Universidade Federal de Roraima
Departamento de Ciência da Computação
Arquitetura e Organização de Computadores

RELATÓRIO TÉCNICO DE IMPLEMENTAÇÃO DOS COMPONENTES

Boa Vista - RR
2024

JASMIM SABINI - 2023010360
VINÍCIUS CAVALCANTE MARTINS - 2023001156

RELATÓRIO TÉCNICO DE IMPLEMENTAÇÃO DOS COMPONENTES

Relatório técnico apresentado ao Prof. Dr. Herbert Oliveira Rocha, como requisito de obtenção de nota parcial na disciplina DCC 301 - Arquitetura e Organização de Computadores.

Boa Vista - RR
2024

SUMÁRIO

1. Introdução

O seguinte relatório técnico visa descrever a resolução dos componentes criados a partir do programa Logisim-evolution versão 3.9.0, para a matéria de Arquitetura e Organização de Computadores, assim como os detalhes sobre o funcionamento de cada componente, pois estão em diferentes tipos de instrução e tamanho em bits, totalizando 17 componentes.

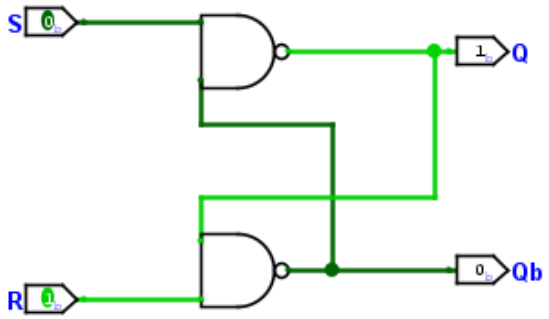
2. Componentes

2.1. Flip - Flop do Tipo D e JK

Flip-flops em síntese são circuitos digitais de memória, onde cada Flip - Flop corresponde a uma memória de 1 bit. O Flip - Flop é o componente de memória mais importante sendo constituído por um conjunto de portas lógicas interconectadas. Existem diversos tipos de flip-flops, cada um deles com suas aplicações. No projeto será visto o circuito Latch (Circuito básico de memória assíncrono) que foi utilizado para criação dos circuitos Flip-Flops e analisaremos a fundo os Flip-Flops D e JK.

2.1.1. Latch com portas NAND

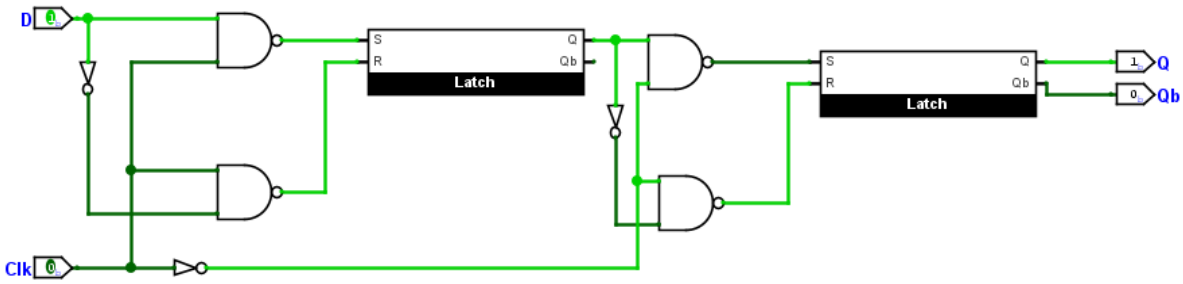
O circuito de Flip-Flop mais simples pode ser implementado utilizando duas portas NAND. As duas portas são interligadas de modo cruzado para fornecer a realimentação necessária para a função de memória.



O latch tem duas entradas: SET e RESET. As saídas das portas são denominadas Q e Q', em condições normais, elas serão o inverso uma da outra. Existem duas possibilidades de estados de saída igualmente prováveis: Q = 0 e Q' = 1, ou Q = 1 e Q' = 0.

2.1.2. Flip - Flop D

O Flip-Flop D utilizou dentro do circuito o Latch com portas NAND. Assim, o circuito possui uma única entrada sendo D, uma entrada de clock e as saídas são Q e Q negado.



Nesse Flip Flop, o estado da saída de Q é igualado ao estado de D quando o clock varia entre 0 e 1. Quando o clock é ativado:

D = 0: Q se torna 0.

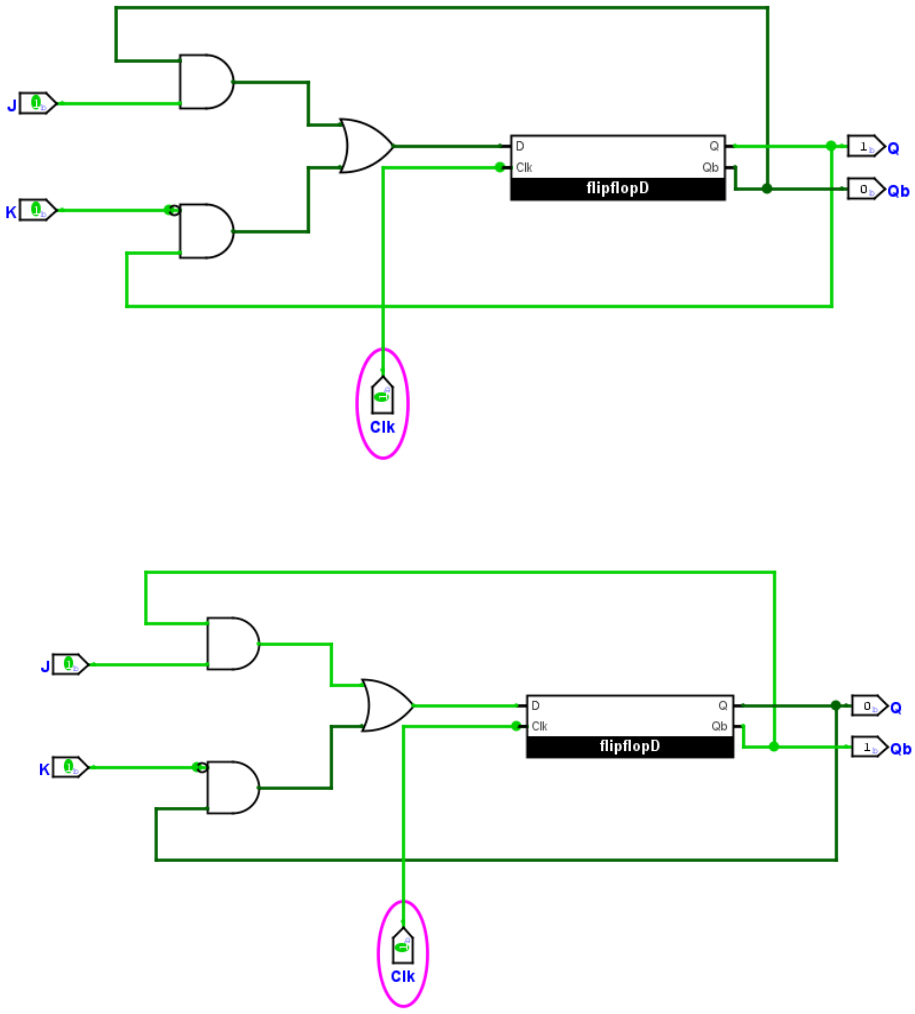
D = 1: Q se torna 1.

D	CLK	SAÍDA
0	↑	Q = 0
1	↑	Q = 1

2.1.3. Flip - Flop JK

O Flip-Flop de tipo JK, funciona de uma maneira mais complexa, utilizando o Flip-Flop D dentro do circuito. O Circuito do Flip-Flop JK, possui as entradas J e K , as saídas são Q e Q

negado, e o clock para alterar o valor dentro do Flip-Flop.

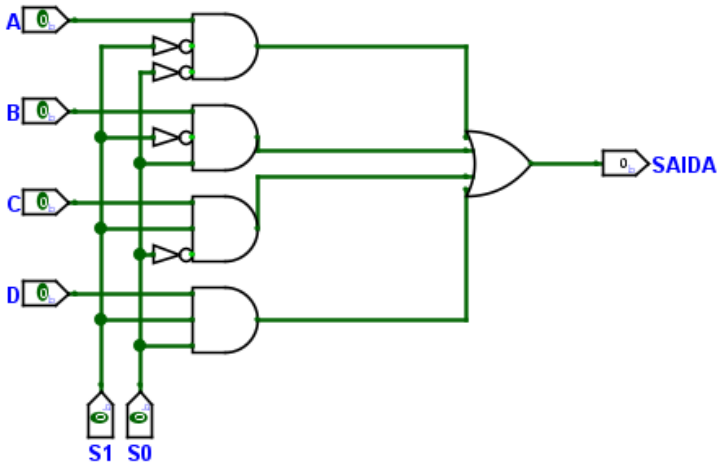


Quando o clock varia de 0 para 1, o valor a ser guardado pelo Flip-Flop alternam se as entradas J e K forem iguais a 1, e permanecerá o mesmo se ambos forem 0. Se as entradas forem diferentes, então o valor torna-se 1 se a entrada J for 1 e 0 se a entrada K for 1.

Operação do Flip Flop JK			
Tabela Verdade			
J	K	$Q_{próx}$	Comentário
0	0	$Q_{anterior}$	mantém (hold)
0	1	0	reseta
1	0	1	seta
1	1	$\overline{Q}_{anterior}$	alterna (Toogle)

2.2. Multiplexador de 4 entradas

O multiplexador é um circuito combinacional que seleciona uma entre várias entradas de dados e direciona essa entrada selecionada para a saída, pode ser entendido como um selecionador de dados. Um multiplexador de 4 entradas possui 4 linhas de entrada, mas apenas uma saída.

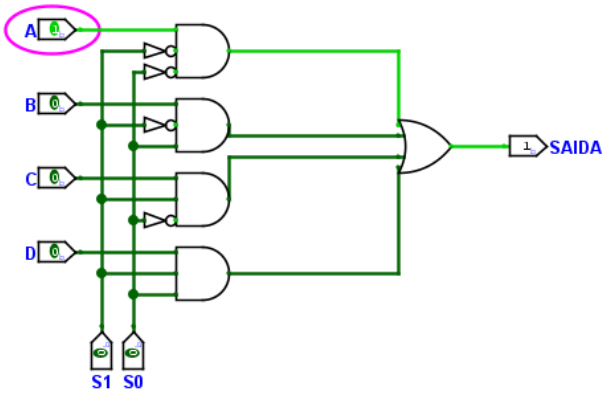


Nesse circuito estão presentes 4 entradas: A ,B ,C e D. O selecionador é formado pelas entradas S0 e S1, onde será selecionado qual das 4 entradas vai passar para a saída.

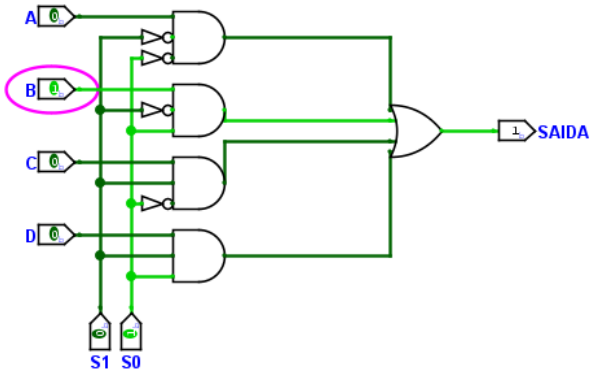
A	B	C	D	S0	S1	Saída
1	0	0	0	0	0	1
0	1	0	0	1	0	1
0	0	1	0	0	1	1
0	0	0	1	1	1	1

Saídas do multiplexador:

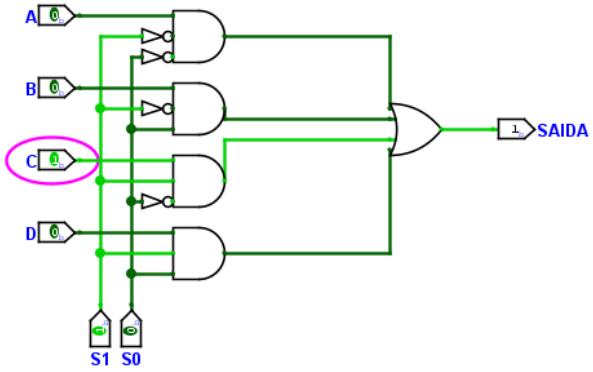
1) S0 = 0 e S1 = 0, saída recebe A.



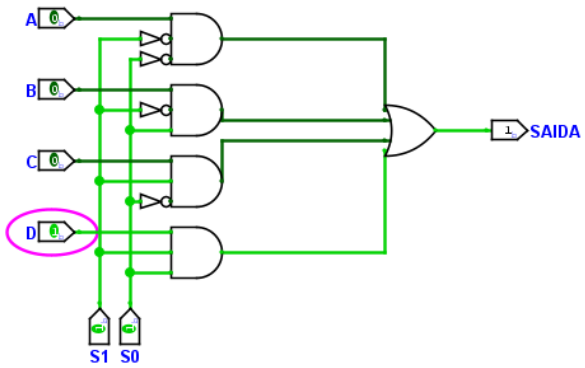
2) S0 = 1 e S1 = 0, saída recebe B.



3) S0 = 0 e S1 = 1, saída recebe C.



4) S0 = 1 e S1 = 1, saída recebe D.



2.3. Porta lógica XOR usando os componentes : AND, NOT e OR

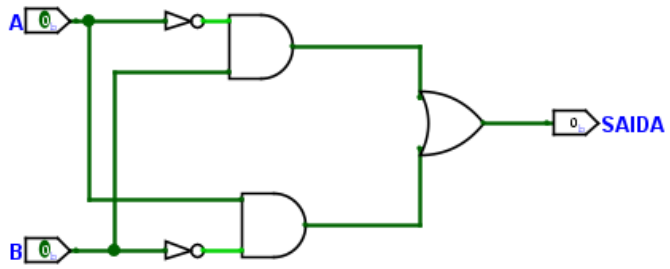
A porta lógica XOR é uma operação lógica que compara duas entradas e retorna um resultado que é verdadeiro (1) somente se apenas uma das entradas for verdadeira. Dessa forma, a porta XOR retorna 1 quando as entradas são diferentes, e retorna 0 quando as entradas são iguais.

A	B	SAÍDA
0	0	0
0	1	1
1	0	1
1	1	0

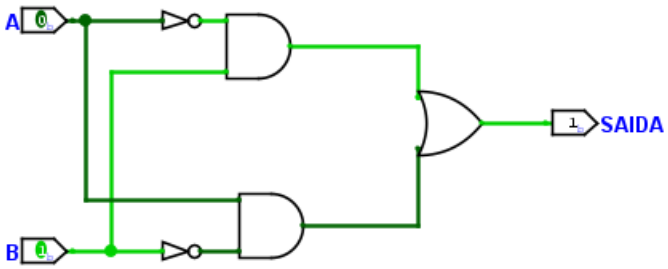
O circuito da porta lógica XOR possui 2 entradas e uma saída, com os componentes, NOT, AND e OR presentes.

Estados:

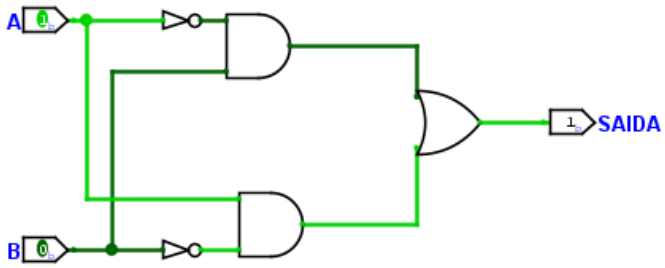
1) A = 0 e B = 0, a saída é 0.



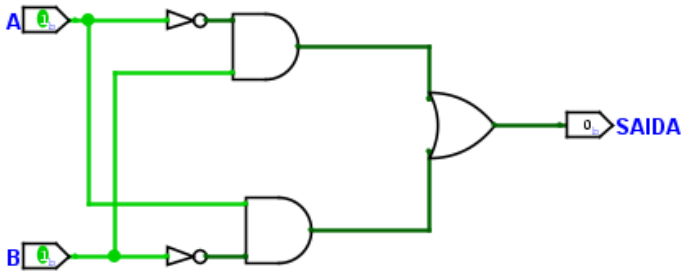
2) A = 0 e B = 1, a saída é 1.



3) A = 1 e B = 0, a saída é 1.



4) A = 1 e B = 1, a saída é 0.

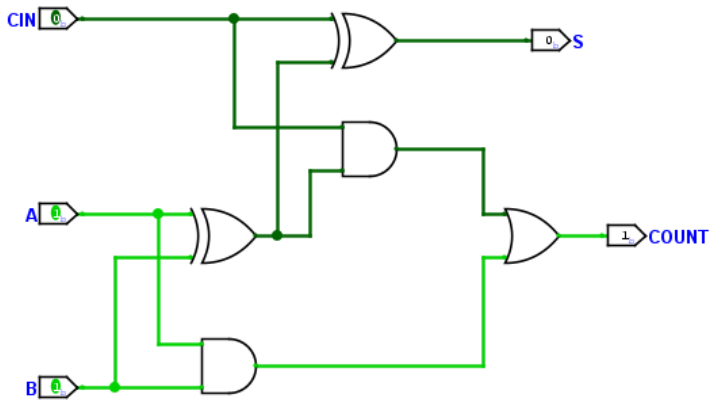


2.4. Somador de 8 bits

O somador de 8 bits é um componente de um circuito que consegue realizar equações de adição, opcionalmente, considera um carry-in (Cin) no início, também gera um carry-out (Cout) se houver um "vai-um" resultante da soma. Este circuito somador pode ser feito por 8 somadores de 1-bit ou 2 somadores de 4 - bits. Para criação deste somador foi realizada a construção de 1 somador de 1-bit, 1 somador de 4-bits e 1 somador de 8-bits. A seguir, todas as etapas da construção do circuito até a conclusão do Somador de 8 bits.

2.4.1. Somador de 1 - bit

O somador de 1 - bit é um circuito combinacional capaz de realizar soma por duas entradas de 1 - bit, entradas A e B, esse somador pode receber um valor extra para a soma como CIN, as saídas são S (soma) e COUT.



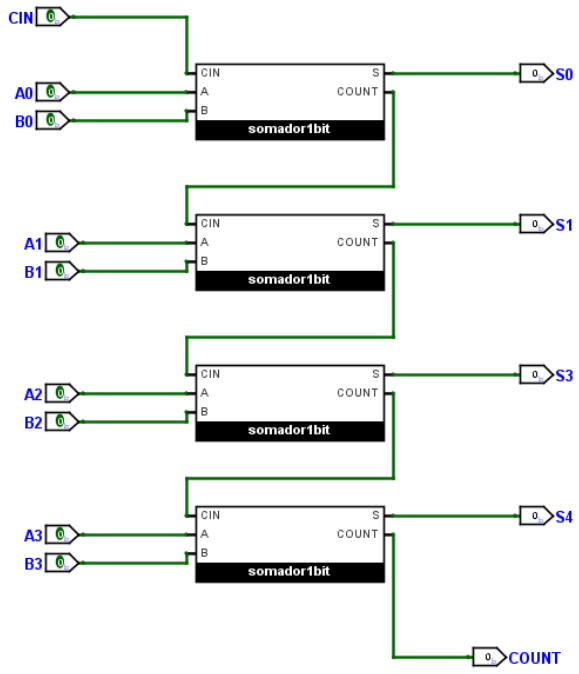
A soma é o resultado da operação de XOR entre A, B e CIN. COUT é o resultado da operação OR de três diferentes resultados: da operação de (A.B), da operação (A.Cin) e (B.Cin).

A	B	CIN	COUT	S
0	0	0	0	0

0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

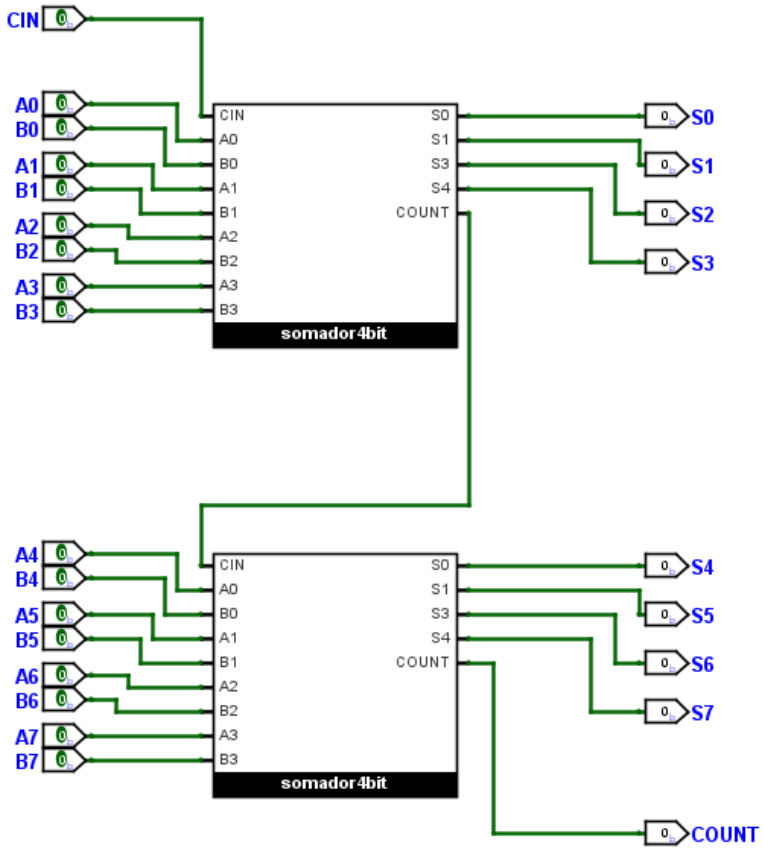
2.4.2. Somador de 4 bits

O somador de 4 bits possui 4 somadores de 1-bit e 9 entradas e 5 saídas. As entradas são A0B0, A1B1, A2B2, A3B3 e o CIN. As saídas são 4 resultados de soma e o COUT. Do modo que o valor do COUT de um somador está conectado ao CIN do próximo somador.



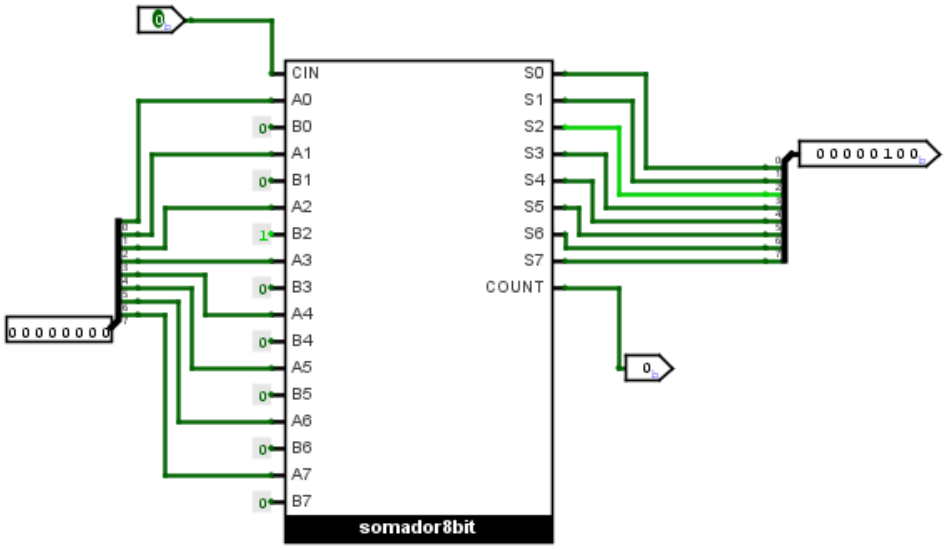
2.4.3. Circuito do somador de 8 bits completo

O somador de 8 bits é composto por 2 somadores de 4 bits. O CIN entra em um dos somadores e o COUT desse somador entra no CIN do outro somador, formando uma saída de 8 resultados de soma, que seriam 8 bits se estivessem juntas.



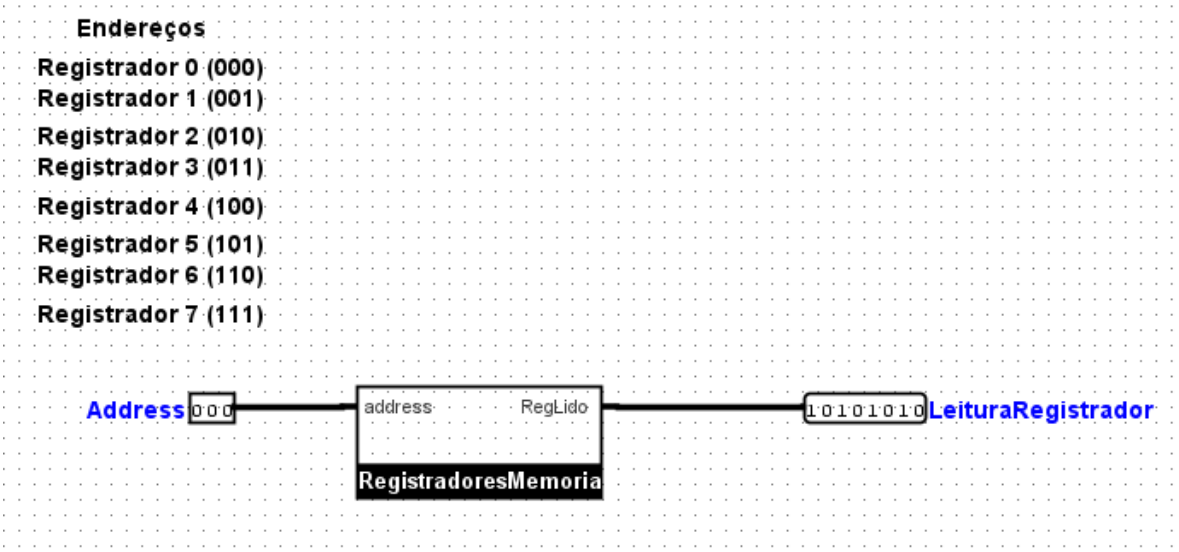
2.4.4. Somador de 8 bits que recebe um valor inteiro e soma com o valor 4.

Modificação do circuito de 8 bits, contabilizando a soma de 8 bits no final. É utilizado uma constante no valor de 4, sendo no circuito como B6, para sempre seja somado valores com 4.



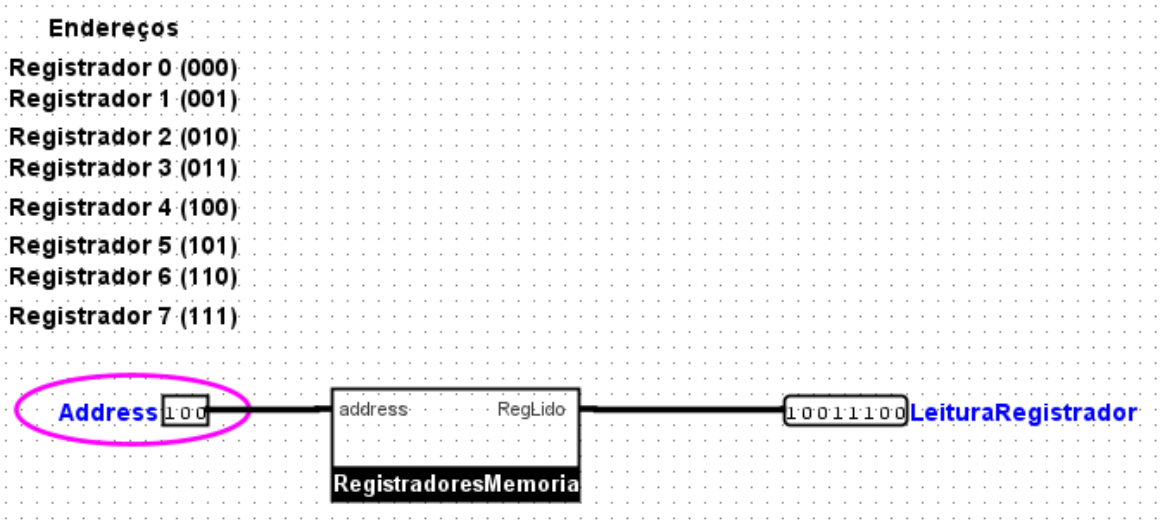
2.5. Memória ROM de 8 bits

A memória ROM é um dispositivo eletrônico no qual a função é apenas a leitura de dados, sendo impossível fazer qualquer alteração nos mesmos. Os valores inseridos na memória ROM são exibidos na saída do componente/circuito.

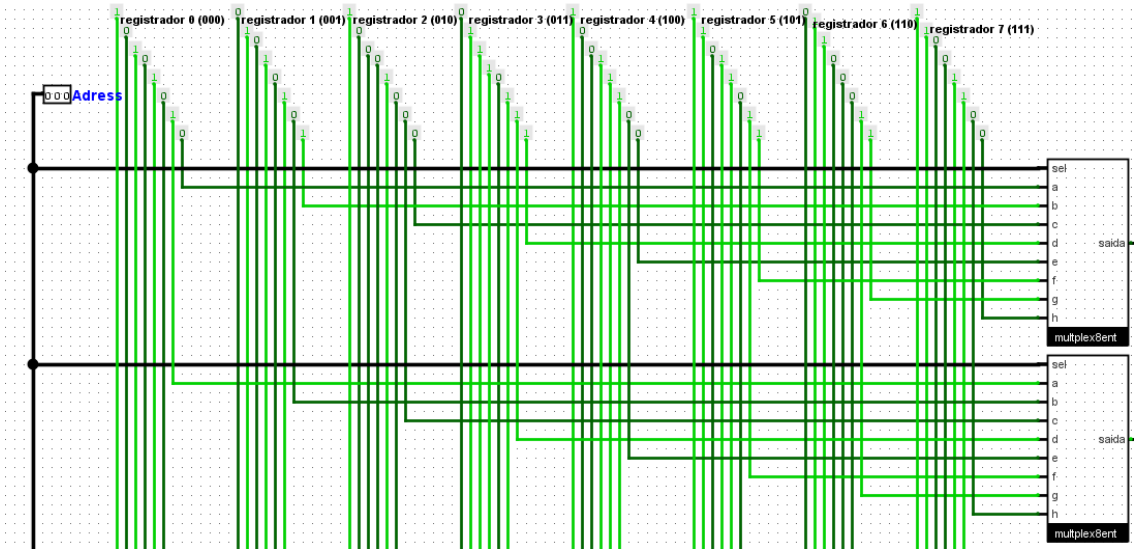


No circuito desenvolvido, é possível selecionar entre 8 endereços diferentes, com cada um sendo correspondente a um registrador (Esses registradores têm a mesma estrutura dos que foram exibidos no banco de registradores). No exemplo acima, o registrador que está sendo acessado é o primeiro, que tem Adress 000.

Exemplo 2 (Endereço 100):



Para o devido funcionamento do circuito, foi necessário montar o subcircuito chamado "Registradores memória" que armazena informações dos 8 registradores:

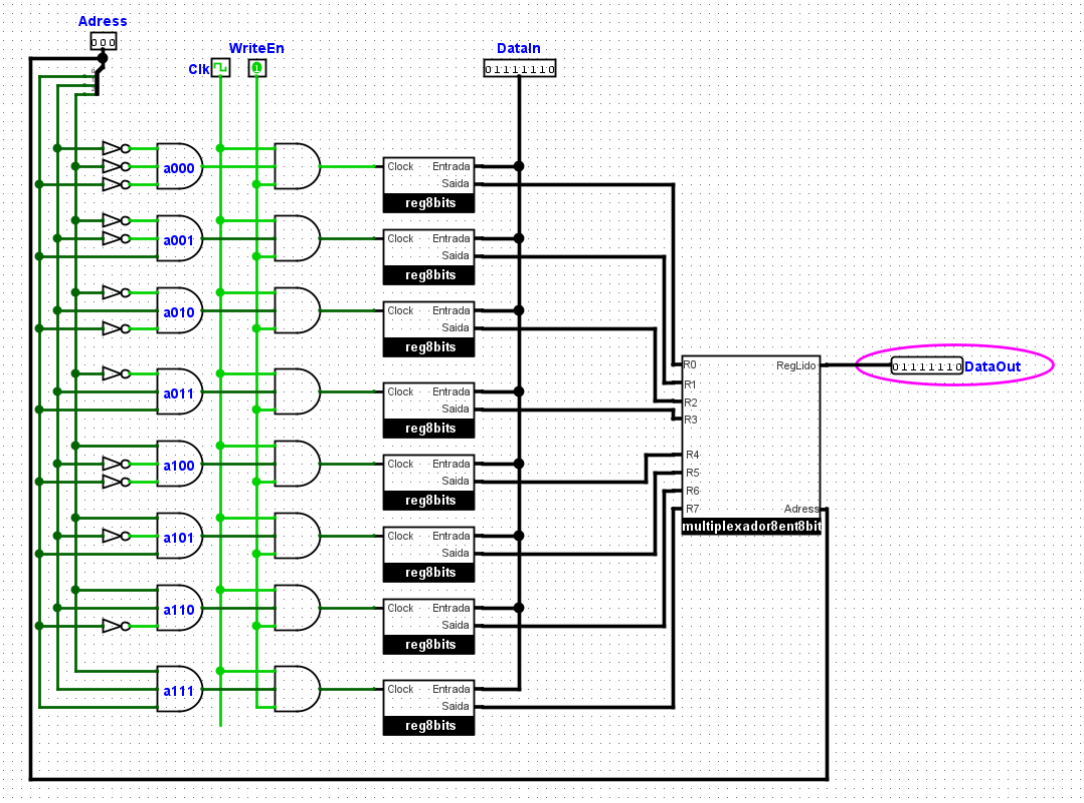


Nesse subcircuito estão presentes 8 multiplexadores de 8 entradas. Cada bit de dados dos 8 registradores estão conectados nas portas de A até H nos multiplexadores, sendo que os bits do Registrador 0 estão todos ligados nas portas A dos multiplexadores. E assim sucessivamente, até o Registrador 7, que está ligado nas portas H.

Com a alteração do valor em "Adress", é selecionado qual informação deve passar pelos multiplexadores de 8 entradas, por isso, se o endereço selecionado for 000, as informações contidas em A, referentes ao registrador 0, são passadas para a saída, que por meio de um distribuidor, une as 8 saídas em apenas uma, com 8 bits de capacidade.

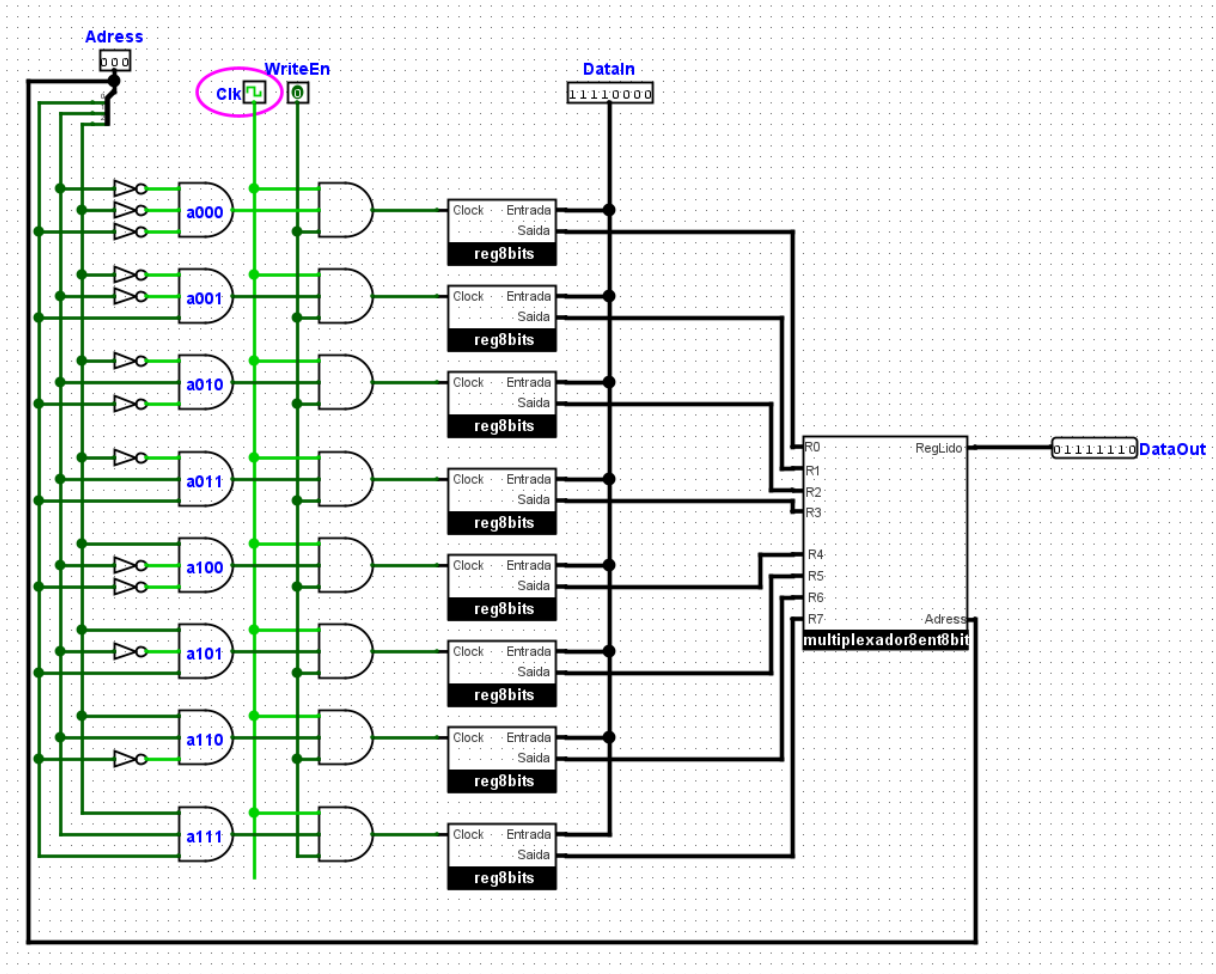
2.6. Memória RAM de 8 bits

A Memória Ram armazena temporariamente informações necessárias para o computador. Representamos o seu funcionamento por meio deste circuito:



Nele, existem 8 registradores de 8 bits ligados por um multiplexador de 8 entradas de 8 bits, que podem ser manipulados. Os endereços de cada registrador estão sendo exibidos no circuito. Para selecionar o registrador desejado, basta mudar o valor da chave de seleção “Adress”. Com um Registrador selecionado, é necessário ligar a função de escrita “WriteEn” para 1. Caso isso não seja feito, o estado de leitura será mantido. A informação que será escrita está em “DataIn”. Com a chave devidamente selecionada e o modo de escrita ligado, é necessário rodar um ciclo de clock para que a mudança seja efetivada. No circuito usado de exemplo, é possível apenas modificar um registrador por vez, porém, as informações inseridas ficam salvas mesmo se o usuário alterar os dados de outro registrador.

Exemplo1 (Alteração de dados com o WriteEn desligado):



Nesse exemplo, tentamos alterar a informação contida no Registrador0, que continha

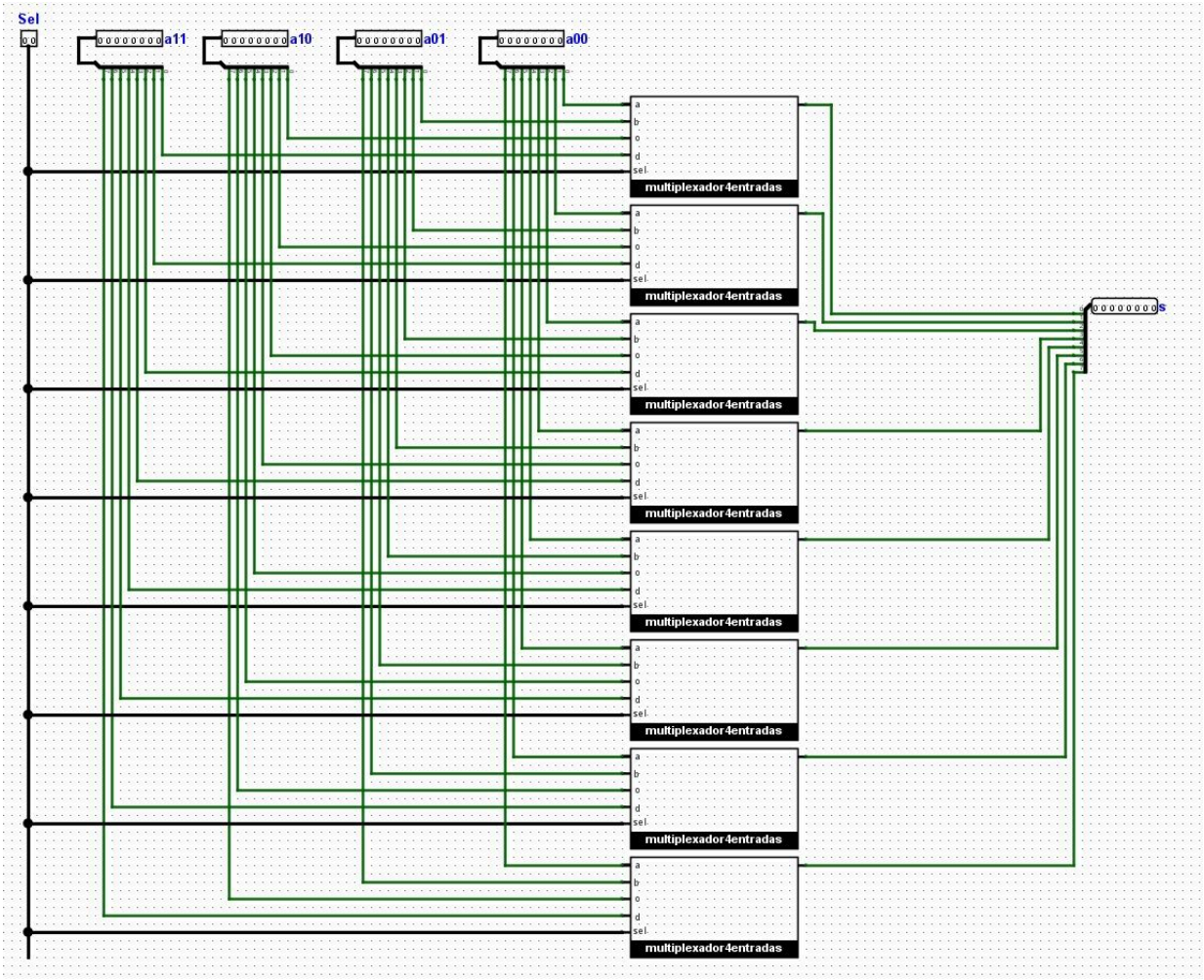
“01111110” armazenado. Ao tentar passar “11110000” para sua memória, mesmo rodando diversos ciclos de clock, não foi possível alterar o valor salvo

2.7. Banco de Registradores de 8 bits

Um banco de registradores é um componente digital composto por um conjunto de registradores que podem ser acessados de forma organizada. Ele é projetado para armazenar pequenos blocos de dados temporariamente e facilitar o acesso rápido durante a execução de operações pelo processador. Para a realização deste circuito foi preciso a construção de um Multiplexador 8 bits e um Registrador 8 bits.

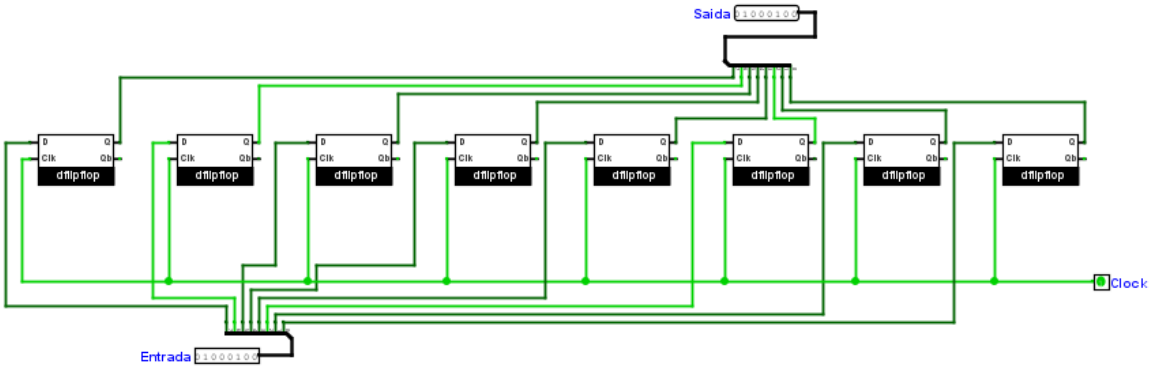
2.7.1. Multiplexador 4 entradas de 8 bits

Para a construção do banco de registradores, foi utilizado um circuito de multiplexador de 8 bits, desenvolvido a partir do multiplexador de 4 entradas implementado anteriormente. Cada multiplexador está conectado a 8 linhas de entrada de dados e é controlado por 3 bits de seleção. As entradas são organizadas em blocos de 8 bits, correspondendo a diferentes portas nos multiplexadores. Os 3 bits de seleção determinam qual bloco de dados será encaminhado para a saída. Essa saída possibilita o processamento simultâneo de todos os 8 bits selecionados, garantindo eficiência e precisão na manipulação dos dados.



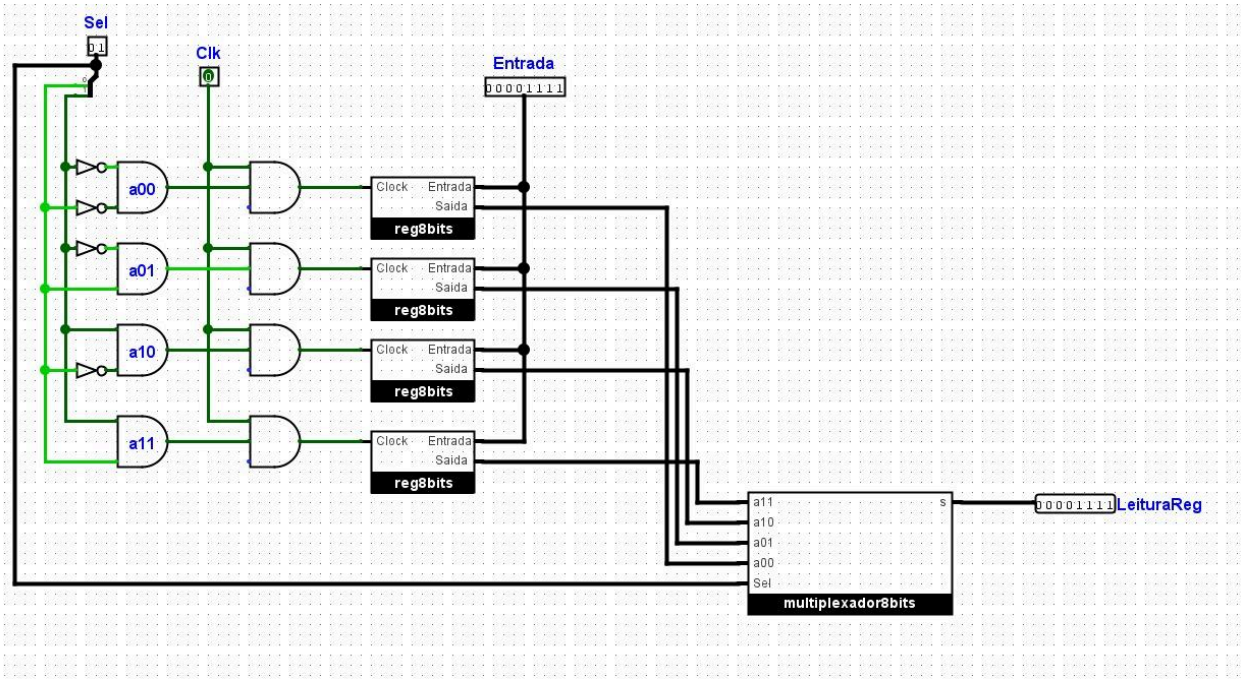
2.7.2. Registrador 8 bits

Neste circuito foram utilizados 8 registradores de 1-bit, utilizando o Flip-Flop tipo D feito anteriormente. Toda subida de clock para 1, o valor é salvo no espaço de memória.



2.7.3. Circuito Banco de Registradores de 8 bits

O sinal Sel (Selector) controla qual dos registradores será acessado para leitura ou escrita. Assim, o circuito de seleção é implementado com portas lógicas AND, cada linha de portas verifica uma combinação específica de valores de seleção, dessa forma apenas um registrador será ativado por vez, dependendo do valor binário de Sel. O clock ativa os registradores para armazenar os dados de entrada (8 bits) apenas no momento correto. Essa entrada é compartilhada entre todos os registradores, mas somente o registrador selecionado armazenará os dados.



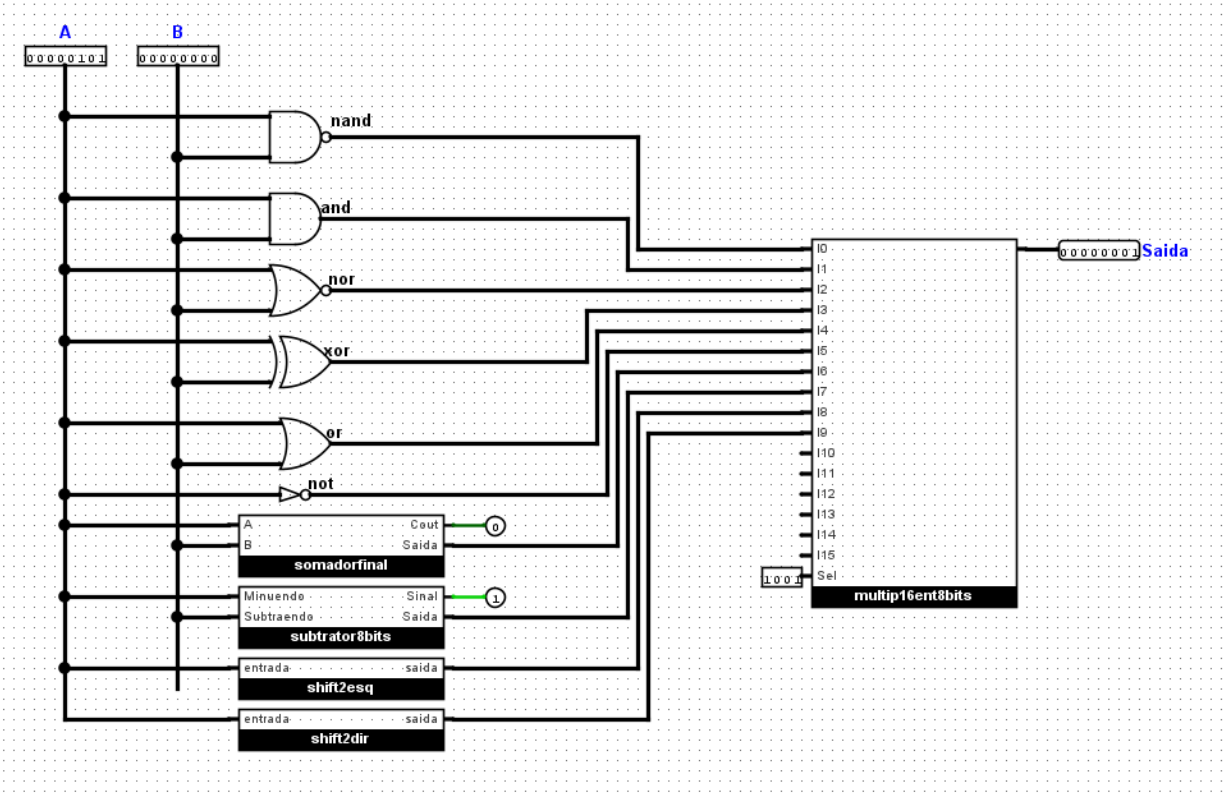
2.8. ULA de 8 bits

Uma Unidade Lógica e Aritmética (ULA) é um componente essencial em circuitos digitais, responsável por realizar operações lógicas e aritméticas sobre dados binários. Normalmente, a ULA recebe dois operandos como entradas (A e B) e utiliza uma entrada auxiliar, chamada seletor, para determinar qual operação será executada.

A construção de uma ULA baseia-se em dois princípios fundamentais: o controle do fluxo de dados e a implementação de circuitos específicos para cada operação. O circuito descrito apresenta uma ULA de 8 bits, com as seguintes características:

1) Entradas: operandos A e B e um seletor de operações.
2) Saída: resultado da operação realizada.
3) Operações disponíveis: portas lógicas AND, OR, NOT, NOR, NAND, XOR, além de operações de soma, subtração, deslocamento de bits à esquerda (SHIFT LEFT) e deslocamento de bits à direita (SHIFT RIGHT).

O seletor controla o tipo de operação a ser realizada. Cada combinação binária no seletor ativa um componente específico da ULA.



No circuito acima é apresentada uma ULA de 8 bits, que possui as entradas A e B. A saída é dada pelo resultado do multiplexador de 16 entradas. Segue abaixo o guia de endereços para selecionar a função desejada

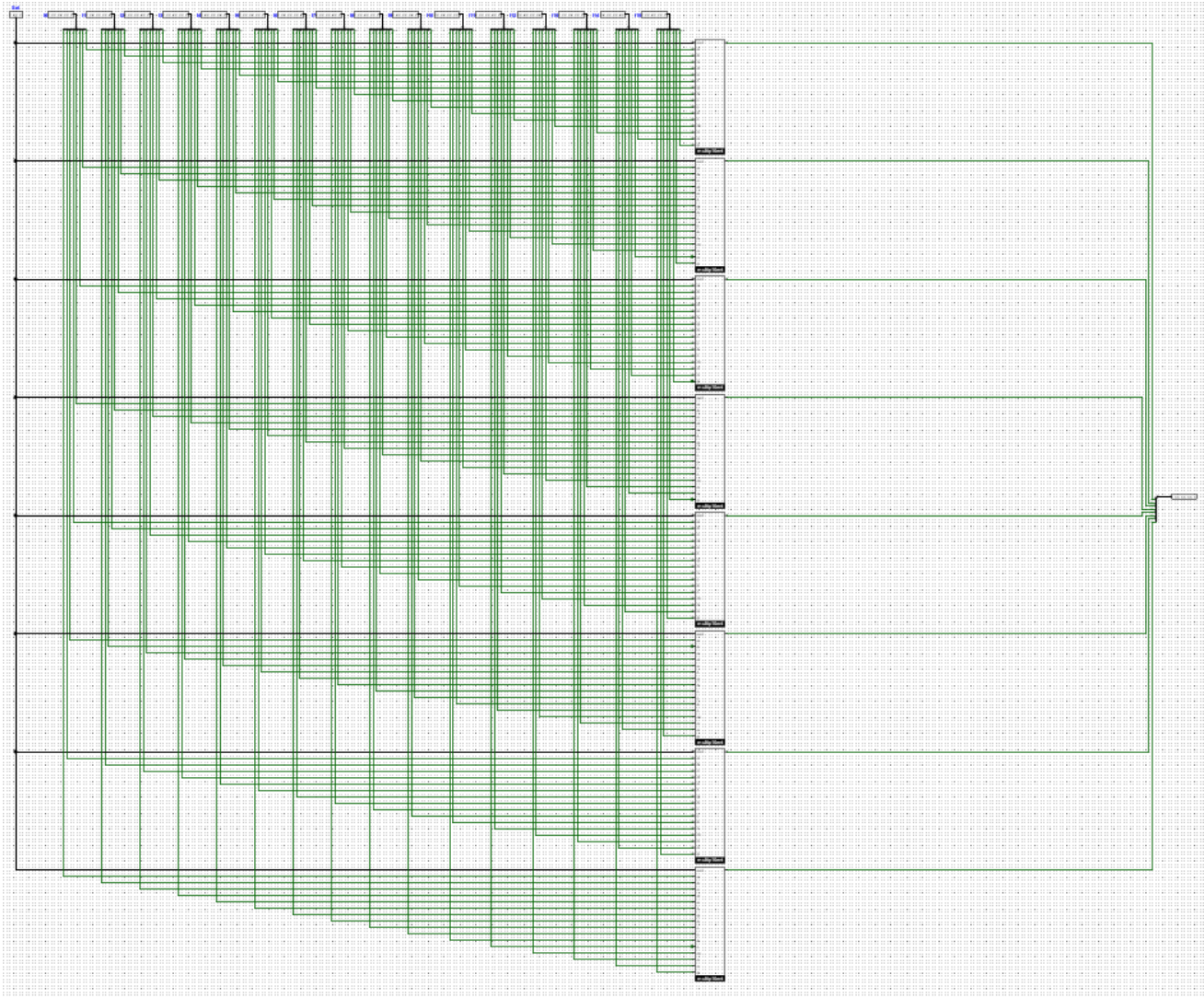
Saídas:

- Ao digitar 0000 em “sel”, vai ser acionado a porta lógica NAND.
- Ao digitar 0001 em “sel”,vai ser acionado a porta lógica AND.
- Ao digitar 0010 em “sel”, vai ser acionado a porta lógica NOR.
- Ao digitar 0011 em “sel”,vai ser acionado a porta lógica XOR.
- Ao digitar 0100 em “sel”, vai ser acionado a porta lógica OR.
- Ao digitar 0101 em “sel”, vai ser acionado a porta lógica NOT.
- Ao digitar 0110 em “sel”, vai ser acionado a porta lógica SOMA.

- Ao digitar 0111 em “sel”, vai ser acionado a porta lógica SUBTRATOR
- Ao digitar 1000 em “sel”, vai ser acionado a porta lógica SHIFT 2 BITS P/ESQUERDA.
- Ao digitar 1001 em “mult”, vai ser acionado a porta lógica SHIFT 2 BITS P/DIREITA.

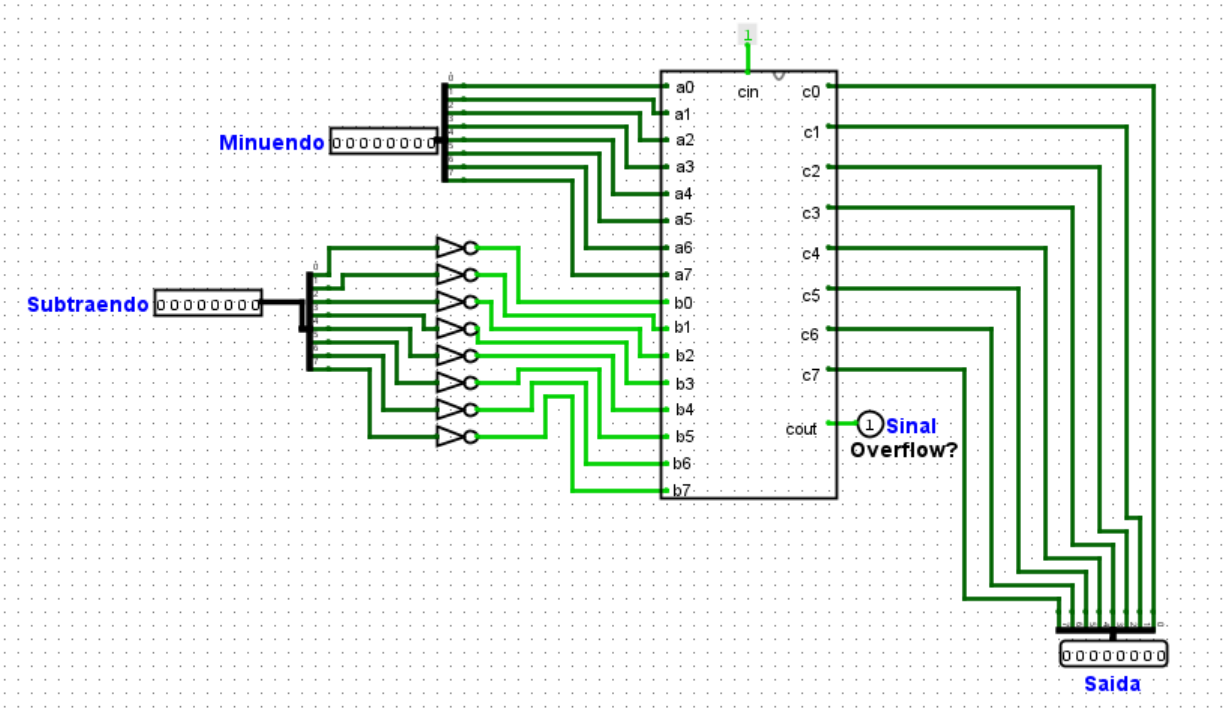
As 6 primeiras funções são possíveis graças às portas lógicas. A função de soma utiliza o somador de 8 bits, que já foi mostrado anteriormente.

2.8.1. Multiplexador 16 Entradas 8 bits



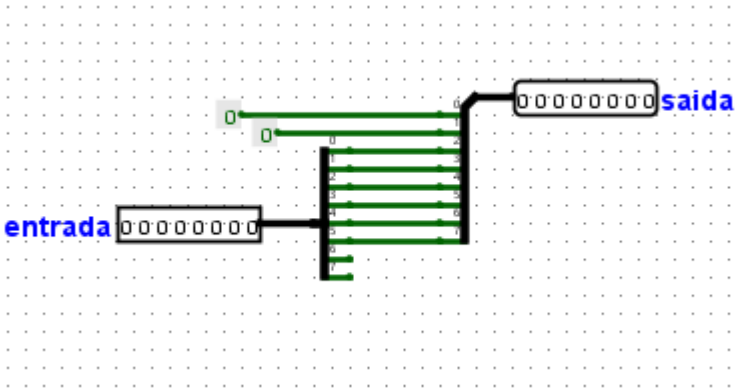
A exibição correta da operação desejada ocorre em decorrência do multiplexador de 16 entradas, que seleciona a saída certa a ser exibida, nele, temos as entradas que vão de I0 a I15, a chave seletora tem 4 bits. Assim, para selecionar I0 devemos selecionar “0000”, e assim respectivamente, até I15, que tem o endereço “1111”.

2.8.2. Subtrator de 8 Bits



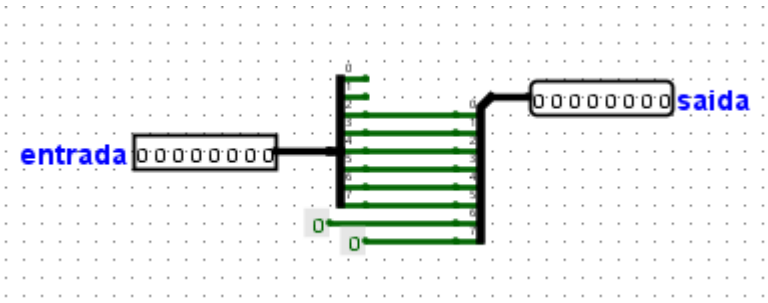
Para subtrair um número em binário, é necessário somar o minuendo ao complemento de 2 do subtraendo. Para simular o complemento de 2, foi alterado o subcircuito já utilizado para fazer o somador de 8 bits, mas adicionamos portas not nas entradas referentes ao subtraendo. Para sempre somar 1 no bit menos significativo, definimos o Carry In para ser constantemente 1.

2.8.3. Shift de 2 bits à esquerda



Para passar uma entrada dois bits à esquerda, utilizamos distribuidores para mover os bits da entrada para frente, definindo os dois bits menos significativos da saída como zero por padrão.

2.8.4. Shift de 2 bits à direita

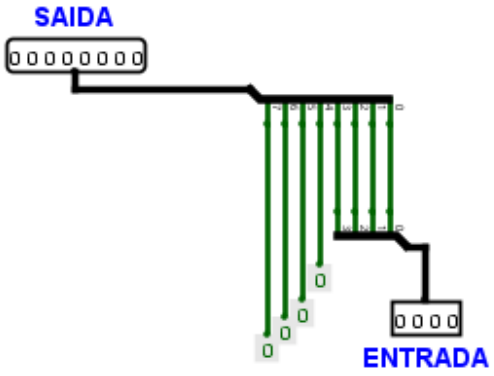


Para passar uma entrada dois bits à direita, utilizamos distribuidores para mover os bits da entrada para trás, definindo os dois bits mais significativos da saída como zero por padrão.

2.9. Extensor de sinal de 4 bits para 8 bits.

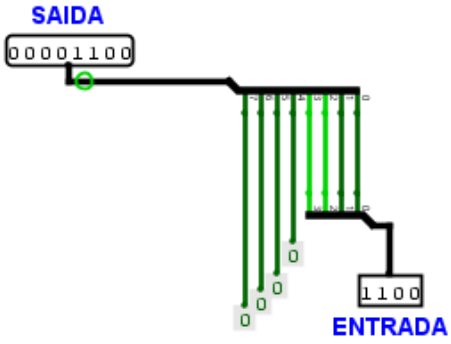
O extensor de sinal é um circuito que converte um valor binário de 4 bits (entrada) em um valor binário de 8 bits (saída). Para transformar para uma largura menor, os bits de mais baixa ordem serão truncados. Para transformar para uma largura maior, os bits menos significativos serão os mesmos. Porém, para os bits de mais alta ordem, eles poderão ser todos iguais a 0, ou todos iguais a 1, ou concordarem com a entrada do bit de sinal (o mais significativo), ou ainda

ter esse valor determinado por uma entrada adicional.



Esse circuito possui uma entrada de 4 bits, uma saída de 8 bits, 2 distribuidores sendo um de 8 bits e outro de 4 bits, e 4 valores de constantes como 0. Quando ocorre a entrada de 4 bits no distribuidor de 4 bits, é passado esses para os bits de mais baixa ordem do distribuidor de 8 bits e os de mais alta ordem recebem 0 para que ocorra a saída do circuito como 8 bits

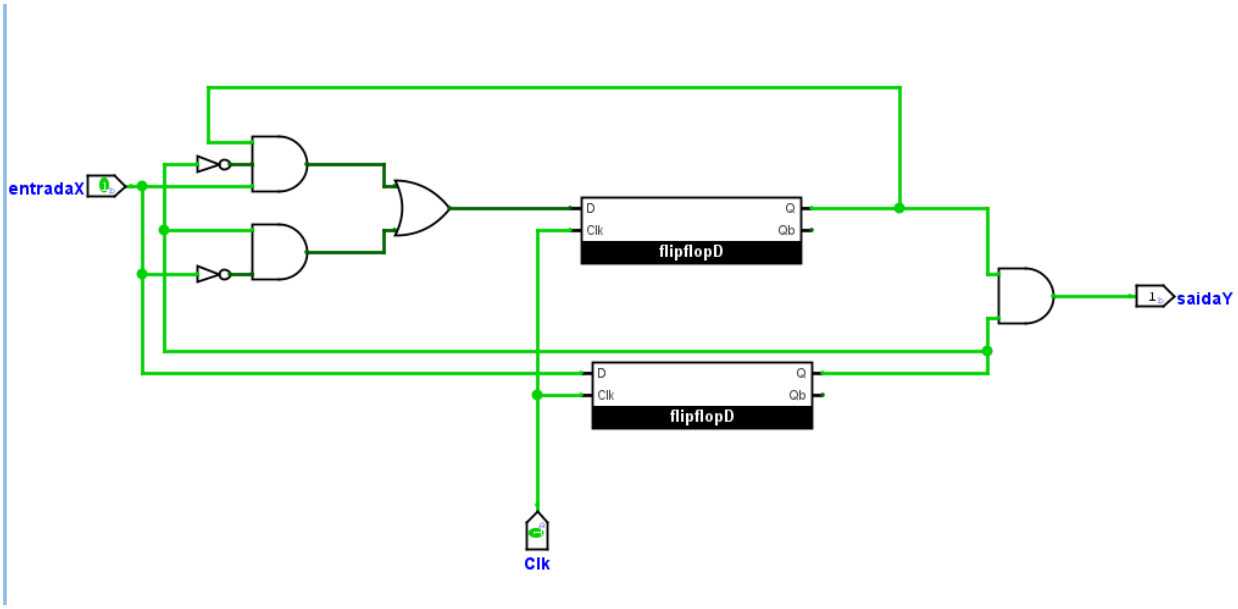
Entrada de 1100 no circuito:



2.10. Máquina de Estados

A Máquina de Estados é um modelo usado para representar o comportamento de sistemas que

podem assumir diferentes condições (estados) e que transitam entre esses estados com base em condições específicas.



Essa Máquina de Estados é um circuito com flip-flops do tipo D (feito anteriormente). Esse circuito é capaz de detectar todas as ocorrências da sequência 101, quando a máquina identifica essa sequência, a saída Y deve ser de nível lógico alto (1).

1) Flip - Flops D: Esses dois flip-flops armazenam o estado atual da máquina. O estado é representado pelos sinais de saída Q dos flip-flops.

2) Portas lógicas (AND, OR, NOT): são usadas para determinar as transições entre os estados com base nas entradas e no estado atual.

3) Entrada X: É a variável de entrada que determina como a máquina transitará entre os estados. X = 0 ou X = 1.

4) Saída Y: É a saída gerada pelo circuito, que depende apenas do estado atual por considerar a Máquina de Moore.

5) Gerador de Clock: controla quando os flip-flops atualizam o estado.

Tabela de transição:

Estado Atual	Estado Próximo (X = 0)	Estado Próximo (X = 1)	Saída (Y)
S_0	S_0	S_1	0
S_1	S_{10}	S_1	0
S_{10}	S_0	S_{101}	0
S_{101}	S_{10}	S_1	1

Funcionamento:

1) Estado S0:

   Se X = 0: Permanece em S0.

   Se X =1: Transita para S1

   Y = 0.

2) Estado S1:

   Se X = 0: Transita para S10.

   Se X = 1: Permanece em S1.

   Y = 0.

3) Estado S10

   Se X = 0: Transita para S0.

   Se X = 1: Transita para S101.

   Y = 0.

   Estado S101

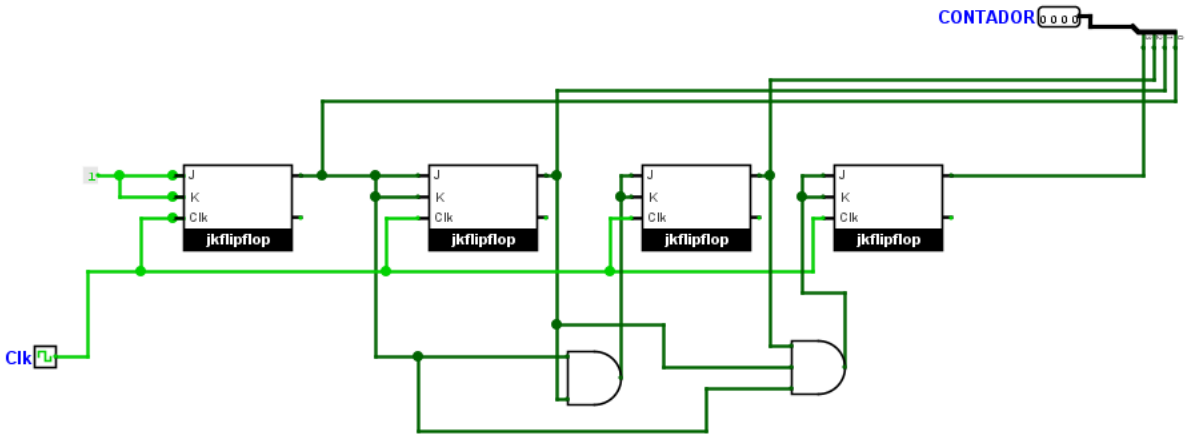
   Se X = 0: Transita para S10.

Se X = 1: Transita para S1.

Y = 1.

2.11. Contador Síncrono

O Contador Síncrono é um circuito digital formado por Flip-Flops JK em paralelos, com todas as entradas clocks conectadas na mesma fonte de clock.



O contador possui uma sequência de 4 Flip-Flops de tipo JK, um pino de uma constante, o pulso de clock e a saída. A cada subida do clock para 1, é aumentado o número da saída. A seguir é apresentado todos os resultados de saída ao pulso do clock:

Pulso de clock	Saída de bits
1	0001
0	0001

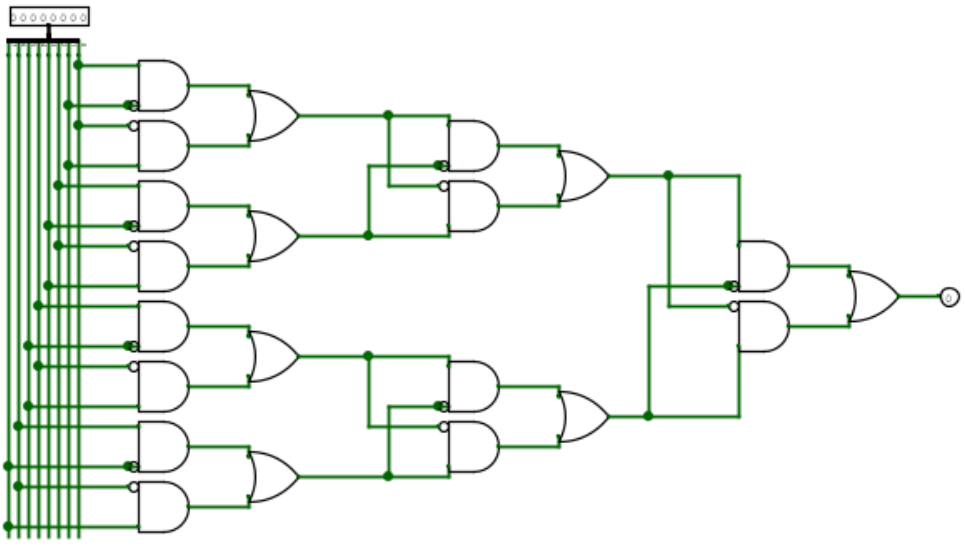
1	0010
0	0010
1	0011
0	0011
1	0100
0	0100
1	0101
0	0101
1	0110
0	0110
1	0111
0	0111
1	1000
0	1000
1	1001
0	1001
1	1010
0	1010
1	1011
0	1011
1	1100
0	1100
1	1101
0	1101
1	1110

0	1110
1	1111
0	1111
1	0000

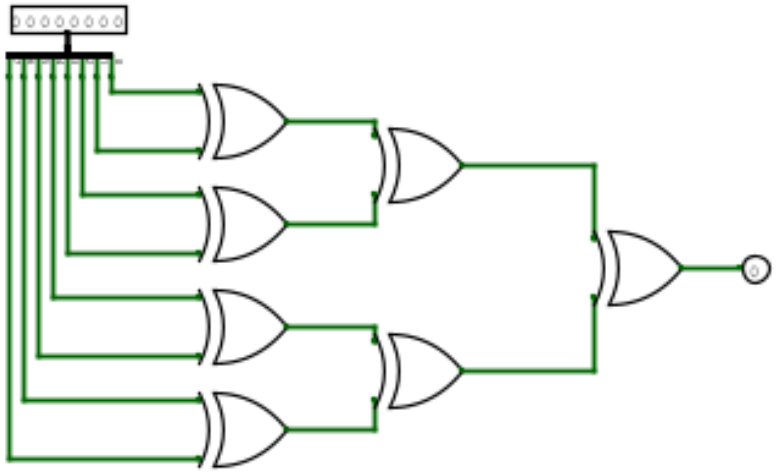
2.12. Detector de Paridade Ìmpar

O Detector de Paridade Ímpar é um circuito digital que verifica se a quantidade de bits em um dado conjunto de entrada é ímpar. Se o número de bits iguais a 1 na entrada for ímpar, a saída será 1; caso contrário, será 0.

1) Detector de Paridade Ímpar com AND, NOT e OR:
Esta é uma adaptação da porta XOR = (A . ~B) + (~A . B)



2) Detector de Paridade Ímpar usando a porta XOR:



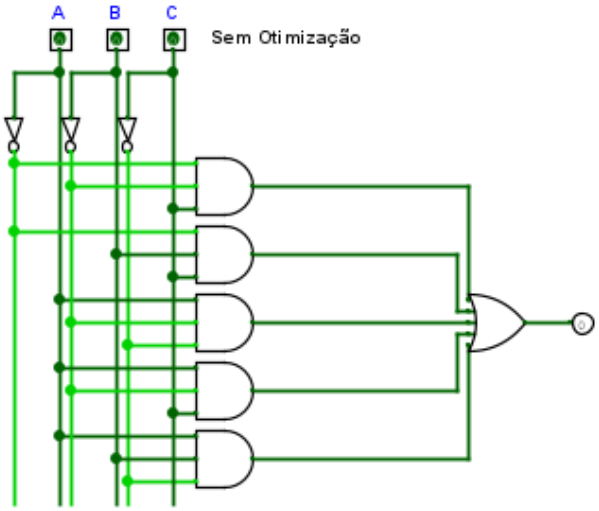
2.13. Circuito Otimizado

Circuitos otimizados são circuitos digitais projetados para atingir maior eficiência em algum aspecto específico. Nesse exemplo ele foi otimizado para reduzir o número de portas lógicas e conexões necessárias, mantendo o comportamento lógico descrito pela tabela verdade.

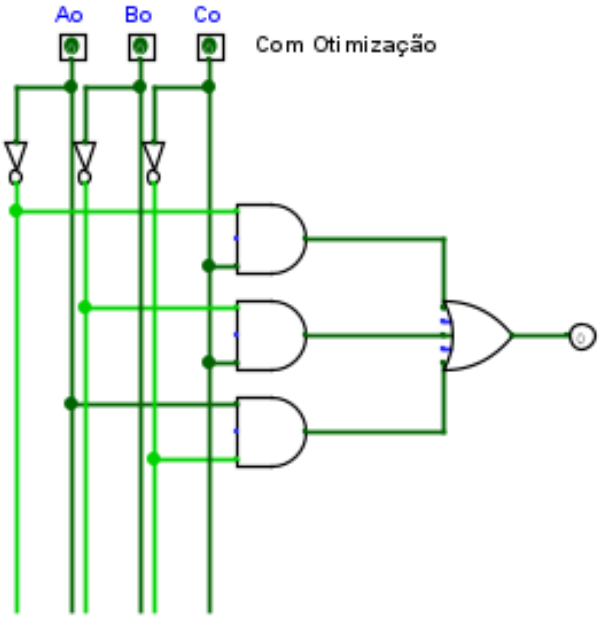
dada a tabela verdade:

A	B	C	||S||
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

1) O circuito sem otimização:



2) O circuito com otimização:

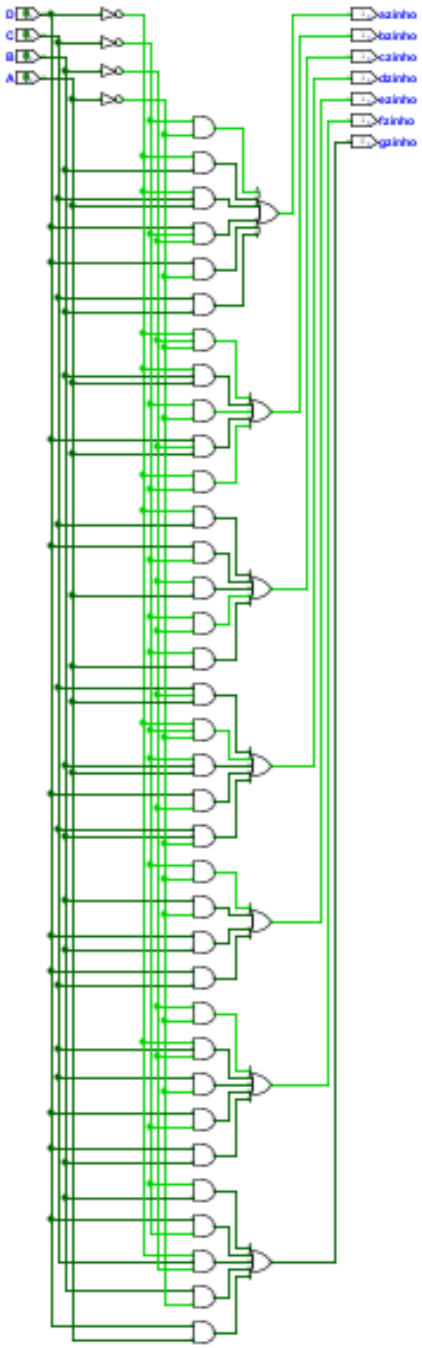


2.14. Decodificador de 7 Segmentos

O decodificador de 7 segmentos é um circuito digital que converte dados binários em sinais elétricos que acendem os segmentos individuais de um display de 7 segmentos, exibindo números ou caracteres.

2.14.1. Conversão de Binário para Hexadecimal

Cada grupo de 4 bits na entrada é mapeado para um dígito hexadecimal correspondente.



2.14.2. Circuito Display 7 Segmentos

O display de 7 segmentos é composto por 7 partes, cada uma identificada pelas letras a, b, c, d, e, f, g. O decodificador, que controla o funcionamento do display, recebe um valor binário como entrada e possui 7 saídas, cada uma conectada diretamente a um segmento do display. Com base no valor recebido, o decodificador ativa os segmentos específicos necessários para formar o número ou caractere correspondente no display.

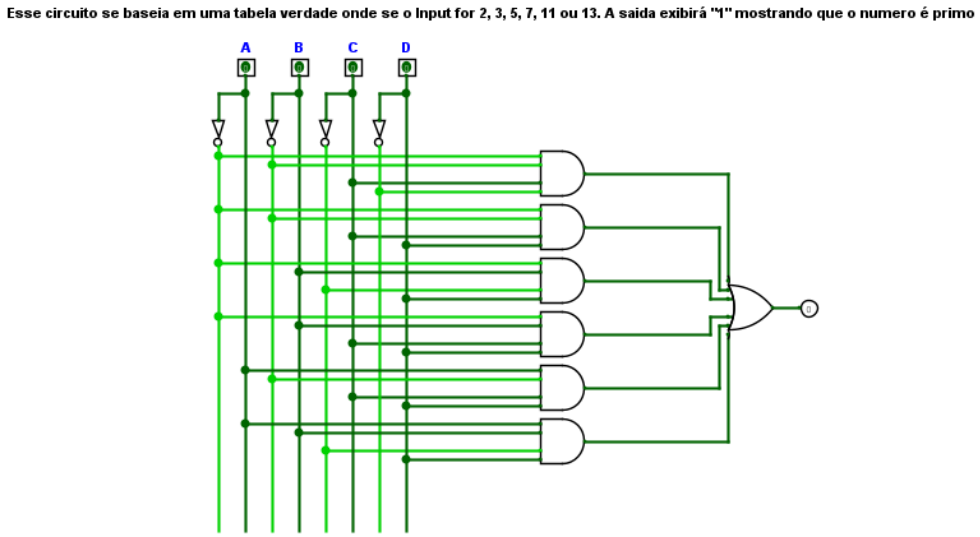
Tabela verdade:

dcba	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0000 (0)	1	1	1	1	1	1	0
0001 (1)	0	1	1	0	0	0	0
0010 (2)	1	1	0	1	1	0	1
0011 (3)	1	1	1	1	0	0	1
0100 (4)	0	1	1	0	0	1	1
0101 (5)	1	0	1	1	0	1	1
0110 (6)	1	0	1	1	1	1	1
0111 (7)	1	1	1	0	0	0	0
1000 (8)	1	1	1	1	1	1	1
1001 (9)	1	1	1	1	0	1	1
1010 (A)	1	1	1	0	1	1	1
1011 (b)	0	0	1	1	1	1	1
1100 (C)	1	0	0	1	1	1	0
1101 (d)	0	1	1	1	1	0	1
1110 (E)	1	0	0	1	1	1	1
1111 (F)	1	0	0	0	1	1	1

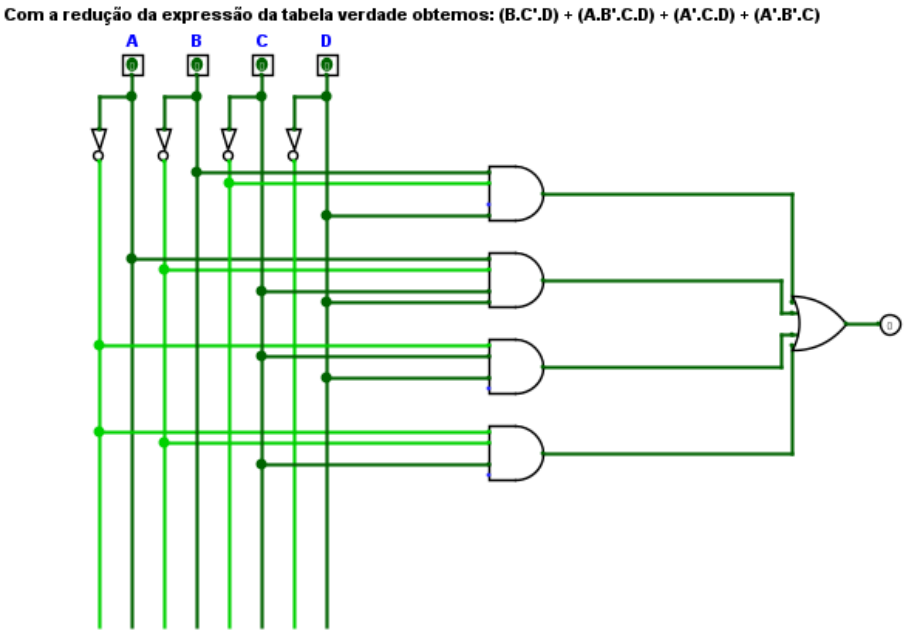
2.15. Detector de Números Primos

Um detector de número primo é um circuito digital que recebe um número como entrada e determina se ele é primo ou não, verificando se o número possui exatamente dois divisores positivos: ele mesmo e 1. Para este circuito foram feitos dois sub circuitos para definir se um número é primo ou não. Um circuito sem otimização e um otimizado.

2.15.1. Primo sem Otimização

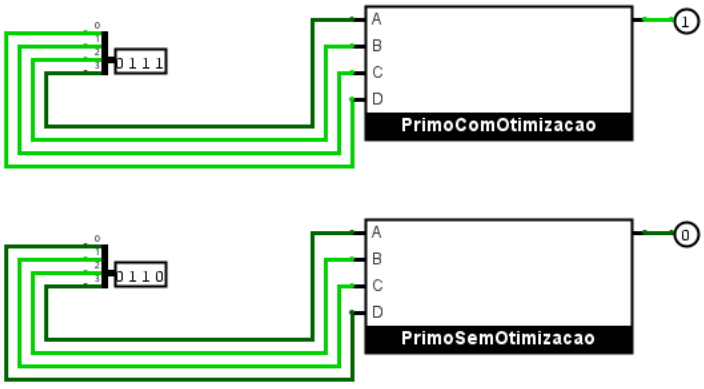


2.15.2. Primo com Otimização



2.15.3. Circuito Detector de Números Primos

Foram feitos dois detectores, um circuito sem otimização e o outro otimizado usando o mapa de karnaugh. Ambos circuitos recebem 4 bits e depois verificam se o número é primo ou não. Se a saída for igual a 1 o número é primo, se a saída for igual a 0 o número não é primo.



3. Considerações Finais

Esse trabalho reforçou a importância do planejamento e da organização no desenvolvimento de circuitos digitais, evidenciando como erros lógicos podem ser rapidamente identificados e corrigidos durante a simulação. Por fim, a experiência proporcionou uma base sólida para projetos futuros na área de Arquitetura e Organização de Computadores, alinhando teoria e prática de forma integrada.

As fotos de todos os componentes individuais estão disponíveis no repositório, sendo acessíveis neste link:

https://github.com/JasmimSabini/AOC_JasmimSabiniViniciusMartins_UFRR_LabCircuitos_2024