

FACULDADE ENGENHEIRO SALVADOR ARENA

Rafael Rupert Barrocal - 081230002

Matheus da Silva Souza - 081230011

Henrique Alves Ferreira - 081230015

Pedro Henrique Simões Reys 081230022

Gabriel Melo Santos - 081230044

DOCUMENTAÇÃO TÉCNICA: CRIPTO-ALU

SÃO BERNARDO DO CAMPO

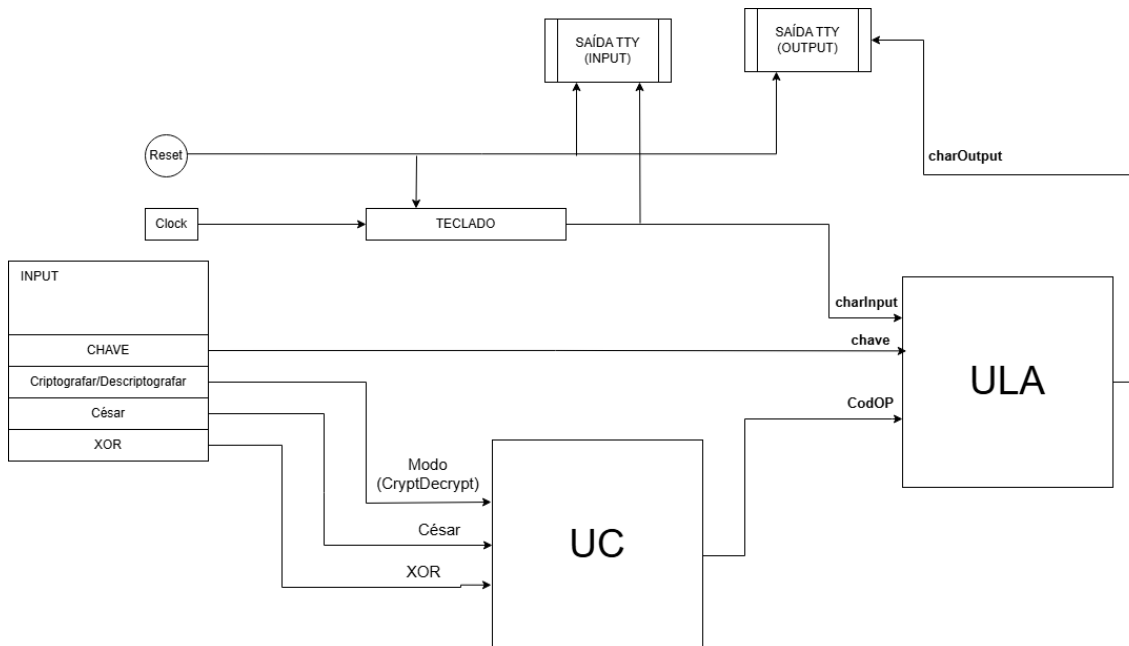
2025

1. INTRODUÇÃO

Este documento apresenta uma documentação técnica abrangente sobre o projeto de um sistema de criptografia desenvolvido no ambiente Logisim. O sistema foi concebido como uma ferramenta educacional que demonstra princípios fundamentais de criptografia digital através da implementação de dois algoritmos distintos: a Cifra de César e a operação XOR. A arquitetura do projeto segue uma abordagem modular hierárquica, onde cada componente possui responsabilidades bem definidas e interfaces claras de comunicação.

O objetivo principal do sistema é processar caracteres ASCII de 7 bits, aplicando transformações criptográficas baseadas na seleção do usuário entre modos de criptografia e descriptografia. A escolha do Logisim como plataforma de desenvolvimento permitiu a modelagem precisa do comportamento dos circuitos digitais, oferecendo uma representação visual clara do fluxo de dados e das operações lógicas envolvidas no processo criptográfico.

2. DIAGRAMA DE BLOCOS



3. DESCRIÇÃO DETALHADA DO PAPEL E FUNCIONAMENTO DE CADA MÓDULO

3.1. ARQUITETURA GERAL DO SISTEMA

A arquitetura do sistema foi organizada em seis circuitos principais interconectados, formando uma estrutura hierárquica coesa. No nível mais alto encontra-se o circuito principal (main), que atua como orquestrador do sistema, integrando todos os módulos e fornecendo as interfaces de usuário. Este circuito principal coordena a Unidade de Controle (UC) e a Unidade Lógica Aritmética (ULA), que por sua vez encapsulam funcionalidades mais especializadas.

A ULA representa o núcleo de processamento do sistema, incorporando três submódulos especializados: o Codificador Caesar (Caesar coder) para operações de criptografia, o Decodificador Caesar (Caesar decoder) para operações de descryptografia, e o Testador de Alfabeto (Alphabet tester) para classificação de caracteres.

O fluxo de dados segue o seguinte padrão: as entradas do usuário são capturadas através do teclado e pinos específicos, processadas pela Unidade de Controle que determina a operação a ser executada, direcionadas para os módulos de processamento apropriados na ULA, e finalmente exibidas nos dispositivos de saída TTY. Esta abordagem garante uma separação clara de responsabilidades entre os módulos.

3.2. MÓDULOS DO SISTEMA

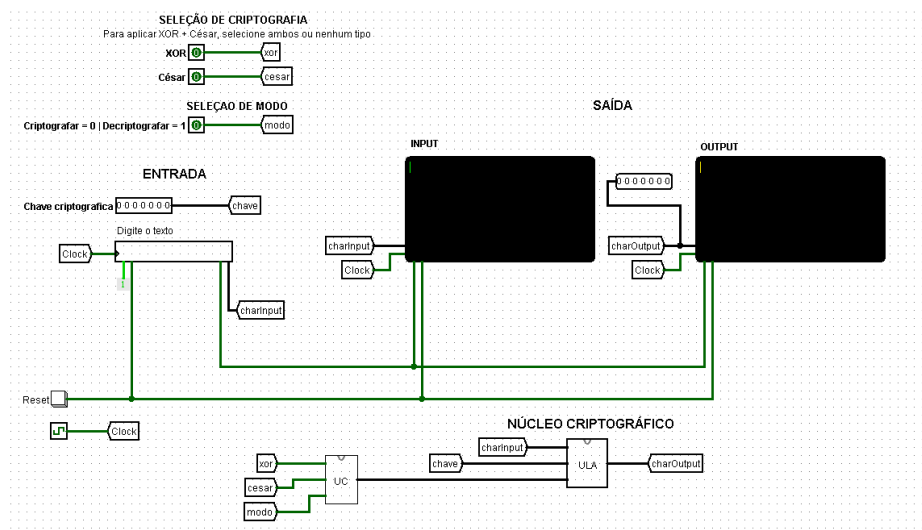
3.2.1. Circuito Principal (main)

O circuito principal é a interface do usuário e o coordenador de todo o sistema. Ele é responsável por:

- Entrada de dados: Através de um teclado, o usuário digita os caracteres que serão processados. Cada caractere é enviado para a ULA (Unidade Lógica e Aritmética) e também exibido em um display TTY de entrada (INPUT).

- Controle de operação: O usuário seleciona o modo de operação (César, XOR ou ambos) e se deseja criptografar ou descriptografar através de pinos de entrada.
- Geração de clock: Um clock central sincroniza todas as operações do sistema, sendo distribuído para todos os módulos e garantindo que as operações ocorram na borda de subida do sinal.
- Reset: Um botão de reset permite reinicializar o sistema a qualquer momento.
- Exibição de resultados: Do displays TTY são utilizados: um para mostrar a entrada (INPUT) e outro para mostrar a saída processada (OUTPUT).

Figura 1



Visão geral do circuito (main) dentro do Logisim

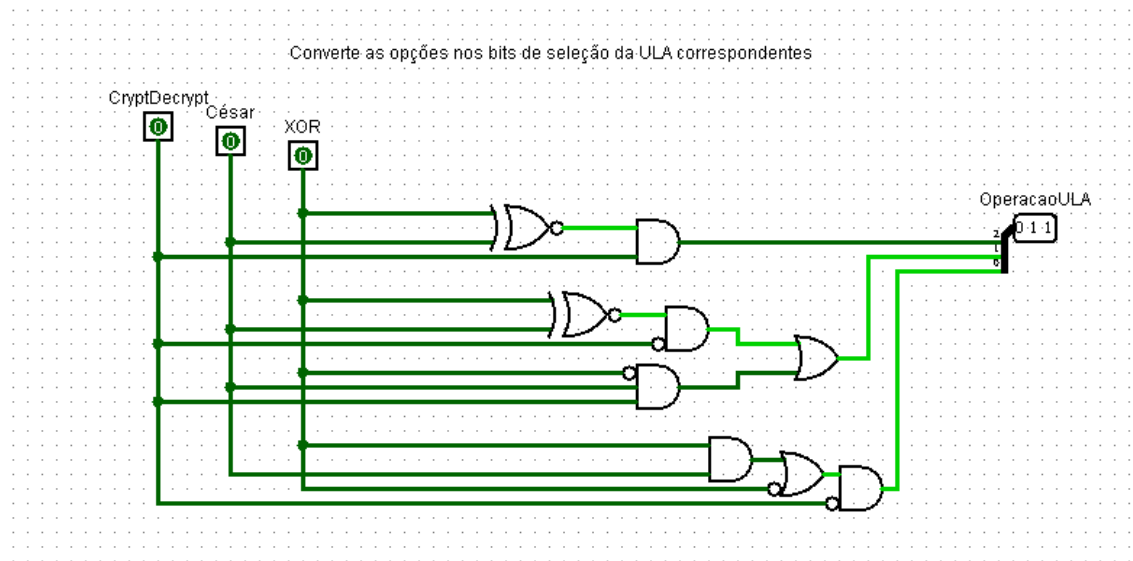
Aspectos técnicos:

- O clock é gerado por um componente Clock do Logisim, com frequência configurável.
- Os túneis charInput, chave, clock, modo, cesar e xor são utilizados para rotear os sinais entre os módulos.

3.2.2. Unidade de Controle (UC)

A Unidade de Controle é o cérebro do sistema. Ela interpreta as entradas do usuário (seleção de algoritmo e modo) e gera os sinais de controle para a ULA.

Figura 2



Visão geral da Unidade de Controle (UC) dentro do Logisim

Entradas:

- César: Ativa (1) ou desativa (0) o algoritmo César.
- XOR: Ativa (1) ou desativa (0) a operação XOR.
- CryptDecrypt: Define o modo (0 para criptografar, 1 para descriptografar).

Saída:

- OperacaoULA: Um sinal de 3 bits que indica à ULA qual operação deve ser realizada.

Funcionamento interno:

- 000: César seguido de XOR, codOP = 011
- 001: operação XOR, codOP = 000
- 010: César para criptografia, codOP = 010

- 011: Criptografia César seguido de XOR, codOP = 011
- 100: XOR seguido de decryptografia César, codOP = 100
- 101: operação XOR, codOP = 000
- 110: César para criptografia, codOP = 010
- 111: XOR seguido de decryptografia César , codOP = 100

Como é possível ver na Figura 2, onde temos uma entrada de 000, CryptDecrypt, César e XOR respectivamente e o código de operação 011 sendo gerado, o qual definira a operação de César seguido de XOR na ULA através da seleção de operação realizada pelo MUX.

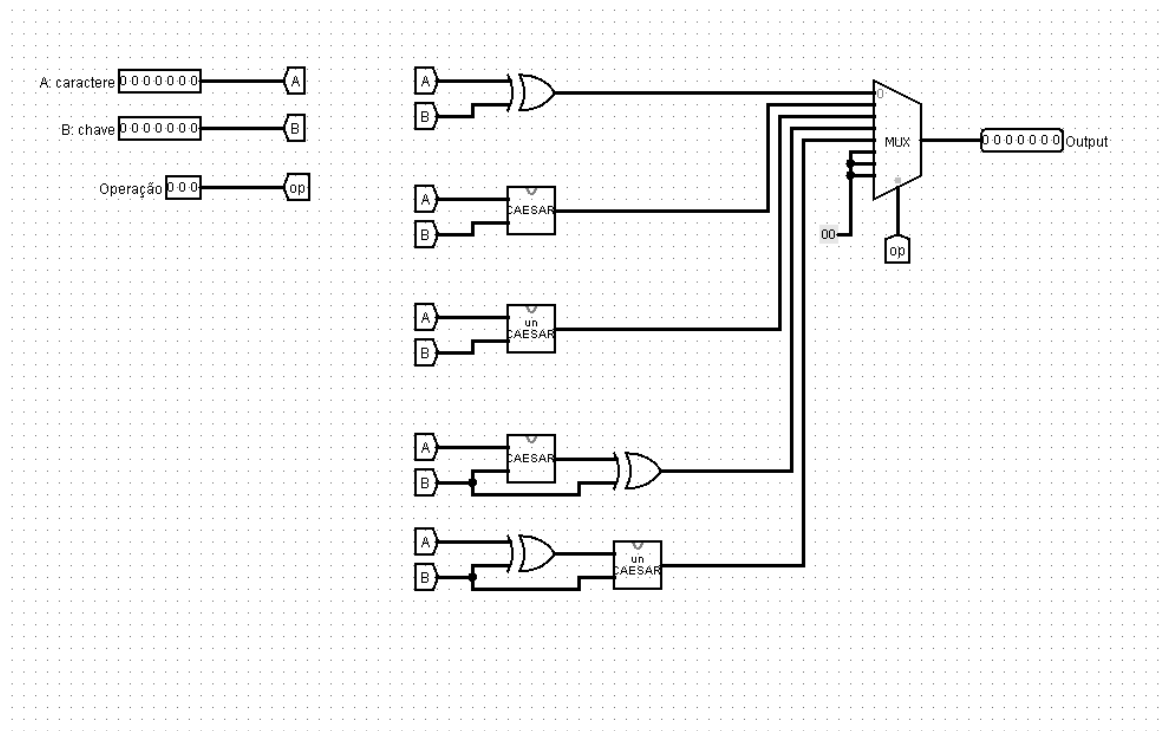
3.2.3. Unidade Lógica e Aritmética (ULA)

A ULA é o núcleo de processamento do sistema. Ela executa as transformações criptográficas de acordo com o sinal de controle recebido da UC.

Arquitetura paralela: A ULA possui vários módulos de processamento que operam simultaneamente:

- I. Dois codificadores Caesar (para criptografia).
- II. Dois decodificadores Caesar (para decryptografia).
- III. Três portas XOR (para operação bit a bit).
- IV. Multiplexador de saída: Um multiplexador de 8 entradas seleciona qual resultado será enviado à saída, baseado no sinal OperacaoULA da UC.

Figura 3



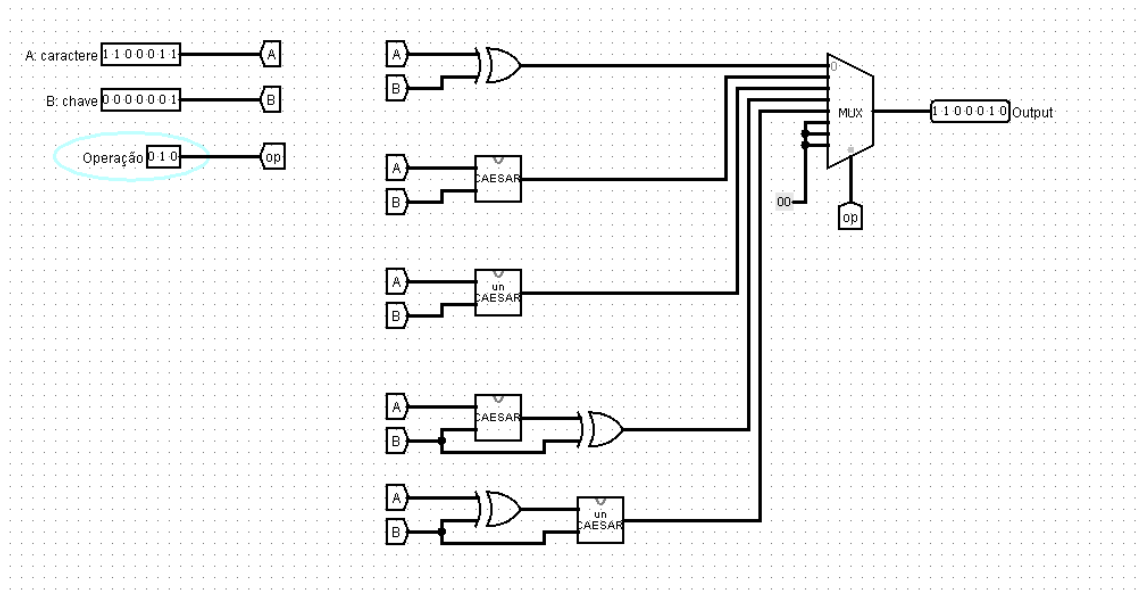
Visão geral da Unidade Lógica e Aritmética (ULA) dentro do Logisim

Fluxo de dados:

Os dados de entrada (caractere e chave) são distribuídos para todos os módulos de processamento. Cada módulo processa os dados independentemente e gera um resultado. O multiplexador seleciona o resultado correto baseado na operação solicitada.

Um exemplo é quando temos como entrada o caractere “c” representado por 1100011, como chave 0000001 representando 1 bit para as operações, e o código que define a operação a ser realizada como 010, o qual define a operação de codificação César adicionando 1 bit, tendo como resultado a letra “d” representada por 1100100 (caractere de entrada somado a 1 bit).

Figura 4



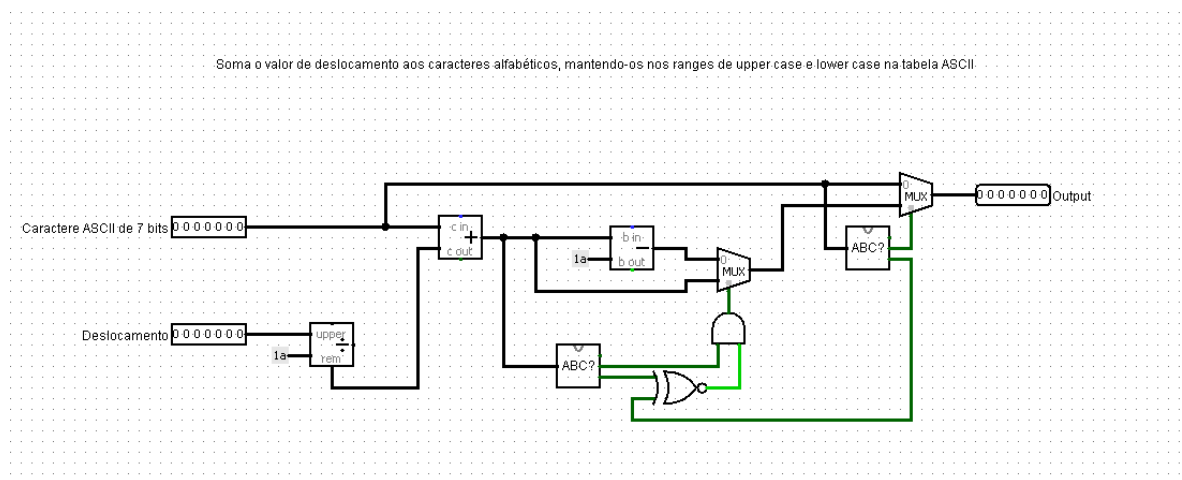
Exemplo de funcionamento da ULA dentro do Logisim

3.2.4. Codificador Caesar (Caesar coder)

Este módulo implementa a criptografia pela Cifra de César.

- Deslocamento de caracteres: O codificador soma o valor do deslocamento (chave) ao caractere de entrada.
- Correção de overflow: Se o resultado ultrapassar o limite das letras (Z ou z), o módulo subtrai 26 para voltar ao início do alfabeto.
- Preservação de não-alfabéticos: Caracteres que não são letras (números, símbolos) não são modificados.

Figura 5



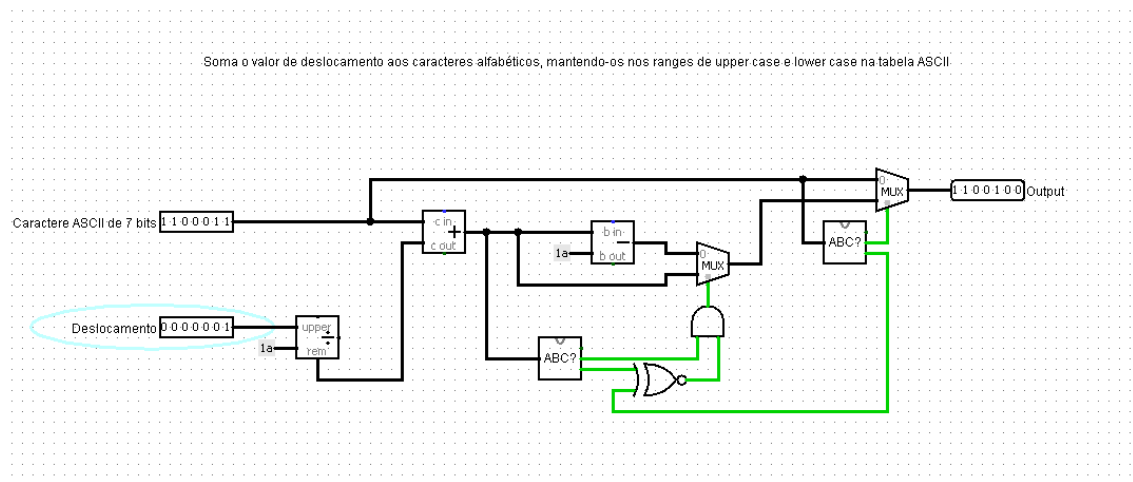
Visão geral do Codificador Caesar dentro do Logisim

Funcionamento técnico:

- I. Um somador realiza a adição do caractere com a chave.
- II. Um comparador verifica se o resultado está fora do intervalo alfabético e ativa a correção.
- III. Um subtrator é usado para corrigir o overflow (subtrai 26).
- IV. Um multiplexador escolhe entre o resultado da adição ou o resultado corrigido.

Um exemplo é quando temos a entrada da letra “c” representada por 1100011 e a chave 0000001, onde teremos a adição de 1 bit tendo como resultado a letra “d” representada por 1100100.

Figura 6



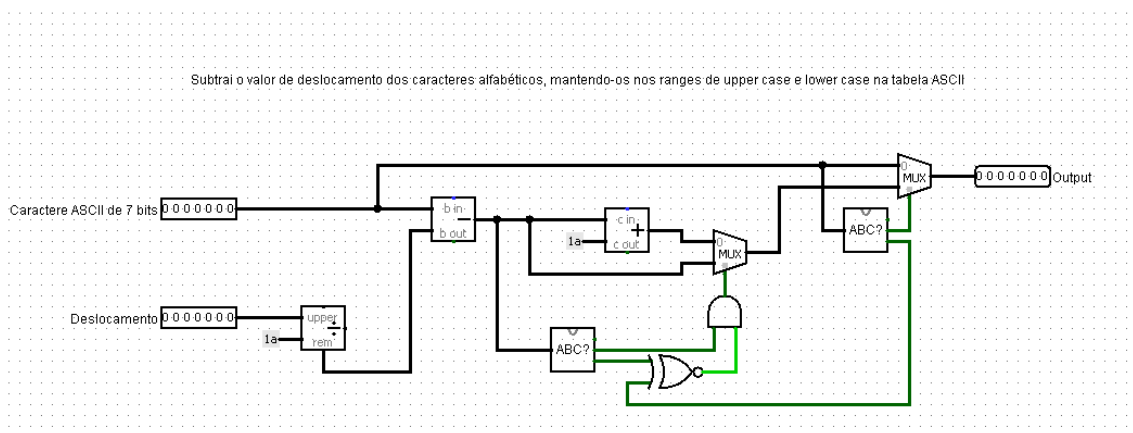
Exemplo de funcionamento do Codificador Caesar dentro do Logisim

3.2.5. Decodificador Caesar (Caesar decoder)

Este módulo implementa a descryptografia pela Cifra de César, realizando a operação inversa do codificador.

- Deslocamento inverso: Subtrai o valor do deslocamento (chave) do caractere cifrado.
- Correção de underflow: Se o resultado for menor que o limite das letras (A ou a), o módulo adiciona 26 para voltar ao final do alfabeto.

Figura 7



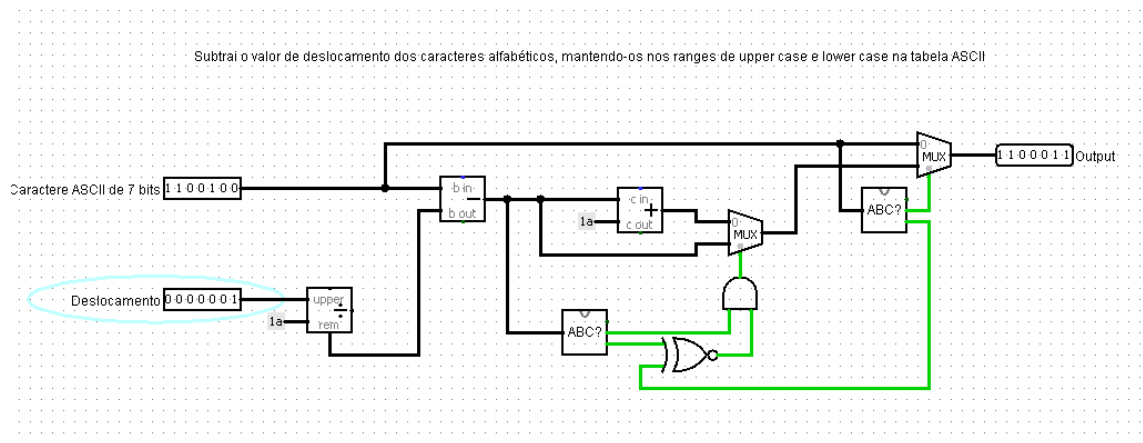
Visão geral do Decodificador Caesar dentro do Logisim

Funcionamento técnico:

- I. Um subtrator realiza a subtração do caractere pela chave.
- II. Um comparador verifica underflow e ativa a correção.
- III. Um somador é usado para corrigir o underflow (adiciona 26).
- IV. A estrutura é espelhada em relação ao codificador

Um exemplo é quando temos a entrada da letra “d” representada por 1100100 e a chave 0000001, onde teremos a subtração de 1 bit tendo como resultado a letra “c” representada por 1100011.

Figura 8



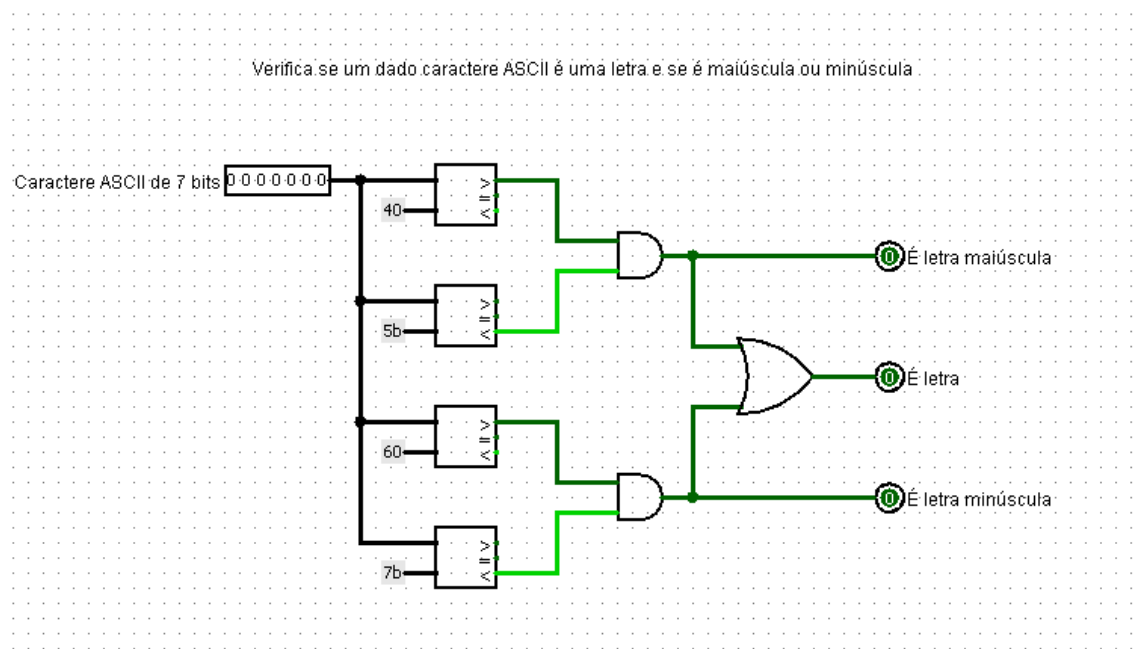
Exemplo de funcionamento do Decodificador Caesar dentro do Logisim

3.2.6. Testador de Alfabeto (Alphabet tester)

Este módulo auxiliar classifica os caracteres em letras maiúsculas, minúsculas ou não alfabéticos.

- Comparadores: Quatro comparadores verificam se o caractere está nos intervalos:
- Maiúsculas: entre 0x41 (A) e 0x5A (Z).
- Minúsculas: entre 0x61 (a) e 0x7A (z).

Figura 9



Visão geral do Testador de Alfabeto dentro do Logisim

Portas lógicas: Combinam os resultados dos comparadores para gerar três sinais:

- É letra maiúscula.
- É letra minúscula.
- É letra (qualquer uma das duas).

Utilização:

O Alphabet tester é usado pelos módulos Caesar para decidir se um caractere deve ser processado ou não.

Um exemplo é na imagem a seguir onde temos a entrada da letra “c” representada por 1100011 e os sinais de confirmação que é letra e que é minúscula sendo gerados.

Verifica-se um dado caractere ASCII é uma letra e se é maiúscula ou minúscula.

Caractere ASCII de 7 bits: 1 1 0 0 0 1 1

40

5b

60

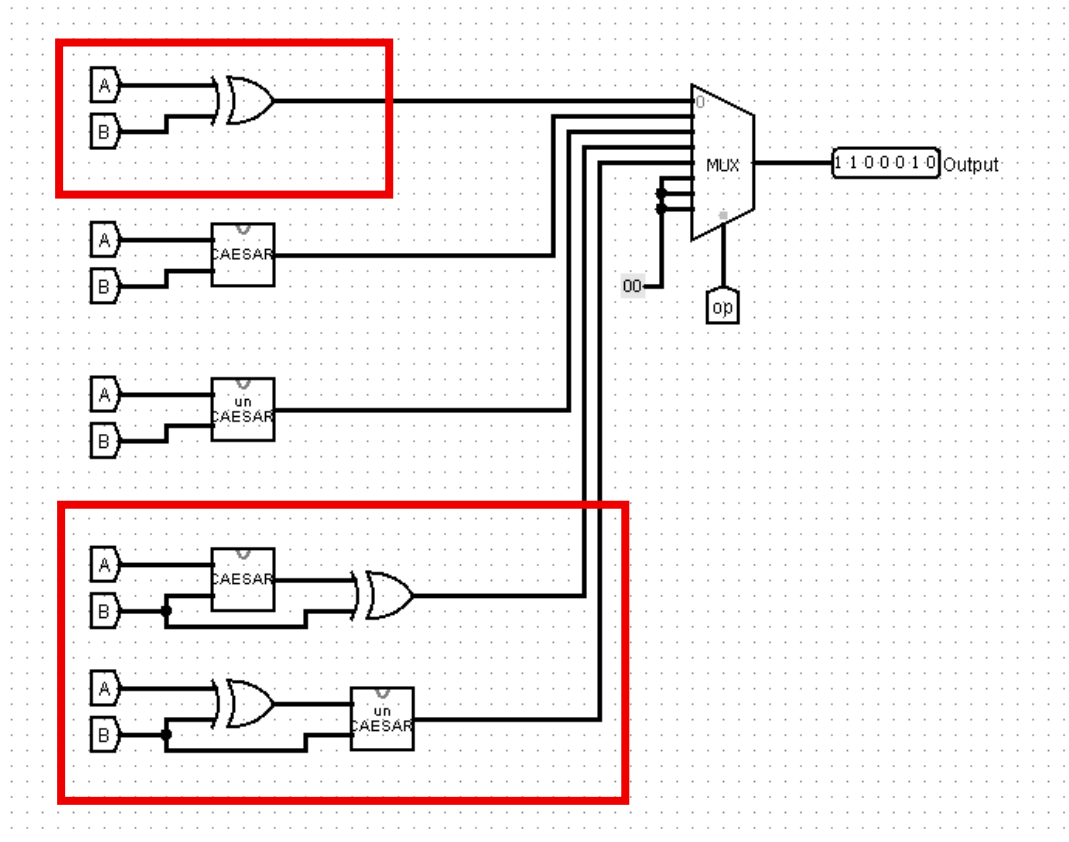
7b

É letra maiúscula

É letra

É letra minúscula

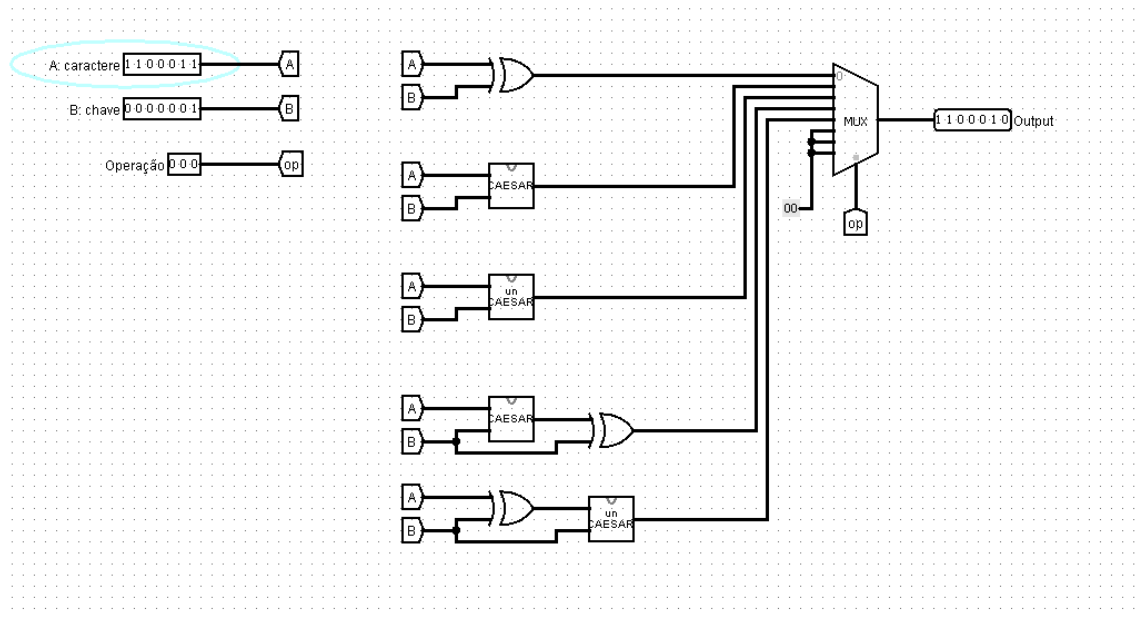
Figura 11



Exemplo da porta XOR dentro do Logisim

Um exemplo é quando temos como entrada o caractere “c” representado por 1100011, como chave 0000001 representando 1 bit para as operações, e o código que define a operação a ser realizada como 000, o qual define a operação XOR com 1 bit de diferença simétrica entre os caracteres, tendo como resultado a letra “b” representada por 1100010. O mesmo acontece quando temos a letra “b” como entrada, onde a saída é a letra “c”, evidenciando a simetria de 1 bit.

Figura 12



Exemplo de funcionamento da porta XOR dentro do Logisim

4. RELATO DAS DECISÕES DE PROJETO E JUSTIFICATIVAS TÉCNICAS

O projeto do sistema Cripto-ALU foi estruturado seguindo uma arquitetura modular hierárquica, decisão técnica fundamentada na necessidade de separação clara de responsabilidades entre os componentes. Esta abordagem permitiu o desenvolvimento, teste e manutenção independente de cada módulo, além de facilitar a reutilização de circuitos em diferentes partes do sistema.

Para a Cifra de César, implementou-se detectores de overflow e underflow com correção modular (± 26), garantindo que os caracteres permaneçam dentro dos intervalos alfabéticos e mantenham a legibilidade do texto. A decisão de preservar caracteres não-alfabéticos através do testador de alfabeto foi crucial para manter a integridade estrutural da mensagem original, evitando corrupção de símbolos e números durante o processamento criptográfico.

A operação XOR foi desenvolvida de forma simétrica, aproveitando a propriedade matemática de que duas aplicações consecutivas com a mesma chave restauram o valor original. Esta escolha técnica simplificou

significativamente a implementação, reduzindo a complexidade do circuito ao eliminar a necessidade de módulos separados para criptografia e descriptografia.

A sincronização via clock único distribuído para todos os módulos garantiu a coordenação temporal das operações, prevenindo condições de corrida e facilitando a depuração do sistema. A centralização do controle na Unidade de Controle desacoplou a lógica operacional do processamento, criando uma interface padronizada que permite futuras expansões e modificações sem impactar a ULA.