Unidade Lógica Aritmética (ULA)

Projeto realizado para cadeira de Projeto de Sistemas Digitais - UNISINOS

Cícero Calsing   
cicerocalsing@hotmail.com Gustavo Parcianello Cardona  
gupcardona@gmail.com

ㅤRafael Klaus  
kp.rafael@gmail.com

*Resumo*—Neste trabalho de Sistemas Digitais tivemos que realizar a criação e implementação de uma Unidade Lógica Aritmética (ULA) nas plataformas Logisim e Tinkercad. Na criação da ULA tínhamos quatro operações lógicas obrigatórias para serem adicionadas, AND, OR, soma e subtração. No decorrer deste artigo será possível entender todas as implementações realizadas nesse processo.

Palavras-chave—ULA, lógica, Sistemas Digitais

# Introdução

ULA se trata de um acrônimo para unidade lógica aritmética. Sua função é efetuar operações, tanto aritméticas (adição, subtração, comparação etc.), quanto lógicas (AND, OR, XOR etc.).

Sua origem está relacionada com John von Neumann. Este matemático húngaro a propôs pela primeira vez em 1946, ao trabalhar na busca de melhorias computacionais relacionadas ao projeto EDVAC, o qual deu origem à renomada Arquitetura de Von Neumann. A ULA era um de seus componentes. A unidade lógica aritmética é utilizada até hoje nos processadores, sendo parte fundamental da arquitetura. Ao longo do tempo, os componentes usados na construção da ULA foram evoluindo e novas formas de construção foram surgindo, porém, ela segue tendo o mesmo papel dentro dos processadores.

Em nosso projeto, construímos uma Unidade Lógica Aritmética que possui as operações de soma, subtração, AND e OR, conforme solicitado pelo professor. A seguir, iniciaremos a demonstração e a explicação do processo de criação e montagem da ULA.

# Operações artiméticas e lógicas

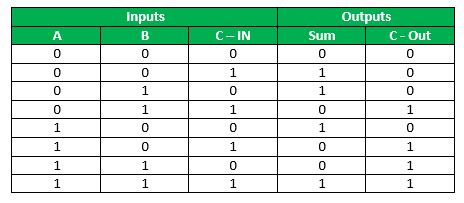
## Somador

Um somador realiza a soma entre dois números binários. Para o trabalho, foram feitos somadores de 4 bits, ou seja, que somam números binários de até 4 dígitos.

1. *Implementação do somador no Logisim*



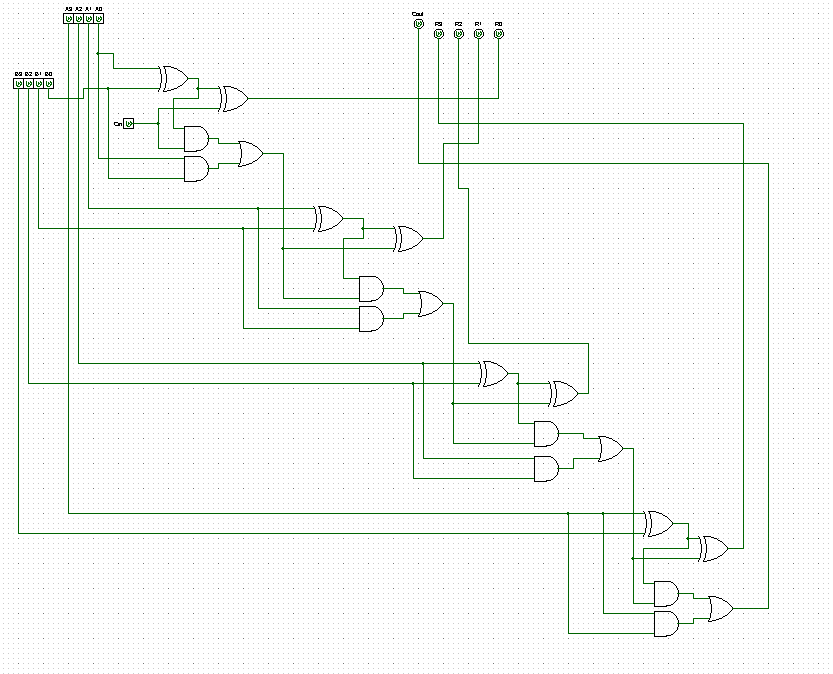
<https://www.geeksforgeeks.org/full-adder-in-digital-logic/>



<https://www.geeksforgeeks.org/full-adder-in-digital-logic/>

Na tabela verdade, A e B se trataria dos 2 bits que se deseja somar. O Carry in (Cin) é o transporte que vem da soma feita anteriormente. Quando fizemos 1 + 1 em binário o resultado é 10, ao somar bit por bit de um valor, entretanto, a soma daqueles dois bits fica em 0, enquanto 1 é transportado para o próximo bit de maior valor (Cout).

Quando se trata de soma entre números binários, na tabela verdade, nota-se que o valor final da soma que aparece no output é como se fosse o resultado de uma lógica XOR entre A, B e Carry in. Logo, um somador possui uma porta XOR para soma de A e B e depois o uma XOR do resultado dessa com o valor de Carry in que viria como Carry out de uma soma anterior. Caso não haja valor transportado da soma anterior, o Cin seria 0. Além da soma, faltaria calcular o Carry out, ou seja, aquilo que esta soma passaria como transporte para uma próxima fase. Para isso, é possível notar que a lógica por trás do Carry out na tabela verdade é o mesmo da porta AND, ou entre A e B, ou entre a soma deles com o Cin.



Captura de tela do somador no Logisim

Logo, é possível notar que para fazer uma soma de mais bits, usa-se um somador para cada dígito da soma, que seria o resultado do algarismo da mesma posição final. É preciso, entretanto, considerar quando a soma de um destes algarismos gera algum transporte. A lógica por trás de um somador de mais bits está aí, já que o Carry out da soma de um algarismo entra como Carry in da soma do próximo algarismo (de valor mais alto). Caso o Carry out dos MSB dos números a serem somados tenha 1 como resultado, o resultado se trata de um número de 5 bits, sendo esse número transportado o MSB da resposta.

1. *Implementação do somador no Tinkercad*

Após nossa equipe ter desenvolvido o circuito teórico no Logisim, começamos a parte de transporte do mesmo para o simulador Tinkercad.

Para recriar o circuito somador teórico no Tinkercad foram usados os seguintes componentes:

* Placa de ensaio;
* Fonte de energia;
* Interruptores DIP DPST de quatro chaves;
* Leds;
* Resistores;
* Portas quad XOR 74HC86;
* Portas quad AND 74HC08;
* Porta quad OR 74HC32.

Começamos a montagem ligando os fios da fonte de energia nas linhas de positivo e negativo da placa de ensaio. Em seguida, conectamos dois DIPs na placa de ensaio para podermos fazer a inserção dos números que irão ser somados.

Na parte de cima dos DIPs, ligamos cada uma das colunas das chaves dos DIPs na linha com corrente positiva, já na parte de baixo, utilizamos interruptores para conectar cada chave dos DIPs na linha com corrente negativa.

Inserimos todas as portas lógicas na placa de ensaio, formando uma linha ao lado dos DIPs. As portas foram colocadas na seguinte ordem: 74HC86, 74HC08, 74HC86, 74HC08, 74HC32.

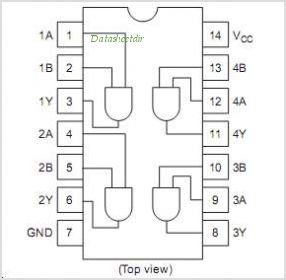
Conectamos as potências das portas na linha de corrente positiva e os solos na linha de corrente negativa. Após isso, iniciamos a montagem dos cabos no circuito para fazer as ligações.

Iniciamos a montagem com os bits menos significativos e fomos seguindo em ordem crescente de significância. Para a implementação correta na montagem dos cabos, foram utilizadas as datasheets de todas as portas lógicas a fim de entender o funcionamento de cada uma. A seguir, serão mostradas as datasheets dos 3 tipos de portas lógicas utilizadas nesse circuito.

Diagrama

Descrição gerada automaticamente

XOR 74HC86



AND 74HC08

Diagrama

Descrição gerada automaticamente

OR 74HC32

Começamos ligando a chave 4 de ambos os DIPs nas entradas 1 e 2 da porta XOR, respectivamente. Após isso, ligamos a saída dessa ligação, a qual é a porta 3, em outra porta de entrada XOR, que seria a porta 4. Então, adicionamos o terceiro DIP ao circuito, ligamos apenas a primeira chave. Esse DIP será usado como Cin no circuito. A parte de baixo da chave ligamos com a linha de corrente negativa e a parte de cima ligamos na outra entrada 5 da porta XOR.

A saída 6 da porta XOR vai será ligada em um resistor, que por sua vez será ligado com o led no terminal 2, a parte catódica do led será ligada na linha de corrente negativa. Esse primeiro led representa o bit menos significativo da soma.

Após isso, ligamos a saída 3 da porta XOR na entrada 1 da porta AND. Com isso, ligamos a primeira chave do DIP que estamos usando como Cin na entrada 2 da porta AND.

Ao mesmo tempo, ligamos as chaves 4 de ambos os DIPs nas entradas 4 e 5 da porta AND, então, as saídas 3 e 6 da porta AND são ligadas as entradas 1 e 2 da porta OR.

Prosseguimos para as chaves de número 3 dos DIPs. Ligamos elas nas entradas 13 e 12 da porta XOR, respectivamente. Em seguida, ligamos a saída 11 na entrada 10 da porta XOR. Na entrada 9, ligaremos a saída 3 da porta OR. Ligamos, então, a saída 8 da porta XOR à um resistor e ligamos o resistor no terminal 2 de um led. A entrada catódica do led é ligada a linha de corrente negativa, com isso, teremos nosso segundo bit.

Seguindo, ligamos a saída 11 da porta XOR na entrada 13 da porta AND e a saída 3 da porta OR na entrada 12 da porta AND. Ligamos também as chaves 3 de ambos os DIPs nas entradas 10 e 9 da porta AND, respectivamente. As saídas 11 e 8 das portas AND serão ligadas nas entradas 4 e 5 da porta OR.

Ligamos então as chaves 2 de ambos os DIPs nas entradas 1 e 2 da segunda porta XOR, ligamos a saída 3 da porta XOR na entrada 4 da mesma e, então, ligamos a saída6 da porta OR na entrada 5 da porta XOR. A saída 6 da porta XOR será ligada a um resistor que será ligado no terminal 2 de um led. A entrada catódica do led será ligada na linha de corrente negativa. Com isso, temos nosso terceiro bit.

Continuando o circuito, ligamos a saída 3 da porta XOR na entrada 1 da segunda porta AND e ligamos a saída 6 da porta OR na entrada 2 da porta AND. É feita também a ligação das chaves 2 de ambos os DIPs nas entradas 4 e 5 da porta AND. As saídas 3 e 6 da porta AND serão ligadas nas entradas 13 e 12 da porta OR, respectivamente.

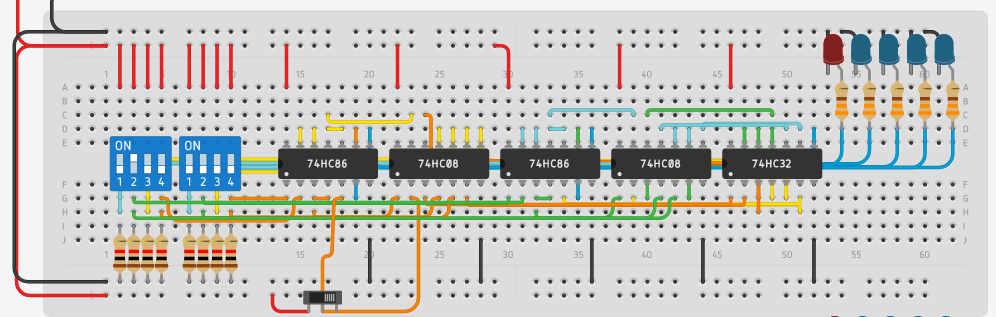
Por fim, ligamos as chaves 1 de ambos os DIPs nas entradas 13 e 12 da porta XOR, respectivamente e, em seguida, ligamos a saída 11 da porta XOR à entrada 10 da mesma.

Ligamos então a saída 11 da porta OR na entrada 9 da porta XOR, a saída 8 da placa é ligada em um resistor que, por sua vez, é ligado no terminal 2 de um led. A entrada catódica do led é ligada na linha de corrente negativa e com isso temos o quarto e último bit da nossa soma.

Continuaremos o circuito a fim de gerar um led para o Cin, que será ativado quando a resposta da soma for maior do que quatro bits.

Para isso, ligaremos a saída 11 da placa XOR na entrada 13 da porta AND e ligaremos a saída 11 da porta OR na entrada 12 da porta AND. Após, ligamos as chaves 1 de ambos os DIPs nas entradas 10 e 9 da porta AND. As saídas 11 e 8 das portas AND serão ligadas nas entradas 10 e 9 da porta OR, respectivamente. A saída 8 da porta OR será ligada em um resistor que, por sua vez, será ligado no terminal 2 de um led. A entrada catódica do led será ligada na linha com corrente negativa, e com isso temos um bit representando o Carry Out do circuito.

Os leds que representam os resultados das somas são os leds azuis e o led que representa o Carry Out é o led vermelho. Após diversos testes, constatamos que o circuito somador está funcionando corretamente e tendo o desempenho esperado.



Captura de tela do somador no Tinkercad

## Subtrator

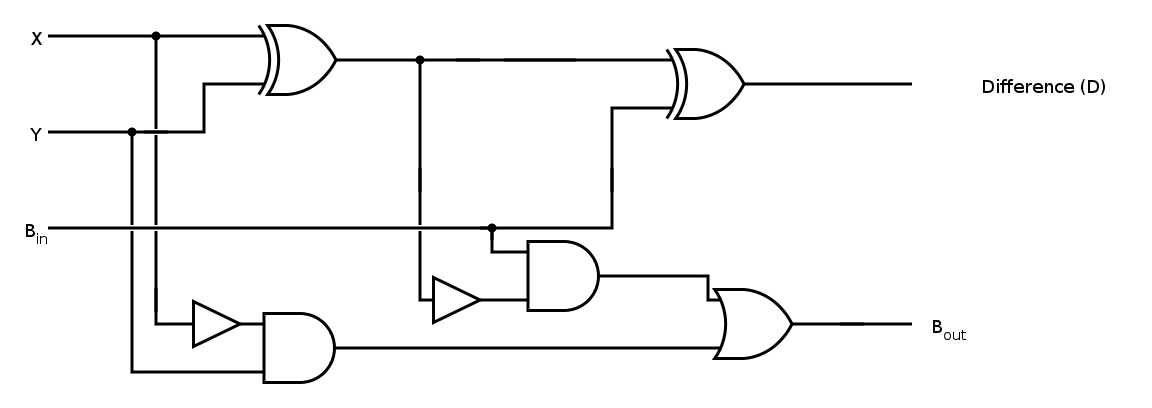
1. *Implementação do subtrator no Logisim*

Um subtrator nos retorna a diferença entre dois números binários. Para o trabalho, foram feitos subtratores de 4 bits, ou seja, que podem mostrar a diferença de números binários de até 4 dígitos.

Uma imagem contendo Tabela

Descrição gerada automaticamente

<https://www.geeksforgeeks.org/full-subtractor-in-digital-logic/>

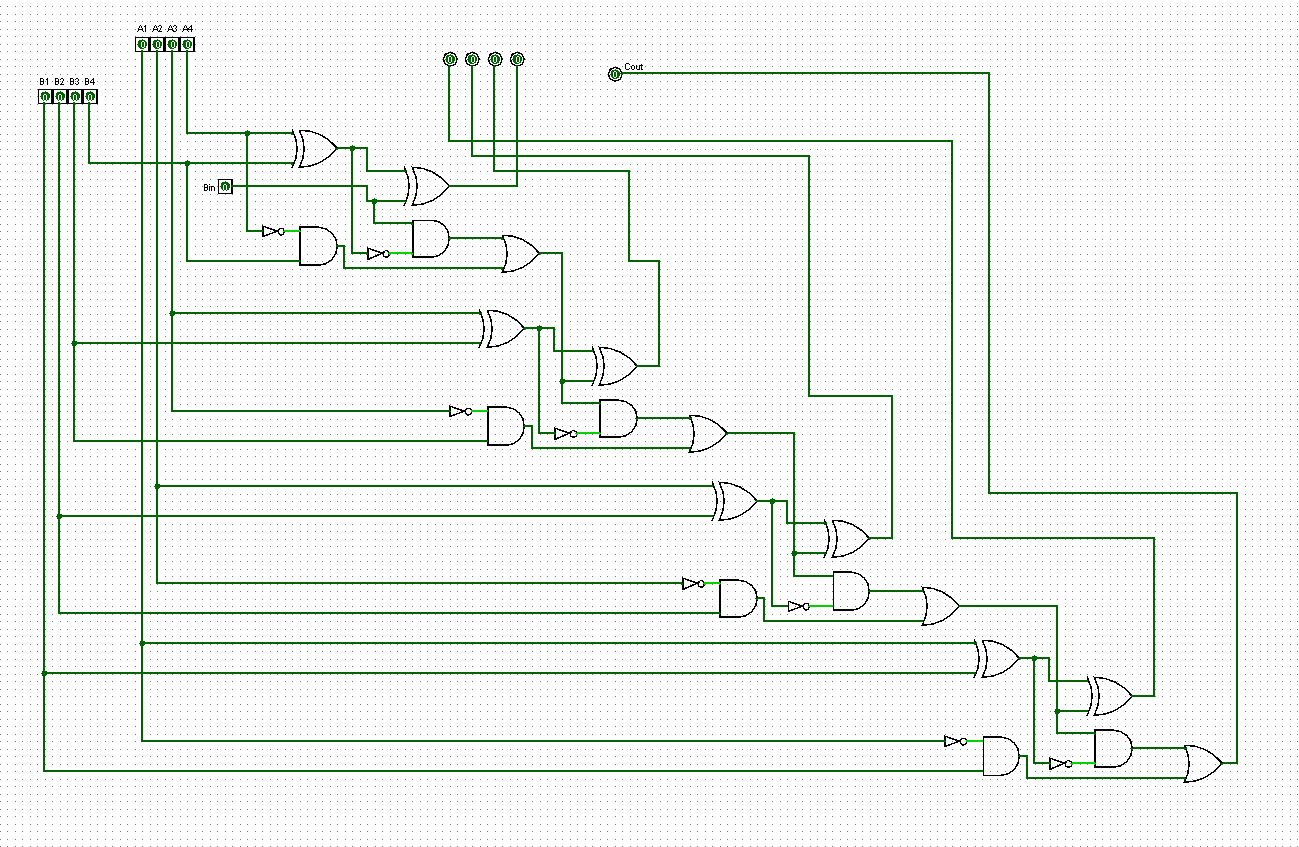


<https://commons.wikimedia.org/wiki/File:Full-sub-Fixed.svg>

O subtrator segue uma lógica bastante parecida com a do somador. A subtração ocorre entre um número A e B. Ademais, é preciso de outro valor relativo ao que se é emprestado e o que se empresta a um algarismo para realizar subtrações quando não é possível fazê-la (0 menos 1).

O valor daquilo que é emprestado para o número é chamado como Borrow out (Bout). Enquanto isso, o Borrow in (Bin) se trata daquilo que o algarismo empresta para outro. No caso da subtração, o Bout de um dígito é passado como Bin para o seu número de posição mais próxima e valor mais alto.

Ao se analisar na tabela verdade o valor relativo à diferença, é possível perceber que se trata da lógica XOR de A e B, seguida do uso do XOR deste resultado anterior com a entrada Bin. Enquanto isso, a saída do Bout é feita por placas AND para ou ver se A e B são 0 e 1 respectivamente, e uma placa and para ver se o valor da diferença entre os algarismos é 0 e o Bin é 1. Uma destas precisas ser 1 para que o subtrator tenha Bout com valor 1. Por fim, ressalta-se que para fazer a conexão entre as subtrações entre algarismos é preciso fazer com que o Bout de um subtrator seja o Bin do subtrator do dígito anterior, de maior posição.



Captura de tela do subtrator no Logisim

1. *Implementação do subtrator no Tinkercad*

Após a criação do circuito subtrator teórico no Logisim, começamos a implementação dele na plataforma simuladora Tinkercad.

Para recriarmos o circuito teórico no simulador Tinkercad, utilizamos os seguintes componentes:

* Placas de ensaio;
* Fonte de energia;
* Interruptores DIPs DPST de quatro chaves;
* Resistores;
* Leds;
* Portas XOR 74HC86;
* Portas inversoras hexadecimais 74HC04;

Para montar o circuito, começamos conectando os cabos de positivo e negativo da fonte de energia em seus respectivos lugares em cima da placa de ensaio. Depois, puxamos os cabos de positivo e negativo da parte de cima da placa e os ligamos com o positivo e o negativo da parte de baixo da placa.

Em seguida, fizemos a ligação entre a placa de ensaio e a placa de ensaio pequena, puxamos os cabos de positivo e negativo da parte de baixo da placa de ensaio e os ligamos com o positivo e negativo da placa de ensaio pequena. Após isso, levamos os cabos de positivo e negativo da parte de cima da placa de ensaio pequena e os ligamos com o positivo e negativo da parte de baixo da placa, ligando assim as duas placas.

Com isso pronto, colocamos os dois DIPs na placa de ensaio e na parte de cima deles, ligamos suas colunas na linha de corrente positiva, na parte de baixo, utilizamos 8 resistores (um para cada coluna) e os ligamos nas colunas dos DIPs e na linha de corrente negativa.

As portas XOR, inversor hexadecimal, AND, XOR, inversor hexadecimal e AND foram colocadas na placa de ensaio em linha, ao lado dos DIPs, nessa exata ordem. A porta OR foi colocada na placa de ensaio pequena.

Começamos, então, a montagem dos fios. Primeiramente ligamos os fios nas saídas da chave 4 de ambos os DIPs e os ligamos nas entradas 1 e 2 da porta XOR, respectivamente. Ligamos a saída 3 da porta XOR na entrada 4 da mesma.

Adicionamos mais um DIP para servir como entrada Cin. Só utilizaremos a primeira chave do DIP. Ligamos a parte de baixo da primeira chave na linha de corrente negativa da placa e, ligamos a parte de cima da chave na entrada 5 da porta XOR. Com isso, ligamos a saída 6 da porta XOR em um resistor e ligamos o resistor no terminal 2 de um led. A entrada catódica do led será ligada na linha de corrente negativa da placa. Com isso, temos o nosso led representando nosso bit menos significativo.

Nosso próximo passo é ligar a saída 3 da porta XOR na entrada 1 do inversor e ligar a saída 2 do inversor na entrada 1 da porta AND. Liguemos a chave 1 do DIP de entrada Cin na entrada 2 da porta AND. Na saída 3 da porta AND ligaremos na entrada 1 da porta OR.

Vamos pegar a chave 4 do segundo DIP e ligá-la na entrada 3 do inversor. Pegaremos a saída 4 do inversor e ligaremos na entrada 4 da porta AND. A chave 4 do primeiro DIP será ligada direto na entrada 5 da porta AND. A saída 6 da porta AND será ligada na entrada 2 da porta OR.

A seguir, ligamos as chaves 3 de ambos os DIPs nas entradas 13 e 12 da porta XOR. A saída 11 da porta XOR será ligada na entrada 10 da mesma e a saída 3 da porta OR será ligada na entrada 9 da porta XOR. Ligamos a saída 8 da porta XOR em um resistor e ligamos o resistor no terminal 2 de um led. A entrada catódica do led será ligada na linha de corrente negativa. Assim, temos o nosso segundo led.

Continuamos ligando a saída 3 da porta OR na entrada 13 da porta AND, ligamos a saída 11 da porta XOR na entrada 5 do inversor e a saída 6 do inversor ligaremos na entrada 12 da porta AND.

A chave 3 do segundo DIP ligaremos na entrada 13 do inversor e na saída 12 do inversor ligaremos na entrada 10 da porta AND. A chave 3 do primeiro DIP ligaremos direto na entrada 9 da porta AND. Ligaremos as saídas 11 e 8 da porta AND nas entradas 4 e 5 da porta OR, respectivamente.

Seguimos ligando as chaves 2 de ambos os DIPs nas portas 1 e 2 do segundo XOR, a saída 3 da porta XOR ligaremos na entrada 4 da mesma. Ligaremos a saída 6 da porta OR na entrada 5 da porta XOR. A saída 6 da porta XOR será ligada em um resistor que será ligado ao terminal 2 do led. A entrada catódica do led será ligada na linha de corrente negativa e, com isso, temos nosso terceiro bit.

A saída 6 da porta XOR deverá ser ligada na entrada 1 da segunda placa AND, a saída 3 da placa XOR deverá ser ligada na entrada 11 do inversor e a saída 10 do inversor será ligada na entrada 2 da porta AND.

Após isso, a chave 2 do segundo DIP será ligada na entrada 9 do inversor e a saída 8 do inversor será ligada na entrada 4 da porta AND. A chave 2 do primeiro DIP será ligada diretamente na entrada 5 da porta AND.

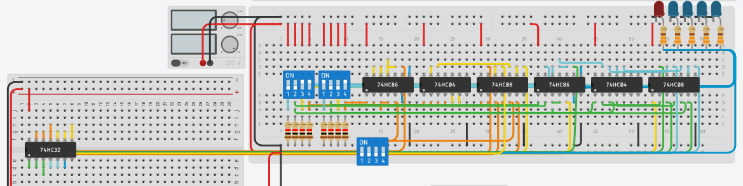
As saídas 3 e 6 da porta AND serão ligadas nas entradas 13 e 12 da porta OR.

Seguindo, ligaremos ambas as chaves 1 dos DIPs nas entradas 13 e 12 da porta XOR, respectivamente.  Ligaremos a saída 11 da porta XOR na entrada 10 da mesma. A saída 11 da porta OR será ligada na entrada 9 da porta XOR. A saída 8 da porta XOR será ligada em um resistor e o resistor será ligado no terminal 2 de um led. A entrada catódica do led será ligada na linha de corrente negativa. Com isso temos o nosso quarto bit pronto.

Seguimos no circuito para implementar o Cout, que seria o bit de saída. Para isso, ligamos a saída 11 da porta OR na entrada 13 da porta AND e, ligaremos a saída 11 da porta XOR na entrada 1 do segundo inversor. A saída 2 do inversor será ligada na entrada 12 da porta AND.

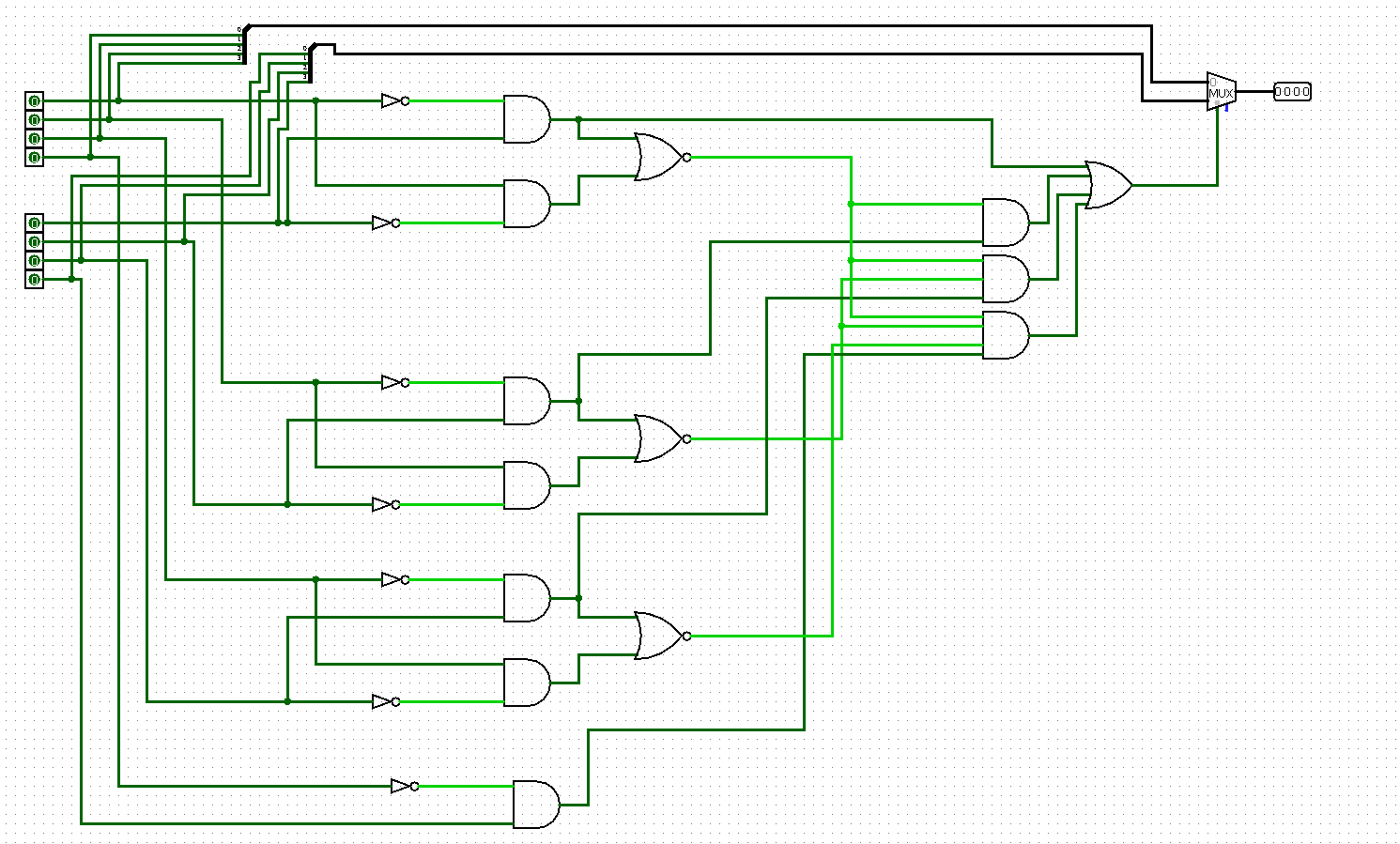
Na sequência, ligaremos a chave 1 do segundo DIP na entrada 3 do inversor e a saída 4 do inversor será ligada na entrada 10 da porta AND. A chave 1 do primeiro DIP será ligada diretamente na entrada 9 da porta AND. As saídas 11 e 8 da porta AND serão ligadas diretamente nas entradas 10 e 9 da porta OR. A saída 8 da porta OR será ligada num resistor, que por sua vez, será ligado no terminal 2 do led. A entrada catódica será ligada na linha de corrente negativa.

Com todas essas implementações temos o subtrator de 4 bits funcionando com a saída carry out.



Captura de tela do subtrator no Tinkercad

## Maior e menor



Captura de tela do Logisim

Para o maior e menor valor, conseguimos realizar apenas a implementação do circuito no Logisim.

Para encontrar o maior valor, se utilizou duas portas AND para os 3 maiores algarismos. Uma que retorna 1 caso A seja maior e outra caso B, seja maior, a saída das duas também leva para uma placa NOR, que retorna 1 se as duas forem iguais. Para ativar a saída que demonstra que B é maior, há uma placa OR que retorna 1 se: o MSB de B for o maior, o MSB for igual e o segundo byte de B for maior, o MSB e o segundo byte serem iguais e o terceiro byte de B for maior e por fim se os 3 primeiros bytes forem iguais e o 4 bytes de B for maior. Essa saída OR condiciona um multiplexador, que passa o valor de B caso o valor da saída OR seja 1, caso contrário (o, A for maior ou de igualdade) retorna o valor de A.

Para ver se 1 byte é maior que o outro, se utiliza a uma placa NOT de um byte com o outro, que, no caso do B, retorna 1 se B for 1 e A for 0.

## Registradores A e B no Logisim

Os registradores servem para salvar, registrar certo número binário determinado. Na implementação dos registradores, tanto A quanto B, utilizamos pouco circuito, por ser um circuito mais simples.

Na implementação do Logisim utilizamos apenas quatro entradas, no registrador A as entradas eram A0, A1, A2 e A3. Além das quatro entradas foram utilizados um Flip-Flop do tipo D para cada bit de entrada. E por fim, utilizamos um Distribuidor para pegar as quatro saídas de dados e transformá-las em apenas uma.

Diagrama

Descrição gerada automaticamente

Para o registrador B foi feita a mesma lógica, apenas alterando as entradas para B0, B1, B2 e B3.

Diagrama

Descrição gerada automaticamente

## Operações lógicas

Para pensar as operações lógicas, basta selecionar os bits de posições semelhantes e colocar a porta lógica que desejas para compará-los. A resposta se trata do valor binário da saída da posição do bit correspondente ao resultado da operação lógica.

Diagrama, Esquemático

Descrição gerada automaticamente

Operação AND no Logisim

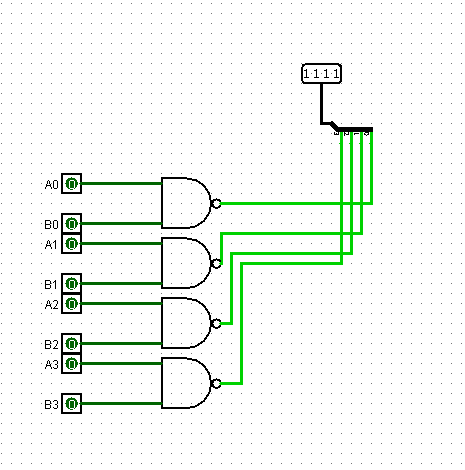


Figura 1Operação NAND no Logisim

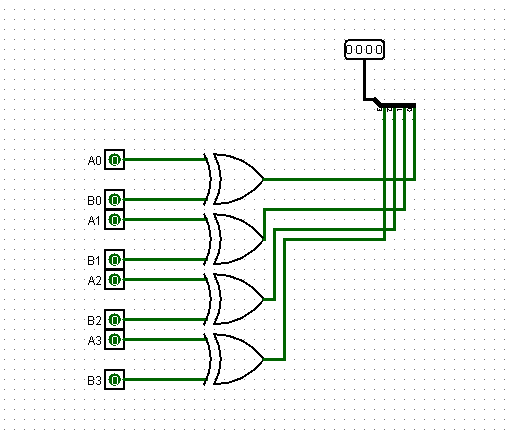


Figura 2Operação XOR no Logisim

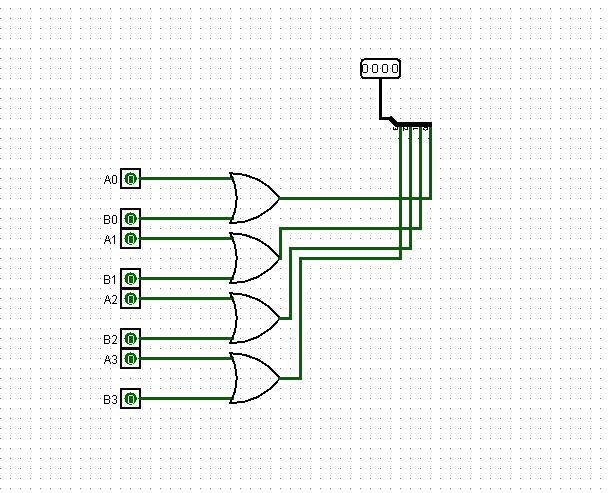


Figura 3Operação OR no Logisim

## Multiplexador

Um multiplexador, ou MUX, é um circuito eletrônico responsável por selecionar um sinal de entrada, para que ele esteja presente na saída. Geralmente, são utilizados em casos em que necessitamos levar um certo número de informações contidas em vários canais, a um só canal, como a nossa ULA.

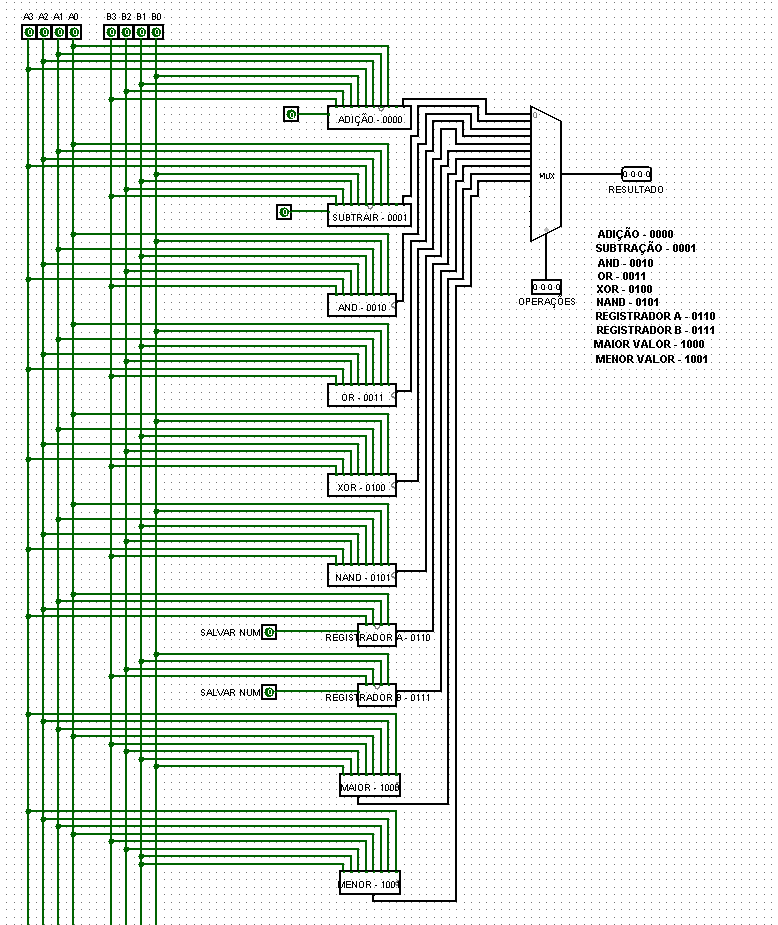
No nosso circuito ele será responsável por direcionar as informações da operação que selecionarmos para a saída.

Desenho de uma pessoa

Descrição gerada automaticamente com confiança média

https://embarcados.com.br/mux/

## ULA no Logisim



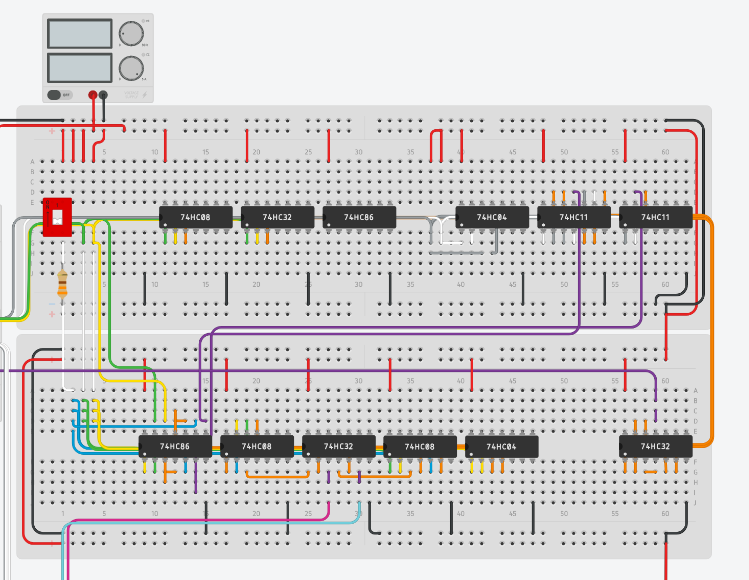
Captura de tela da ULA no Logisim

Para ULA no Logisim, foi juntado todas as operações feitas no Logisim. Por questões de espaço foram modificados para o circuito ficar mais compactado. Na saída de cada circuito se utilizou um distribuidor para que 4 bits separados da resposta fossem armazenados em conjunto. A saída desses circuitos foi para um multiplexador, pelo qual, através do número colocado, pode-se escolher qual tipo de operação se deseja realizar, para que a saída do MUX seja seu resultado.

## ULA no Tinkercad

Para a ULA no Tinkercad, não conseguimos implementar todas as operações que realizamos no Logisim. No Tinkercad aplicamos somente as operações básicas e necessárias, devido à dificuldade de implementação.

Para implementar a unidade lógica aritmética no Tinkercad, diferentemente do Logisim, fizemos ULAs de 1 bit com um MUX de 4 operações, e multiplicamos 4 vezes, para assim conseguirmos o funcionamento completo da ULA.



ULA 1 bit

Diagrama

Descrição gerada automaticamente

ULA com 4 operações pronta

A conexão entre as ULAs foi feita da seguinte forma, a primeira ULA recebe o Cin por meio de um switch, já as outras ULAs recebem como Cin o Cout da ULA anterior.

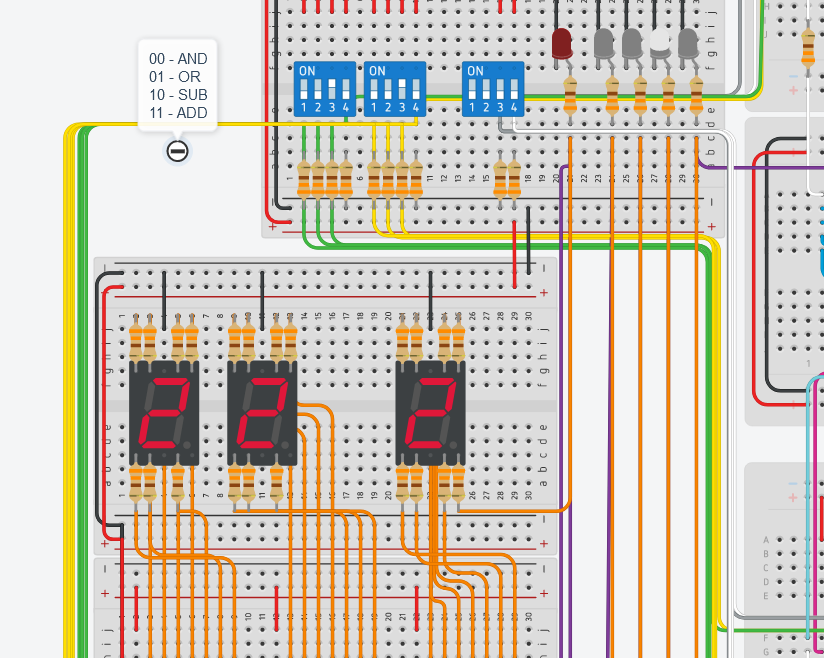
Como escolhemos realizar ULAs de 1 bit para o Tinkercad, todas as operações, consequentemente também foram de 1 bit. A lógica das operações mostradas anteriormente e as de 1 bit são a mesma, apenas reduzindo a quantidade de bits utilizadas.

Para escolher as operações adicionamos um DIP switch, onde 00 será a operação AND, 01 será OR, 10 a subtração e 11 a adição.

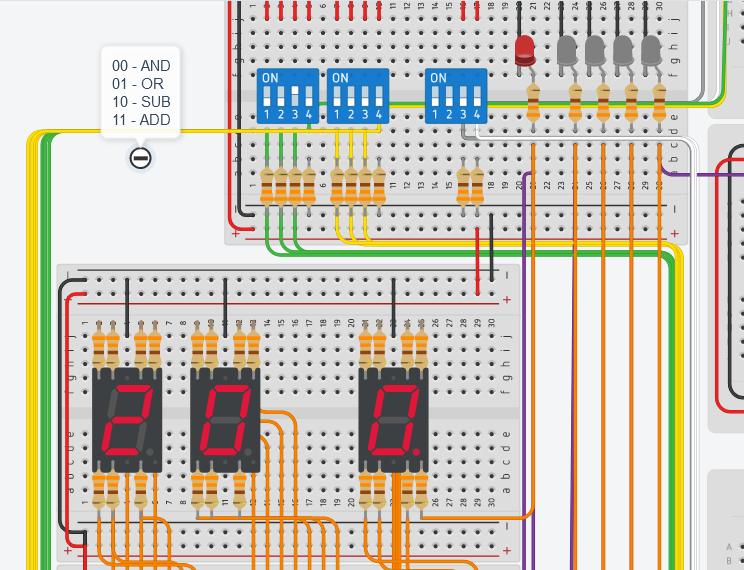
Os números são selecionados de forma binária por 2 DIP switches, um deles controla as entradas A e o outro controla as entradas B. Ambos São conectados à visores de 7 segmentos para mostrar o número selecionado.

Em sequência mostraremos imagens da ULA em funcionamento. O visor da direita é referente às entradas A, ou seja, as operações são feitas a partir dela (subtração ocorre da direta para a esquerda nos visores, por exemplo). O ponto do visor serve para mostrar quando o Cout estiver ativo.

### AND

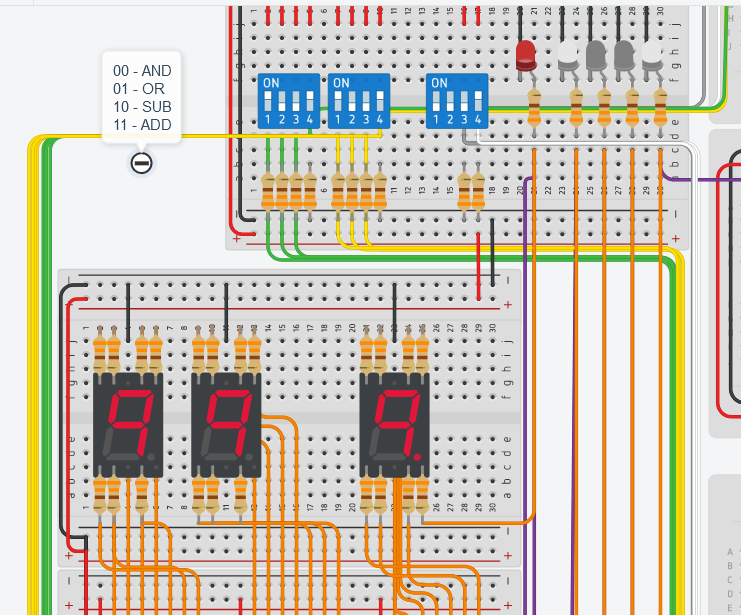


Captura de tela da ULA na operação AND

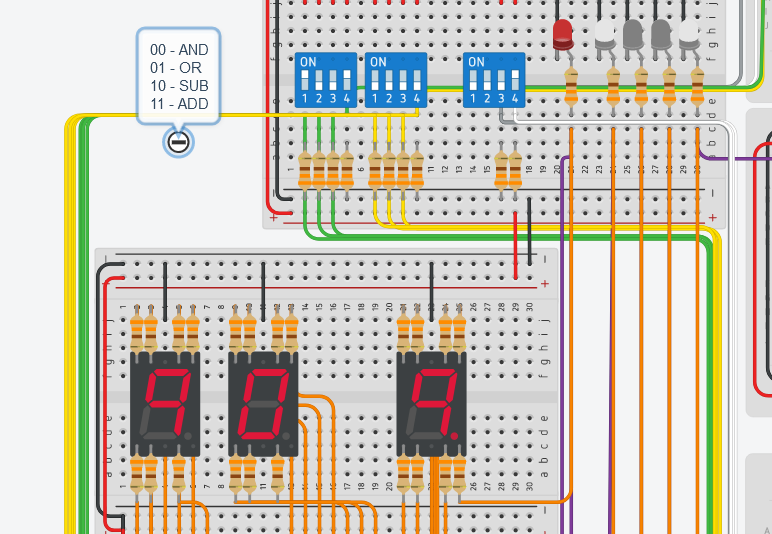


Captura de tela da ULA na operação AND

### OR

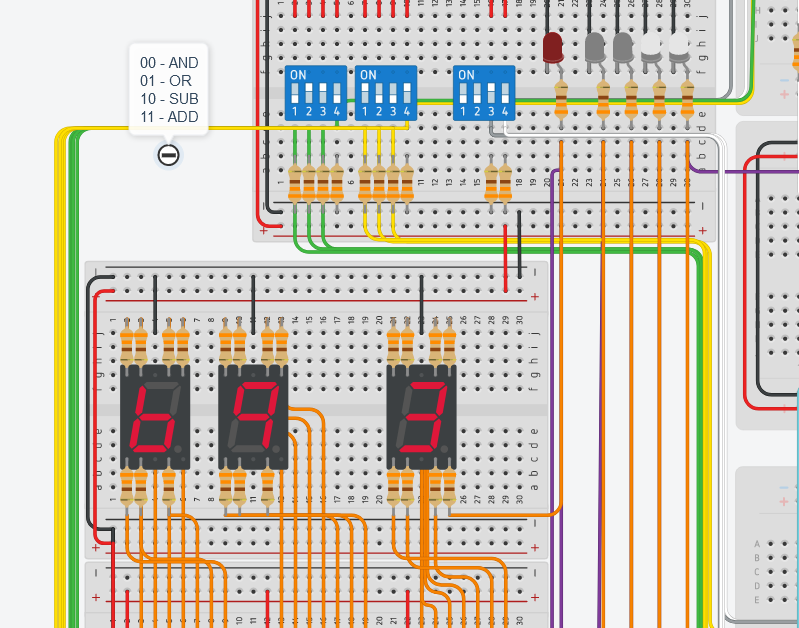


Captura de tela da ULA na operação OR



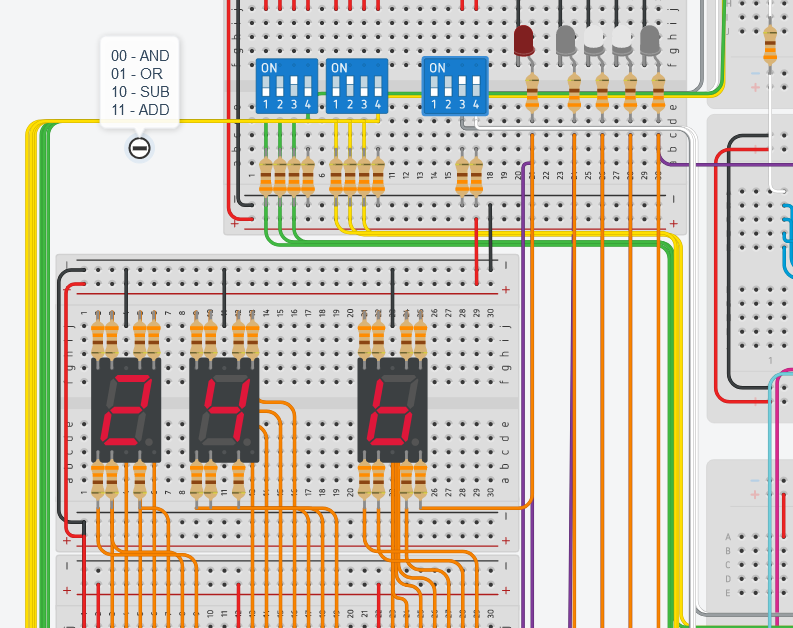
Captura de tela da ULA na operação OR

### Subtração



Captura de tela da ULA na operação de subtração

### Adição



Captura de tela a ULA na operação de adição

##### Referências

1. MULTIPLEXER Digital Electronics. [*S. l.*], 2021. Disponível em: https://www.javatpoint.com/multiplexer-digital-electronics. Acesso em: 22 nov. 2022.
2. 1 Bit Full Adder using Logic Gates. [*S. l.*], 2021. Disponível em: https://www.iitg.ac.in/cseweb/vlab/Digital-System Lab/fa\_lg.php?id=9. Acesso em: 22 nov. 2022.
3. 1 Bit Full Adder using Logic Gates. [*S. l.*], 2021. Disponível em: https://lacriptadelhacker.wordpress.com/2019/12/06/montando-una-alu-con-logisim/. Acesso em: 22 nov. 2022.
4. ROMBAUTS, Karl. **Building an 8-bit computer in Logisim (Part 2— Arithmetic)**. [*S. l.*], 2020. Disponível em: https://medium.com/@karlrombauts/building-an-8-bit-computer-in-logisim-part-2-arithmetic-ae7861c82e79. Acesso em: 22 nov. 2022.
5. CIRCUITOS Lógicos. [S. l.], 1 jun. 2011. Disponível em: https://www.ufjf.br/daniel\_silveira/files/2011/06/aula\_101.pdf. Acesso em: 22 nov. 2022.
6. INTRODUÇÃO a Multiplexador. [S. l.], 3 out. 2015. Disponível em: https://embarcados.com.br/mux/. Acesso em: 22 nov. 2022.
7. ULA. [S. l.], 1 fev. 2010. Disponível em: https://www.eletrica.ufpr.br/ogouveia/te130/files/ULA.pdf. Acesso em: 22 nov. 2022.
8. CIRCUITOS Aritméticos. [S. l.], 1 fev. 2011. Disponível em: https://www.cin.ufpe.br/~voo/sd/Aula5.pdf. Acesso em: 22 nov. 2022.
9. CIRCUITOS Multiplicador. [S. l.], 5 abr. 2010. Disponível em: https://www.cin.ufpe.br/~if675ec/arquivos/aulas/2003-2/unidade1/aula13\_CircuitoAritmetico\_Multiplicador.pdf. Acesso em: 22 nov. 2022.