UNIVERSIDADE FEDERAL DO CEARÁ

ENGENHARIA DA COMPUTAÇÃO CAMPUS SOBRAL

RELATÓRIO SOBRE AS FUNCIONALIDADES DOS CIRCUITOS LÓGICOS E SUAS CARACTERÍSTICAS.

DISCIPLINA: ARQUITETURA E ORGANIZAÇÃO DE COMPUTADORES

FRANCISCA JANNIELLY GARCIA DA COSTA

KLAYVER XIMENES CARMO

GABRIEL ALBUQUERQUE ARAUJO

SOBRAL - CE

2020

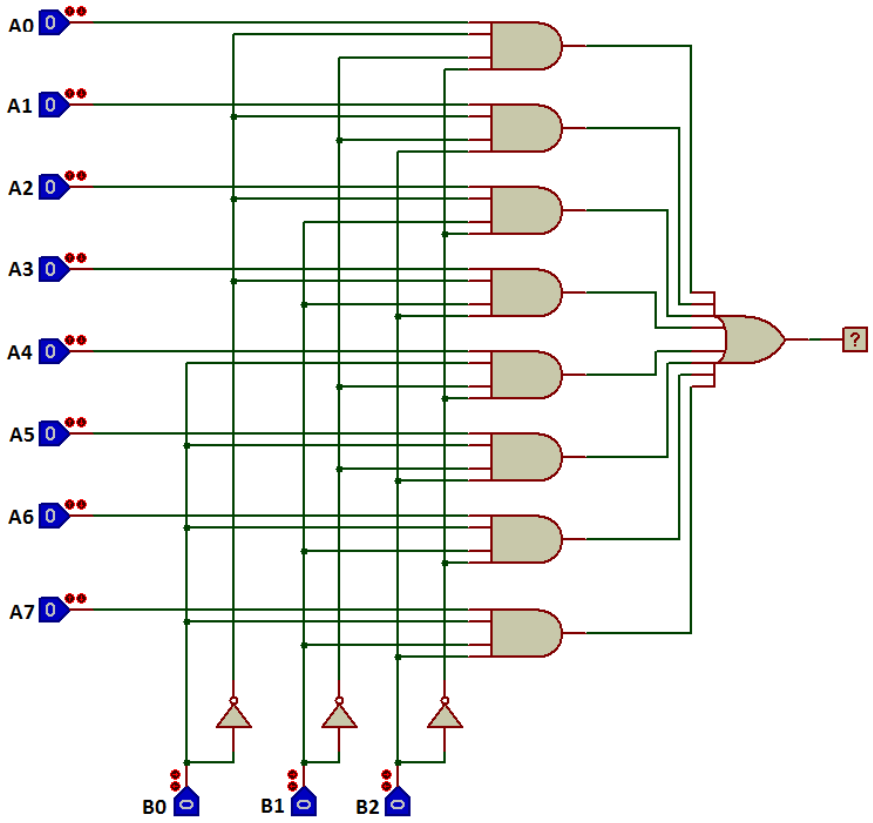
SUMÁRIO

1. MULTIPLEXADOR DE 8 ENTRADAS

Um multiplexador é um circuito que possui 2^n entradas de dados, uma única saída e n chaves de seleção o qual selecionam uma das 2^n entradas de dados. Um multiplexador é composto por portas AND, NOT e OR.

Seu funcionamento é bem simples, cada combinação nas chaves de seleção seleciona uma das entradas de dados, onde terá seu valor exibido na saída.

Figura: 1.1

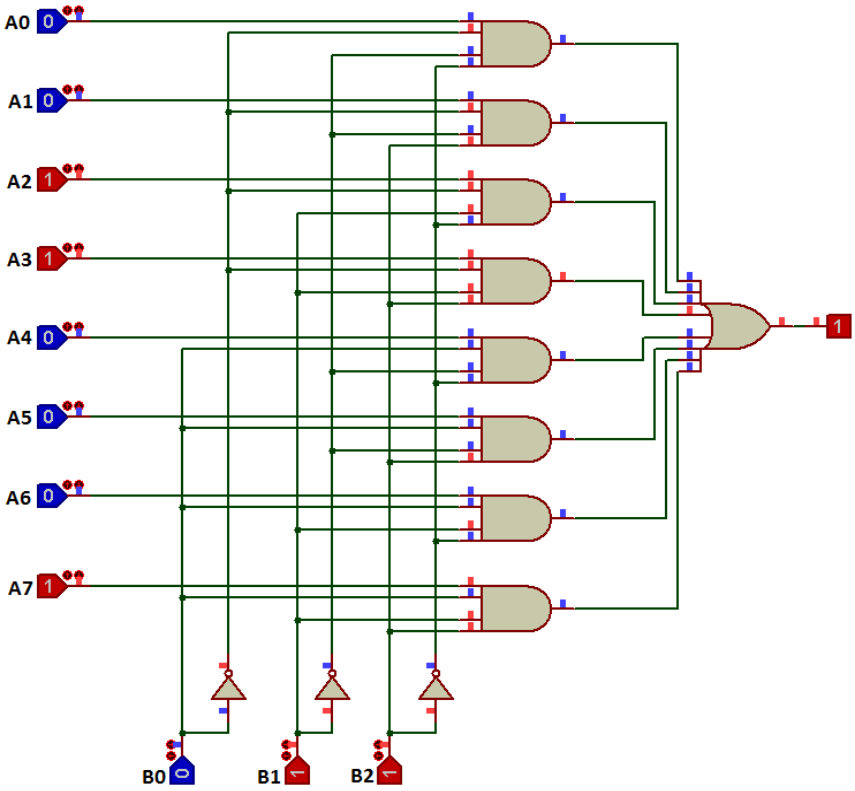


Fonte: Autor

A figura 1.1 é um exemplo de um multiplexador de 8 entradas (A0 - A7) e 3 chaves de seleção (B0 – B2). A seleção de cada uma das entradas é determinada pela combinação das chaves em binário. O valor 000 em binário nas chaves de seleção irá selecionar primeira entrada de dados A0, já o valor 001 corresponde a segunda entrada A1, logo seguindo com essa lógica a última entrada de dados a ser selecionada será dada pelo valor nas chaves de 111.

Exemplo de seleção de portas:

Figura: 2.2



Fonte: Autor

Na figura 2.2, a entrada de dados A2 está selecionada, O valor das chaves de seleção é 011 em binário que corresponde a 3 em decimal.

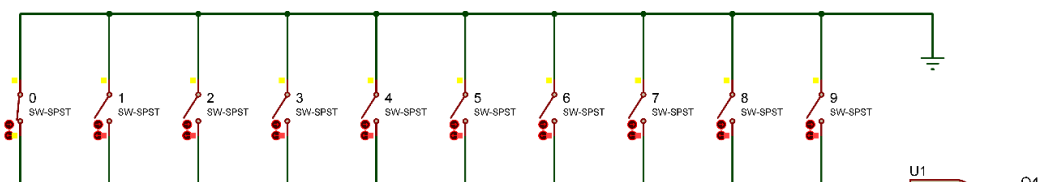
2. CODIFICADOR DECIMAL EM BINÁRIO

O circuito codificador decimal em binário, faz a conversão de um número de base decimal para a base binária.

Seu funcionamento é simples, onde cada interruptor (switch) corresponde a um valor decimal, de 0 a 9. O conjunto de saídas irá assumir o valor binário equivalente a entrada, com 4 dígitos, que é o necessário para a representação dos 10 primeiros números decimais. A lógica do sistema é centrada nas portas lógicas NAND, que funcionam como uma porta lógica AND, mas com uma inversão do sinal na saída.

Os critérios da porta NAND são os seguintes: a saída será 0 se todos os valores de entrada forem 1; se possuir uma entrada sendo 0, ou todas, a saída será 1.

Figura: 2.1

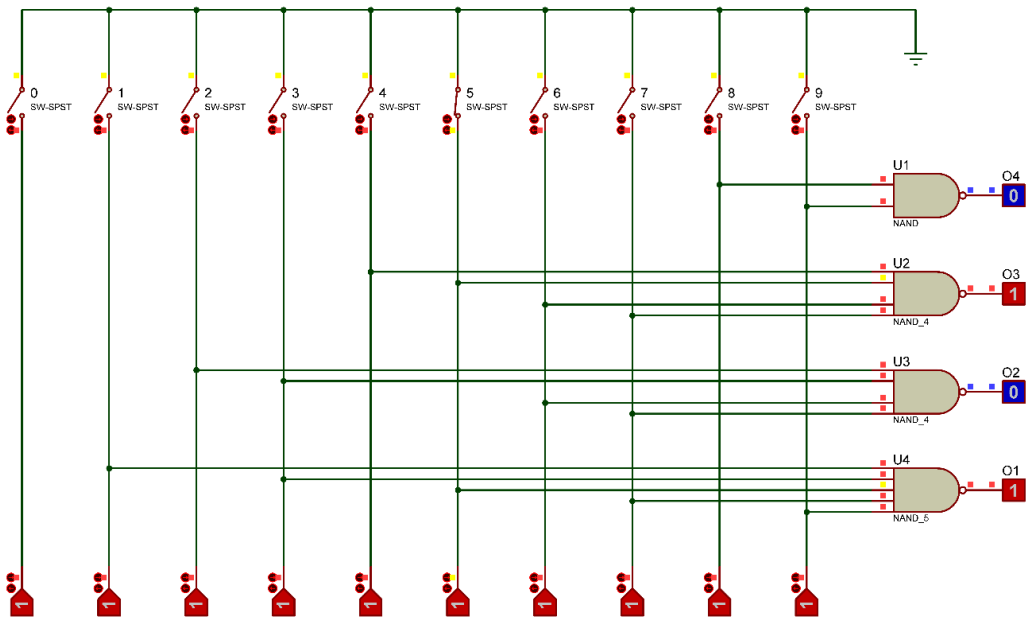


Fonte: Autor

Como mostrado na imagem 2.1, o circuito é ligado em nível lógico alto (1), com um interruptor (switch) em cada número decimal ligado com uma extremidade no terra (0).

No número 0 em decimal, podemos representar em base binária de acordo com a imagem 2.1, tendo como entrada nas portas lógicas o nível lógico alto (1), assim, saindo 0 nos quatro dígitos de saída, representando o número 0 em binário.

Figura 2.2 Fonte: Autor

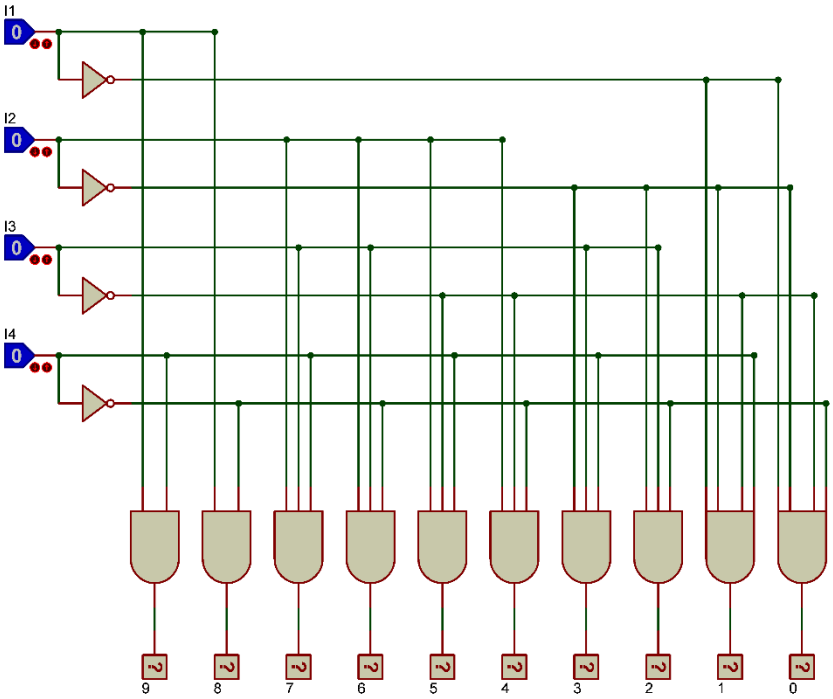


Na imagem 2.2 é representado a conversão do número 5 para binário, onde seu switch correspondente é fechado, tendo uma entrada diferente de 1 nas portas 1 e 3 (de baixo para cima), com isso, a saída será 0101, representando o número 5 em binário.

3. DECODIFICADOR BINÁRIO

O decodificador tem o funcionamento inverso do codificador, converte o número de base binária para decimal.

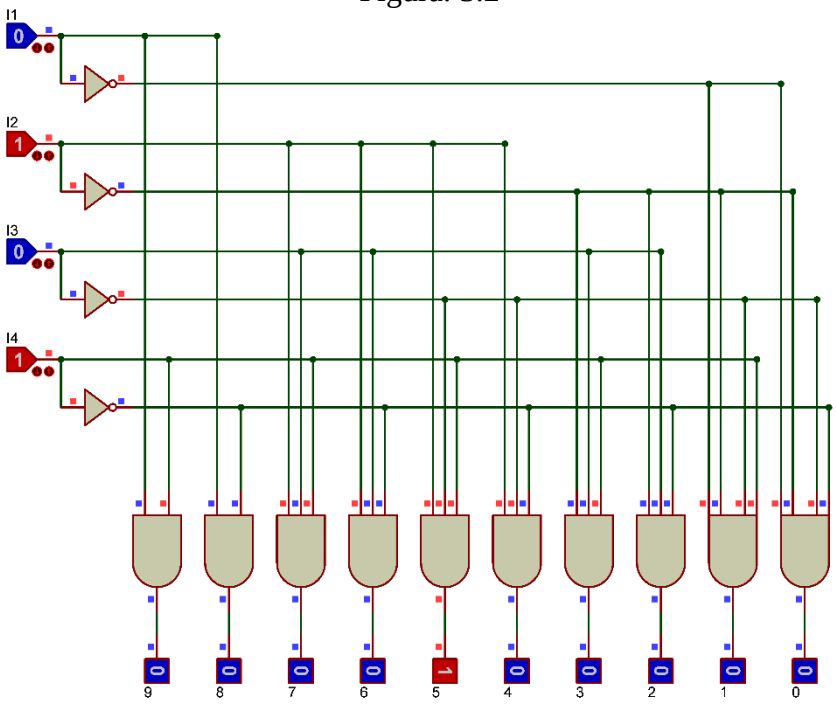
Figura: 3.1



Fonte: Autor

Sua ação é bem simples, como mostrado na figura 3.1, sua saída depende completamente dos sinais de entrada.

Figura: 3.2

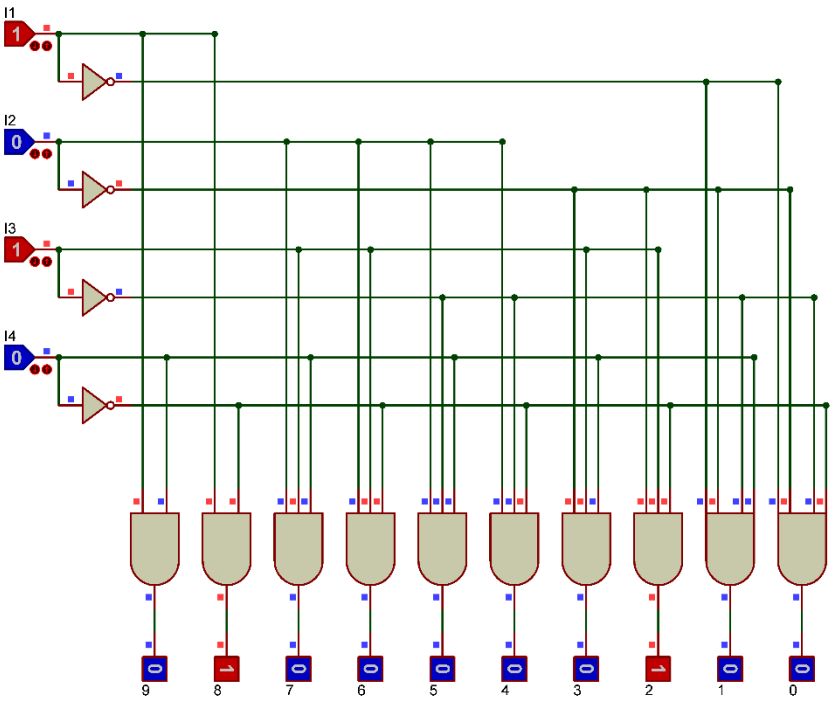


Fonte: Autor

Na figura 3.2 é possível observar a conversão do número “0101” em binário para decimal, onde está com nível lógico alto a saída de número 5.

Como visto, o circuito é centrado na porta lógica AND, que tem saída 1, caso todas as entradas também forem de nível lógico alto (1).

Figura: 3.3 Fonte: Autor

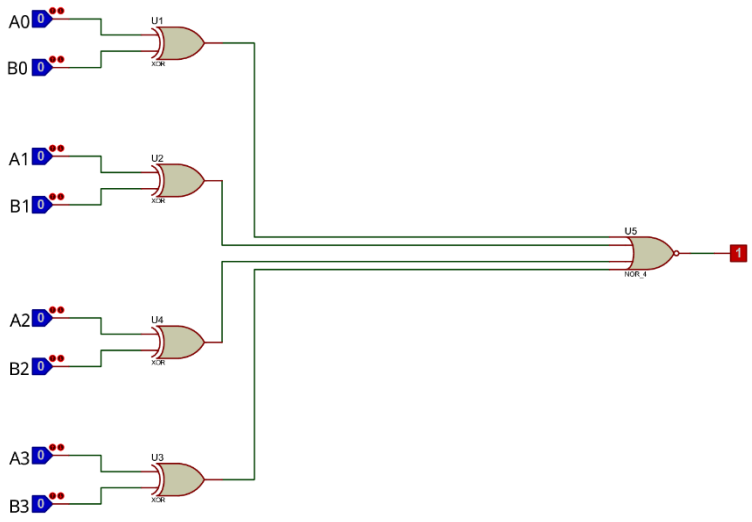


Outro exemplo é com a primeira e a terceira entrada em nível lógico alto (1010), como mostrado na figura 3.3, tendo como saída os números 8 e 2, que somados dão o número 12 em decimal.

4. CIRCUITO COMPARADOR

O circuito comparador, tem como função comparar 2 sequências de binário e para isso utiliza portas lógicas XOR e NAND.

Figura: 4.1



Fonte: Autor

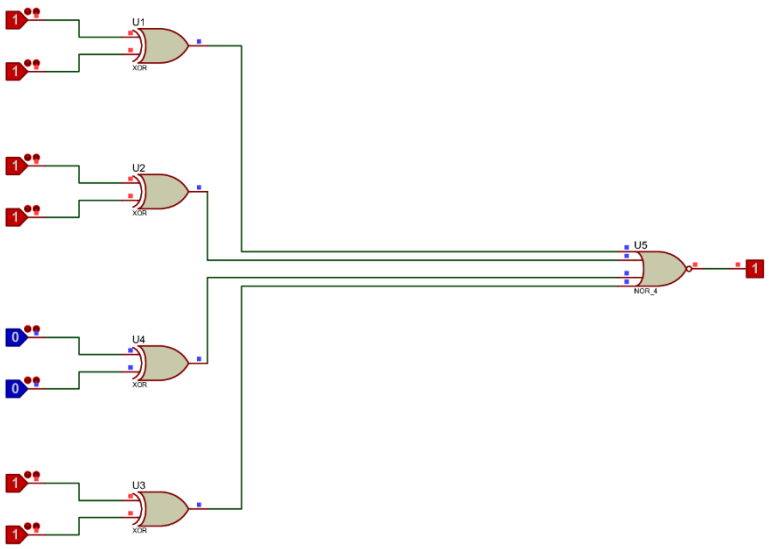
Nas entradas temos A e B, onde cada algarismo deve ser colocado respectivamente. Caso as sequências em binários sejam iguais tem saída valor 1, Caso contrário 0.

Comparando dois valores:

A = 1101

B = 1101

Figura: 4.2



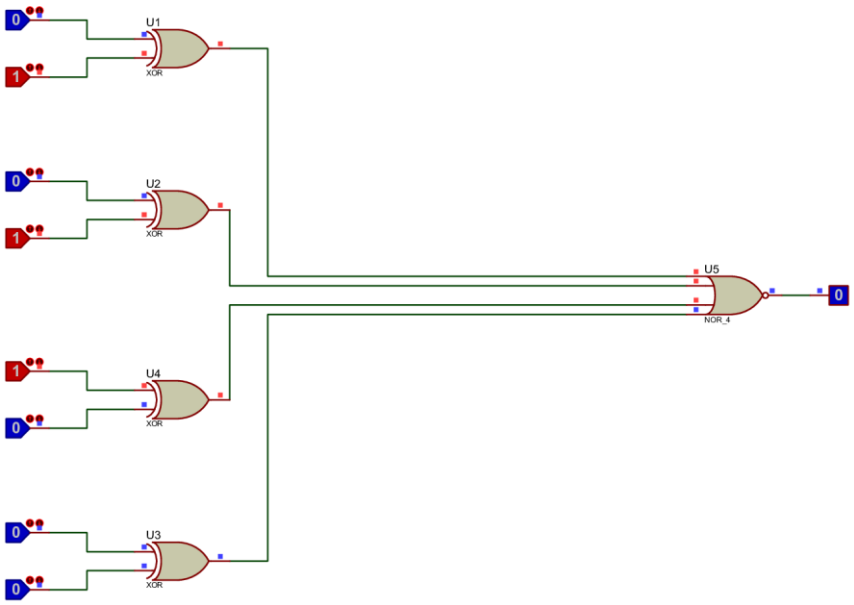
Fonte: Autor

Na figura 4.2 verificamos que o resultado deu 1 já que os bits são iguais.

A = 0010

B = 1100

Figura: 4.3



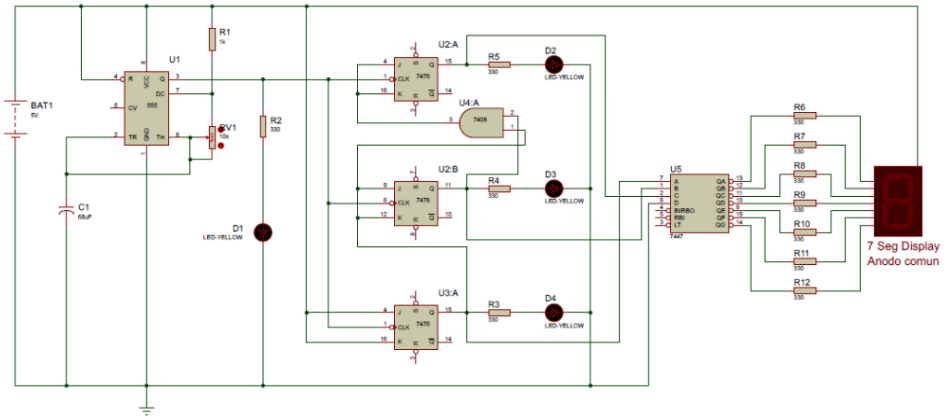
Fonte: Autor

Já na figura 4.3 observamos que o resultado é 0 pois o as sequencias em binário são diferentes.

5. CIRCUITO CONTADOR DE 3 BITS EM BINÁRIO

O circuito contador, é o que o próprio nome diz, um contador que vai de 0 até a quantidade de combinações e bits desejadas. Neste caso, são 3 bits. 2^3 é 8, então irá de 0 a 7.

Figura: 5.1



Fonte: Autor

Para entender seu funcionamento, seguiremos a seguinte lógica: o circuito começa contando em 0 e vai até 7. Que não 8 combinações de 3 bits. (2^3=8). E por esse motivo foram usador apenas 3 flip-flops. Fazendo uma tabela para entender melhor seu funcionamento.

Estado Atual		
E1	E2	E3
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Estado Futuro		
F1	F2	F3
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1
0	0	0

Para saber as conexões que cada entrada dos flip-flops irão receber teremos que fazer uma tabela pensando nas possibilidades de acordo com as duas tabelas

anteriores. Se o estado futuro atual for zero, e o estado futuro quero que seja 0, então a entrada J recebe 0 e não importa se 0 ou 1 entrar na K. E assim por diante, com isso construímos a tabela abaixo:

Atual	Futuro	J	K
0	0	0	Não importa
0	1	1	Não importa
1	0	Não importa	1
1	1	Não importa	0

Agora, olhando e comparando cada coluna das duas primeiras tabelas “Estado Atual” e “Estado Futuro”, e baseando-se na lógica da tabela acima, podemos construir uma tabela com as entradas para cada um dos flip-flops, como a tabela abaixo:

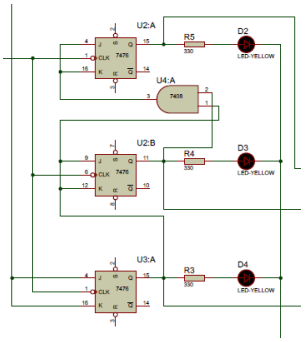
Entradas Flip-flop					
J1	K1	J2	K2	J3	K3
0	Não importa	0	Não importa	1	Não importa
0	Não importa	1	Não importa	Não importa	1
0	Não importa	Não importa	0	1	Não importa
1	Não importa	Não importa	1	Não importa	1
Não importa	0	0	Não importa	1	Não importa
Não importa	0	1	Não importa	Não importa	1
Não importa	0	Não importa	0	1	Não importa
Não importa	1	Não importa	1	Não importa	1

Usando o mapa de Karnaugh para encontrar a função de cada entrada, usando apenas o estado atual como base, teremos:

- J1 = 1 / k1 = 1;
- J2 = E3 / K2 = E3;
- J3 = E2.E3 / K3 = E2.E3;

Essas funções resultarão nessas conexões:

Figura: 5.2

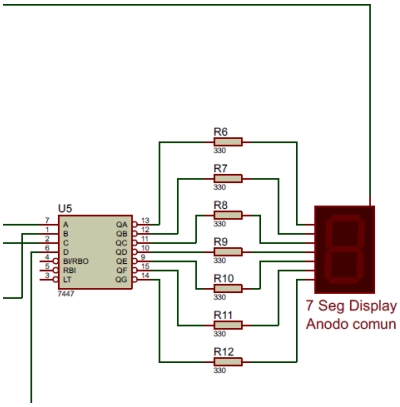


Fonte: Autor

Na figura 5.3, mostra a conexão do display de 7 seguimentos com o CI, as conexões são feitas de acordo com o padrão. (7447).

Essa parte é responsável por gerar o positivo e o negativo para cada seguimento, resultando nos números de 0 a 7.

Figura: 5.3



Fonte: Autor

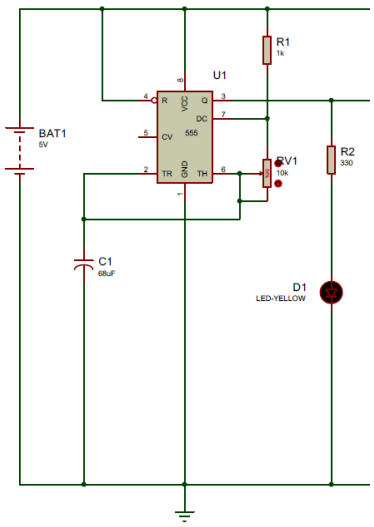
As conexões com os flip-flops são feitas da seguinte forma:

Flip-flop	CI
E1	C
E2	B
E3	A

E o pino “D”, faz conexão com o terra.

A figura abaixo (5.4) faz menção a parte responsável por gerar os pulsos de clock, acoplado com um CI555, que “alimenta” as entradas de clock de cada flip-flop.

Figura: 5.4



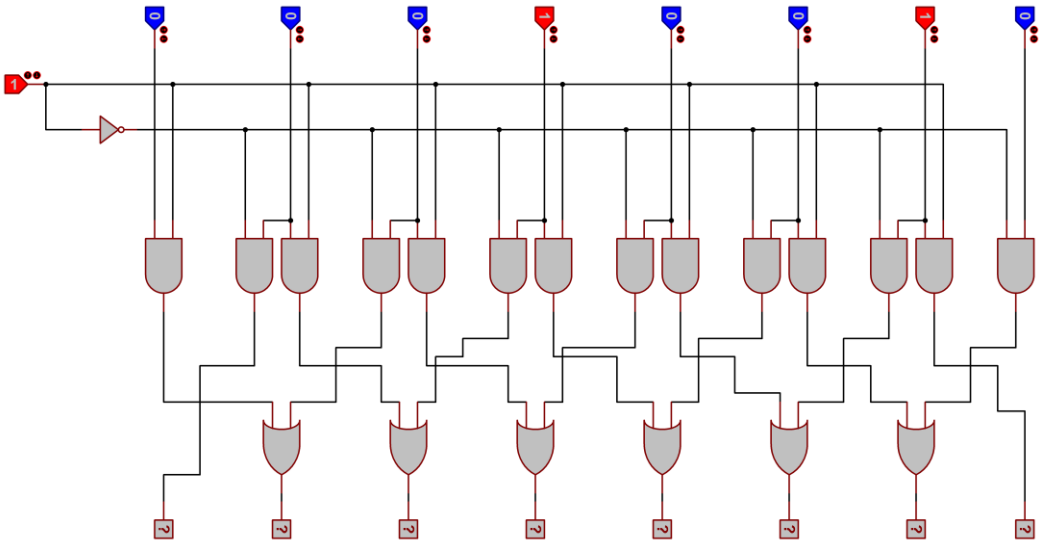
Fonte: Autor

Desta forma, o gerador de pulsos de clock faz com que haja variação nos bits, para fazer as combinações em cada flip-flop gerando (0-7), os números são mandados para o CI 7447, e mostrados no display de 7 seguimentos.

6. CIRCUITO DESLOCADOR DE 8 BITS

Um circuito deslocador possui N entradas e saídas, ambas equivalentes. Os valores que entram são mandados para as linhas de saída e deslocados 1 bit para a direita ou para esquerda. A escolha do sentido de deslocamento. Depende diretamente do estado da linha de controle.

Figura: 6.1 Fonte: Autor



O sistema contém 8 entradas, uma entrada de controle, 14 AND’s, 6 OR e uma NOT. A organização é como está na imagem. Essa entrada de controle que é a principal responsável pelo deslocamento de cada bit, caso seja 0, os bits são deslocados para a esquerda, e caso 1, para a direita.

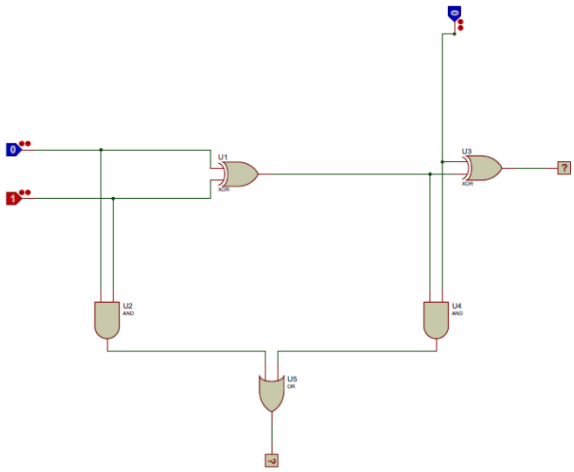
Ex: caso o número seja 00010010 e a entrada de controle seja 0, como já explicado os bits serão deslocados para esquerda. Quem faz o deslocamento são as portas AND’s e OR’s, mas elas dependem diretamente do valor de controle e de seu negativo.

7. SOMADOR COMPLETO DE 1 BIT

Realiza operação de adição com apenas um bit, a diferença entre ele e somador de 1 bit simples, é que no somador de 1 bit completo tem as saídas de “vai um” e “vem um”, para operações do tipo,

Ex: 1 + 1 = 0 1, mas como é de 1 bit, a saída é 0 e “vai um” é 1.

Figura: 7.1



Fonte: Autor

Como visto na figura 7.1 Os dois bits que se quer somar são representados por duas “logicstates”, ambas são entradas de uma porta XOR(u1) , a saída dessa XOR(u1) é uma das entradas de outra porta XOR(u4), esta XOR(u4) tem como segunda entrada um “vem um”, que pode ser um “vai um” de uma soma anterior.

O “vai um” da soma é feito por duas AND e um OR. A AND(u2), tem como entradas os mesmos logicstates da XOR(u1). Já a AND(u4) tem como entrada as mesmas de XOR(u4). Ambas as AND’s são entradas de uma OR(u5), que terá “vai um” como saída.

Ex: 1 + 1.

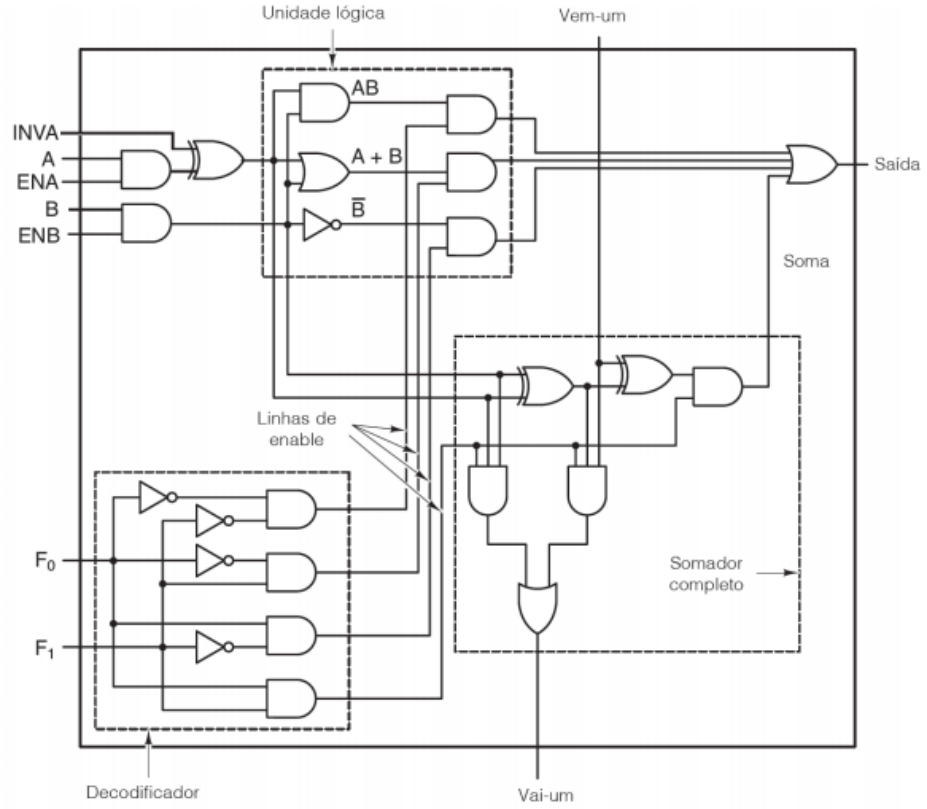
Uma XOR com entradas iguais, terá 0 como saída e consequentemente este zero será uma das entradas da próxima XOR. Vamos supor que o “vem um” seja 0 também. Então logicamente a saída principal deste sistema será 0.

A parte do “vai um” é composta por duas AND’s, a primeira terá 1 e 1 como entradas, resultando 1. A segunda AND terá como entrada a saída da primeira XOR que é 0 e um suposto “vem um” que é 0 também. Uma OR com 0 e 0 resultará em 1. Então 1+1 em binário é 0 e vai 1.

8. ULA DE 1 BIT

A unidade lógica e aritmética (ULA), é um circuito digital que realiza operações de adição ou subtração e booleana. A ULA é uma peça fundamental da unidade central de processamento (UCP), e até dos mais simples microprocessadores. As operações são realizadas de dois registradores fontes do banco de registradores, e com a escrita do resultado no registrador de destino.

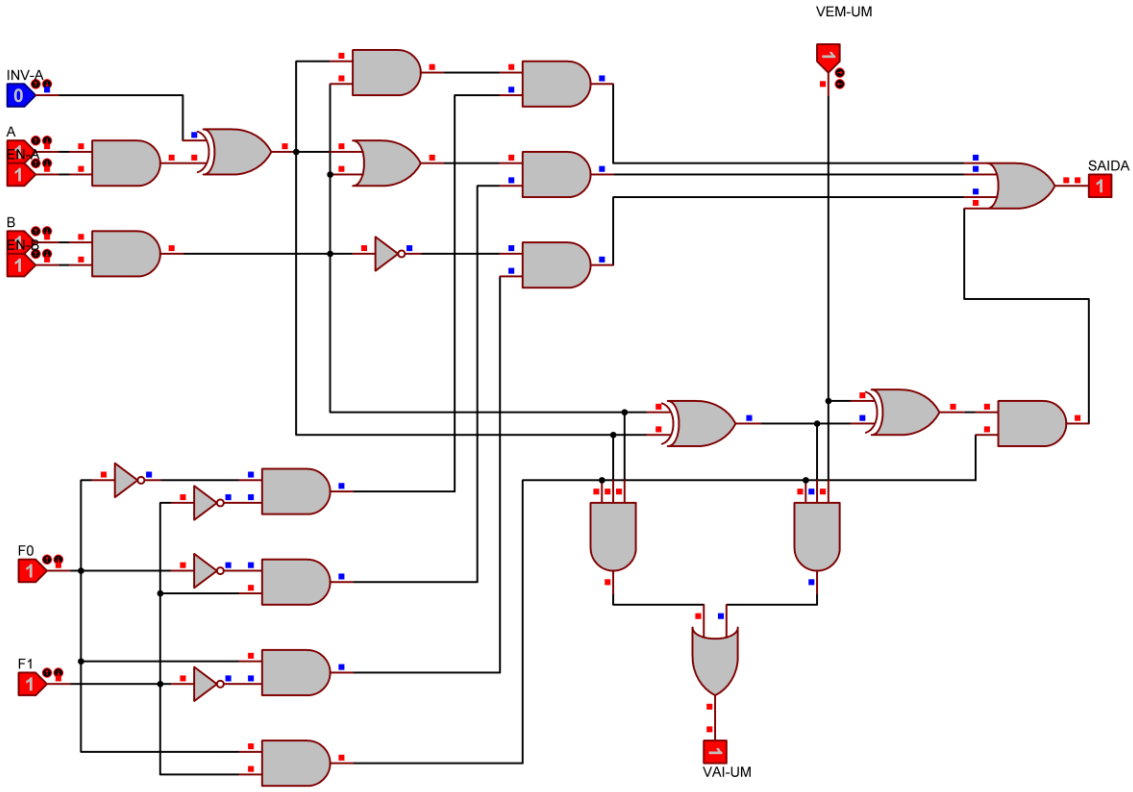
Figura: 8.1 Fonte: Slide aula4 Arquitetura



Como podemos ver na figura 8.1, a ULA tem praticamente 3 grupos de circuitos principais.

O primeiro é o grupo de unidade lógica, na qual será feito as operações com base nos bits de dados de entrada, o segundo grupo é o decodificador, que de acordo com a combinação dos bits de entrada selecionará a operação que será feita, já o terceiro grupo é o somador completo, para caso precise repassar um dos valores de saída para a entrada de outra ULA.

Figura: 8.2

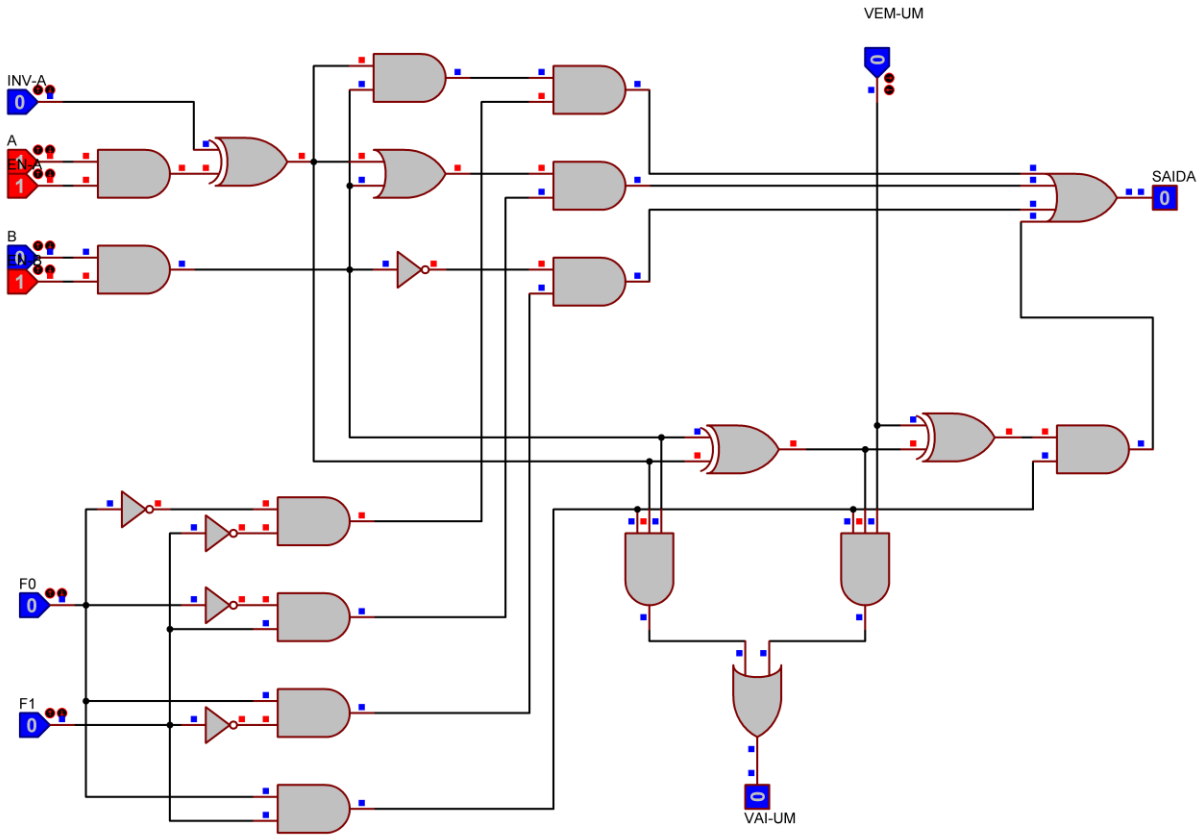


Fonte: Autor

Na figura 8.2 temos uma simulação da soma de três bits binários de corrente alta, na qual a soma tem como resultado 1 e vai um.

A combinação do decodificador (1 e 1), é a correspondente da operação de soma, também percebemos que veio um bit na entrada “vem-um”, e foi acionado mais dois bits na soma, com isso temos a soma de 3 bits

Figura: 8.3



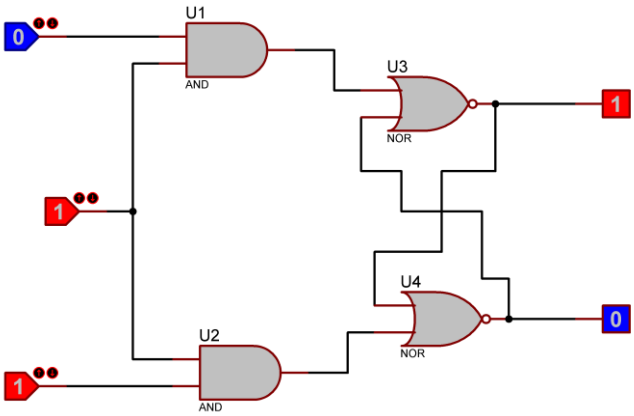
Fonte: Autor

Outro exemplo de operação AND, onde na figura 8.3 temos a operação com os bits binários 1 e 0, e o resultado será 1, como o apresentado.

9. LATCH SR SEM CLOCK

É um circuito muito usado para armazenar o estado (0 ou 1) até que chegue novas combinações para uma nova mudança de estado. Como podemos ver na figura abaixo:

Figura: 9.1



Fonte: Autor

O seu funcionamento dá-se da seguinte forma:

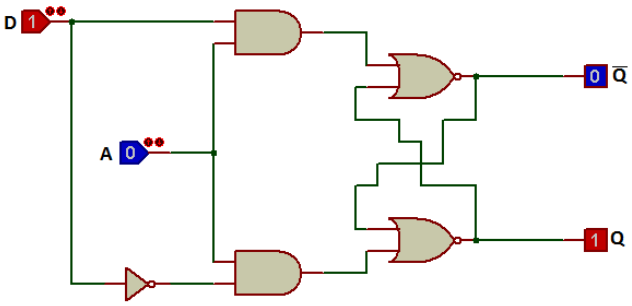
Temos duas portas NOR que são inter alimentadas, e duas entradas SET(S) e Reset(R). Uma “obriga” a latch sem 0 e a outra 1. Esse esquema faz com que o estado seja armazenado até que haja mudança nas entradas.

Ex: O valor de R é 0 e já tinha o 1 armazenado. Então a NOR de baixo terá 1 e 0 como entrada, que resultará em 0. O 0 é uma das entradas da primeira NOR, juntamente com, vamos supor que S = 0, então resultará em 1. 1 é justamente o valor armazenado. Então até que haja mudança em S ou R, o estado continuará sendo 1.

10. LATCH D SEM CLOCK

Latch D sem clock é um circuito capaz de armazenar dados temporariamente. Um Latch D sem clock é composto por portas logicas NOT, AND e NOR.

Figura: 10.1

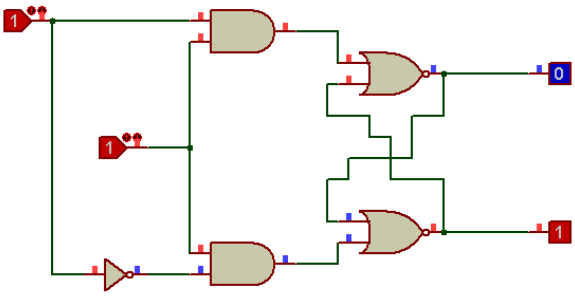


Fonte: Autor

Seu funcionamento é bem simples, caso a chave A esteja com valor 1, o valor de D que irá definir se irá armazenar ou zerar o valor guardado: D=1 armazena e D=0 reseta. Para manter armazenado sem a influência da chave D, desativamos a chave A e o valor armazenado não será modificado independentemente do valor de D.

Exemplo armazenando um valor:

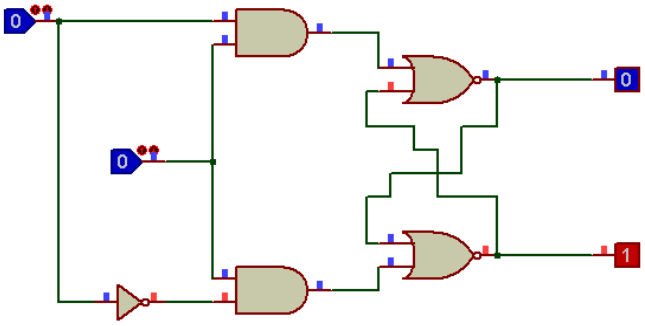
Figura: 10.2



Fonte: Autor

Como vemos na imagem 10.2, o valor 1 está armazenado.

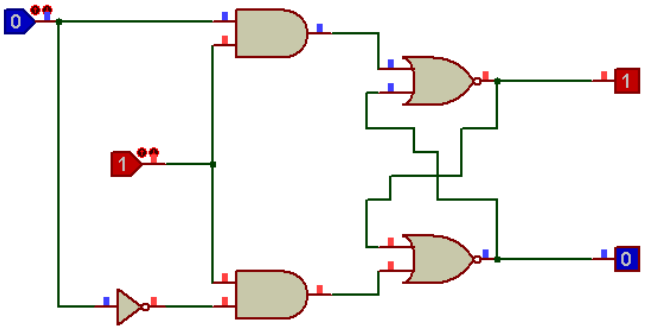
Figura: 10.3



Fonte: Autor

Nota-se que na figura 10.3 o valor 1 continua armazenado independentemente do valor de D está alterada, pois a chave A está com valor 0.

Figura: 10.4



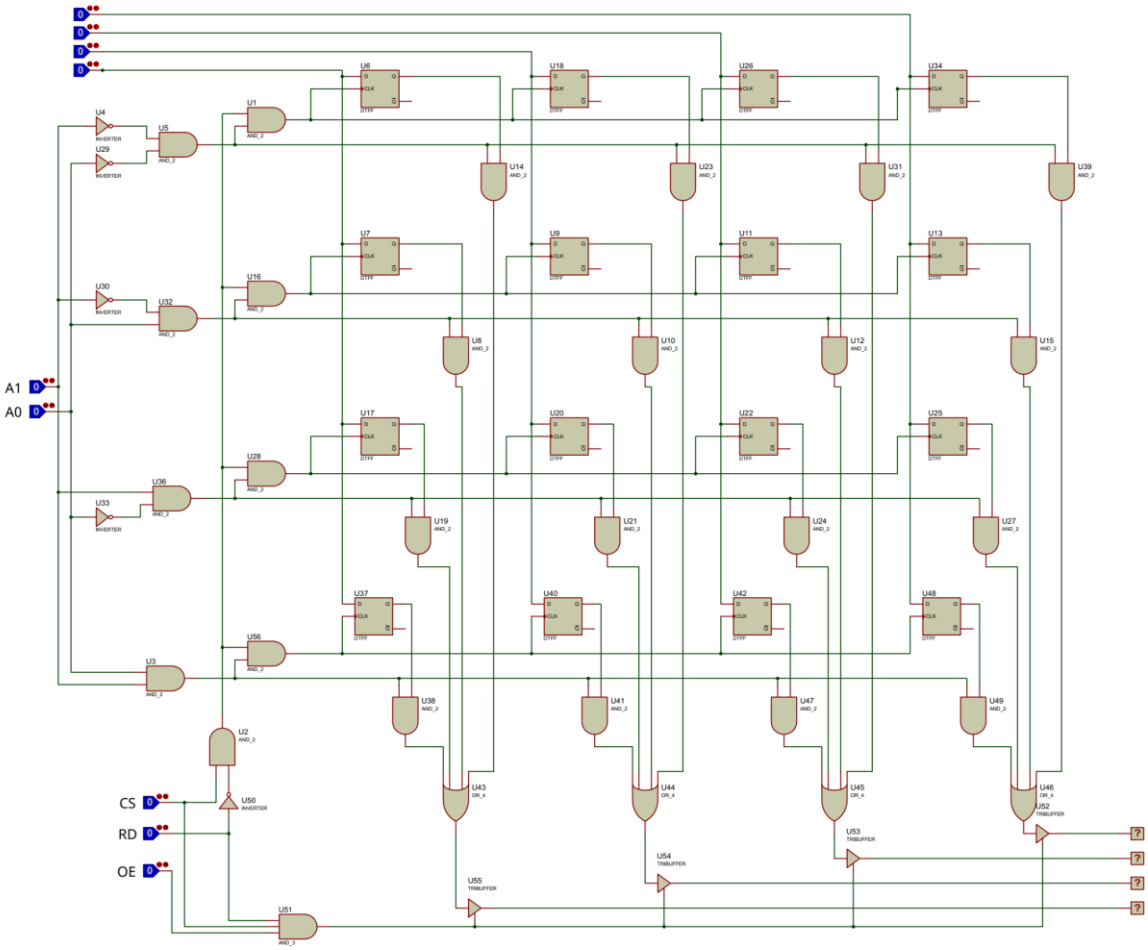
Fonte: Autor

Na figura 10.4, como a chave A está ativada e o valor da porta D é 0, o valor armazenado é resetado.

11. CIRCUITO LÓGIO PARA UMA MEMÓRIA DE 4 X 4

Uma memória 4x4 é um dispositivo capaz de armazenar dados. Sua estrutura 4x4 o torna capaz de armazenar até 4 palavras de 4 bits cada, onde cada palavra fica guardada em uma linha. A memória e composta por entradas de dados, entrada de controle e saída de dados. Possui portas logicas como AND, NOT, OR e Flip-flops D.

Figura: 11.1



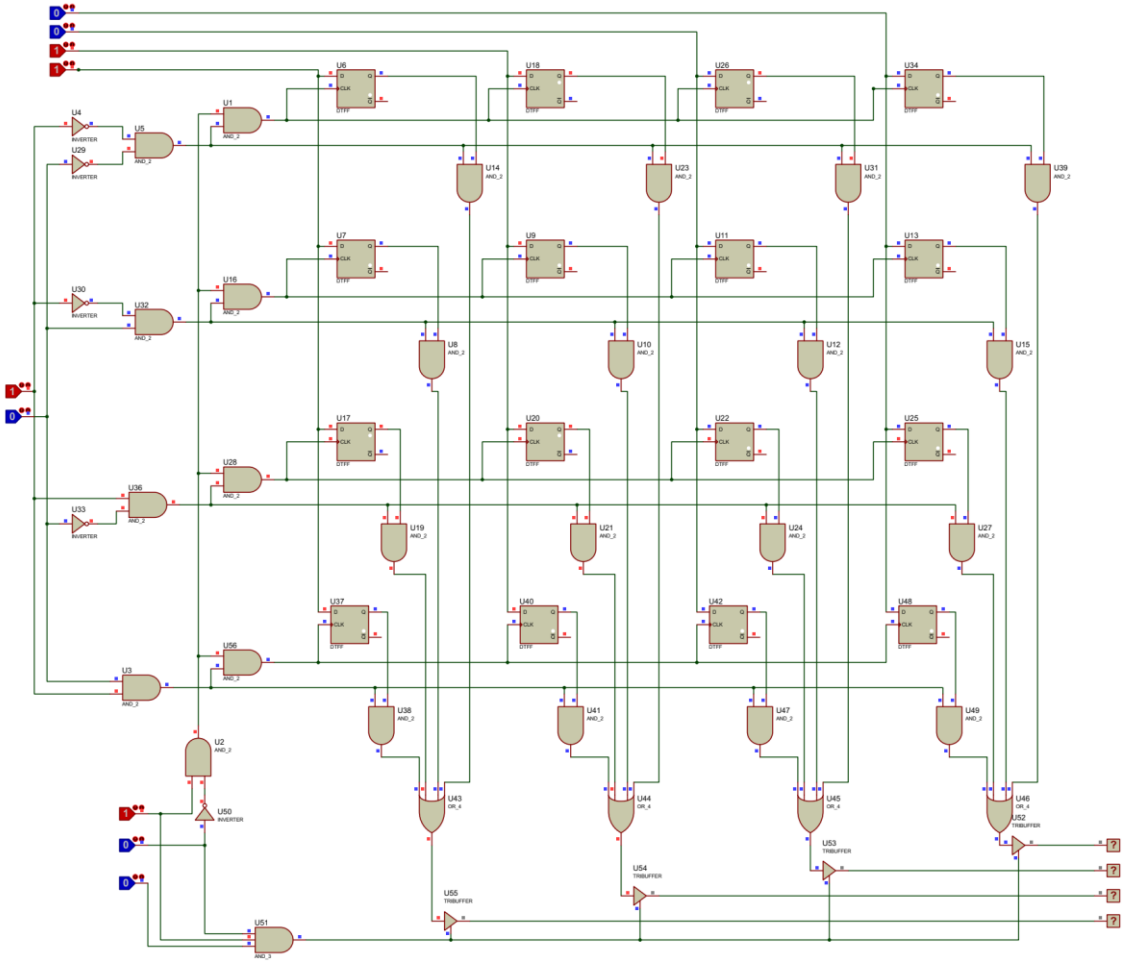
Fonte: Autor

As entradas de controle são responsáveis por: seleção de linha da memória, ativar/desativar memória, seleção de leitura ou escrita e habilitação da saída de dados.

- As chaves A0 E A1 ficam responsáveis pela seleção de linhas, onde cada combinação de chaves selecionará uma linha. Como possui 2 chaves, são 4 combinações possíveis: 00, 01, 10 e 11.
- A chave CS (CHIPSET) fica responsável pela ativação da memória, pois pode-se ter casos onde possui várias memórias, e essa chave determina qual é a ativa.
- Já a chave RD (READ) tem como função determinar se é leitura ou escrita da memória, caso 0 é escrita e 1 leitura.
- A chave OE tem como função habilitar a saída de dados.

Exemplo de escrita de dados:

Figura: 11.2

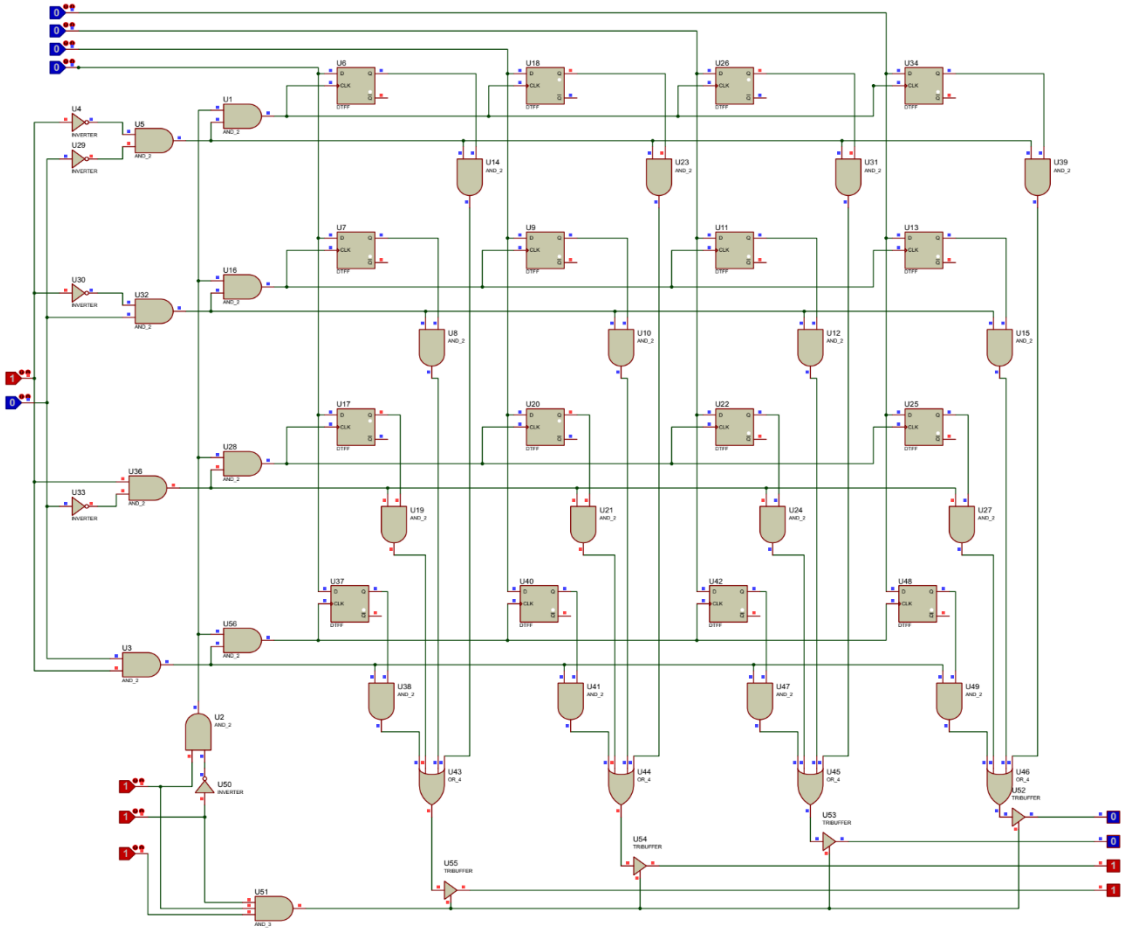


Fonte: Autor

Na figura 11.2 observa-se a palavra 0011 sendo escrita na linha 01, também se nota que na linha 00 possui a palavra 0110 já escrita. Como é uma escrita, a chave RD possui o valor 0 e a chave de saída também possui valor 0.

Exemplo de leitura de dados:

Figura: 11.3



Fonte: Autor

Na imagem 11.3, observa-se a memória possui a palavra 0011 salvo na linha 01. Para obter a leitura da palavra salva, primeiramente é selecionada nas chaves A0 e A1 qual linha deve ser lida e logo após ativa-se as chaves: CS, RD (escrita ou leitura) e OE (habilitação da saída). Nota-se na saída de dados a palavra buscada (0011).

12. REFERÊNCIAS

- Latch e Flip-Flop - Conceitos importantes. Disponível em:
  < https://www.embarcados.com.br/latch/#LATCHSR-BASICO>

- Organização Estruturada de Computador – Aula 05. Disponível em:
  < http://www.dpi.inpe.br/~carlos/Academicos/Cursos/ArqComp/aula_5bn1.html>

- Youtube – Circuito contador síncrono de 3 bits – Flip-flop J-K . Disponível em:
  <https://www.youtube.com/watch?v=FANMaccckNM>