

**UNIVERSIDADE FEDERAL DE RORAIMA**

**CENTRO DE CIÊNCIA E TECNOLOGIA - CCT**

**CURSO DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO**

**Vinícius Nolêto de Araújo e Alex Silva Costa**

**LABORATÓRIO DE CIRCUITOS**

BOA VISTA, RR

2022

**Vinícius Nolêto de Araújo e Alex Silva Costa**

**LABORATÓRIO DE CIRCUITOS**

Avaliação de Barramento de Circuitos Digitais, apresentado como requisito de obtenção de nota parcial da disciplina de Arquitetura e Organização de Computadores - DCC 301.

Orientador: Prof. Dr. Hebert Oliveira Rocha

BOA VISTA, RR

2022

**LISTA DE ILUSTRAÇÕES**

|  |  |  |
| --- | --- | --- |
| Figura 1 - | Figura1…………………………………………………...... | **11** |
| Figura 2 - | Figura2……………………………………………….......... | **11** |
| Figura 3 - | Figura3…………………………………………………...... | **11** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# 

# 

# 

# 

**LISTA DE TABELAS**

|  |  |  |
| --- | --- | --- |
| Tabela 1 - | Tabela1…………………………………………….. | 8 |
|  |  |  |

# 

# 

# 

# **SUMÁRIO**

**[SUMÁRIO](#_absqj4iefmp2) 5**

**[1. INTRODUÇÃO](#_o083ftjjbrar) 6**

**[2. COMPONENTES](#_b2s9ui9tvhvi) 5**

[2.1. REGISTRADORES FLIP FLOP D E JK](#_1oeqqr7htrtu) 5

[2.1.1. Descrição pinos e lógica](#_3p71wvcd5sq6) 5

[2.2. MULTIPLEXADOR 4 OPÇÕES DE ENTRADA](#_airg3fofun10) 5

[2.2.1. Descrição pinos e lógica](#_ecmulxww882y) 5

[2.3. PORTA LÓGICA DO XOR](#_1kc7h3wd8u85) 6

[2.3.1. Descrição pinos e lógica](#_h8n913mgw0qg) 6

[2.4. SOMADOR 8 BITS (de acordo com a descrição do projeto)](#_o0kmji7y9zec) 6

[2.4.1. Descrição pinos e lógica](#_vamlr0vj2d4b) 6

[2.5. MEMÓRIA ROM](#_o4r5tly0l223) 7

[2.5.1. Descrição pinos e lógica](#_jftv7h4x883t) 7

[2.6. MEMÓRIA RAM](#_6g58ww3xy3ss) 7

[2.6.1. Descrição pinos e lógica](#_njhape12c895) 7

[2.7. BANCO DE REGISTRADORES](#_9kliq0ochmh0) 7

[2.7.1. Descrição pinos e lógica](#_cipu8o7so39o) 7

[2.8. SOMADOR 8 BITS](#_y4r2l7f28f0u) 8

[2.8.1. Descrição pinos e lógica](#_abom61ko1ih3) 8

[2.9. UNIDADE DE CONTROLE 16 BITS](#_7inp4qphtbrk) 8

[2.9.1. Descrição pinos e lógica](#_8m5x7acf9px9) 8

[2.10. ULA 8 BITS](#_jxg026gz0cg2) 8

[2.10.1. Descrição pinos e lógica](#_7srkmcv6dwya) 9

[2.11. EXTENSOR DE SINAL](#_xcmyp1hmh4ns) 9

[2.11.1. Descrição pinos e lógica](#_ooxzt7my7g6z) 9

[2.12. MÁQUINA DE ESTADO](#_jx0nefj4w2gp) 9

[2.12.1. Descrição pinos e lógica](#_hoguxcczu0eg) 9

[2.13. CONTADOR SÍNCRONO](#_atnzh8xshx2v) 9

[2.13.1. Descrição pinos e lógica](#_7aeijkwduja) 10

**[3. CONSIDERAÇÕES FINAIS](#_kn1kc3jrpzfn) 10**

**[4. REFERÊNCIAS](#_z7rqvd2hfb0p) 11**

# **INTRODUÇÃO**

# 

# **COMPONENTES**

## *REGISTRADOR FLIP-FLOP* JK E *FLIP-FLOP* D

Figura 1 - *Flip-Flop* JK

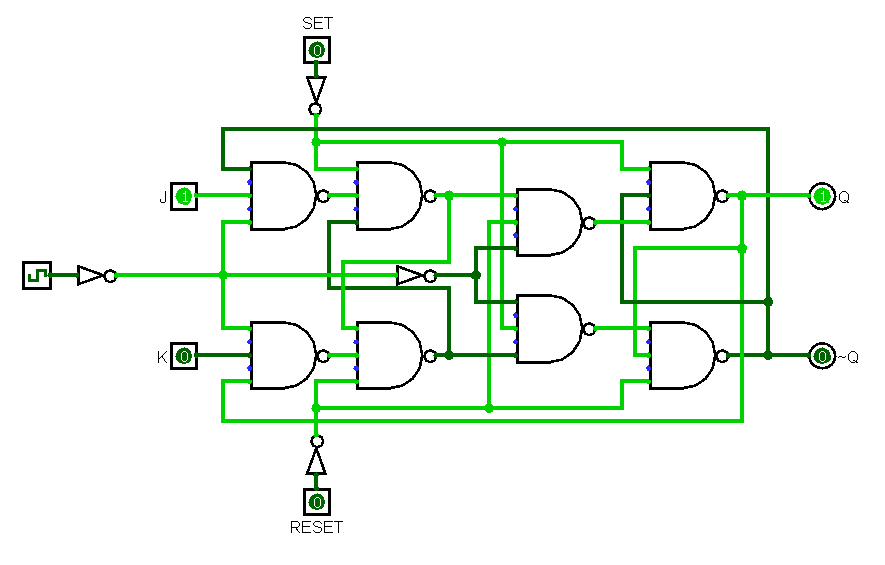
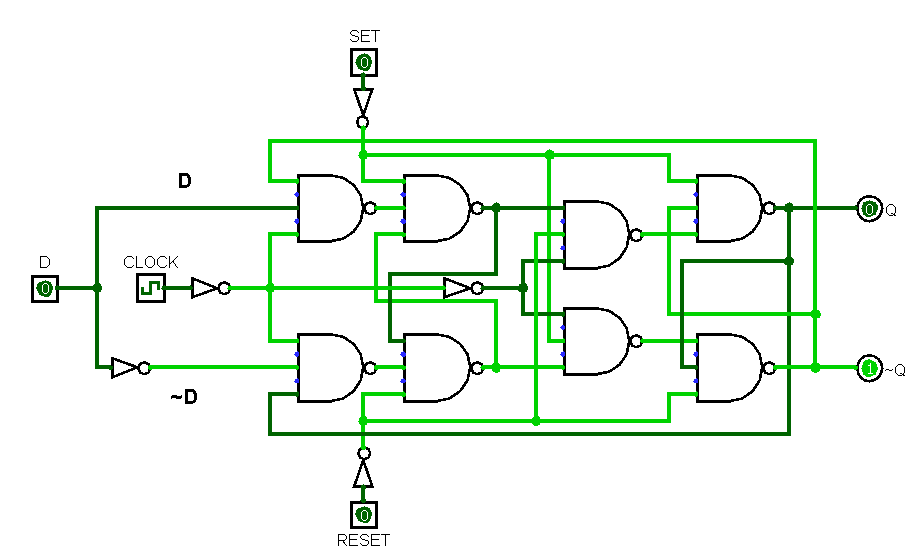


Figura 2 - *Flip-Flop* D

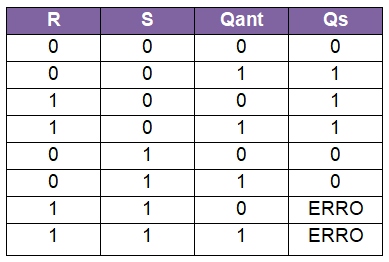


Os *Flip-Flops* são componentes da arquitetura de computadores que possuem em sua tabela-verdade uma característica destoante de outros componentes, a presença do último valor de saída efetuado pelo componente como próximo valor de saída em determinados casos, mesmo sem conhecimento do valor específico do mesmo, ou seja, desta forma é possível armazenar valores anteriores e trabalhar com os mesmos, podendo alterná-los, negá-los ou mantê-los para o funcionamento de demais circuitos.

### **Descrição pinos e lógica**

O *Flip-Flop* JK possui quatro entradas: J, K, *Preset* e *Clear*, fora o Clock, sendo as entradas *Preset* e *Clear* correspondentes às respectivas entradas *SET* e *RESET* na Figura 1, além disto possui dois valores de saída: Q e ~Q. Por outro lado o *Flip-Flop* D possui apenas três entradas D, *Preset* e *Clear*, fora o Clock, e, de forma parecida com o componente *Flip-Flop* JK desenvolvido, as entradas *Preset* e *Clear* são correspondentes às respectivas entradas *SET* e *RESET* na Figura 2, e também possui dois valores de saída: Q e ~Q. O modelo escolhido de *Flip-Flop* JK e *Flip-Flop D* foi o modelo Chefe-Servente, que divide o *Flip-Flop* em duas partes, como visto na Figura 3, onde existem dois *Flip-Flop* RS, uma versão mais simples de *Flip-Flop*, conectados, onde o Clock do *Flip-Flop* Servente é negado, impedindo o erro no caso de entradas 1 e 1, originalmente presente em *Flip-Flops* RS, como é demonstrado na Tabela 1.

Tabela 1 - Tabela-Verdade Flip-Flop RS Completa



* + 1. **Testes do componente**

Ao realizar os testes foi possível gerar as tabelas-verdade do circuito do *Flip-Flop* JK e do circuito do *Flip-Flop* D e, após isto, simplificá-lo, como pode ser visto nas tabelas Tabela 2, Tabela 3, Tabela 4 e Tabela 5, onde o valor Qant está relacionado ao valor de Q anterior e Qs está relacionado saída do valor de Q atual.

Tabela 2 - Tabela-Verdade Flip-Flop JK Completa

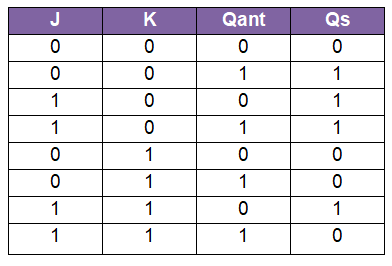


Tabela 3 - Tabela-Verdade Flip-Flop JK Simplificada

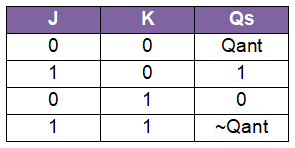


Tabela 4 - Tabela-Verdade Flip-Flop D Completa

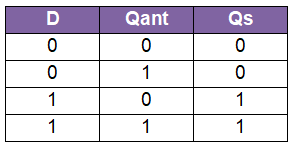
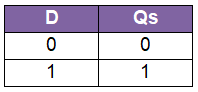


Tabela 5 - Tabela-Verdade Flip-Flop D Simplificada



## MULTIPLEXADOR

Figura 3 - Multiplexador

## Questão 2 - Multiplexador

O multiplexador é um componente extremamente comum em arquiteturas de computadores, pois é respondável por controlar o fluxo de dados em circuitos digitais.

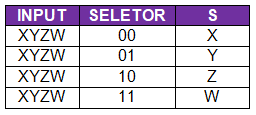
### **Descrição pinos e lógica**

O multiplexador possui duas entradas, *INPUT* e o SELETOR, possuindo, respectivamente 4 e 2 bits cada, além disto possui uma saída, S. O componetnte é responsável pelo desvio de fluxo em um circuito, ou seja, o valor de cada saída em um multiplexador será igual ao input apenas caso o valor do seletor desvie a corrente no sentido do mesmo.

* + 1. **Testes do componente**

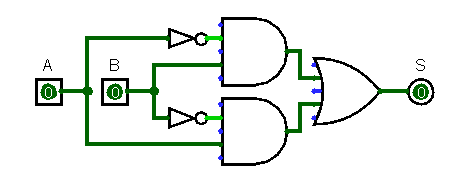
Ao realizar os teste com o multiplexador foi possível gerar um tabela-verdade como pode ser visto nas tabelas Tabela 6, sendo “X”, “Y”, “Z” e “W” os valores dos bits de INPUT, que variam de acordo com a entrada do dada pelo funcionamento do circuito.

Tabela 6 - Tabela-Verdade Multiplexador



## PORTA LÓGICA DO XOR

Figura 4 - *Xor*



O componente *Xor* é um componente comummente utilizado em diversas arquiteturas de circuitos digitais, pois o mesmo obriga que apenas um dos valores seja verdadeiro para que a saída também seja.

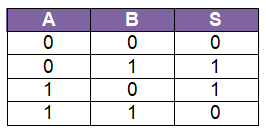
### **Descrição pinos e lógica**

O componente *Xor*, também conhecido como “ou exclusivo” funciona com a junção de dois componentes do tipo E, onde as entradas são negadas de forma alternada e seus resultados após o componente E são conectados à um único componente OU, desta forma, OU o primeiro valor (neste caso, o input A) é igual a 1 e o segundo valor (neste caso, o input B) é igual a 0, OU o primeiro valor é igual a 0 e o segundo valor é igual a 1, para que a saída final seja verdadeira.

* + 1. **Testes do componente**

A tabela verdade do componente *XOR* já era previamente conhecida, pois o mesmo é utilizado em diversas outras arquiteturas, como é demonstrado por [Metrópole Digital](https://materialpublic.imd.ufrn.br/), na publicação “[Circuitos](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4) *[Exclusive-OR](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4)* [(](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4)*[XOR](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4)*[) e](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4) *[Exclusive-NOR](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4)* [(](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4)*[XNOR](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4)*[)](https://materialpublic.imd.ufrn.br/curso/disciplina/2/17/7/4)”, e, após teste, foi possível obter os mesmos resultados, como é demonstrado na Tabela 8.

Tabela 7 - Tabela-Verdade XOR



## SOMADOR 8 BITS **(de acordo com a descrição do projeto)**

Figura 5 - Circuito Completo Somador 8 bits

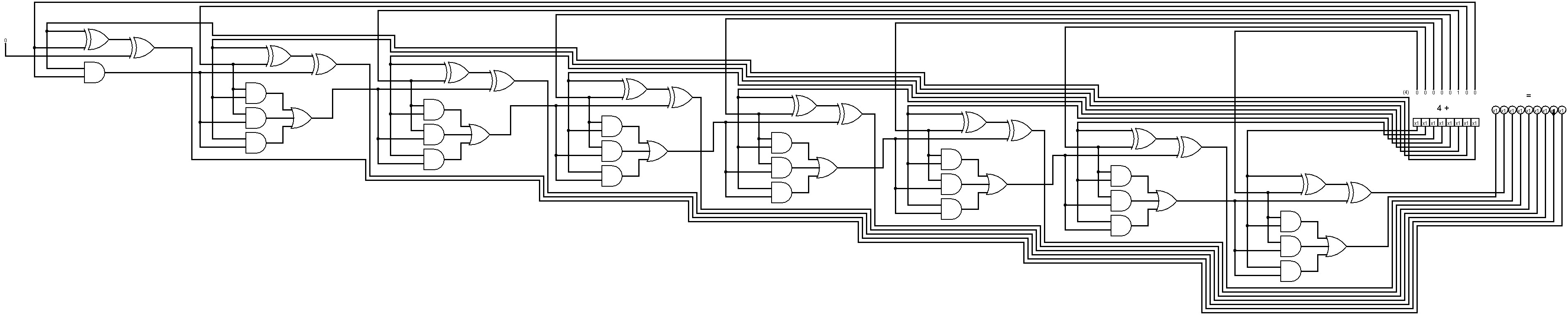
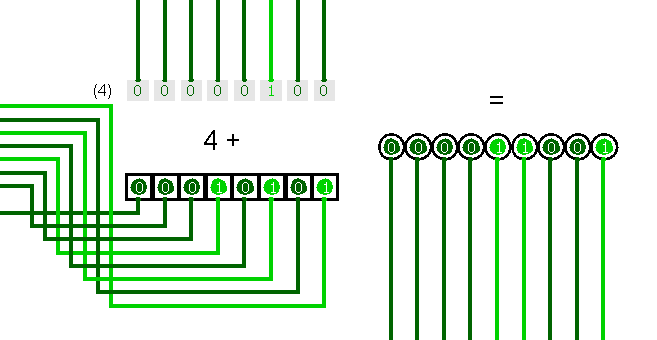


Figura 6 - Valores de Entrada Somador 8 bits



O Somador é um componente aritmético que soma valores binários utilizado para efetuar cálculos pelo computador, sendo que o seu tamanho varia de acordo com a quantidade de bits dos valores de entrada.

### **Descrição pinos e lógica**

O funcionamento do Somador de 8 bits pode ser entendido a partir do funcionamento de uma secção do mesmo, como pode ser visto na Figura 7, que possui três valores de entrada de 8 bits, A, B, e *Cin*, sendo A e B os valores de entrada atuais e *Cin* o valor excedente da soma da secção anterior, além disto possui duas saídas de 1 bit, S e Cout, onde o valor de S corresponde ao valor da soma e Cout corresponde ao valor excedente da soma atual. Neste circuito o componente *XOR 1* impede que soma de dois valores iguais a 1 provenientes de A e B resultem em 1, pois 1+1 em binário, diferente do sistema decimal comummente utilizado, resulta em 10, não em 2, pois o valor 2 não existe no sistema binário, da mesma forma o componente *XOR 2* impede que, caso o valor do *XOR 1* seja igual a 1 e o valor de *Cin* seja igual a 1, que o valor de S seja igual a 1. O Grupo de E determina o valor de *Cout*, de forma que caso o valor de A **e** B sejam iguais a 1, ou caso A **ou** B sejam iguais a 1 e *Cin* seja igual a 1, então o *Cout* será igual a 1, como pode observado na Tabela 9.

Figura 7 - Secção de Circuito Somador 8 bits

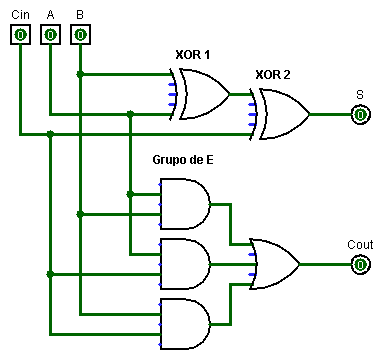
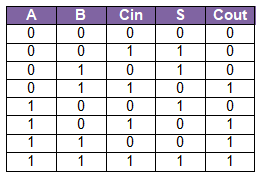


Tabela 9 - Tabela-Verdade Secção de Circuito Somador 8 bits



* + 1. **Testes do componente**

O circuito do Somador de 8bits é formado por diversas secções de somador, presente na Figura 7, encadeadas, como demonstrado na Figura 5, de forma que 8 valores de 1 bit de entrada (ou uma única entrada de 8 bits) e o valor fixo de 100 (que equivale ao valor 4 em binário), são somados, como demonstrado na Figura 6.

## MEMÓRIA ROM

Figura 8 - Circuito Completo Memoria ROM

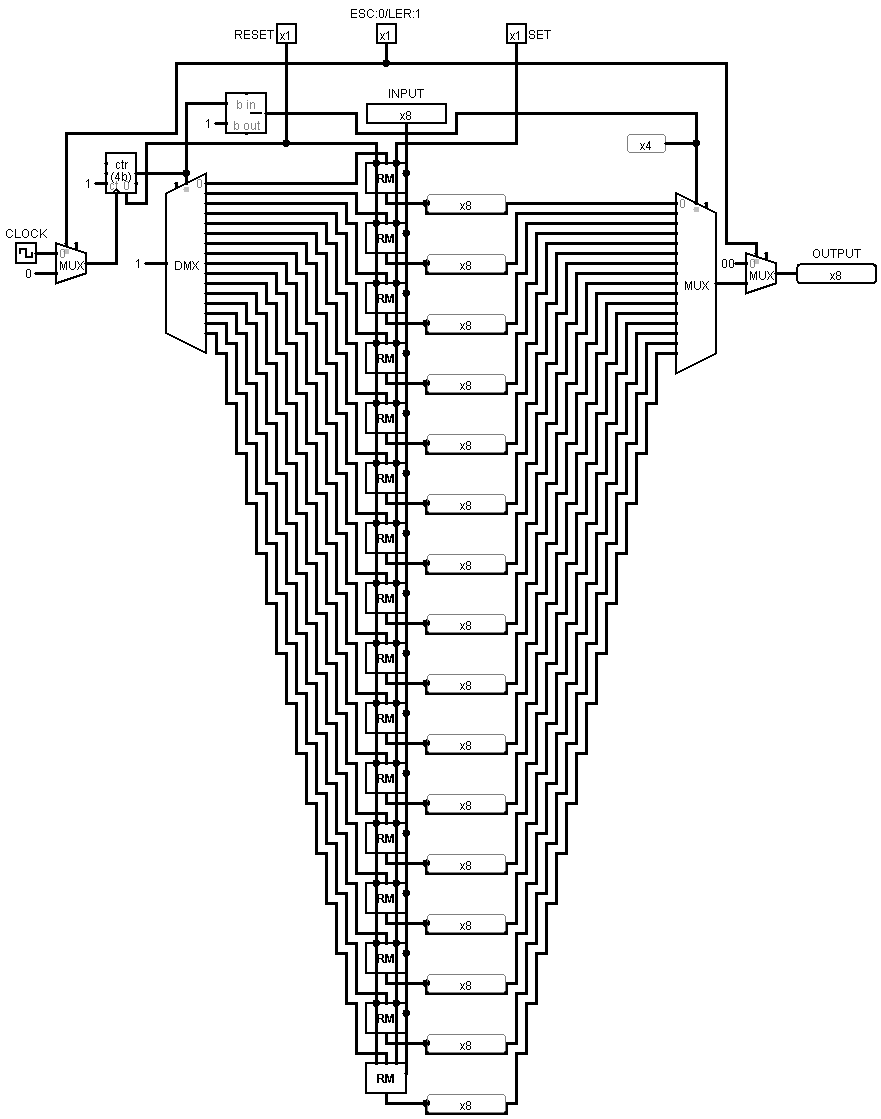


Figura 9 - Entradas e Saídas Memória ROM

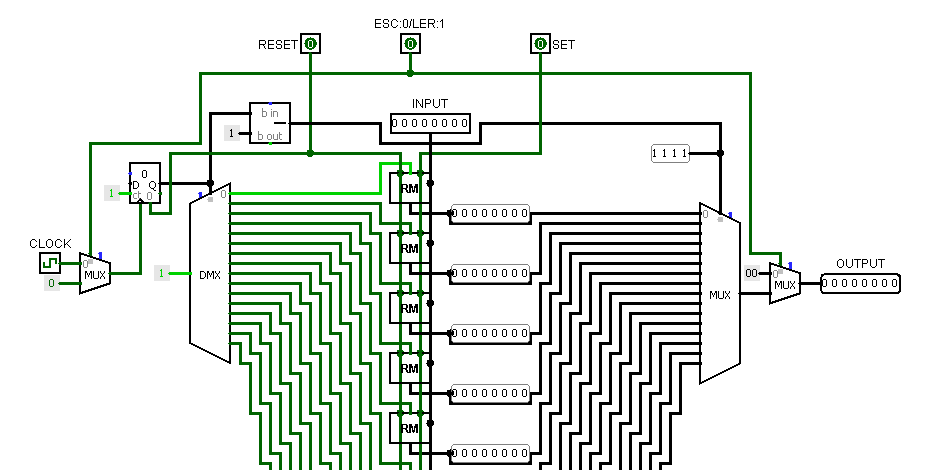
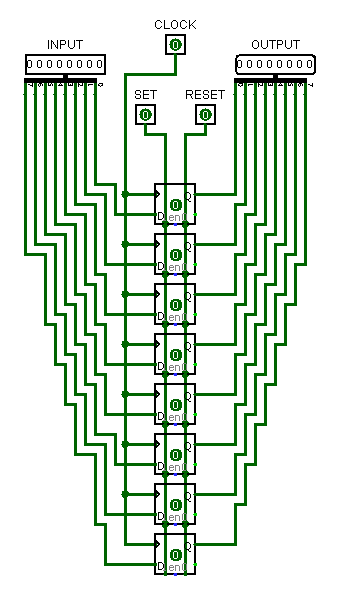


Figura 10 - Valor de Memória



A Memória ROM é uma memória que guarda instruções de forma sequencial, gerando algo parecido com um “histórico”, conceito comum em outras tecnologias, de forma que as instruções anteriores possam ser consultadas.

### **Descrição pinos e lógica**

A Memória ROM, que possui o circuito presente na Figura 8, é uma memória sequencial, ou seja, guarda os valores de entrada em sequencia, neste caso uma pseudo-intrução de 8 bits foi adotada, com 16 espaços de memória disponíveis. O fluxo do circuito se inicia na entrada Ler/Esc, presente na Figura 9, que determina se a operação será de leitura ou escrita de valores na memória, em caso de escrita, o fluxo pode ser observado iniciando no multiplexador conectado com o Clock, que, caso a entrada Ler/Esc seja 0, o Contador Síncrono de 4 bits é incrementado, determinando o seletor no desmultiplexador com entrada constante 1, armazenando o valor de INPUT no Valor de Memória, representado na Figura 10, correto, salvando bit a bit em cada *Flip-Flop* *D* presente no mesmo. Caso a entrada Ler/Esc seja igual a 1, o valor atual do Contador Síncrono é decrementado em 1, enviando para o valor de saída OUTPUT a última instrução registrada.

* + 1. **Testes do componente**

Após testes um problema de escrita de valores pôde ser percebido, pois o valor de entrada do principal do desmultiplexador, assim como o seu Seletor eram energizados ao mesmo tempo, gerando um erro no programa e salvando o mesmo valor em dois Valores de Memória diferentes, no entanto, ao energizar apenas o seletor do desmultiplexador com o valor do Contador Síncrono e atribuindo a entrada principal o valor fixo 1.

## MEMÓRIA RAM

Figura 11 - Memória RAM

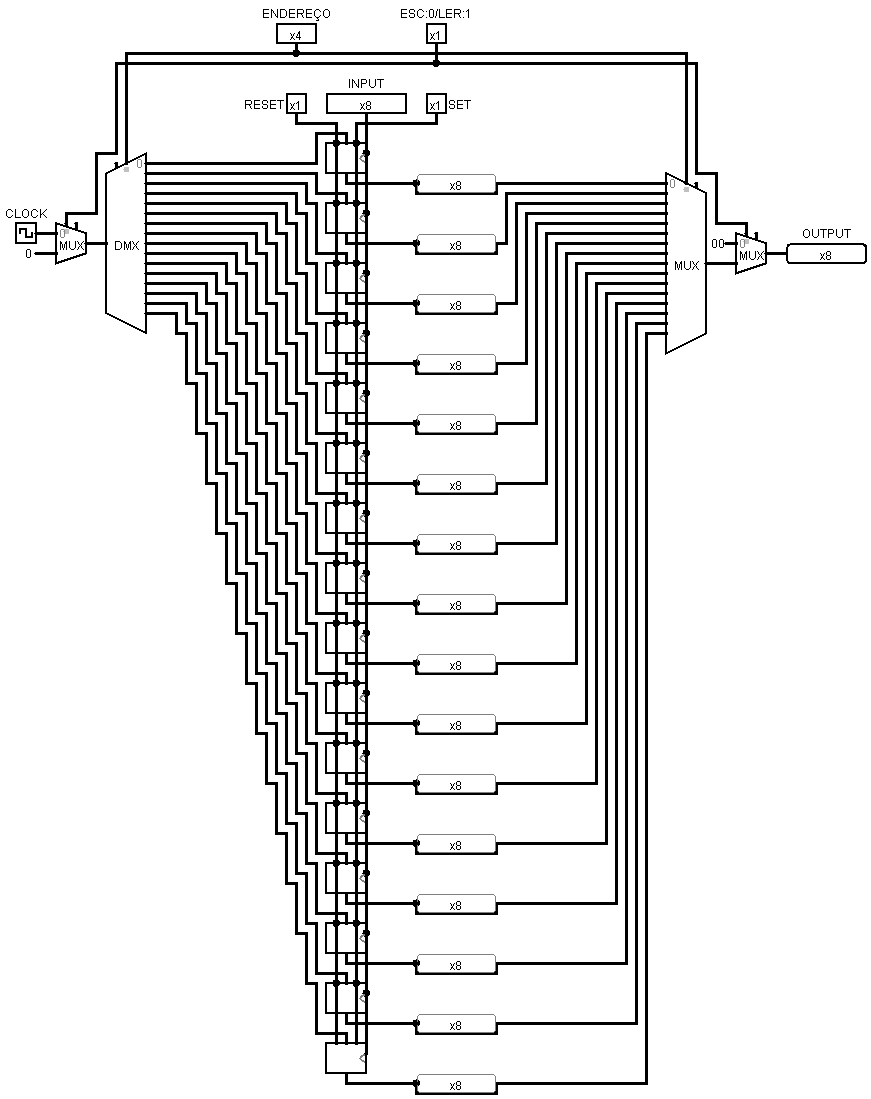
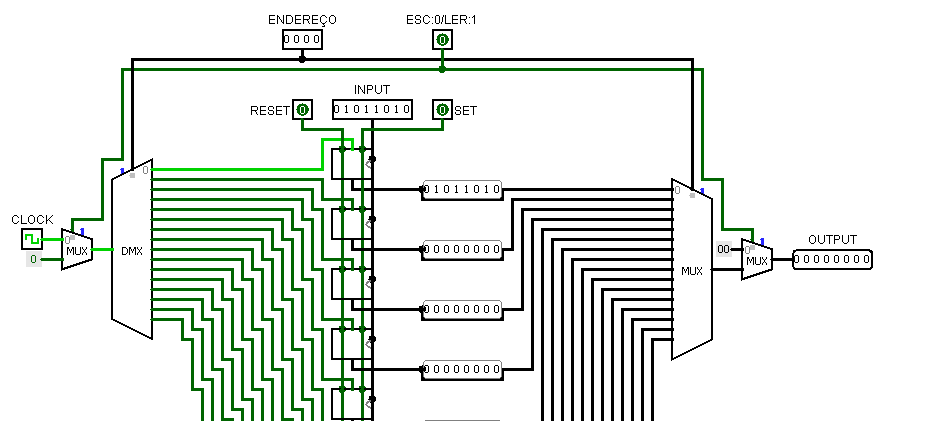


Figura 12 - Entradas e Saídas Memória RAM



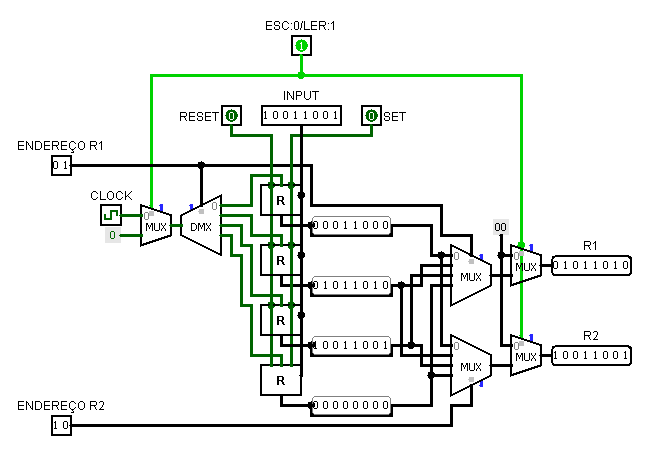
Diferente da Memória ROM, a Memória RAM é uma memória que guarda instruções de forma não-sequencial, ou seja, o local onde o valor será armazenado é designado pela entrada de Endereço.

### **Descrição pinos e lógica**

A Memória RAM é uma memória não-sequencial, ou seja, guarda os valores de entrada de acordo com o endereço que lhe é atribuída, neste caso um valor de 8 bits foi adotado, com 16 espaços de memória disponíveis. O fluxo do circuito se inicia na entrada Ler/Esc, presente na Figura 12, que determina se a operação será de leitura ou escrita de valores na memória, em caso de escrita, o fluxo pode ser observado iniciando no multiplexador conectado com o Clock, que, caso a entrada Ler/Esc seja 0, o valor de Clock é atribuído ao desmultiplexador e o sinal é enviado ao endereço determinado pela entrada Endereço, armazenando o valor de INPUT no Valor de Memória, presente na FIgura 10 designado. Caso a entrada Ler/Esc seja igual a 1 o valor de Endereço é atribuído ao seletor do multiplexador conectado com os Valores de Memória, que irá se conectar com multiplexador que conectado com o OUTPUT, que irá selecionar a saída da memória por conta da entrada Ler/Esc, que é conectada ao seu seletor e irá enviar o valor para a saída OUTPUT.

## BANCO DE REGISTRADORES

Figura 13 - Banco de Registradores



O Banco de Registradores armazena valores para serem utilizado em tempo de execução, realizando operações lógicas e aritméticas com o mesmos.

### **Descrição pinos e lógica**

O Banco de Registradores, representado na Figura 13, tem seu funcionamento parecido com a Memória RAM, usando de demultiplexadores e multiplexadores, com o auxílio de entradas de endereço para armazenas valores em Rgistradores, que possuem o mesmo circuito do Valor de Memória, presente na Figura 10, sendo sua maior diferença a quantidade de entradas e saídas, pois o mesmo possui duas entradas de endereço, ENDEREÇO R1 e ENDEREÇO R2, ambos com 2 bits, uma entrada de valor, INPUT e uma entrada de controle, Ler/Esc, além disto possui duas saídas de valores, R1 e R2. Quando o valor de Ler/Esc é igual a 0 o valor de ENDEREÇO R1 é utilizado para selecionar o Registrador para ser salvo o valor de INPUT, por outro lado caso o valor de Ler/Esc seja igual a 1, o valor da saída R1 será igual ao valor do Registrador no endereço da entrada ENDEREÇO R1, da mesma forma o valor da saída R2 será igual ao valor do Registrador no endereço da entrada ENDEREÇO R2.

## SOMADOR 8 BITS

Figura 14 - Circuito Completo Somador 8 bits

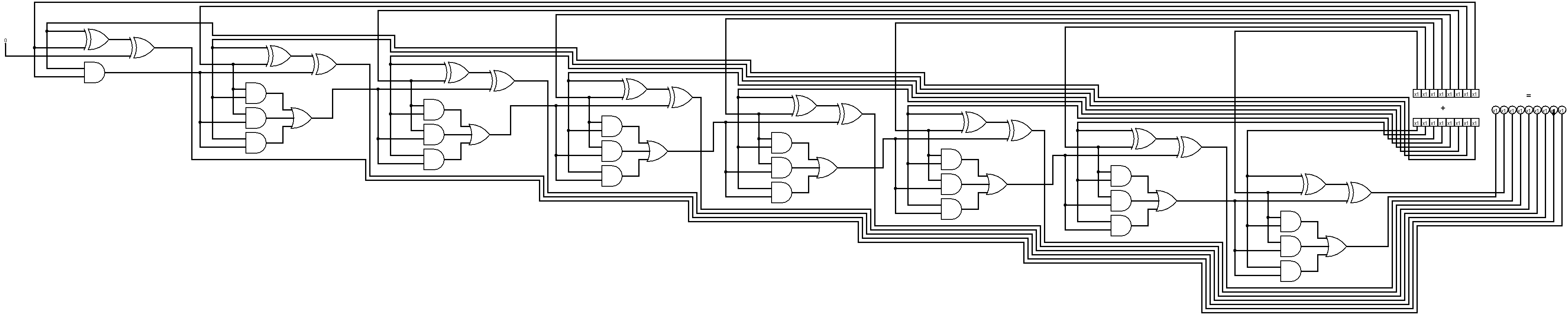
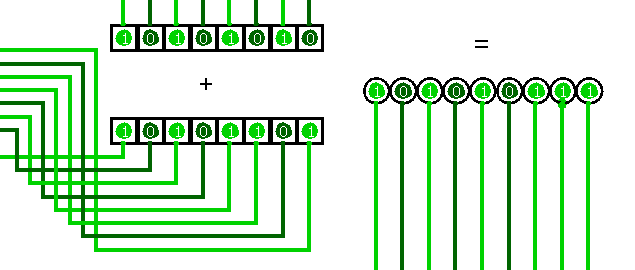


Figura 15 - Valores de Entrada Somador 8 bits



Assim como no ponto 2.4, que também se trata de um Somador 8 bits, o Somador 8 bits de dois valores quaisquer de 8 bits é um componente aritmético que soma valores binários utilizado para efetuar cálculos pelo computador.

### **Descrição pinos e lógica**

Da mesma forma que no ponto 2.4.1 o funcionamento do Somador de 8 bits pode ser entendido a partir do funcionamento de uma secção do mesmo, possuindo como único diferencial neste caso que não há valores fixos de entrada, podendo somar dois valores de 8 bits quaisquer.

* + 1. **Testes do componente**

O circuito do Somador de 8bits é formado por diversas secções de somador, presente na Figura 7, encadeadas, como demonstrado na Figura 14, onde dois valores de 8 bits quaisquer são somados, como demonstrado na Figura 15.

## UNIDADE DE CONTROLE 16 BITS

Definir o conceito aqui

### **Descrição pinos e lógica**

Definir aqui a descrição dos pinos, lógica do componente e a imagem no fim desta secção.

* + 1. **Testes do componente**

Definir aqui os testes(imagem) e colocar a descrição(Descrição dos testes, apresentando os pinos de entrada, as conexões ativas e o resultado dos pinos de saída.)

## ULA 8 BITS

Definir o conceito aqui

### **Descrição pinos e lógica**

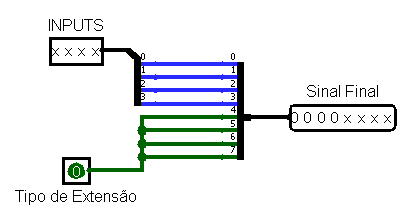
Definir aqui a descrição dos pinos, lógica do componente e a imagem no fim desta secção.

* + 1. **Testes do componente**

Definir aqui os testes(imagem) e colocar a descrição(Descrição dos testes, apresentando os pinos de entrada, as conexões ativas e o resultado dos pinos de saída.)

## EXTENSOR DE SINAL 4 PARA 8 BITS

Figura X - Extensor de Sinal



O Extensor de Sinal de 4 para 8 bits é útil para transformar um valor previamente obtido para um tamanho maior de instrução, quando isto é necessário para o funcionamento do circuito, existindo dois tipos de extensão: extensão de valor 0 e extensão de valor 1, completando o bits excedentes com o valor do tipo de extensão.

### **Descrição pinos e lógica**

O Extensor de Sinal possui duas entradas Tipo de Extensão, que possui 1 bit e INPUTS, que possui 4 bits e apenas uma saída, Sinal Final, que possui 8 bits, de forma que os bits da entrada INPUTS ocupam os quatro últimos bits do Sinal Final e o valor repetido da entrada Tipo de Extensão nos quatro primeiros bits do Sinal Final.

* + 1. **Testes do componente**

O valor “XXXX” da entrada INPUTS significam que dependem do funcionamento do circuito em conexão com o extensor de sinal, desta forma os resultados do Extensor de Sinal serão “0000XXXX” se o valor de Tipo de Extensão for 0 e serão “1111XXXX” se o valor de Tipo de Extensão for 1.

## MÁQUINA DE ESTADO

Figura X - Máquina de Estados Herbert

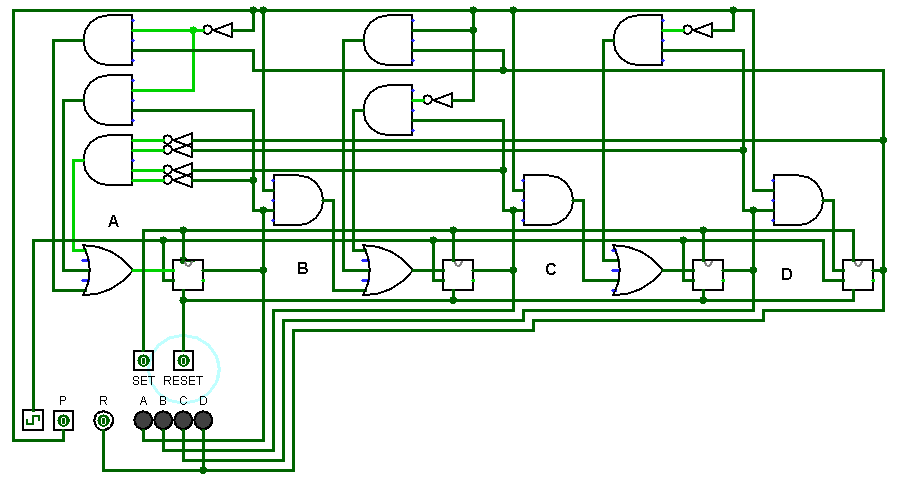
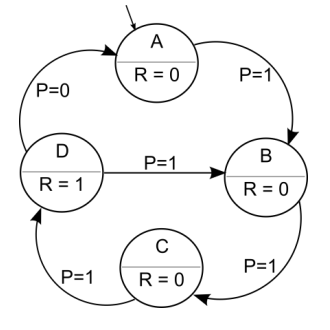


Figura X - Conceito Máquina de Estados Herbert



Uma Máquina de Estados é, resumidamente, uma máquina que devolve determinados valores dependendo do momento e se move de diferentes formas dependendo de uma determinada entrada. Esta Máquina de Estados não é nomeada, por isto lhe nomeei “Maquina de Estados Herbert”.

### **Descrição pinos e lógica**

A Máquina de Estados, presente na Figura X, é baseada no conceito de máquina de estados da Figura X, desta forma pressupõe-se as verdades: “Caso não haja estado atual, A torna-se o estado atual”, “Se o estado atual for A e P = 1, o estado atual torna-se B”, “Se o estado atual for B e P = 1, o estado atual torna-se C”, “Se o estado atual for C e P = 1, o estado atual torna-se D”, “Se o estado atual for D e P = 1, o estado atual torna-se B”, “Se o estado atual for D e P = 0, o estado atual torna-se A”, “Se A, ou B, ou C forem verdadeiro e P = , o determinado valor verdadeiro mantém-se verdadeiro”, “Se o estado atual for diferente de D, então R=0” e, por último, “Se o estado atual for D, então R=1”.

* + 1. **Testes do componente**

Figura X - Máquina de Estados Herbert (Caso 1)

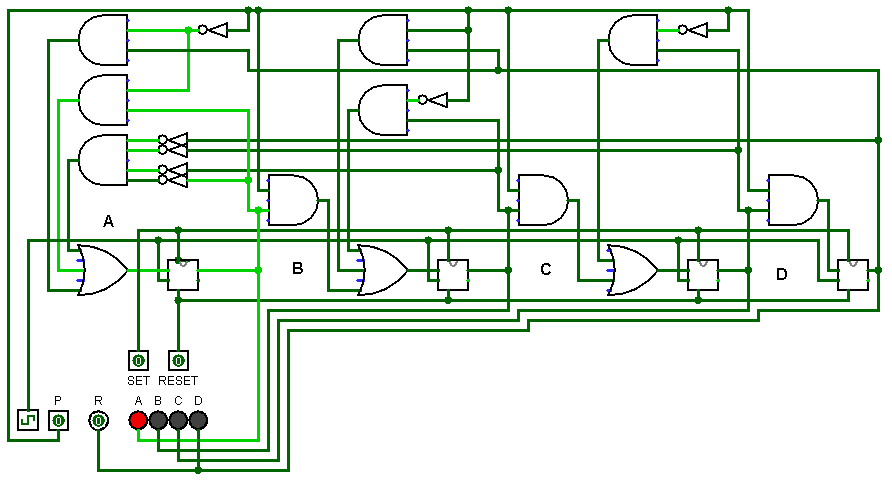


Figura X - Máquina de Estados Herbert (Caso 2)

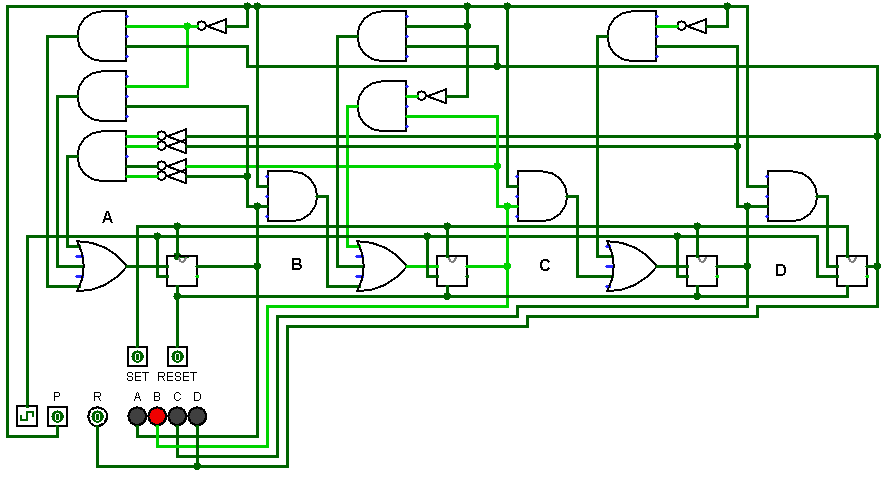


Figura X - Máquina de Estados Herbert (Caso 3)

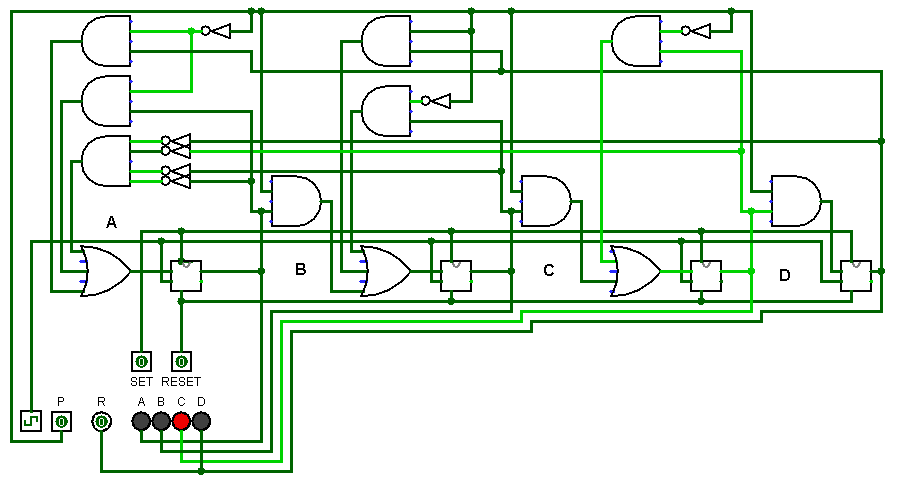


Figura X - Máquina de Estados Herbert (Caso 4)

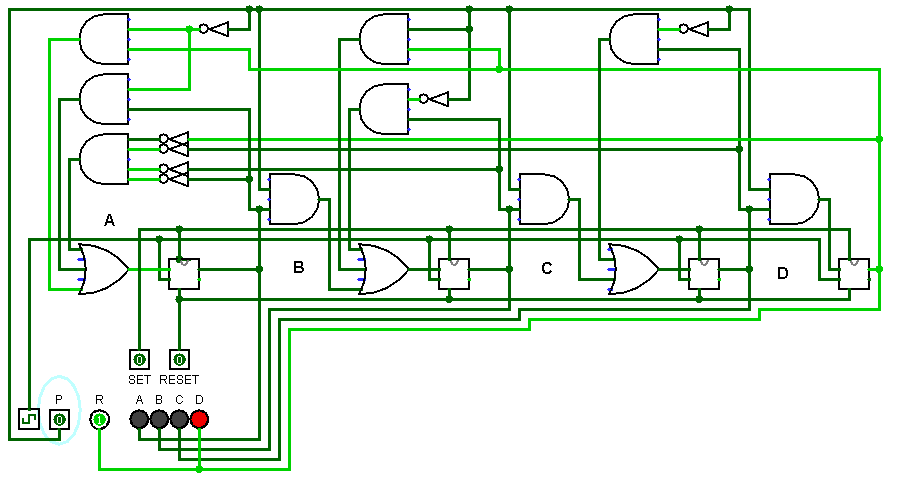


Figura X - Máquina de Estados Herbert (Caso 5)

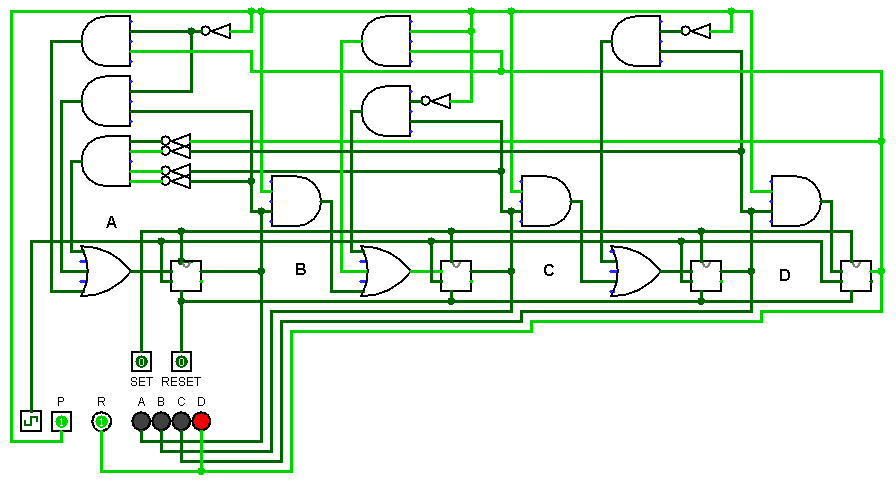
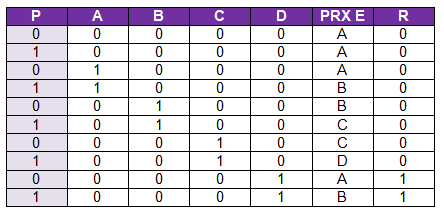


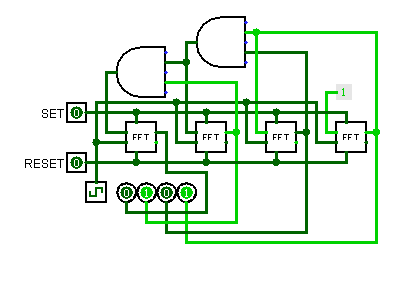
Tabela X - Tabela-Verdade Máquina de Estados Herbert



Seguindo as regras impostas pelo Conceito de Máquina de Estado da Figura X para cada *Flip-Flop D* presente no sistema (representando os estados), há condições para que os mesmos sejam iguais a 1, sendo elas, para o *Flip-Flop D* do estado A: Todos os outros *Flip-Flop D*  de estado serem iguais a 0, **ou** o *Flip-Flop D* de A ser igual a 1 **e** P for 0, **ou** o *Flip-Flop D* de D for 1 **e** P for 0. Para o *Flip-Flop D* do estado B: O *Flip-Flop D* de B ser igual a 1 **e** P for 0, **ou** o *Flip-Flop D* de A ser igual a 1 **e** P for 1, **ou** o *Flip-Flop D* de D for 1 **e** P for 1. Para o *Flip-Flop D* do estado C: O *Flip-Flop D* de C ser igual a 1 **e** P for 0, **ou** o *Flip-Flop D* de B ser igual a 1 **e** P for 1. Para o *Flip-Flop D* do estado D: O *Flip-Flop D* de C ser igual a 1 **e** P for 1. Por último, o valor de R é igual ao valor do *Flip-Flop D* do estado D. O funcionamento da Máquina de Estados Herbert pode ser descrita por meio da Tabela X, onde a coluna PRX E equivale ao próximo estado a ser ativado.

## CONTADOR SÍNCRONO

Figura X - Contador Síncrono



O Contador Síncrono é um componente que realiza uma contagem bit a bit de um valor com a quantidade de bits delimitada pela quantidade de *Flip-Flops* *T* presentes em seu circuito.

### **Descrição pinos e lógica**

O contador Síncrono desenvolvido possui 4 bits e, consequentemente, quatro *Flip-Flops* *T*, demonstrado na Figura X, que, após ser desenvolvido e testado gerou a tabela-verdade Tabela X e, após simplificação, gerou a Tabela X, desta forma o contador atribui o valor 1 a entrada do *Flip-Flop T* apenas quando todos os valores anteriores são iguais a 1, alternando-o, com exceção do último *Flip-Flop T, que sempre alterna o seu valor*.

Figura X - *Flip-Flop T*

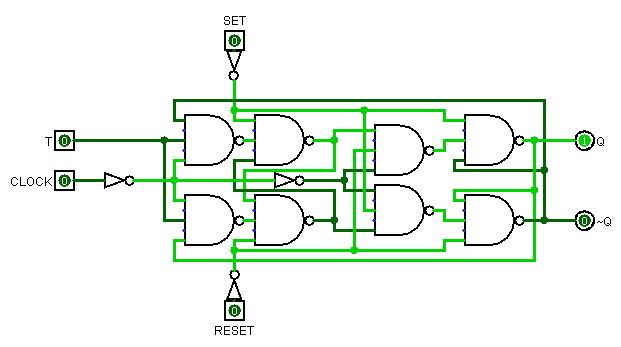


Tabela X - Tabela-Verdade *Flip-Flop T* Completa

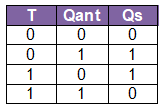
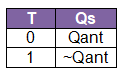


Tabela X - Tabela-Verdade *Flip-Flop T* Simplificada



* + 1. **Testes do componente**

Os valores do Contador Síncrono são obtidos pela propriedade de alternação de valores anteriores do *Flip-Flop T*, de forma que é possível atribuir entradas iguais a 1 quando valores também sejam iguais a 1, de forma parecida com um Somador.

# **CONSIDERAÇÕES FINAIS**

# 

# **REFERÊNCIAS**

MOL, Rian. Entendendo os Flip-FLop.**Flip e Flop, 2021**. Disponível em: https://www.filipeflop.com/blog/entendendo-o-flip-flops/. Acesso em: 20 nov. 2022.