

UNIVERSIDADE FEDERAL DE MATO GROSSO

CAMPUS UNIVERSITÁRIO DO ARAGUAIA

Instituto de Ciências Exatas e da Terra

*Curso de Bacharelado em Ciência da Computação*

**Relatório do trabalho da**

**disciplina de Lógica Digital 2023/1**

Dicente: Anna Bheatryz Martins Santos;

Fernanda Lima de Souza

Docente:Prof. Dr. Linder Cândido da Silva

UNIVERSIDADE FEDERAL DE MATO GROSSO

CAMPUS UNIVERSITÁRIO DO ARAGUAIA

Instituto de Ciências Exatas e da Terra

*Curso de Bacharelado em Ciência da Computação*

Anna Bheatryz Martins Dos Santos

Fernanda Lima De Souza

Projeto: Trabalho de **Lógica Digital.**

O objetivo do projeto é realizar a parte

prática das aulas fornecidas

teoricamente em sala de aula,

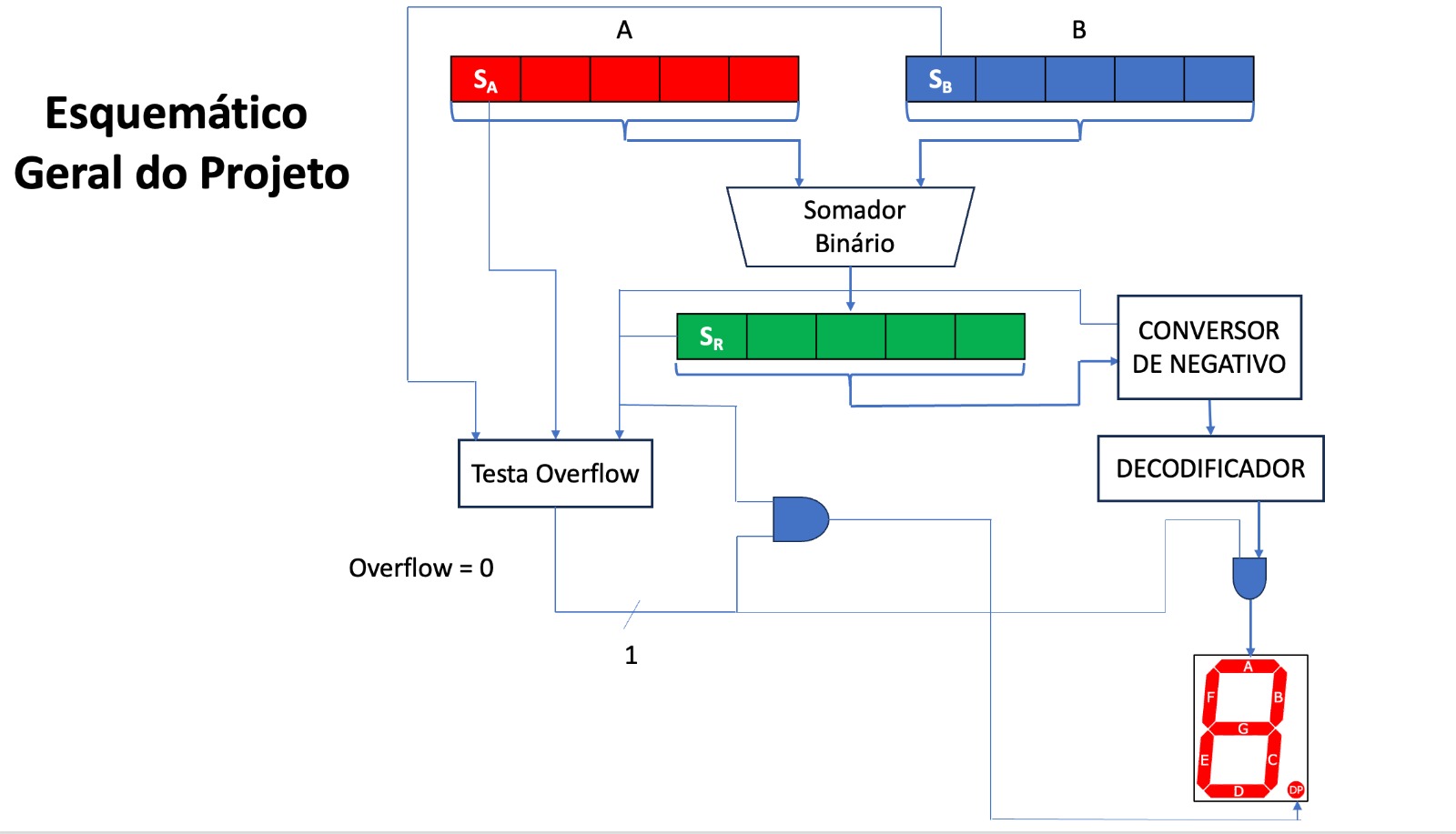
projetando circuitos digitais combinacionais.

Barra do Garças/ MT

2023

1. Introdução...................................................................................................
2. Somador Binário.........................................................................................
3. Verificador de Overflow...............................................................................
4. Conversor de negativo................................................................................
5. Decodificador..............................................................................................
6. Integração dos Componentes ....................................................................

Introdução:



# 1° Passo - Somador Binário

Um somador binário completo de 5 bits é um circuito lógico que realiza a adição de dois números binários de 5 bits, incluindo a consideração do transporte de um bit para o próximo estágio. Detalhes de funcionamento:

Entradas:

* Dois números binários de 5 bits, A e B.
* Um bit de entrada de transporte, Cin (Carry-in), que é inicialmente 0 ou 1, dependendo das operações anteriores.

Saídas:

* A soma binária de A e B, representada como S (uma saída de 5 bits).
* Um bit de transporte de saída, Cout (Carry-out), que é transferido para o próximo somador binário, se existir.

Funcionamento:

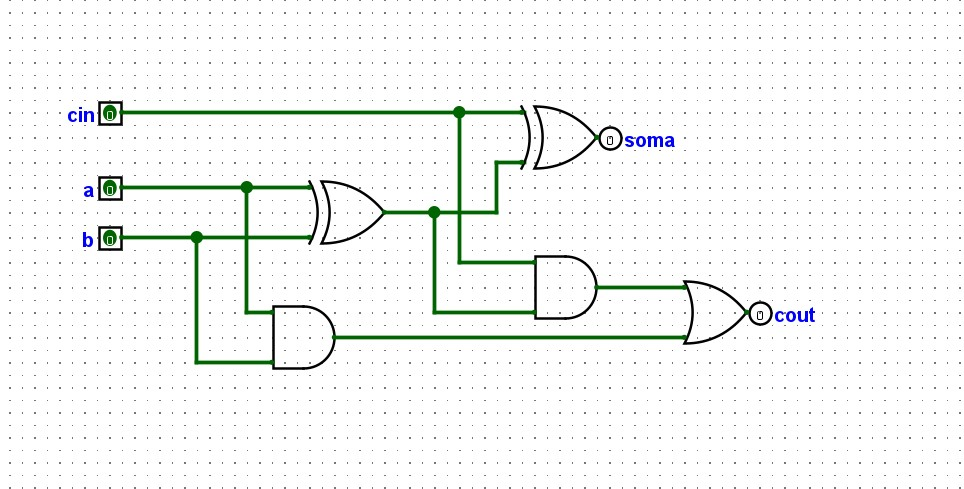
O somador de 5 bits é composto por cinco estágios, onde cada estágio somará um par de bits correspondentes de A e B, juntamente com o transporte de entrada (Cin) do estágio anterior. O somador de um único estágio é chamado de somador completo de 1 bit, que foi implementado usando portas XOR, AND e OR.

A saída do primeiro estágio (menos significativo) é a primeira posição de S, e o Cout desse estágio é o Cin do próximo estágio. Esse processo se repete para cada par de bits correspondentes, gerando assim a saída completa de S (um número binário de 5 bits) e o Cout final (Carry-out).

Portas Utilizadas:

Cada estágio do somador completo de 1 bit requer 2 portas XOR, 2 portas AND e 1 porta OR. Para um somador binário completo de 5 bits, são necessários 5 desses estágios, resultando em 10 portas XOR, 10 portas AND e 5 portas OR, além das portas utilizadas para conectar as saídas dos estágios anteriores. O somador binário foi feito através de abstraído em “caixas”, a partir de 5 somadores de um bit, na qual foram conectados em cascata, com o carry de saída de cada somador completo interligado ao carry de entrada de cada somador completo da cadeia que vinha após o outro, cada somador completo foi criado a partir de um meio somador.

Portanto, um somador binário completo de 5 bits neste caso é composto por vários somadores completos de 1 bit conectados em cascata para realizar adições binárias de números de 5 bits.



Somador de 1 bit.

# 2° Passo - Verificador de Overflow

Para criar um verificador de overflow em um circuito combinacional que funcione com números em complemento de 2 a 5 bits, você pode seguir os passos indicados e utilizar portas XOR e um somador de 5 bits. Vou detalhar o processo passo a passo:

\*\*Passo 1: Inverter os bits usando portas XOR\*\*

Primeiro, você precisa inverter todos os bits do número de entrada. Para fazer isso, você pode usar portas XOR. O número de entrada tem 5 bits, então você precisará de 5 portas XOR para inverter cada bit individualmente.

Suponhamos que o número de entrada seja A[4:0], onde A[4] é o bit mais significativo e A[0] é o bit menos significativo. As portas XOR funcionam da seguinte forma:

- A[4] é diretamente conectado a uma porta XOR.

- A[3] é diretamente conectado a outra porta XOR.

- E assim por diante, até A[0] que é conectado a uma porta XOR.

O resultado de cada porta XOR será a inversão do bit correspondente. Portanto, você obterá o valor negativo em complemento de 2.

\*\*Passo 2: Adicionar 1 ao resultado usando um somador de 5 bits\*\*

Agora que você tem o número em complemento de 2, você precisa adicionar 1 a ele para converter de volta para o formato original. Você pode usar um somador de 5 bits para isso. Suponha que o número em complemento de 2 invertido seja B[4:0]. Você precisa somar 1 a B.

Para isso, você pode usar um somador de 5 bits, onde a entrada A é B e a entrada B é o número 1 em formato binário (00001).

A saída do somador fornecerá o resultado final, que estará no formato original, e agora você pode verificar o overflow.

\*\*Passo 3: Verificar o overflow\*\*

Para verificar o overflow, você precisa comparar os bits mais significativos do número de entrada original (antes da inversão) e o resultado final do somador. Se esses bits forem diferentes, houve um overflow.

Suponha que o número de entrada original seja C[4:0] e o resultado do somador seja D[4:0]. Para verificar o overflow, compare C[4] com D[4].

Se C[4] for igual a D[4], não houve overflow. Caso contrário, se C[4] for diferente de D[4], ocorreu um overflow.

Isso é tudo o que é necessário para criar um verificador de overflow para números em complemento de 2 de 5 bits. Certifique-se de que todas as portas XOR e o somador de 5 bits estejam corretamente interligados e que a comparação do bit mais significativo seja feita para verificar o overflow.

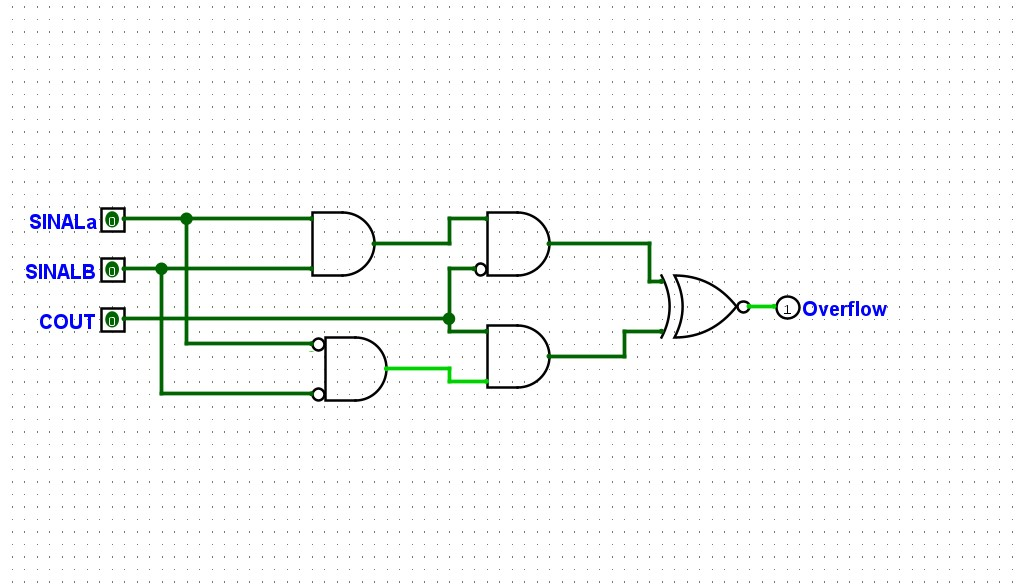


Tabela verdade do Over Flow:

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | COUT | OVERFLOW |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

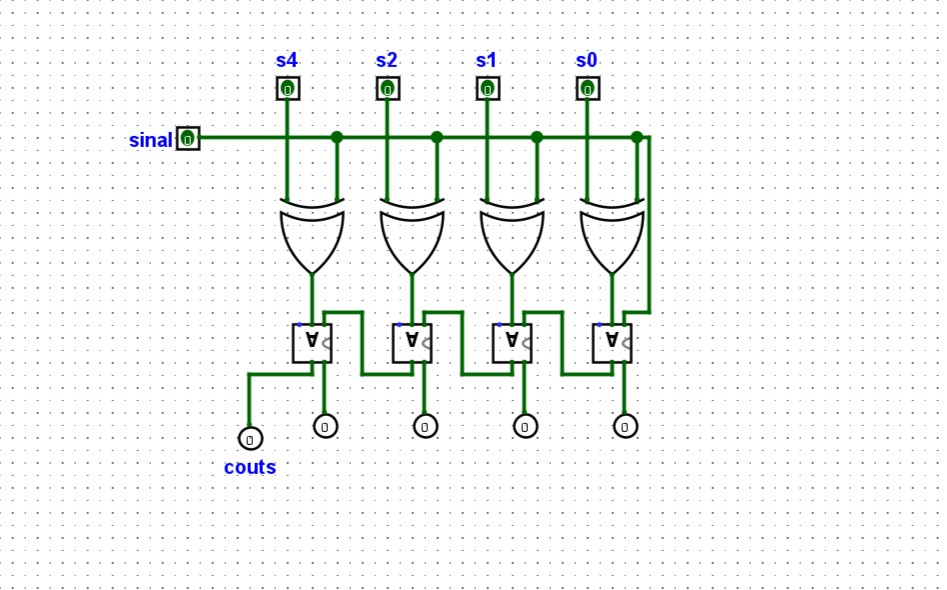
3° Passo - Conversor de negativo

O conversor de negativo é o circuito que realizará a tarefa de inverter um número em complemento de dois que está negativo — ou seja, que possui o bit mais significativo em 1. Dessa forma, função do conversor é fazer a troca do número negativo para o positivo, transformando 1's em 0's e vice-versa. Em seguida é somado 1 ao bit menos significativo do número invertido.

Entradas: Utilizamos os 4 bits de magnitude do resultado da soma, e o bit mais significativo como uma constante de comparação para inverter os bits.

Saídas: Quando realizamos a troca dos bits podemos prosseguir. Sendo assim, não vamos precisar mais do bit mais significativo pois sabemos que o resultado será 0, sendo necessário apenas os 4 bits de saída do Cin dos respectivos somatórios binários.

Portas utilizadas: Usamos 4 Portas XOR conectadas em uma das entradas dos 4 somadores binários, conectamos o bit mais significativo no Cin (carry in) do somador com os bits menos significativos, ligando o Cout (carry out) do mesmo, no Cin (carry in) do próximo somador.

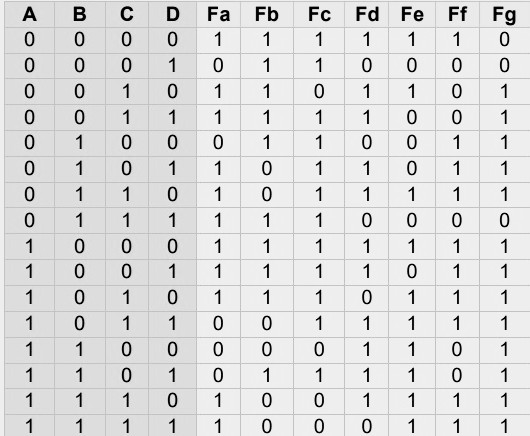


4° Passo-Decodificador Binário Para 7 Segmentos

O decodificador transforma o número binário utilizando o código BCD, para conseguirmos visualizar o resultado da soma do nosso display de 7 segmentos.

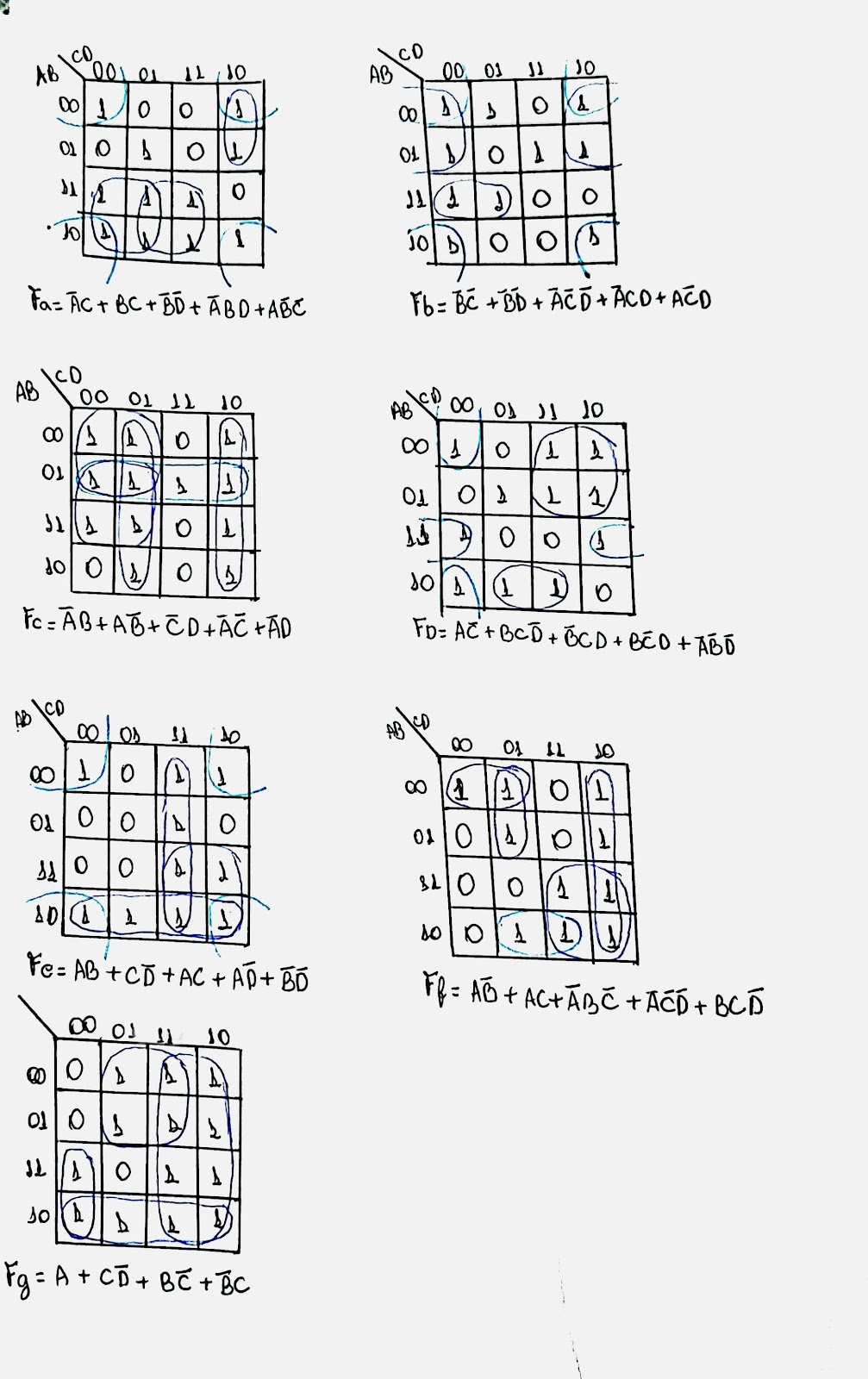
Construímos da seguinte forma:

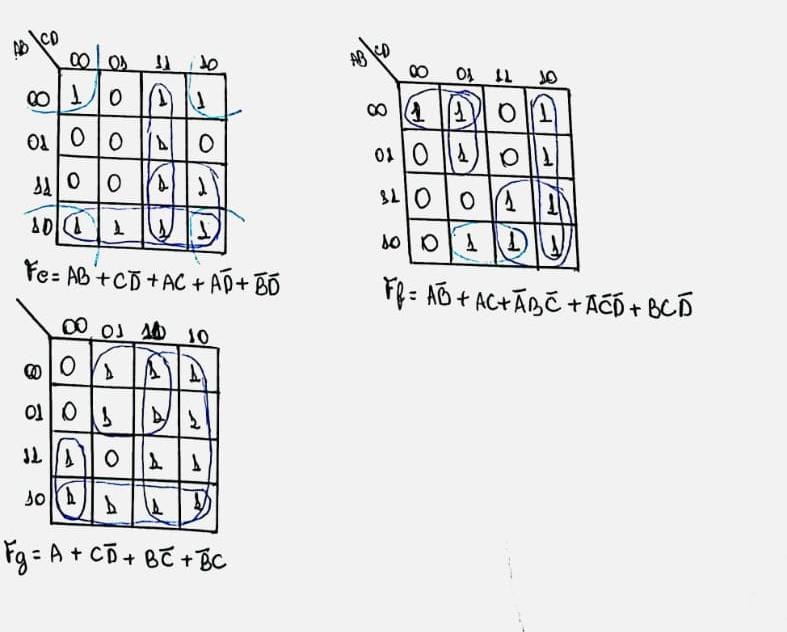
Passo 1: Montamos a tabela verdade:



Passo 2: Funções booleanas

Através da tabela verdade usamos o resultado para encontrar as nossas funções booleanas usando o mapa de karnaugh.





Passo 3: Montamos o circuito

Usamos 33 portas AND e 7 portas OR

Entradas: Usamos as saídas do conversor binário.

Saídas: Às 7 saídas foram conectadas em um circuito de portas AND para o display de 7 segmentos ser apagado se o resultado for maior que F(15), após ser conectados o circuito e ligado diretamente no display. Outro detalhe do display é o ponto decimal, que terá a função de nos mostrar se o resultado da soma será negativo se estiver ligado, se não o resultado é positivo.



Resultado de todos os circuitos integrados:

