Cada nó representa um estado do sistema. Há uma aresta de um estado para outro se é possível uma transição de um para o outro. O rótulo nas arestas indica entrada x e saída z (leia-se x / z). Como há apenas uma variável de entrada, que pode tomar os valores ou 0 ou 1, então há exatamente 2 arestas que saem de cada nó.



Diagramas de estado são uma representação equivalente à tabela de estados.

1.4 Projeto de circuitos seqüenciais

Projeto de circuitos seqüenciais é um processo inverso ao da análise. No entanto, o ponto de partida em geral não é uma tabela ou diagrama de estados, e sim uma descrição funcional do circuito.

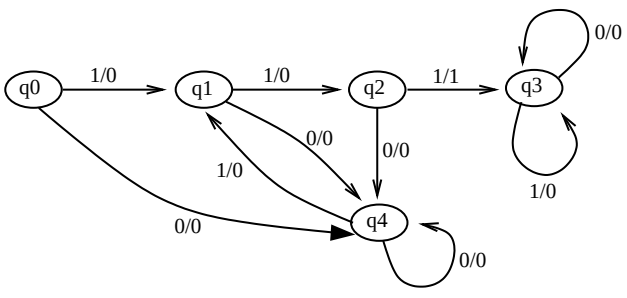
As etapas que fazem parte de projeto de circuitos seqüenciais são:

- Descrição funcional
- Tabela de estados (que pode ser obtida a partir do diagrama de estados ou não)
- Tabela minimal de estados
- Tabela de transição
- Equação das entradas dos flip-flops
- Circuito

Novamente, introduzimos esses conceitos através de um exemplo.

Exemplo: Detector de início de mensagem. Considere uma linha de transmissão de sinal, denotado por x, sincronizada com o clock. Uma ocorrência de 3 bits 1 consecutivos é considerado início de mensagem. Desejamos projetar um circuito síncrono que detecta um início de mensagem. Suponha que existe algum mecanismo que coloca o sistema detector de início de mensagem em um estado q_0 a cada final de mensagem e suponha que incialmente o sistema encontra-se no estado q_0.

a) **Diagrama de estados**: uma possível forma de se começar o projeto de circuitos seqüenciais é construindo-se o diagrama de estados correspondente ao comportamento funcional desejado.



A detecção de início de mensagem é equivalente a atingir o estado q_3 no diagrama acima.

b) **Tabela de estados**: o diagrama acima pode ser equivalentemente representado pela tabela de estados a seguir:

Estado	Entrada	
	0	1
q_0	$q_4/0$	$q_1/0$
q_1	$q_4/0$	$q_2/0$
q_2	$q_4/0$	$q_3/1$
q_3	$q_3/0$	$q_3/0$
q_4	$q_4/0$	$q_1/0$

c) **Tabela minimal de estados**: na tabela de estados acima, o estado q_0 é equivalente ao estado q_4. Isso poderia ser percebido até no próprio diagrama de estados. No entanto, em um caso genérico, nem sempre o diagrama de estados é gerado e, além disso, a equivalência de estados pode não ser tão óbvia. De qualquer forma, nesta etapa reduz-se a tabela de estados a uma tabela minimal, ou seja, eliminam-se os estados equivalentes. Para não gerar confusão na identificação dos estados, na tabela minimal de estados é aconselhável a utilização de outros nomes para os estados. Desta forma, em vez da notação q_i para os estados, passaremos a utilizar as letras a, b, c, d. Fazendo $a = q_4$, $b = q_2$, $c = q_1$ e $d = q_3$ temos:

Estado	Entrada	
	0	1
a	$a/0$	$c/0$
b	$a/0$	$d/1$
c	$a/0$	$b/0$
d	$d/0$	$d/0$

d) **Associação de estados**: se o número de estados na tabela minimal de estados é m, então serão necessários r flip-flops para armazenar qualquer um desses estados, onde r é tal que $2^{r-1} < m \leq 2^r$.

O problema de associação de estados consiste em definir qual das 2^r combinações de valores binários será utilizado para representar cada um dos estados do sistema. No exemplo que estamos considerando, como são 4 estados então são necessários $r = 2$ flip-flops e existem as três seguintes possíveis associações:

Estados	Associação		
	1	2	3
a	00	00	00
b	01	11	10
c	11	01	01
d	10	10	11

As demais associações são equivalentes a um desses três no sentido de que correspondem a uma rotação vertical ou à complementação de uma ou ambas as variáveis e, portanto, em termos de circuito resultante teriam o mesmo custo.

e) **Tabelas de transição**: para cada uma das associações consideradas, pode-se gerar uma tabela de transição. Em cada tabela de transição, as atribuições de estado estão listadas seguindo a ordem do gray-code $(00 - 01 - 11 - 10)$. Uma tabela de transição mostra qual será o próximo estado em função do estado e entrada atuais.

Associação 1			
Estado	$y_1\, y_0$	$y_1^*\, y_0^*$	
		$x = 0$	$x = 1$
a	00	00	11
b	01	00	10
c	11	00	01
d	10	10	10

Associação 2			
Estado	$y_1\, y_0$	$y_1^*\, y_0^*$	
		$x = 0$	$x = 1$
a	00	00	01
c	01	00	11
b	11	00	10
d	10	10	10

Associação 3			
Estado	$y_1\, y_0$	$y_1^*\, y_0^*$	
		$x = 0$	$x = 1$
a	00	00	01
c	01	00	10
d	11	11	11
b	10	00	11

f) **Equação das entradas dos flip-flops**: a equação das entradas dos flip-flops pode ser gerada a partir da análise das tabelas de transições. Vejamos como se realiza esse processo para a associação 3 do item anterior.

Primeiramente, observe que sabemos que do estado $y_1\, y_0$ o circuito irá para o estado $y_1^*\, y_0^*$ e que cada variável de estado (no caso, y_1 e y_0) corresponde a um flip-flop. Assim, o que queremos descobrir é a expressão que descreve o sinal de entrada desses flip-flops para que a transição (mudança de estado) desejada ocorra.

Vamos analisar inicialmente o estado y_1. Restringindo a tabela de transição da associação 3 à variável y_1, temos:

y_1	y_1^*	
	$x = 0$	$x = 1$
a 0	0	0
c 0	0	1
d 1	1	1
b 1	0	1

Suponha que usaremos flip-flops JK neste circuito. Então, qual deve ser o valor de J e K para que a transição $y_1 \rightarrow y_1^*$ ocorra ? Para isso, recordemos a tabela do flip-flop JK. A tabela abaixo à esquerda descreve o comportamento do flip-flop JK e a tabela da direita mostra em quais condições ocorre a transição $Q \rightarrow Q^*$. O símbolo $\times$ indica uma entrada don't care.

JK	Q^*	
	$Q=0$	$Q=1$
00	0	1
01	0	0
10	1	1
11	1	0

$Q \rightarrow Q^*$	J	K
$0 \rightarrow 0$	0	$\times$
$0 \rightarrow 1$	1	$\times$
$1 \rightarrow 0$	$\times$	1
$1 \rightarrow 1$	$\times$	0

Então podemos montar uma tabela para J_1 e K_1 (as entradas J e K do primeiro flip-flop cujo estado é denotado por y_1), indicando os valores que produzirão a transição $y_1 \rightarrow y_1^*$.

Associação 3	y_1	y_1^*		J_1		K_1	
y_1y_0		$x=0$	$x=1$	$x=0$	$x=1$	$x=0$	$x=1$
a 00	0	0	0	0	0	$\times$	$\times$
c 01	0	0	1	0	1	$\times$	$\times$
d 11	1	1	1	$\times$	$\times$	0	0
b 10	1	0	1	$\times$	$\times$	1	0

As colunas y_1 e y_1^* simplesmente reproduzem uma das tabelas acima. Na primeira linha é indicada a transição de $y_1 = 0$ para $y_1^* = 0$ tanto quando $x = 0$ como quando $x = 1$. Essa transição ocorre quando $J_1 = 0$, independente do valor de K_1. Na segunda linha, quando $x = 0$, $y_1 = 0$ muda também para $y_1^* = 0$. Mas se $x = 1$ então $y_1 = 0$ muda para $y_1^* = 1$ e isso ocorre quando $J_1 = 1$, independente do valor de K_1. E assim por diante.

Usando procedimento similar aos mapas de Karnaugh, das colunas correspondentes a J_1 o único 1 na coluna $x = 1$ pode ser agrupado com o don't care $\times$ logo abaixo dele para obtermos $J_1 = x\,y_0$. Das correspondentes a K_1, o também unico 1 na coluna $x = 0$ pode ser agrupado com o don't care $\times$ logo abaixo del (na primeira linha da tabela) para obtermos $K_1 = \overline{x}\,\overline{y}_0$.

De forma análoga, repetimos o processo para y_0. Restringindo a tabela de transição da associação 3 à variável y_0, temos:

y_0	y_0^*	
	$x=0$	$x=1$
a 0	0	1
c 1	0	0
d 1	1	1
b 0	0	1

Portanto, as tabelas para J_0 e K_0 serão respectivamente

y_1y_0	J_0	
	$x=0$	$x=1$
00	0	
01	$\times$	$\times$
11	$\times$	$\times$
10	0	1

y_1y_0	K_0	
	$x=0$	$x=1$
00	$\times$	$\times$
01	1	1
11	0	0
10	$\times$	$\times$

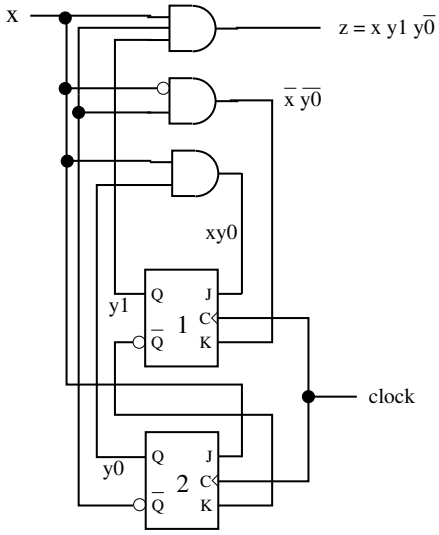
De onde obtemos que $J_0 = x$ e $K_0 = \overline{y}_1$.

Finalmente, a expressão para a saída z é dada por $x\, y_1\, \overline{y}_0$ (pois existe uma única situação em que a saída do circuito é 1; justamente quando ele se encontra no estado b e a entrada x é $= 1$. A expressão segue do fato de termos associado ao estado b o par $y_1 y_0 = 10$).

Procedimento similar pode ser aplicado para as associações 1 e 2. A associação 1 resulta em um circuito de custo (em termos de número total de portas lógicas) equivalente ao da associação 3 e a associação 2 resulta em um circuito de custo ligeiramente maior.

g) **O circuito!**

As equações obtidas para a associação 3 correspondem ao seguinte circuito.



Exercícios:

1. Desenho o diagrama de funcionamento do flip-flop JK ao longo do tempo, quando o mesmo é controlado por um sinal de clock. Suponha que inicialmente $J = K = Q = 0$. No diagrama, contemple todas as possíveis combinações de valores para J e K.

2. Desenho o diagrama de funcionamento do circuito contador incremental/decremental módulo 2^3 quando o sinal $up = 0$, para 10 pulsos do clock. Supondo que inicialmente o contador está com valor 0, qual é a seqüência de valores do contador?

3. Desenho o diagrama de funcionamento do circuito contador incremental módulo 2^3 assíncrono, para 10 pulsos do clock. Supondo que há um pequeno atraso até a saída dos flip-flops mudarem de valor desde a descida do clock, qual é a seqüência de valores do contador? Quais são os estados transitórios?

4. Escreva a equação das entradas dos flip-flops, a equação dos próximos estados e da saída do circuito para a associação 1 no caso do exemplo de projeto de circuitos visto acima.

Referências Bibliográficas

[Hill and Peterson, 1993] Hill, F. J. and Peterson, G. R. (1993). *Computer Aided Logical Design with Emphasis on VLSI*. John Wiley & Sons, fourth edition.

[Nelson et al., 1995] Nelson, V. P., Nagle, H. T., Carroll, B. D., and Irwin, J. D. (1995). *Digital Logic Circuit Analysis and Design*. Prentice-Hall.

Última revisão em 13 de junho de 2007.