**CORREÇÃO DE ERROS**

**CONCEITO**

* A transmissão de informação em formato digital (binário) é umas das operações mais frequentes em Sistemas Digitais.
* Devido a interferência externa, ruídos, atenuação do sinal, etc, o sinal pode ser corrompido e, consequentemente, a informação transmitida torna-se incorreta.
* A detecção e correção de erros lida com mecanismos para atenuar tais problemas por meio do envio de dados adicionais que permitem a verificação e possivelmente a correção de erros de transmissão.

**Método da paridade**

Bit de paridade

* Um bit extra é anexado ao conjunto de bits do código a ser transmitido

Paridade par

* O bit extra assume o valor 0 ou 1 de modo que o total de bits 1 seja par

P 0 1 1 1 0 0 0 -> 1 0 1 1 1 0 0 0

P 0 1 1 1 1 0 0 -> 0 0 1 1 1 1 0 0

Paridade impar

* O bit extra assume o valor 0 ou 1 de modo que o total de bits 1 seja impar

P 0 1 1 1 0 0 0 -> 0 0 1 1 1 0 0 0

P 0 1 1 1 1 0 0 -> 1 0 1 1 1 1 0 0

CÓDIGO ASCII COM PARIDADE PAR

100 0111 0100 0111

110 1111 0110 1111

110 1100 0110 1100

010 0000 1010 0000

110 0100 1110 0100

110 1111 0110 1111

010 0000 1010 0000

101 0110 0101 0110

110 0101 0110 0101

111 0010 0111 0010

110 0100 1110 0100

110 0001 1110 0001

110 1111 0110 1111

Gerador de paridade par

A B C D P

0 0 0 0 0

0 0 1 0 1

0 1 0 0 1

0 1 1 0 0

0 0 0 1 1

0 0 1 1 0

0 1 0 1 0

0 1 1 1 1

1 0 0 0 1

1 0 1 0 0

1 1 0 0 0

1 1 1 0 1

1 0 0 1 0

1 0 1 1 1

1 1 0 1 1

1 1 1 1 0

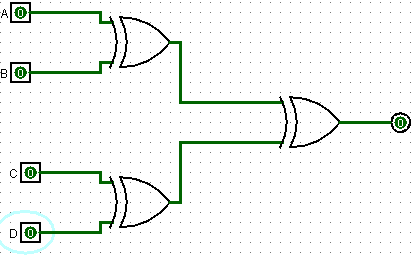
Após extraímos os mintermos da tabela verdade chegamos a seguinte fórmula: A (+) B (+) C (+) D

Obs: (+) = <->’ ( o ou exclusivo é equivalente a negação da bi-implicação )

CIRCUITO

O circuito de um gerador de paridade par funciona da seguinte maneira:

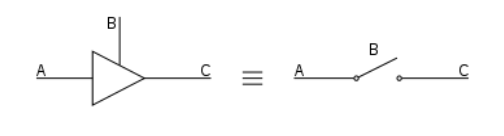
* A e B se tornam entradas da primeira porta ou exclusivo
* B e C se tornam entradas da segunda porta ou exclusivo
* A saída da primeira porta ou exclusivo e a saída da segunda porta ou exclusivo se tornam as entradas da terceira porta ou exclusivo
* A saída da terceira porta ou exclusiva é o valor de P

****

**Saída 3-state**

* Portas lógicas com saída 3-state permitem a geração de valores 0,1 ou Z.
* O estado Z é considerado como uma saída desconectada do circuito, pois se apresenta em outro estado de impedância.
* Basicamente, se a saída for Z não acontece nada, ou seja, a corrente não passa pelo circuito
* A real intenção do terceiro estado (Z) é remover efetivamente a influência de uma dada parte do circuito restante.

A = entrada

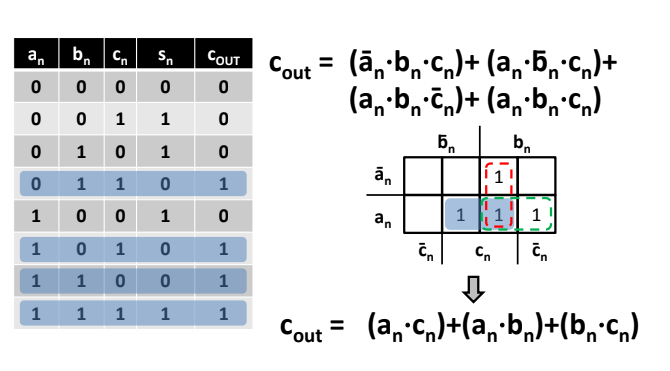
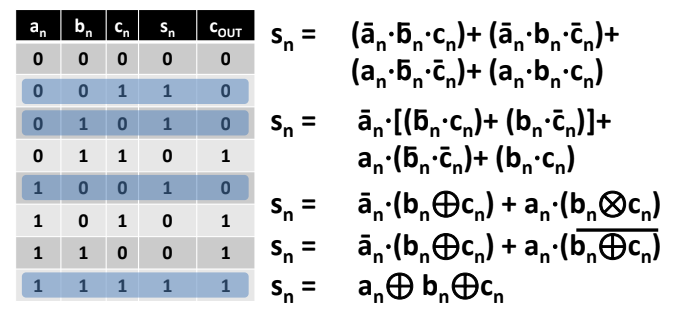
B = chave seletora

Se B = 1, C = A

Se B = 0, C = Z

**SOMADOR COMPLETO**

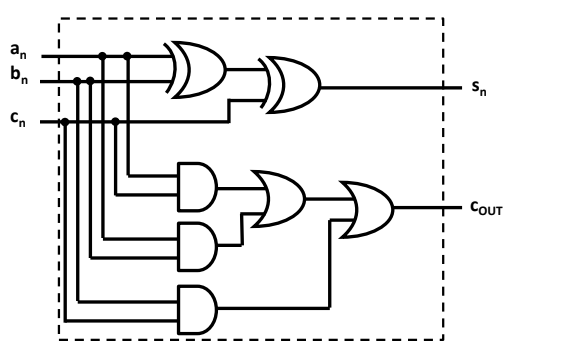
**CONCEITO**

O somador completo é composto de três entradas, an, bn e cn, e duas saídas, Sn e a COUT. an, bn e cn são os valores que serão somados, de acordo com a tabela verdade há 8 possiblidades diferentes para tais somas. O sn é o resultado da soma, e o COUT é correspondente ao “vai um” da soma. Ao extrairmos da tabela verdade os mintermos de Sn chegamos a fórmula Sn = (an (+) bn (+) cn), e ao extrairmos os mintermos de COUT chegamos a fórmula COUT = (an.cn) + (an.bn) + (bn.cn).

OBS: 1 + 1 = 0, vai 1; 1 + 0 = 1.

**CIRCUITO**

O circuito em um somador completo funciona da seguinte maneira:

Na primeira parte do circuito:

* an e bn se tornam entradas para a primeira porta ou exclusivo
* A saída da primeira porta ou exclusivo e cn se tornam entradas para a segunda porta ou exclusivo
* A saída da segunda porta ou exclusivo é o valor de Sn, ou seja, a resposta da soma

Já na segunda parte do circuito:

* an e cn se tornam entradas para a primeira porta and
* an e bn se tornam entradas para a segunda porta and
* bn e cn se tornam entradas para a terceira porta and
* A saída da primeira porta and, e a saída da segunda porta and se tornam entradas para a primeira porta or
* A saída da primeira porta or e a saída da terceira porta and se tornam entradas para a segunda porta or
* A saída é o valor de COUT, ou seja, o valor do “vai um”.

**SOMADOR DE OITO BITS**

**CONCEITO**

Um somador de 8 bits funciona basicamente desta forma:

c7 c6 c5 c4 c3 c2 c1 1 1 1 1 1 1 1

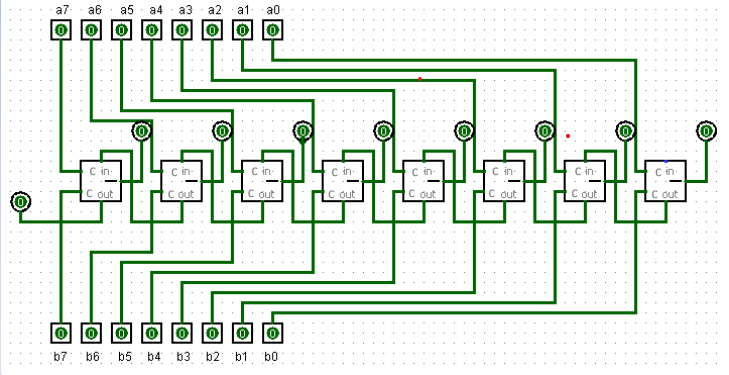
+ a7 a6 a5 a4 a3 a2 a1 a0 + 1 0 1 1 0 0 0 1

b7 b6 b5 b4 b3 b2 b1 b0 0 0 0 0 1 1 1 1

s7 s6 s5 s4 s3 s2 s1 s0 1 1 0 0 0 0 0 0

O somador de oito bits possui 16 entradas (8 para os bits do primeiro valor, e 8 para os bits do segundo valor), nove saídas (8 para o resultado da soma de cada bit, e uma para o último valor de COUT), e oito somadores completos.

**CIRCUITO**



O circuito de um somador de oito bits funciona da seguinte maneira:

* a0 e b0 se tornam as entradas do oitavo somador completo, sua primeira saída é o s0, sua segunda saída é o c1.
* A1, b1 e c1 e tornam as entradas do sétimo somador completo, sua primeira saída é o s1, sua segunda saída é o c2.
* A2, b2 e c2 e tornam as entradas do sexto somador completo, sua primeira saída é o s2, sua segunda saída é o c3.
* A3, b3 e c3 e tornam as entradas do quinto somador completo, sua primeira saída é o s3, sua segunda saída é o c4.
* A4, b4 e c3 e tornam as entradas do quarto somador completo, sua primeira saída é o s4, sua segunda saída é o c5.
* A5, b5 e c4 e tornam as entradas do terceiro somador completo, sua primeira saída é o s5, sua segunda saída é o c6.
* A6, b6 e c5 e tornam as entradas do segundo somador completo, sua primeira saída é o s6, sua segunda saída é o c7.
* A7, b7 e c6 e tornam as entradas do primeiro somador completo, sua primeira saída é o s7, sua segunda saída é o c8.

S7, s6, s5,s4,s2,s1,s0 é o resultado da soma de a7,a6,a5,a4,a3,a2,a1,a0 com b7,b6,b5,b4,b3,b2,b1,b0.

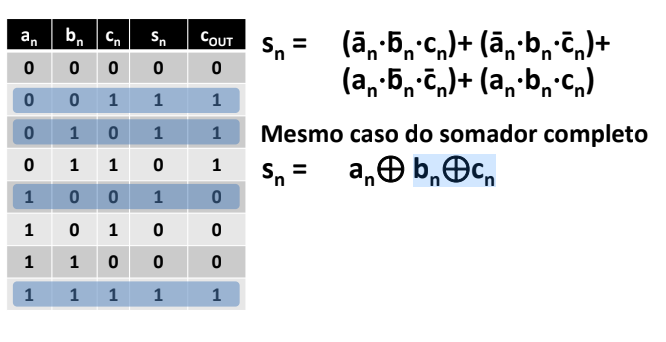
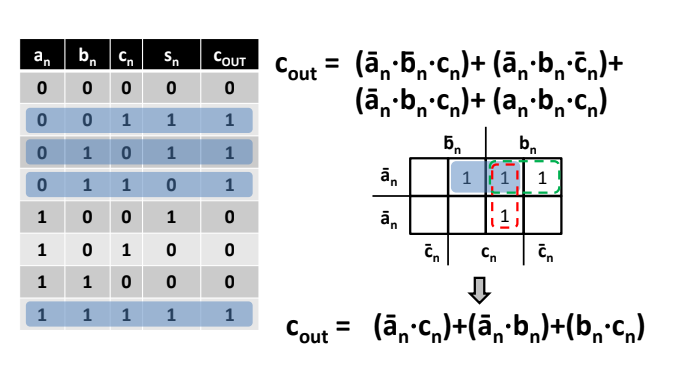
Note que os valores de todos os “vai um” das somas foram extremamente importantes para o circuito, pois se o COUT não existisse o resultado da soma seria completamente diferente.

**SUBTRATOR COMPLETO**

**CONCEITO**

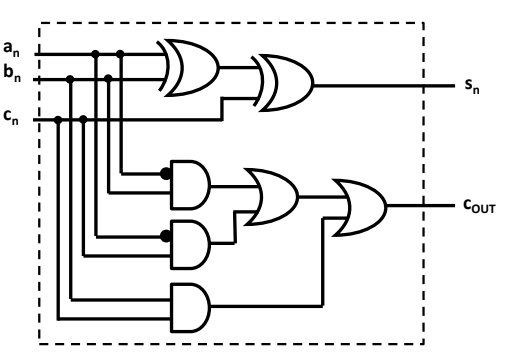
O subtrator completo é composto de três entradas, an, bn e cn, e duas saídas, Sn e a COUT. an, bn e cn são os valores que serão subtraídos, de acordo com a tabela verdade há 8 possiblidades diferentes para tais subtrações. O sn é o resultado da subtração, e o COUT é correspondente ao “vai um” da subtração. Ao extrairmos da tabela verdade os mintermos de Sn chegamos a fórmula sn = (an (+) bn (+) cn), e ao extrairmos os mintermos de COUT chegamos a fórmula COUT = (an'.cn) + (an'.bn) + (bn.cn).

A única diferença entre o somador completo e o subtrator completo está no valor de COUT, o valor de COUT é diferente no subtrator completo devido ao fato das operações de subtração com binários possuírem uma peculiaridade no “vai um”, exemplo 0 - 1, = 1 e vai um; 0 – 1 – 1, = 0, e vai um.

****

OBS: 0 – 1 = 1, vai um;

**CIRCUITO**

O circuito em um subtrator completo funciona da seguinte maneira:

Na primeira parte do circuito:

* an e bn se tornam entradas para a primeira porta ou exclusivo
* A saída da primeira porta ou exclusivo e cn se tornam entradas para a segunda porta ou exclusivo
* A saída da segunda porta ou exclusivo é o valor de Sn, ou seja, a resposta da subtração

Já na segunda parte do circuito:

* an’ e cn se tornam entradas para a primeira porta and
* an’ e bn se tornam entradas para a segunda porta and
* bn e cn se tornam entradas para a terceira porta and
* A saída da primeira porta and, e a saída da segunda porta and se tornam entradas para a primeira porta or
* A saída da primeira porta or e a saída da terceira porta and se tornam entradas para a segunda porta or
* A saída é o valor de COUT, ou seja, o valor do “vai um”.

**SUBTRATOR DE OITO BITS**

**CONCEITO**

Um subtrator de 8 bits funciona basicamente desta forma:

a7 a6 a5 a4 a3 a2 a1 a0 1 0 1 1 0 0 0 1

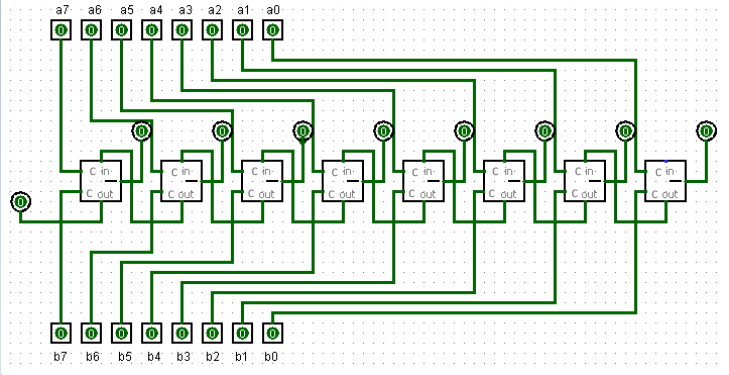
- c7 c6 c5 c4 c3 c2 c1 - 1 1 1 0

b7 b6 b5 b4 b3 b2 b1 b0 0 0 0 0 1 1 1 1

s7 s6 s5 s4 s3 s2 s1 s0 1 0 1 0 0 0 1 0

O subtrator de oito bits possui 16 entradas (8 para os bits do primeiro valor, e 8 para os bits do segundo valor), nove saídas (8 para o resultado da subtração de cada bit, e uma para o último valor de COUT), e oito subtratores completos.

**CIRCUITO**



O circuito de um subtrator de oito bits funciona da seguinte maneira:

* a0 e b0 se tornam as entradas do oitavo subtrator completo, sua primeira saída é o s0, sua segunda saída é o c1.
* A1, b1 e c1 se tornam as entradas do sétimo subtrator completo, sua primeira saída é o s1, sua segunda saída é o c2.
* A2, b2 e c2 se tornam as entradas do sexto subtrator completo, sua primeira saída é o s2, sua segunda saída é o c3.
* A3, b3 e c3 se tornam as entradas do quinto subtrator completo, sua primeira saída é o s3, sua segunda saída é o c4.
* A4, b4 e c3 se tornam as entradas do quarto subtrator completo, sua primeira saída é o s4, sua segunda saída é o c5.
* A5, b5 e c4 se tornam as entradas do terceiro subtrator completo, sua primeira saída é o s5, sua segunda saída é o c6.
* A6, b6 e c5 e tornam as entradas do segundo subtrator completo, sua primeira saída é o s6, sua segunda saída é o c7.
* A7, b7 e c6 e tornam as entradas do primeiro subtrator completo, sua primeira saída é o s7, sua segunda saída é o c8.

s7, s6, s5,s4,s2,s1,s0 é o resultado da subtração de a7,a6,a5,a4,a3,a2,a1,a0 com b7,b6,b5,b4,b3,b2,b1,b0.

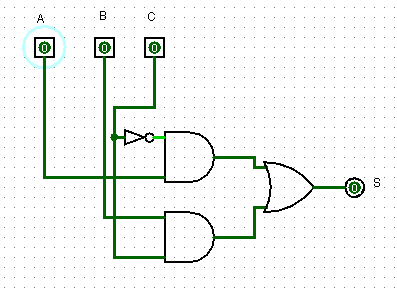
Note que os valores de todos os “vai um” das subtrações foram extremamente importantes para o circuito, pois se o COUT não existisse o resultado da subtração seria completamente diferente.

**MUX**

**CONCEITO**

O multiplexador é composto de entradas, uma saída e chaves correspondentes a quantidade de entradas, se há duas entradas, há uma chave seletora, se há 4 entradas há duas chaves seletoras, se há 8 entradas há três chaves seletoras etc. A chave seletora serve para uma informação ser acessada, ou seja, ela serve para selecionar dentre várias entradas somente uma. Em um MUX 4-1, se C1 = 0 e C2 = 0, as chaves irão acessar o valor de I1 e com isso a informação conseguirá percorrer o circuito e chegar na saída.

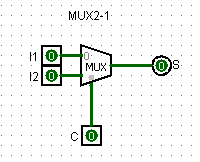
**CIRCUITO MUX 2-1**

1. A primeira entrada e a primeira possiblidade para a chave se tornam entradas de uma porta and
2. A segunda entrada e e a segunda possibilidade para a chave se tornam entradas de uma porta and
3. Uma porta or realiza a disjunção entre os dois valores obtidos nas duas portas and
4. Desta maneira, a entrada correspondente ao valor selecionado nas chave consegue chegar no final do circuito.

A B C S

... ... 0 A

... .... 1 B

A B C C’ (A.C’) (B.C) S

0 0 0 1 0 0 0

0 1 0 1 0 0 0

1 0 1 0 0 0 0

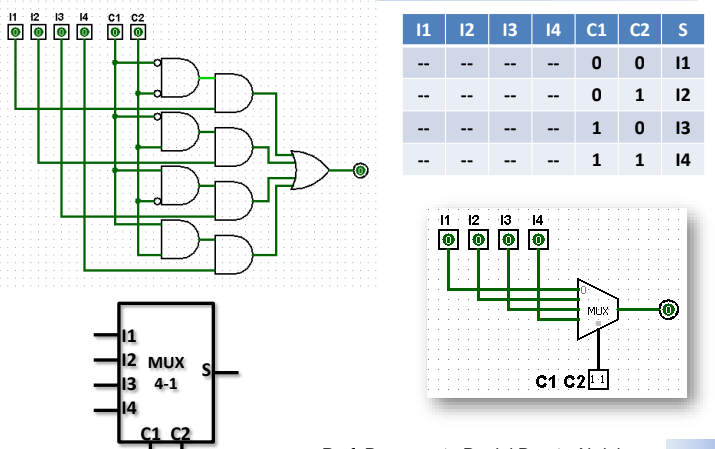
1 1 1 0 0 1 1

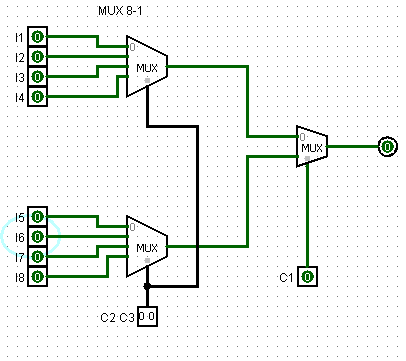
**CIRCUITO MUX 4 -1**

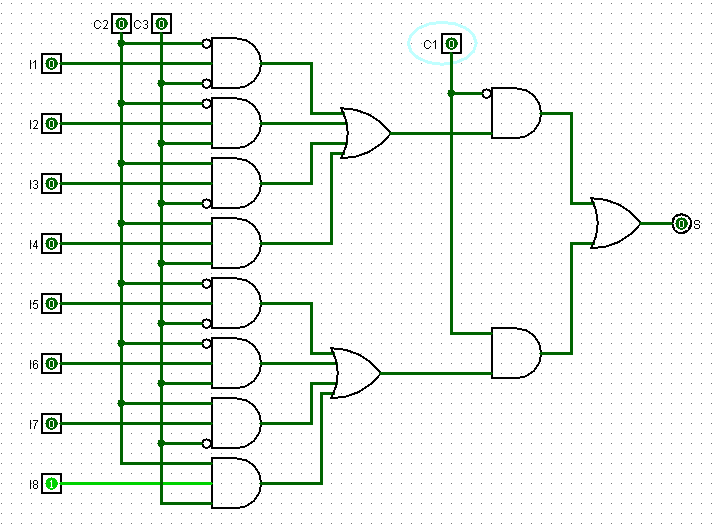
O circuito em um MUX 4-1 funciona da seguinte maneira:

1. Todas as possibilidades para as chaves são unidas em quatro portas and
2. Cada saída das portas and é unida com uma das quatro entradas em quatro portas and,
3. Uma porta or realiza a disjunção entre todos os valores obtidos nas últimas quatro portas and.
4. Desta maneira, a entrada correspondente ao valor selecionado nas chaves consegue chegar no final do circuito.

Basicamente, multiplexadores permitem a seleção de uma entre várias possíveis entradas.



**CIRCUITO MUX 8-1**

Dois MUX 4-1 e um MUX 2-1 são utilizados na construção de um MUX 8-1.

O circuito em um MUX 8-1 funciona da seguinte maneira:

1.C2 e C3 são as chaves do primeiro e do segundo MUX

No primeiro MUX 4-1:

2.Cada possiblidade para as chaves e cada uma das quatro primeiras entradas são unidas em quatro portas and

3.Uma porta or realiza a disjunção entre todos os valores obtidos nas últimas quatro portas and.

No segundo MUX 4-1:

4.Cada possiblidade para as chaves e cada uma das quatro últimas entradas são unidas em quatro portas and

5.Uma porta or realiza a disjunção entre todos os valores obtidos nas últimas quatro portas and.

6.C1 se torna a chave do MUX 2-1.

7.C1 controla se o segundo MUX 2-1 irá acessar a saída do primeiro MUX 4-1 ou do segundo MUX 4-1

No MUX 2-1:

8.A saída da primeira porta or e a primeira possiblidade para a chave se tornam entradas da primeira porta and

9.A saída da segunda porta or e a segunda possibilidade para a chave se tornam entradas da segunda porta and

10.Uma porta or realiza a disjunção entre os dois valores obtidos nas duas portas and

11.Desta maneira, a entrada correspondente ao valor selecionado nas chaves consegue chegar no final do circuito.

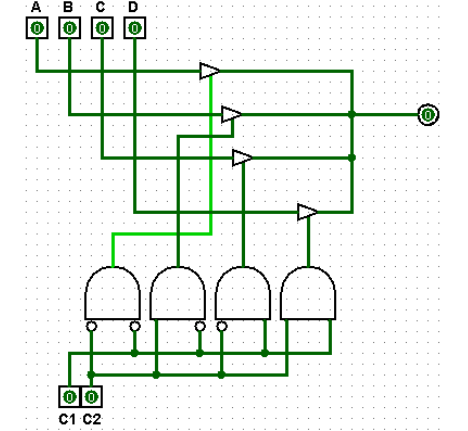
**MUX via Buffers 3-state**

**CONCEITO E CIRCUITO**

O que diferencia o MUX 3-state de um MUX normal é que no lugar de determinadas portas and utiliza-se buffers 3-state, a porta or também se torna desnecessária. As chaves continuam selecionando qual informação será acessada, as possibilidades para as chaves continuam sendo unidas em portas and, mas desta vez a saída de cada porta and se torna a chave seletora de um buffer 3-state conectado à entrada correspondente as chaves do circuito.

Se C1 = 0 e C2 = 0 e A = 1, a chave C do buffer 3-state de A será 1, se C = 1, a informação de A será percorrida por todo o circuito, portanto a saída do buffer e do circuito é o valor de A.

Se C1 = 1 e C2 = 1 e A = 1, a chave C do buffer 3-state de A será 0, se C = 0, a informação de A não será percorrida por todo o circuito, porém a chave C do buffer 3-state de D será 1, portanto a saída do buffer 3-state e a saída do circuito é o valor de D.



**DEMUX**

**CONCEITO**

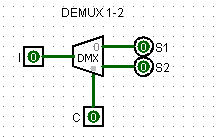
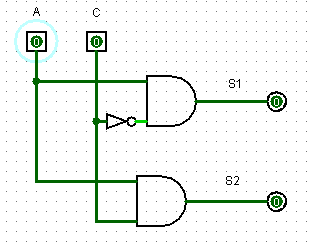
O demultiplexador é composto de uma entrada, saídas e chaves correspondentes a quantidade de saídas, se há duas saídas, há uma chave, se há quatro saídas há duas chaves, se há oito saídas há três chaves etc. A chave seletora serve para uma informação ser direcionada para uma saída específica, ou seja, ela serve para selecionar dentre várias saídas somente uma. Em um DEMUX 1-4, se C1 = 0 e C2 = 0, as chaves irão direcionar a informação para a saída S1.

**CIRCUITO DEMUX 1 – 2**

1. A única entrada e cada possiblidade para a chave C1 são unidas em duas portas and

2. A primeira e segunda saída do circuito são dispostas após cada porta and

3. Desta maneira, a saída correspondente ao valor selecionado nas chaves consegue receber a informação

A C S1 S2

... 0 A 0

... 1 0 A

S1 S2

A C C’ A.C’ A.C

0 0 1 0 0

1 0 1 1 0

0 1 0 0 0

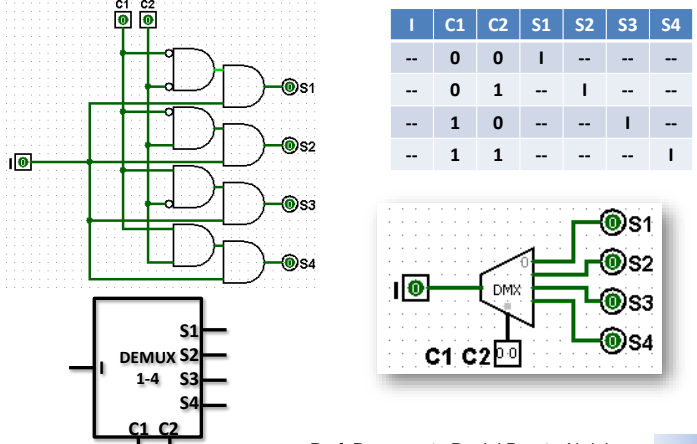
1 1 0 0 1

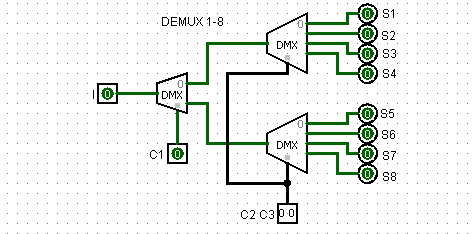
**CIRCUITO DEMUX 1 - 4**

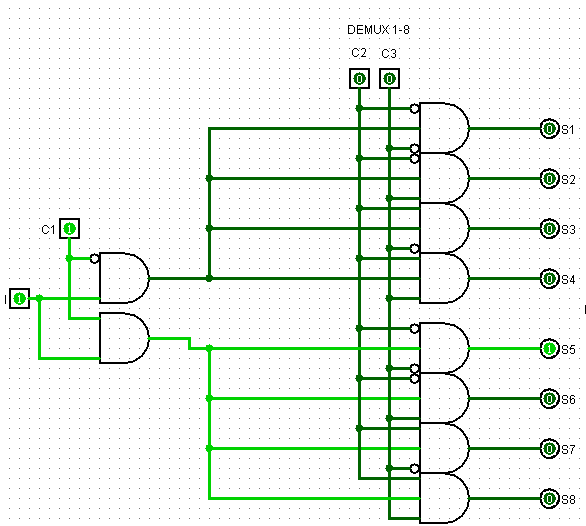
O circuito em um DEMUX 1-4 funciona da seguinte maneira:

1. Todas as possibilidades para as chaves são unidas em quatro portas and
2. Cada saída das portas and é unida com a única entrada em quatro portas and
3. As respectivas saídas são dispostas após as uniões.
4. Desta maneira, a saída correspondente ao valor selecionado nas chaves consegue receber a informação.

Basicamente, demultiplixadores permitem o roteamento de um único canal de informação para diferentes canais.



**CIRCUITO DEMUX 1-8**

Dois DEMUX 4-1, e um DEMUX 2-1 são utilizados na construção de um DEMUX 8-1.

1.No primeiro DEMUX 2-1:

2.C1 é a chave do DEMUX 2-1

3.C1 controla se o primeiro MUX 2-1 irá direcionar a informação para o primeiro DEMUX 4-1 ou para o segundo DEMUX 4-1

4. A única entrada e a primeira possiblidade para a chave C1 se tornam entradas de uma porta and

5. A única entrada e a segunda possibilidade para a chave C1 se tornam entradas de uma porta and

7.C2 e C3 são as chaves dos dois DEMUX 4-1

7.No primeiro DEMUX 4-1:

9. A saída da primeira porta and e cada possibilidade para as chaves C2 e C3 são unidas em quatro portas and

10. A primeira, segunda, terceira e quarta saída do circuito são dispostas após cada porta and

11.No segundo DEMUX 4-1

12. A saída da segunda porta and e cada possiblidade para as chaves C2 e C3 são unidas em quatro portas and

14.A quinta, sexta, sétima e oitava saída do circuito são dispostas após cada porta and

16. Desta maneira, a saída correspondente ao valor selecionado nas chaves consegue receber a informação.

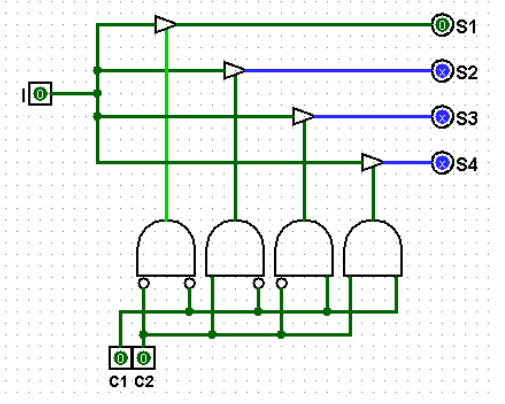
**DEMUX com Buffers 3-state**

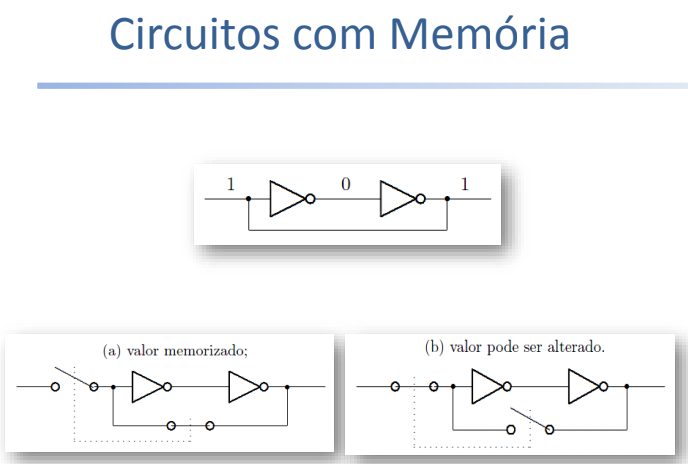
**CONCEITO E CIRCUITO**

O que diferencia um DEMUX 3- state de um DEMUX normal é que no lugar de determinadas portas and utiliza-se buffers 3 state. As chaves continuam selecionando a saída para qual a informação será direcionada, as possibilidades para as chaves continuam sendo unidas em portas and, mas desta vez a saída de cada porta and se torna a chave seletora de um buffer 3-state conectado a única entrada do circuito.

Se C1 = 0 e C2 = 0 e I = 1, a chave C do buffer 3-state de S1, será 1, se C = 1, a informação de I será direcionada para a saída S1.

Se C1 = 0 e C2 = 0 e I = 1, a chave C do buffer 3-state de S2,S3,S4 será 0, se C = 0, a saída de S2,S3,S4 será Z, ou seja, as saídas S2,S3,S4 possuem uma saída de alta impedância porque nenhuma corrente foi passada por elas, ou seja, nesta parte do circuito não aconteceu nada.

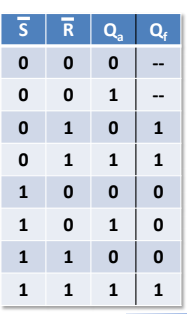




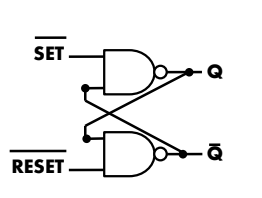
**LATCHES(FLIP FLOP RS)**

**CONCEITO**

O latch é composto de duas entradas, a SET e a RESET, e duas saídas, a Q e a Q'. Ele é um circuito sequencial, ou seja, a saída depende de suas entradas no instante atual e nos instantes anteriores. Portanto, latches são circuitos que possuem memória. Além de ser um circuito sequencial, ele também é um circuito sincrono, ou seja, as saídas podem mudar de estado a qualquer momento. O efeito de armazenamento em um latch é alcançado graças a uma ligação retroalimentada, ou seja, cada saída é ligada como entrada em algum ponto anterior do circuito.

**CIRCUITO LATCHES**

S’ R’ Qa Qa’ Qf Qf’

0 0 0 1 - - - -

0 0 1 1 - - - -

0 1 0 1 1 0

0 1 1 0 1 0

1 0 0 1 0 1

1 0 1 1 0 1

1 1 0 1 0 1

1 1 1 0 1 0

O circuito em um Latch com portas nand funciona da seguinte maneira: o valor de Q e RESET’ se tornam entradas na segunda porta nand do latch, seu resultado, ou seja, Q’, e o SET’ se tornam as entradas da primeira porta nand do latch. O funcionamento básico de um latch é exatamente esse, cada saída se torna uma das entradas da outra porta nand.

Se SET’ = 0 , RESET’ = 1, Q = 0, então Q’ = 1, Qf = 1, Qf’ = 0, ou seja os valores das saídas foram invertidos e ocorreu um set no valor de Q e um reset no valor de Q’.

Se SET’ = 1, RESET’ = 0, Q = 1, então Q’ = 1, Qf = 0, Qf’ = 1, ou seja, os valores das saídas foram invertidos e ocorreu um reset no valor de Q e um set no valor de Q’.

Obs = Se SET’ = 0 e RESET’ = 0, os valores de Q E Q’ não ficarão invertidos, portando, tecnicamente tal possiblidade não faz sentido e está incorreta.

**FLIP\_FLOP RS COM CLOCK**

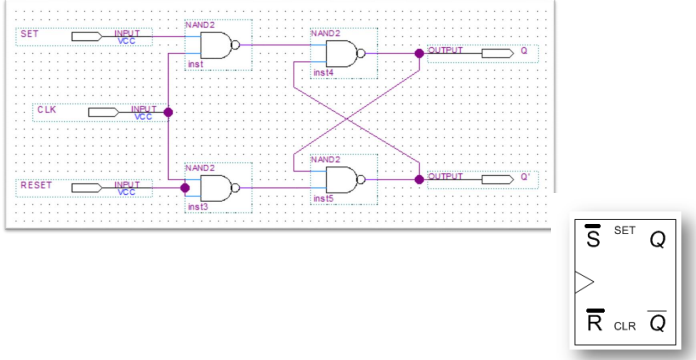
**CONCEITO E CIRCUITO**

O que diferencia o FLIP FLOP com clock do FLIP FLOP normal é que agora o circuito possui mais duas portas nand (a primeira possui como entrada o SET’ e o CLK, a segunda possui como entrada o RESET’ e o CLK) e mais uma entrada, ou seja, o CLK. O clock funciona como uma chave seletora para o circuito, o valor do clock irá definir se o circuito irá funcionar ou não, ou seja, o clock define se a corrente necessária para o circuito funcionar corretamente irá percorrer o circuito ou não.

Se CLK = 0, as portas nand da esquerda sempre irão produzir saída 1, ou seja, o FLIP FLOP sempre irá ficar no modo estável, isto é, as saídas anteriores irão se manter, independente do valor de SET’ e RESET’.

Ex: Se CLK = 0 , SET =1, RESET = 1, então a saída do primeiro e segundo nand é 1, no circuito do latch a saída do segundo nand será 1, e com isso a saída do primeiro nand será 0. Ou seja, o valor de Q não se alterou, Qa era 0 e continuou sendo 0.

Se CLK = 1, o circuito passa a funcionar como um FLIP FLOP RS normal.



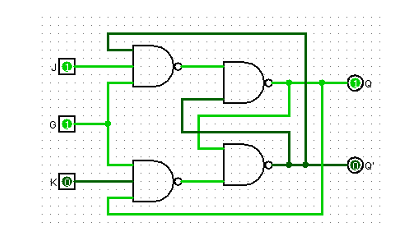
**FLIP FLOP JK**

**CONCEITO E CIRCUITO**

Há duas características que diferenciam um flip flop JK de um FLIP FLOP com clock:

1. J, Q’, e CLK são as entradas da primeira porta nand localizada antes do latch
2. K, Q e CLK são as entradas da segunda porta nand localizada antes do latch

J K Qa Qf Obs

0 0 0 0 MANTEM

0 0 1 1 MANTEM

0 1 0 0 RESET

0 1 1 0 RESET

1 0 0 1 SET

1 0 1 1 SET

1 1 0 1 INVERTE

1 1 1 0 INVERTE

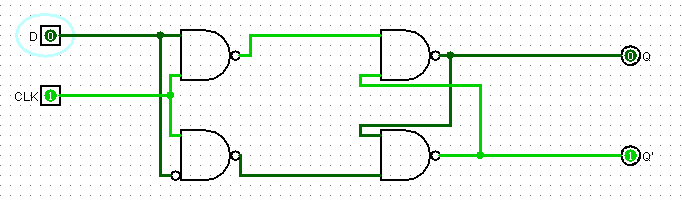
**FLIP FLOP TIPO D**

**CONCEITO E CIRCUITO**

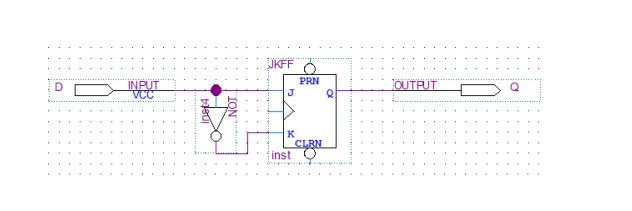
O circuito de um flip flop D funciona da seguinte maneira: a informação (D) entra no circuito, se ramificada em dois caminhos, um que possui uma porta not, e outra que não possui, o caminho que não possui se direciona a entrada J de um flip flop JK, e o caminho que possui a porta not se direciona a entrada K do flip flop Jk.

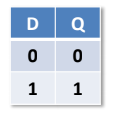
O flip flop JK possui mais uma entrada, a CLK, tal entrada que irá controlar se o circuito irá funcionar normalmente ou se as saídas antigas serão mantidas.

Como o flip flop JK sempre irá receber a informação e a informação negada, suas possibilidades de entrada sempre serão 0 e 1 ou 1 e 0.

 Se CLK = 1, tais possiblidades sempre produzirão as mesmas saídas:

Se D = 0 e D’ = 1, então Qf = 0,

 Se D = 1 e D’= 0, então Qf = 1.



**FLIP FLOP TIPO T**

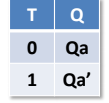
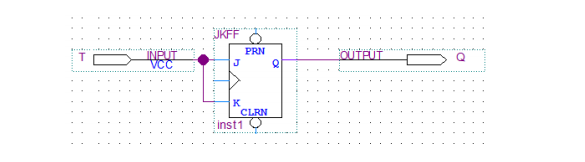
**CONCEITO E CIRCUITO**

O circuito de um flip flop T funciona da seguinte maneira: a informação (T) entra no circuito, se ramifica em dois caminho, o primeiro caminho se direciona a entrada J do flip flop JK, e o segundo caminho se direciona a entrada K do flip flop JK.

O flip flop JK possui mais uma entrada, a CLK, tal entrada que irá controlar se o circuito irá funcionar normalmente ou se as saídas antigas serão mantidas.

Como o flip flop JK sempre irá receber uma entrada igual a outra, suas possibilidades de entrada sempre serão J = 0 e K = 0 ou J = 1 e K = 1.

Se CLK = 1, tais possiblidades irão produzir as seguintes saídas:

Se J = 0, K = 0 e Qa = 0, então Qf = 0

Se J = 0, K = 0 e Qa = 1, então Qf = 1

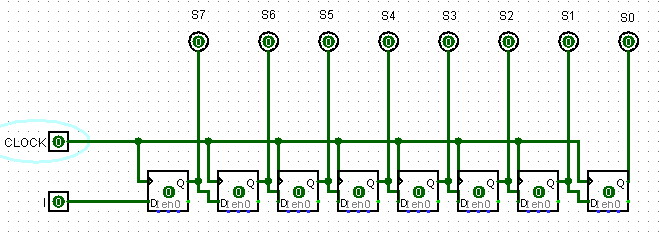
Se J = 1, K = 1 e Qa = 0, então Qf = 1

Se J = 1, K = 1 e Qa = 1, então Qf = 0

Ou seja, como podemos perceber

Se T = 0, então Qf= Qa

Se T = 1, então Qf = Qa’

**CONVERSOR SÉRIE PARELELO**

**CONCEITO**

O conversor série paralelo é composto de duas entradas, a I e a clock., uma saída e flips flops do tipo D correspondentes a quantidade de bits que serão enviados para o circuito, se 8 bits forem enviados 8 flips flops do tipo D serão utilizados.

O dado de entrada é fornecido bit a bit a cada ciclo do clock.

OBS:

Na comunicação serial, os bits são transmitidos um de cada vez, ou seja, bit a bit.

Já na comunicação paralela, os bits são transmitidos simultaneamente.

**CIRCUITO**

Os pulsos do clock definem se os flip flops irão funcionar normalmente ou se somente irão manter os valores antigos.

* Em um flip flop tipo D:

Se CLOCK = 1 e D = 0, a saída é 0

Se CLOCK = 1 e D = 1, a saída é 1

Se I = 1, e CLK =0, nada ocorre, as saídas anteriores se manterão.

Se I = 1, e CLK = 1, I entra no flip flop e logo após vai para a primeira saída e para o próximo flip flop,

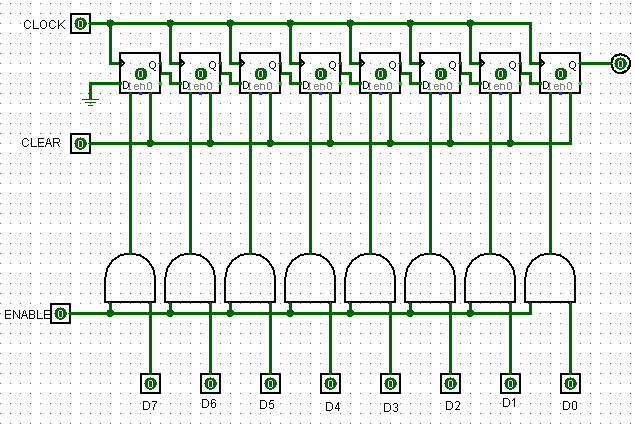
Se I = 0, e CLK = 1, o novo I vai para o primeiro flip flop, e o antigo I saí do segundo Flip Flop, vai para a segunda saída e para o próximo flip flop.

Se I = 0, e CLK = 0, nada ocorre, as saídas anteriores se manterão.

Ou seja, o circuito funciona basicamente desta maneira: a cada pulso do clock os flips flops D fazem com que o bit atual seja deslocado para a direita. Se 8 bits serão transmitidos, irá ocorrer 8 pulsos de clock.

OBS:

Os bits são inseridos da direita para a esquerda, exemplo para inserir 101010, envia-se o digito menos significativo, desta maneira os flip flops conseguem deslocar os bits para direita e obter como saída 101010 em paralelo.

**CONVERSOR PARALELO – SERIE**

**CONCEITO**

O conversor paralelo-série é composto pelas entradas da informação em paralelo, por três entradas que são obrigatórias neste circuito, a CLOCK, a CLEAR e a ENABLE, por uma saída, e por flip flops do tipo D correspondentes a quantos bits a informação possui, se possui 8 bits necessita-se de 8 flip flops etc.

* A clock irá apenas definir se o circuito funcionará como se espera ou não.
* A clear irá zerar todos os valores presentes nos flip flops se for necessário.
* A enable irá definir se as informações poderão ser enviadas para os flip flops ou não.

**CIRCUITO**

O circuito de um conversor paralelo – serie funciona da seguinte maneira:

* Neste circuito os bits são “lidos” da esquerda pra direita, ou seja, no circuito as entradas serão colocadas do digito menos significativo para o mais significativo.
* Cada bit da informação e o enable se tornam entradas de uma porta and, se há 8 bits, existem 8 portas and
* Se enable = 1, as saídas das portas and irão ser direcionadas para a entrada preset dos flip flops,, ou seja, a saídas das portas and se tornarão as saídas pré definidas dos flip flops. Ou seja, se enable = 1, a informação será transmitida e armazenada nos flip flops.
* O primeiro flip flop recebe na entrada D o valor de um Terra, ou seja, 0.
* A saída do primeiro se torna a entrada D do segundo, a saída do segundo se torna a entrada D do terceiro, etc.
* Em um flip flop tipo D:

Se CLOCK = 1 e D = 0, a saída é 0

Se CLOCK = 1 e D = 1, a saída é 1

* Com base nesse raciocínio, concluímos que para os bits saírem dos flips flops bit a bit é preciso somente dar pulsos de clock.
* Se a informação presente nos flips flops for 10101010:

No primeiro pulso:

A saída do primeiro flip flop se torna 0

A saída do segundo flip flop se torna 1

A saída do terceiro flip flop se torna 0

A saída do quarto flip flop se torna 1

A saída do quinto flip flop se torna 0

A saída do sexto flip flop se torna 1

A saída do sétimo flip flop se torna 0

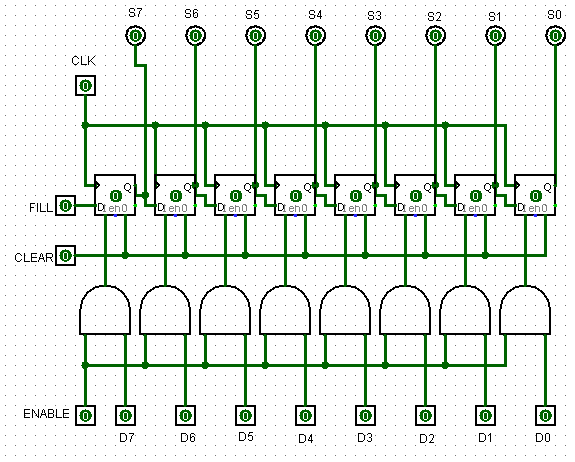
A saída do oitavo flip flop e do circuito se torna 1

Com isso, o bit mais significativo da informação sai do circuito.

Nos pulsos seguintes ocorre o mesmo procedimento, ou seja, os bits são deslocados para a direita

até chegarem no final do circuito.

Desta maneira, os bits que estavam em paralelo se transformam em bits em série.

**DESLOCAMENTO PARA DIREITA**

**CONCEITO**

O circuito de um deslocador para direita é composto pelas entradas da informação em paralelo, por quatro entradas que são obrigatórias neste circuito, a CLOCK, a CLEAR e a ENABLE, e a FILL, pelas saídas da informação em paralelo, e por flip flops do tipo D correspondentes a quantos bits a informação possui, se possui 8 bits necessita-se de 8 flip flops etc.

* A clock irá apenas definir se o circuito funcionará como se espera ou não.
* A clear irá zerar todos os valores presentes nos flip flops se for necessário.
* A enable irá definir se as informações poderão ser enviadas para os flip flops ou não.
* A FILL define se os bits vagos serão preenchidos com 0 ou com 1.

**CIRCUITO** O circuito de um deslocador para direita funciona da seguinte maneira:

* Neste circuito os bits são “lidos” da esquerda pra direita, ou seja, no circuito as entradas serão colocadas do digito menos significativo para o mais significativo.
* Cada bit da informação e o enable se tornam entradas de uma porta and, se há 8 bits, existem 8 portas and
* Se enable = 1, as saídas das portas and irão ser direcionadas para a entrada preset dos flip flops,, ou seja, a saídas das portas and se tornarão as saídas pré definidas dos flip flops. Ou seja, se enable = 1, a informação será transmitida e armazenada nos flip flops.
* O primeiro flip flop recebe na entrada D o valor do FILL, ou seja, 0 ou 1.
* A saída do primeiro flip flop se torna a entrada D do segundo, a saída do segundo se torna a entrada D do terceiro, etc.
* A saída de cada flip flop se torna uma das saídas do circuito, se a informação possui 8 bits, há oito entradas e oito saídas.

Em um flip flop tipo D:

Se CLOCK = 1 e D = 0, a saída é 0

Se CLOCK = 1 e D = 1, a saída é 1

* Com base nesse raciocínio, concluímos que para os bits se deslocarem para a direita é preciso somente dar pulsos de clock.
* Se a informação presente nos flips flops for 10101010:

No primeiro pulso:

A saída do primeiro flip flop se torna 0

A saída do segundo flip flop se torna 1

A saída do terceiro flip flop se torna 0

A saída do quarto flip flop se torna 1

A saída do quinto flip flop se torna 0

A saída do sexto flip flop se torna 1

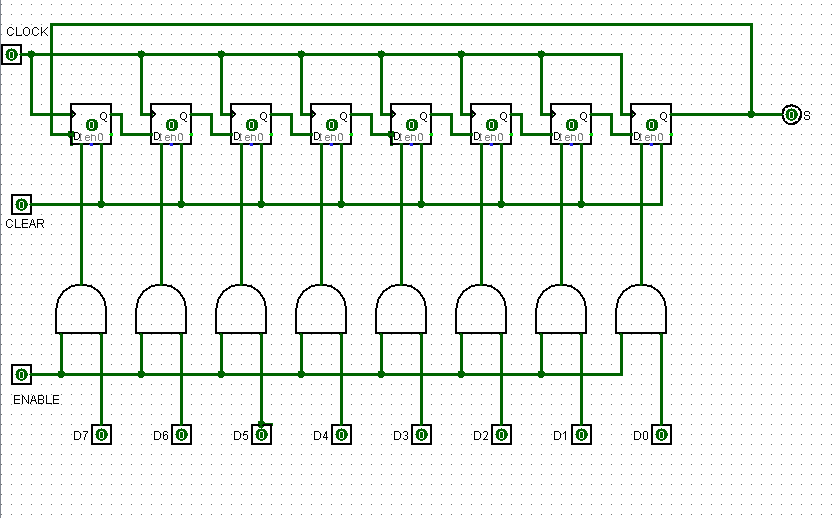
A saída do sétimo flip flop se torna 0

A saída do oitavo flip flop e do flip flop se torna 1

Nos pulsos seguintes ocorre o mesmo procedimento, ou seja, os bits são deslocados para a direita

até chegarem no final do circuito e serem removidos, bit a bit, um por um

**DESLOCAMENTO CIRCULAR**

****

**CONCEITO**

O circuito de um deslocador circular é composto pelas entradas da informação em paralelo, por três entradas que são obrigatórias neste circuito, a CLOCK, a CLEAR e a ENABLE, por uma saída que armazenará a saída do último flip flop que será enviado para o primeiro flip flop, e por flip flops do tipo D correspondentes a quantos bits a informação possui, se possui 8 bits necessita-se de 8 flip flops etc.

* A clock irá apenas definir se o circuito funcionará como se espera ou não.
* A clear irá zerar todos os valores presentes nos flip flops se for necessário.
* A enable irá definir se as informações poderão ser enviadas para os flip flops ou não

**CIRCUITO**

O circuito de um deslocador circular funciona da seguinte maneira:

* Neste circuito os bits são “lidos” da esquerda pra direita, ou seja, no circuito as entradas serão colocadas do digito menos significativo para o mais significativo.
* Cada bit da informação e o enable se tornam entradas de uma porta and, se há 8 bits, existem 8 portas and
* Se enable = 1, as saídas das portas and irão ser direcionadas para a entrada preset dos flip flops,, ou seja, a saídas das portas and se tornarão as saídas pré definidas dos flip flops. Ou seja, se enable = 1, a informação será transmitida e armazenada nos flip flops.
* O primeiro flip flop recebe na entrada D a saída do último fip flop
* A saída do primeiro flip flop se torna a entrada D do segundo, a saída do segundo se torna a entrada D do terceiro, etc.
* O deslocamento circular se diferencia porque a saída do último flip flop se torna a entrada D do primeiro flip flop.

Em um flip flop tipo D:

Se CLOCK = 1 e D = 0, a saída é 0

Se CLOCK = 1 e D = 1, a saída é 1

* Com base nesse raciocínio, concluímos que para os bits se deslocarem para a direita é preciso somente dar pulsos de clock.
* Se a informação presente nos flips flops for 10101010:

No primeiro pulso:

A saída do primeiro flip flop se torna 0

A saída do segundo flip flop se torna 1

A saída do terceiro flip flop se torna 0

A saída do quarto flip flop se torna 1

A saída do quinto flip flop se torna 0

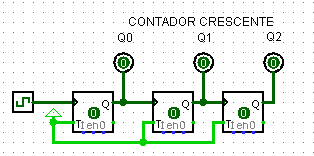
A saída do sexto flip flop se torna 1

A saída do sétimo flip flop se torna 0

A saída do oitavo flip flop e se torna 1 e a entrada do primeiro flip flop se torna 1

Nos pulsos seguintes ocorre o mesmo procedimento, ou seja, os bits são deslocados para a direita, porém ao invés do bit deslocado no final ser removido ele é enviado para o primeiro flip flop, fazendo assim que o deslocamento se torne circular.

CONTADOR CRESCENTE



CONCEITO

O contador crescente possui duas entradas (a CLOCK, e uma fonte, ou seja, 1.), saídas e flip flops tipo T correspondentes a quantidades de bits que serão contados. A entrada CLOCK é ligada no primeiro flip flop, a saída Q do primeiro flip flop se torna a entrada CLOCK do segundo flip flop...

O flip flop de um contador crescente tem uma peculiaridade, normalmente o flip flop funciona se o clock possui valor 1, ou seja, normalmente o flip flop é sensível a mudança de estado de 0 para 1, ou seja, é sensível a curva ascendente. Porém, os flips flops de um contador crescente foram configurados para serem sensíveis a mudança de estado de 1 para 0, ou seja, são sensíveis a curva descendente. Portanto, quando o valor do clock passar de 1 para 0, o flip flop funcionará normalmente, caso contrário, a corrente não irá passar pelo circuito do flip flop e a saída antiga será mantida.

Um flip flop tipo T funciona basicamente da seguinte maneira:

Se T = 0, então Qf = Qa

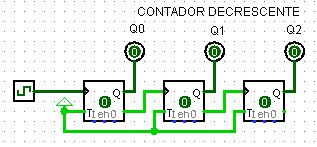
Se T = 1, então Qf = Qa’

Como o T neste circuito sempre vai ser 1, a saída do flip flop se ocorrer a mudança de estado 1 -> 0, sempre vai ser o Qa’.

**CIRCUITO**

* Passo a passo do circuito de um contador crescente de 3 bits (conta de 0 a 7)
* CLK1 = 0, Q0/CLK2 = 0, Q1/CLK3 = 0, Q2 = 0
* CLK1 = 1, Q0/CLK2 = 0, Q1/CLK3 = 0, Q2 = 0
* CLK1 = 0, Q0/CLK2 = 1, Q1/CLK3 = 0, Q2 = 0\*
* CLK1 = 1, Q0/CLK2 = 1, Q1/CLK3 = 0, Q2 = 0
* CLK1 = 0, Q0/CLK2 = 0, Q1/CLK3 = 1, Q2 = 0\*
* CLK1 = 1, Q0/CLK2 = 0, Q1/CLK3 = 1, Q2 = 0
* CLK1 = 0, Q0/CLK2 = 1, Q1/CLK3 = 1, Q2 = 0 \*
* CLK1 = 1, Q0/CLK2 = 1, Q1/CLK3 = 1, Q2 = 0
* CLK1 = 0, Q0/CLK2 = 0, Q1/CLK3 = 0, Q2 = 1\*
* CLK1 = 1, Q0/CLK2 = 0, Q1/CLK3 = 0, Q2 = 1
* CLK1 = 0, Q0/CLK2 = 1, Q1/CLK3 = 0, Q2 = 1\*
* CLK1 = 1, Q0/CLK2 = 1, Q1/₢LK3 = 0, Q2 = 1
* CLK1 = 0, Q0/CLK2 = 0, Q1/CLK3 = 1, Q2 = 1\*
* CLK1 = 1, Q0/CLK2 = 0, Q1/CLK3 = 1, Q2 = 1
* CLK1 = 0, Q0/CLK2 = 1, Q1/CLK3 = 1, Q2 = 1\*

**CONTADOR DECRESCENTE**

****

**CONCEITO**

O contador decrescente possui duas entradas (a CLOCK, e uma fonte, ou seja, 1.), saídas e flip flops tipo T correspondentes a quantidades de bits que serão contados. A entrada CLOCK é ligada no primeiro flip flop, a saída Q’ do primeiro flip flop se torna a entrada CLOCK do segundo flip flop...

O flip flop de um contador decrescente tem uma peculiaridade, normalmente o flip flop se o clock possui valor 1, ou seja, normalmente o flip flop é sensível a mudança de estado de 0 para 1, ou seja, é sensível a curva ascendente. Porém, os flips flops de um contador decrescente foram configurados para serem sensíveis a mudança de estado de 1 para 0, ou seja, são sensíveis a curva descendente. Portanto, quando o valor do clock passar de 1 para 0, o flip flop funcionará normalmente, caso contrário, a corrente não irá passar pelo circuito do flip flop e a saída antiga será mantida.

Um flip flop tipo T funciona basicamente da seguinte maneira:

Se T = 0, então Qf = Qa

Se T = 1, então Qf = Qa’

Como o T neste circuito sempre vai ser 1, a saída do flip flop se ocorrer a mudança de estado 1 -> 0, sempre vai ser o Qa’.

**CIRCUITO**

* Passo a passo do circuito de um contador decrescente de 3 bits (conta de 7 a 0)
* CLK1 = 0, Q0 = 0, Q0’/CLK2 = 1, Q1 = 0, Q1’/CLK3 = 1, Q2 = 0
* CLK1 = 1, Q0 = 0, Q0’/CLK2 = 1, Q1 = 0, Q1’/CLK3 = 1, Q2 = 0
* CLK1 = 0, Q0 = 1, Q0’/CLK2 = 0, Q1 = 1, Q1’/CLK3 = 0, Q2 = 1 \*
* CLK1 = 1, Q0 = 1, Q0’/CLK2 = 0, Q1 = 1, Q1’/CLK3 = 0, Q2 = 1
* CLK1 = 0, Q0 = 0, Q0’/CLK2 = 1, Q1 = 1, Q1’/CLK3 = 0, Q2 = 1\*
* CLK1 = 1, Q0 = 0, Q0’/CLK2 = 1, Q1 = 1, Q1’/CLK3 = 0, Q2 = 1
* CLK1 = 0, Q0 = 1, Q0’/CLK2 = 0, Q1 = 0, Q1’/CLK3 = 1, Q2 = 1 \*
* CLK1 = 1, Q0 = 1, Q0’/CLK2 = 0, Q1 = 0, Q1’/CLK3 = 1, Q2 = 1
* CLK1 = 0, Q0 = 0, Q0’/CLK2 = 1, Q1 = 0, Q1’/CLK3 = 1, Q2 = 1\*
* CLK1 = 1, Q0 = 0, Q0’/CLK2 = 1, Q1 = 0, Q1’/CLK3 = 1, Q2 = 1
* CLK1 = 0, Q0 = 1, Q0’/CLK2 = 0, Q1 = 1, Q1’/CLK3 = 0, Q2 = 0\*
* CLK1 = 1, Q0 = 1, Q0’/CLK2 = 0, Q1 = 1, Q1’/₢LK3 = 0, Q2 = 0
* CLK1 = 0, Q0 = 0, Q0’/CLK2 = 1, Q1 = 1, Q1’/CLK3 = 0, Q2 = 0\*
* CLK1 = 1, Q0 = 0, Q0’/CLK2 = 1, Q1 = 1, Q1’/CLK3 = 0, Q2 = 0
* CLK1 = 0, Q0 = 1, Q0’/CLK2 = 0, Q1 = 0, Q1’/CLK3 = 1, Q2 = 0\*
* CLK1 = 1, Q0 = 1, Q0’/CLK2 = 0, Q1 = 0, Q1’/CLK3 = 1, Q2 = 0
* CLK1 = 0, Q0 = 0, Q0’/CLK2 = 1, Q1 = 0, Q1’/CLK3 = 1, Q2 = 0 \*