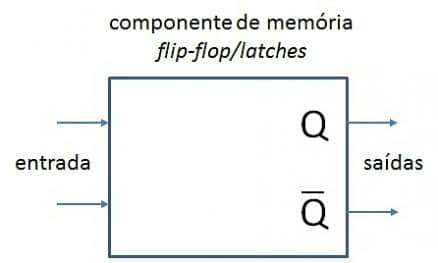
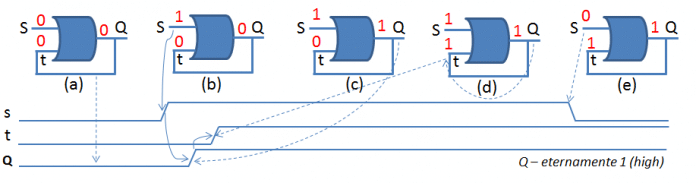
**Latch**

Latch é um elemento primitivo de memória amplamente utilizado em microeletrônica. É um circuito constituído por portas lógicas, capaz de armazenar um bit de informação, onde as saídas em um momento dependem dos valores de entrada do instante mais os valores anteriores de saída, isto é, do seu estado atual, e onde as saídas mudam a qualquer instante de tempo, podendo ter ou não variáveis de controle. Ou seja, um latch armazena uma informação. Um circuito sequencial tem as suas saídas (output) dependentes tanto do estado atual ou do estado corrente da entrada (input).



Uma vez que este estado é alterado torna-se necessário de alguma forma armazenar essa informação até que futuramente novas combinações nas entradas voltem a ocorrer e mudem este estado. Portanto, para esse armazenamento são utilizadas células de memória, no caso latches ou flip-flops.

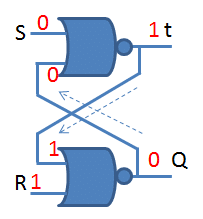
O método básico para armazenamento de um bit em um circuito digital é a realimentação deste circuito. O exemplo abaixo ilustra todos os passos dessa realimentação. Em (a) tanto as entradas quanto a saída da porta lógica OR estão em zero. No item (b), a entrada S é alimentada, e a saída é alterada para nível lógico 1 em (c). Em (d) a outra entrada do circuito é realimentada pela saída. Por fim no item (e) a entrada S é novamente colocada em nível lógico zero e a saída continua em 1.



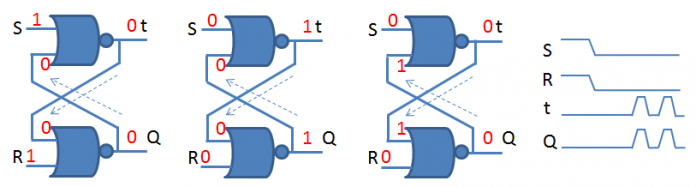
O maior problema é que quando a entrada S muda de estado (e) para 1 a saída Q permanecerá sempre em 1. Isto ocorre devido à realimentação (ao menos que o circuito seja desligado).

**LATCH SR BÁSICO**

Uma solução para resolver o problema descrito anteriormente foi o desenvolvimento do latch. O latch SR funciona da seguinte forma: a porta S (SET) obriga o latch a ser 1, enquanto a porta R (RESET) obriga o latch a ser 0.



Quando ambas as entradas S e R ficam em 1 ao mesmo tempo pode ocorrer um problema de indefinição do estado, ou seja, a saída Q do latch SR vai começar a oscilar quando S e R voltarem a ser zero. Isso para um circuito digital não é aceitável.



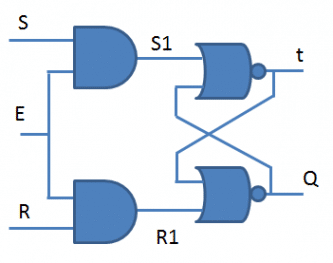
O que realmente ocorre em um circuito real são atrasos nas linhas que interligam os sistemas, portanto em algum momento, depois de um período de oscilação, uma das portas poderá estar atrasada ou adiantada o suficiente para tornar o circuito estável. Essa indefinição é chamada de condição de corrida (race condition), e torna o latch SR básico muito instável para projetos digitais.

**Latch NOR SR**

O funcionamento do latch NOR SR é idêntico ao do latch NAND SR, ao final de seu processamento, este, resulta na seguinte tabela verdade logo abaixo, quando o Reset e o set são iguais a 1, tem-se uma combinação limitada, este estado é proibido, pois esta consegue quebrar a equação lógica Q = NOT Q.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabela Verdade** | | | | | | | |
| **S** | **R** | **Qpróximo** | **Ação** | **Q** | **Qpróximo** | **S** | **R** |
| 0 | 0 | Q | Continua no mesmo estado | 0 | 0 | 0 | x |
| 0 | 1 | 0 | RESET | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | SET | 1 | 0 | 0 | 1 |
| 1 | 1 | X | Não permitido | 1 | 1 | X | 0 |

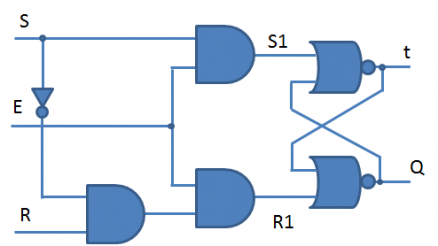
**LATCH SR Sensível a Nível**

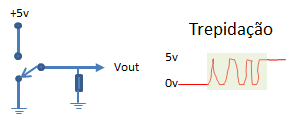
Um latch SR Sensível a Nível é um latch cujas portas de entradas S e R somente são habilitadas quando a terceira porta E (Enable) é habilitada. Em termos gerais, somente colocamos a entrada E em 1 quando realmente tivermos seguros que as entradas S e R estão estáveis.

**LATCH SR tipo D**

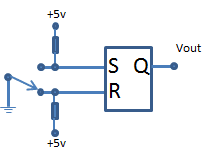
Um latch com uma configuração mais avançada é o latch SR do tipo D. Este latch evita definitivamente o problema de trepidação. O latch SR tipo D difere do latch SR Sensível a Nível por conta da inclusão de uma porta inversora entre as entradas S e R. Dessa forma, devido ao atraso inerente à porta NOT, estamos garantindo que as entradas S e R não serão 1 ao mesmo tempo, desde de que a porta de enable esteja desabilitada.

O latch tipo D é, portanto, o ponto de partida para estruturas mais complexas para armazenamento de um bit, ou seja, com a combinação de dois latches do tipo D podemos projetar um flip-flop do tipo D.

Uma aplicação interessante para um latch é a supressão de ruídos de contato (chaves mecânicas), chamado de debouncing ou contact bounce.

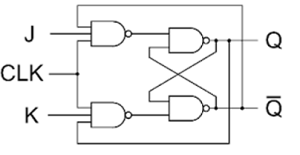


Uma forma de evitarmos o problema de trepidação é com a inserção de um latch na saída da chave.



**Flip‐flop J‐K com clock**

Mais versátil e mais usado que o R‐S, pois não tem estados ambíguos. Bastante utilizado em contadores binários.



**Flip-Flops**

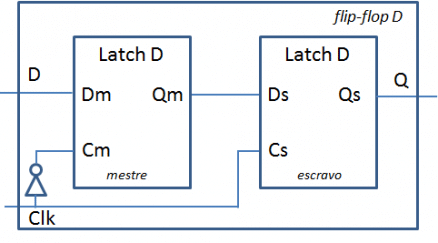
O flip-flop serve como memória de um bit, onde as entradas podem ser um ou dois sinais de entrada, um sinal clock, e um sinal de saída. Alguns flip-flops têm um clear que é responsável por limpar a saída atual. Basicamente um Flip-Flop serve para guardar estados 0 ou 1.

Os flip-flops são implementados na forma de circuito integrado, ou seja, as mudanças que ocorrem em alguns “componentes” do circuito, interferem nos resultados de entrada ou saída. Ou seja, a pulsação ou mudança no sinal do clock faz com que aconteça uma ação no flip-flop, baseado nos valores dos sinais de entrada e em sua equação característica.

Nos latches, o estado se altera durante o clock ativo. Nos flip-flops, o estado é alterado apenas durante uma das bordas do clock.

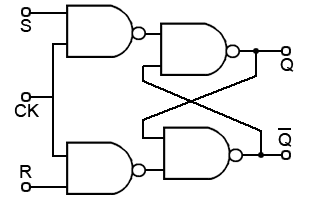
**Flip-flop tipo D**

Um flip-flop do tipo D é um bloco de armazenamento de 1 bit, sensível à borda. Quando a entrada Clk variar, seja na subida ou na descida do sinal, o sinal da entrada é transmitido para a saída. Um flip-flop do tipo D é composto de dois latches do tipo D ligados em modo mestre/escravo.



**Flip-flop SR**

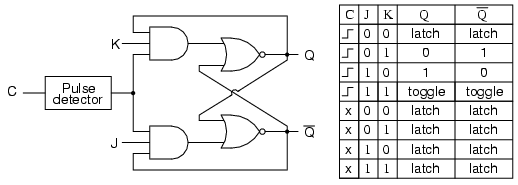
O flip-flop SR (ou RS) possui duas entradas, S (set) e R (reset), além do clock. Sua saída é 1 quando S = 1 e R = 0 (ação set), e 0 quando S = 0 e R = 1 (ação reset). Quando ambas entradas forem 0, a saída é a saída anterior, isto é, o estado atual. Quando ambos S e R são 1, o flip-flop tem comportamento inesperado, chamado de estado proibido.



**Flip-flop JK**

Flip-flop JK é um flip-flop que pode memorizar um único bit de informação e onde o próximo estado de saída é caracterizado como uma função das duas entradas presentes e do estado presente. São largamente utilizados em contadores e nada mais são que flip-flops S-R com realimentação. Quando se aplica uma borda de subida na entrada J, sua saída Q vai a nível lógico 1 (se já não estiver nele) e, ao se aplicar uma borda de descida na entrada K, sua saída Q vai a nível 0. É um aprimoramento do flip-flop S-R, pois, enquanto as combinações "J = 1, K = 0" e "J = 0, K = 1" são respectivamente set e reset, "J = K = 1" não leva a um estado proibido, mas inverte o flip-flop, tornando-se um flip-flop T.

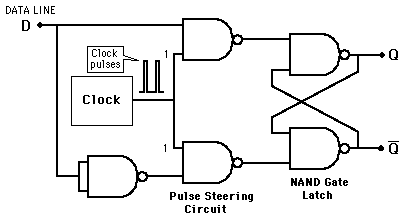
|  |  |  |  |
| --- | --- | --- | --- |
| **J** | **K** | **Qpróximo** | **Ação** |
| 0 | 0 | Q | Mantém o estado |
| 0 | 1 | 0 | RESET |
| 1 | 0 | 1 | SET |
| 1 | 1 | NOT(Q) | Inverte o estado |

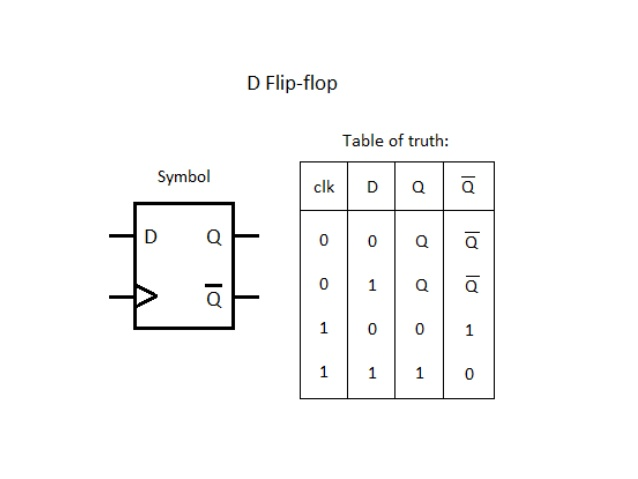


**Flip-flop D**

Flip-flop tipo D (data) é um circuito síncrono de memória com uma entrada (D), um sinal de clock e saídas Q e QQ, sendo esta última responsável por realimentar o circuito com o valor memorizado. Possui uma entrada que se liga diretamente à saída quando há alteração no clock. Quando esta alteração ocorre, o flip-flop D assume o valor 1 se D = 1 ou 0 se D = 0, independentemente do valor atual. Pode-se interpretar este flip-flop como uma primitiva linha de atraso ou hold de ordem zero, pois a informação é ligada na saída um ciclo após seu recebimento na entrada. É o mais econômico e eficiente flip-flop em número de transistores e área de silício.

|  |  |  |
| --- | --- | --- |
| D | Q | **Qpróximo** |
| 0 | 0 | **0** |
| 0 | 1 | **0** |
| 1 | 0 | **1** |
| 1 | 1 | **1** |





**Contadores**

Um contador é um circuito digital que evolui sob o comando de um clock, de forma que seus estados reproduzam uma sequência pré-determinada. Contadores digitais são utilizados principalmente para contagens, geração de palavras, divisão de frequências, medição de frequência e tempo. São basicamente divididos em duas categorias: contadores assíncronos e síncronos.

**Contadores assíncronos**

Nesse tipo de contador, o pulso de clock é aplicado apenas no primeiro bloco flip-flop, enquanto para os demais blocos, a saída do bloco anterior é que vai servir como clock. Os principais tipos de contadores assíncronos são:

* Contadores de Pulso
* Contadores de Década
* Contador Sequencial de O a N
* Contadores Assíncronos Decrescentes
* Contadores Assíncronos Up-Down (crescente/decrescente)

**Contadores síncronos**

Nesses contadores o clock entra em todos os flip-flops simultaneamente. Para que haja mudanças de estado, deve-se então estudar o comportamento das entradas J e K dos vários flip-flops, para que se tenha nas saídas as sequências desejadas. Para isso, deve-se escrever a tabela verdade das entradas J e K dos flip-flops para que esses assumam os estados seguintes. O contador síncrono é o mais completo contador, ele tem condições de gerar qualquer tipo de sequência binária, ou seja, é um gerador de palavras e consequentemente de códigos binários.