



FCI – Faculdade de Computação e Informática

PROJETO 2 - AB&CD

DESCRITIVO TÉCNICO

Unidade Lógica e Aritmética de 8 bits



Índice

1. Apresentação
2. Descritivo Técnico
3. Elaboração do Projeto
4. Simulações e avaliação de resultados
5. Conclusões finais
6. Referências Bibliográficas
7. Anexos



1. Apresentação

Uma **Unidade Lógica Aritmética** (ULA, ou em inglês *Arithmetic and Logic Unit*, ALU), é um componente digital crucial dentro de uma **Unidade Central de Processamento** (CPU), responsável por executar as operações lógicas e aritméticas necessárias. A ULA das CPUs modernas é altamente poderosa e complexa. Além da ULA, as CPUs também incluem uma **Unidade de Controle** (UC). A maior parte das operações de uma CPU é realizada pela ULA, que manipula dados nos **registradores** de entrada, que por sua vez são pequenas áreas de armazenamento temporário integradas à CPU.

A UC instrui a ULA sobre qual operação executar sobre esses dados, e a ULA armazena o resultado em um registrador de saída, também conhecido como acumulador (ACC). A UC coordena o fluxo de dados entre esses registradores, a ULA e a memória principal do computador.

A ULA, por sua vez, pode ser subdividida em duas unidades básicas:

- **Unidade de Operações Aritméticas** (UA), que realiza operações como adição, subtração, transferência de dados, incremento e decremento;
- **Unidade de Operações Lógicas** (UL), responsável por operações bit-a-bit (ou no inglês *bitwise*) como AND, OR, XOR e XNOR.

Para esse projeto foi construída uma ULA simples de 8-bits, usando o programa CEDAR Logic. O projeto inclui uma interface integrada que possibilita uma fácil entrada e saída de dados, garantindo a funcionalidade completa do circuito da ULA.

2. Descritivo Técnico

A ULA construída nesse projeto possui uma palavra de dados (tamanho dos dados operados) de 8-bits, contendo dois registradores de entrada (X e Y) e um de saída (G), todos com tamanho de 8-bits. As operações realizadas foram divididas em duas unidades: **unidade aritmética** (UA) e **unidade lógica** (UL).

Na UA são realizadas as operações de soma, subtração, incremento, decremento e transferência de dados usando o circuito integrado 74283, que é um circuito somador completo. As operações são selecionadas com base em 4 chaves que o usuário pressiona, que enviam sinais de controle para multiplexadores que selecionam os operandos referentes a cada operação. Por exemplo, para a operação de decremento **DEC G, X** os multiplexadores selecionam X para o operando B e -1 para o operando A, ocasionando em $B - 1$. Já na UL são realizadas as operações lógicas mais simples: AND, OR, XOR e XNOR, usando portas lógicas básicas.

Para a entrada de dados são usados teclados matriciais e chaves de seleção.



Nos teclados o usuário pode selecionar os operandos de entrada (ou seja, os valores dos registradores X e Y), e por meio das chaves é possível enviar sinais de controle à ULA.

Nas chaves de seleção é possível selecionar uma das 12 operações. Há um multiplexador conectado à quarta chave de seleção, pois é ela que escolhe qual unidade da ULA será usada. Também há chaves ao lados dos teclados para informar à ULA que o operando é negativo. Na tabela abaixo estão as operações da ULA:

| S ₃ S ₂ S ₁ S ₀ | Operação | Mnemônico |
|-------------------------------------------------------------|----------------|-------------|
| 0 0 0 0 | G = X | MOV G, A |
| 0 0 0 1 | G = X + 1 | INC G, A |
| 0 0 1 0 | G = X - 1 | DEC G, A |
| 0 0 1 1 | G = X + Y | ADD G, A, B |
| 0 1 0 0 | G = Y | MOV G, B |
| 0 1 0 1 | G = Y + 1 | INC G, B |
| 0 1 1 0 | G = Y - 1 | DEC G, B |
| 0 1 1 1 | G = X + Y' + 1 | SUB G, A, B |
| 1 0 0 0 | G = X and Y | AND G, A, B |
| 1 0 0 1 | G = X or Y | OR G, A, B |
| 1 0 1 0 | G = X xor Y | XOR G, A, B |
| 1 0 1 1 | G = X xnor Y | XNR G, A, B |

Tabela 1 - Operações da ULA

Vale ressaltar que os teclados matriciais funcionam no formato BCD, ou seja, são três teclados, um para as centenas, outro para as unidades e mais um para as dezenas, para cada um dos operandos (dessa forma, cada operando tem dois teclados), que enviam o número de 8-bits com sinal (ou seja, de -128 a 127) inserido para um codificador, que codifica os dados em binário e os envia para os registradores para serem processados pela ULA.

Para a saída de dados são usados 3 displays de 7 segmentos e 9 LEDs. Os displays mostram os dados em formato BCD. Portanto há um decodificador que recebe os dados do registrador de saída G e os decodifica para formato BCD para ser mostrado nos três displays (e em mais 3 LEDs, usados para mostrar o sinal do número). Os 9 LEDs superiores mostram o valor de saída em binário, tendo um nono LED para mostrar o valor do sinal Carry-out provindo da UA.

Um detalhe importante é que caso a unidade lógica (UL) seja usada os displays de 7 segmentos não serão usados (mostrarão zero). Adicionalmente são usados alguns LEDs para mostrar alguns erros que podem ocorrer, como o usuário inserir um dígito inválido (de A a F) ou um número inválido (por exemplo -130) ou quando o resultado de alguma operação está errado (ou seja, ocorre um overflow).

3. Elaboração do projeto

O projeto contempla várias partes diferentes que são integradas para que haja



um funcionamento completo, tendo sido dividido em páginas, para uma melhor organização. Essas partes são descritas nesta seção.

3.1 Fase 1: Teclados e registradores

Para que o usuário possa utilizar a ULA foi adicionada uma interface simplificada ao projeto, se encontrando na primeira aba no CEDAR.

Para a inserção de dados nos registradores X e Y para o processamento na ULA são utilizados 6 teclados matriciais, 3 para cada registrador. Como dito anteriormente, esses teclados emitem os dados inseridos em formato BCD, podendo obter dados de -128 a 0, e de 0 a 127 (valores fora da escala ou o uso de dígitos hexadecimais irão sinalizar erros por meio de LEDs). O botão a esquerda dos teclados indica se o número é negativo, dessa forma podendo inserir valores de -128 a 0. A ULA processa os dados em binário, portanto os dados inseridos em BCD são convertidos para binário por meio de um codificador e enviados para os registradores para processamento. Para selecionar de operações da ULA são usadas 4 chaves, de S_0 a S_3 , onde é S_3 que escolhe a unidade a ser usada (UA caso 0, ou UL caso 1).

Para a visualização do dado no registrador G são usados displays de 7 segmentos e LEDs, como já dito. Como os displays mostram os dados em formato BCD existe uma etapa de conversão de binário para BCD antes de mostrar nos displays, podendo ser de -128 a 127 (existem três LEDs ao lado dos displays para mostrar o sinal), necessitando então de três displays.

O resultado também é mostrado nos LEDs superiores, porém em formato binário, mostrando os 8-bits e o valor do sinal de C_{out} , provindo da UA. Caso a UL seja usada, os displays não são usados e o sinal de C_{out} não é mostrado. Ainda são usados três LEDs para a detecção de erros: **O_XERR** e **O_YERR** sinalizam quando o usuário insere um dígito ou valor inválido, já **O_SUMERR** sinaliza que resultado da operação resultou em overflow (como por exemplo, $127 + 1 = -128$).

Abaixo está o diagrama da interface principal contendo os teclados e displays.

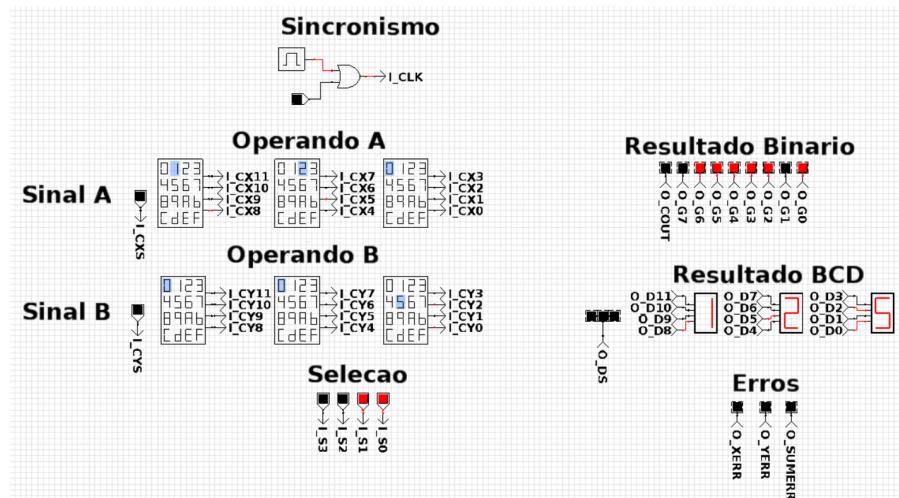


Figura 1 - Diagrama da Interface Principal

Os dados que saem dos conversores BCD para binário (provindo originalmente dos teclados) chegam aos registradores X (ou A) e Y (ou B), ambos de 8-bits. O resultado da ULA também é salvo no registrador G (ou ACC).

Abaixo está representado os três registradores usados:

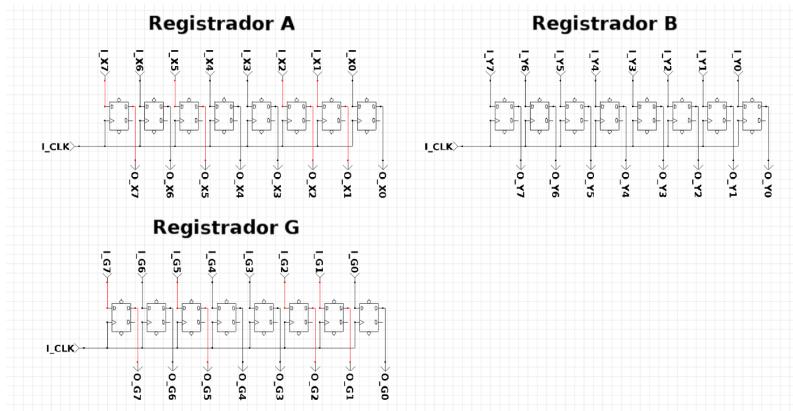


Figura 2 - Diagrama dos Registradores de 8-bits

Vale notar que os dados dos registradores são atualizados com base no sinal de sincronismo provindo da interface com o usuário, que emite um pulso de sincronização com base no clock de 100 Hz ou no apertar da chave de sincronismo, na parte superior da interface, como mostra a figura 1.

3.2 Fase 2: Unidade Aritmética

As operações aritméticas são realizadas na UA, sendo selecionada a operação desejada por meio das chaves de seleção na interface principal. Caso a chave S_3 seja desativada a UA será utilizada.

Toda a operação da UA envolve o CI 74283, de 4-bits (dois são usados para conseguir um processamento de 8-bits, conectando o C_{out} do primeiro ao C_{in} do segundo). Os valores dos registradores de entrada são passados aos



multiplexadores, que selecionam que operandos a serem usados na operação com base nas chaves de seleção e repassam para os dois CI's, conectando às entradas A e B os valores escolhidos para a operação. O resultado final em S é repassado ao registrador de saída G. O bit V de overflow é usado para sinalizar erro na soma (sinal O_SUMERR). Na tabela abaixo é possível ver quais operações são realizadas para cada conjunto de entradas de seleção, bem como os operandos usados.

| Código de seleção $S_2\ S_1\ S_0$ | Operação desejada $G(A + B + C_{in})$ | Entradas necessárias ao somador | | |
|--------------------------------------|---------------------------------------------|---------------------------------|---|----------|
| | | A | B | C_{in} |
| 0 0 0 | X (transferir) | 0000 0000 | X | 0 |
| 0 0 1 | X + 1 (incrementar) | 0000 0000 | X | 1 |
| 0 1 0 | X - 1 (decrementar) | 1111 1111 | X | 0 |
| 0 1 1 | X + Y (adicionar) | Y | X | 0 |
| 1 0 0 | Y (transferir) | 0000 0000 | Y | 0 |
| 1 0 1 | Y + 1 (incrementar) | 0000 0000 | Y | 1 |
| 1 1 0 | Y - 1 (decrementar) | 1111 1111 | Y | 0 |
| 1 1 1 | X + Y' + 1 (subtração por complemento de 2) | Y' | X | 1 |

Tabela 2 - Operações da Unidade Aritmética

Como os operandos mudam conforme a operação, foram acoplados multiplexadores a entrada do CI, que escolhem os operandos com base em um sinal de seleção baseado nos códigos de operações:

No operando A existem 4 opções possíveis, portando 2 bits de seleção:

- 00: 0000 0000
- 01: Y
- 10: Y'
- 11: 1111 1111

No operando B existem 2 opções possíveis, portando 1 bit de seleção:

- 0: X
- 1: Y

Adicionalmente, o sinal C_{in} também precisa ser selecionado conforme a operação, portanto também é um 1 bit de seleção: 1 para sim e 0 para não.

Abaixo está a tabela verdade construída para os bits de seleção:

| S2 | S1 | S0 | A1 | A0 | B | C |
|----|----|----|----|----|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 |

Tabela 3 - Tabela-verdade da seleção de operandos



Com base na tabela foi montado as expressões lógicas de seleção e criado o diagrama do circuito da unidade aritmética, como mostra a figura abaixo:

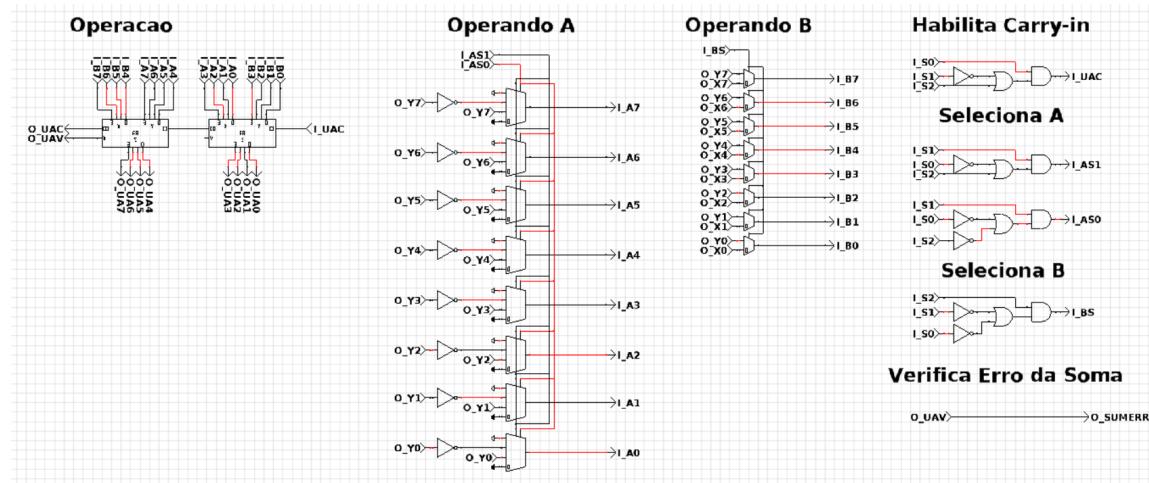


Figura 3 - Diagrama da Unidade Aritmética (UA)

3.3 Fase 3: Unidade Lógica

As operações lógicas são realizadas na UL, sendo selecionada a operação desejada por meio das chaves S_1 e S_0 . Caso a chave S_3 seja ativada a UL será utilizada e o bit C_{out} não será usado, bem como os displays. A UL, como visto na tabela 1, consegue fazer quatro operações:

- AND ($S_1S_0 = 00$)
- OR ($S_1S_0 = 01$)
- XOR ($S_1S_0 = 10$)
- XNOR ($S_1S_0 = 11$)

Da mesma forma que antes, se 4 operações são necessárias, são necessários 2 bits de seleção, que no caso são S_0 e S_1 . Como pode ser visto na figura abaixo, são realizadas as 4 operações sobre os valores em X e Y, e os seus resultados passam por multiplexadores, que repassam as entradas às saídas com base nos bits de seleção acima.

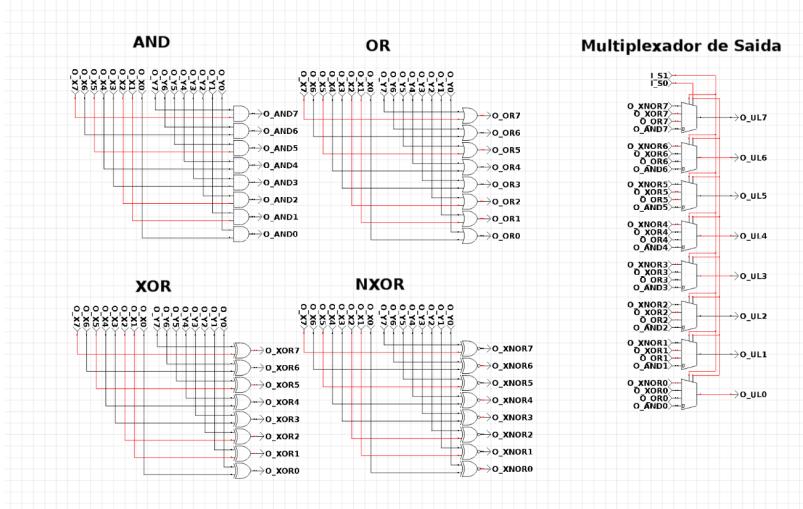


Figura 4 - Diagrama da Unidade Lógica (UL)

3.4 Fase 4: Integração Unidade Aritmética e Lógica

Ao integrar a UA e a UL obtém-se a ULA completa, como é mostrado na figura abaixo.

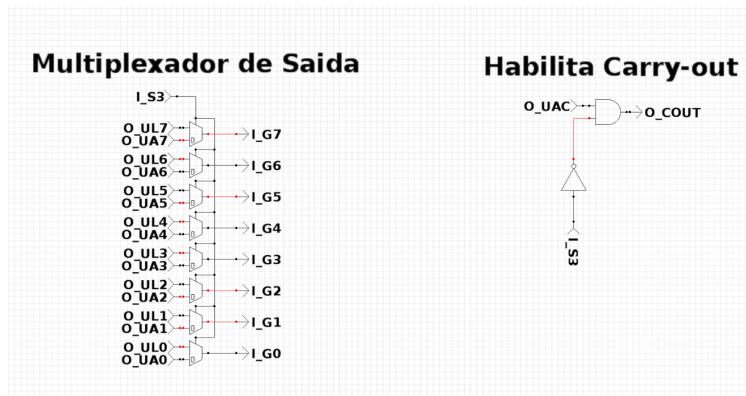


Figura 5 - Diagrama da Integração da Unidade Lógica e Aritmética

É possível perceber que há um multiplexador que, com base no valor de S_3 , escolhe o valor de saída para o registrador G. Ou seja, são processados os dados na UA e na UL, e esses dados são repassados para o registrador G.

Como dito anteriormente, caso seja usada a UL ($S_3 = 1$) o C_{out} deve ser ignorado e não deve aparecer no LED. Portanto é feito uma lógica para habilitar ou não o sinal de C_{out} , com base em S_3 .

3.5.3. Decodificador a partir de memória ROM

Para obter os dados (pelo teclado) e mostrar os dados (nos displays) são usados valores BCD. A ULA processa dados binários, portanto deve haver uma conversão.

Abaixo são falados os dois conversores usados, ambos tendo sido baseados



em memórias ROMs, onde o valor de entrada (valor a ser decodificado) representa o endereço e o valor de saída (valor decodificado) é o conteúdo daquele endereço.

3.5.3.1 Decodificação de binário para BCD

Para a decodificação de binário em BCD foi usado uma memória ROM 12x16 (12 bits de endereço para 16-bits de valor), onde a entrada é um valor binário que corresponde a um endereço na memória, e a saída é o valor da célula selecionada, correspondente a representação BCD do valor de entrada.

Abaixo está mostrado o diagrama do circuito do decodificador.

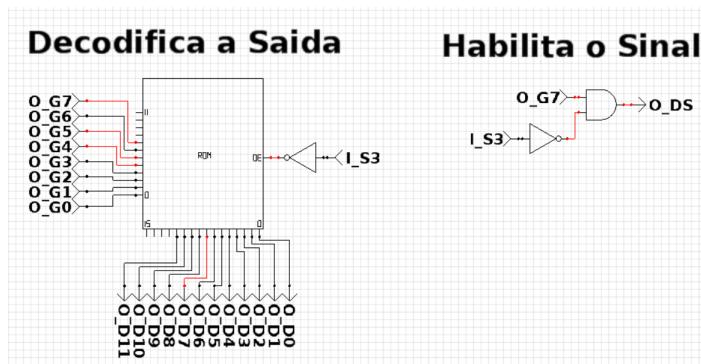


Figura 6 - Diagrama do Decodificador binário para BCD

Como exemplo, caso o resultado da ULA seja 33, será escolhido na memória a posição de endereço 0x33, com conteúdo 51, o BCD equivalente.

Como os displays só funcionam se a UA for escolhida, o decodificador só funcionará se S_3 for 0 (ou seja, ativa o pino Output Enable, OE). Também há um sinal de controle que habilita o sinal (caso o número seja negativo), baseado no bit mais significativo do resultado, caso a operação seja aritmética.

3.5.3.2 Codificação de BCD para binário

Para a codificação de BCD para binário usado na entrada foi usado uma memória ROM 16x16 (16 bits de endereço para 16-bits de valor), onde a entrada é um valor BCD de três dígitos (ou seja, 12-bits, os 4 bits restantes são usados para o sinal) que corresponde a um endereço na memória, e a saída é o valor da célula selecionada, correspondente a representação binária do valor de entrada. Como tem duas entradas então foram necessários dois codificadores.

Abaixo está mostrado o diagrama do circuito dos codificadores.

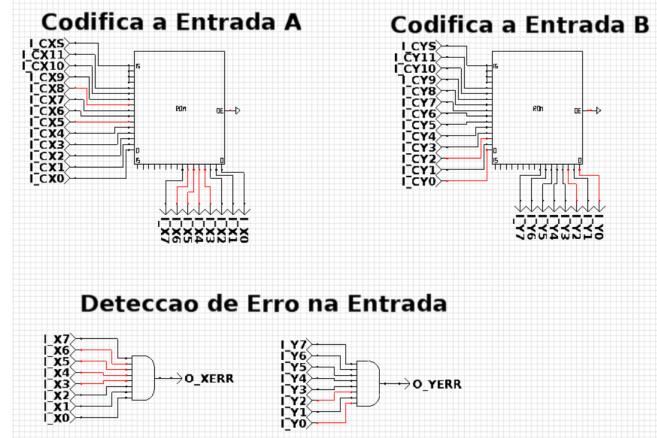


Figura 7 - Diagrama do Codificador BCD para binário

O sinal é aplicado aos 4 bits superiores do “endereço”, prefixando os valores negativos com 0xF. Como exemplo, caso seja inserido 51, será escolhido na memória a posição de endereço 0x51, com conteúdo 0x33, o hexadecimal equivalente. Caso seja inserido -128, será escolhida a posição 0xF128 com conteúdo 0x80.

As posições referentes à entradas inválidas foram preenchidas com 0xFFFF, o que significa que caso seja inserido alguma entrada inválida na saída terá um valor 0xFFFF. O valor de saída é verificado abaixo dos codificadores: caso seja igual a 0xFFFF então houve um erro na entrada dos dados e um sinal será emitido para um dos LEDs na entrada. Perceba que cada operando de entrada tem um LED sinalizando isso.

3.5.3.3 Memória ROM (Read Only Memory)

Cada um dos dois tipos de conversores tem um mapa de memória diferente, já que o decodificador mapeia binário para BCD e o codificador mapeia de BCD para binário. Portanto os mapas de memória refletem essa característica.

Abaixo está o mapa de memória do decodificador de binário para BCD, acoplado à saída da ULA, que converte os valores binários processados na ULA para valores BCD a serem mostrados nos displays de 7 segmentos.

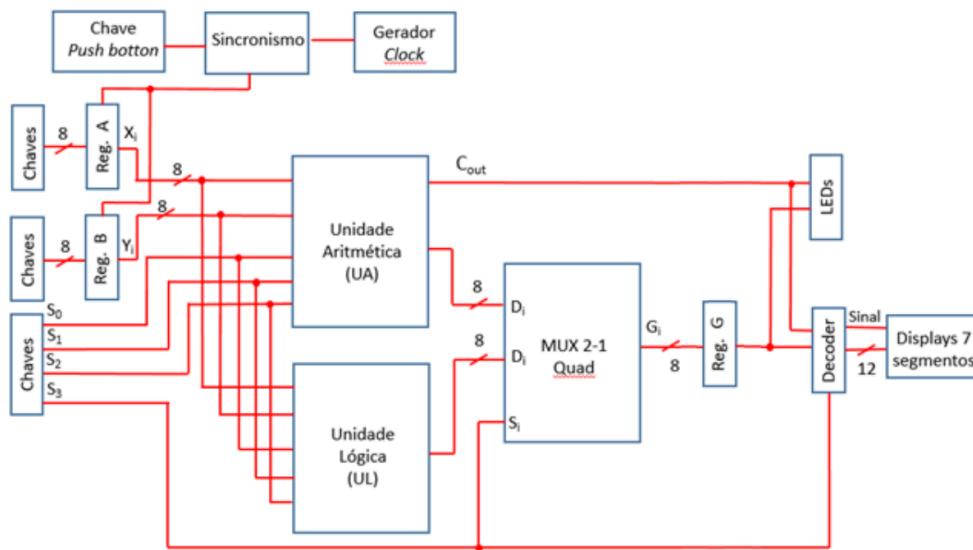


Figura 10 - Diagrama do projeto completo da ULA

3.6.1 Lista de componentes

Abaixo está a lista completa dos componentes usados no projeto.

| Part Number CI | Quantidade | Descrição |
|----------------|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CI 74283 | 2 | Um somador completo de 4 bits com duas entradas de 4 bits (A0-A3, B0-B3), uma entrada de carry (C _{in}) e uma saída de carry (C _{out}). |

Tabela 4 - Lista de componentes lógicos do projeto da ULA

4. Simulações e avaliação de resultados

Abaixo estão os resultados dos testes realizados no projeto, tanto da UA quanto da UL, de forma a testar todas as operações.

4.1 Testes da Unidade Aritmética

Abaixo estão os testes da unidade aritmética (UA). Todos os testes realizados estão alinhados com os resultados esperados, conforme mostra a tabela abaixo.

| S ₃ S ₂ S ₁ S ₀ | Operação | X _i (X ₇ X ₆ X ₅ X ₄ X ₃ X ₂ X ₀) | Y _i (Y ₇ Y ₆ Y ₅ Y ₄ Y ₃ Y ₂ Y ₁ Y ₀) | LEDs (C _{out} G ₇ G ₆ G ₅ G ₄ G ₃ G ₂ G ₁ G ₀) | DISPLAY |
|-------------------------------------------------------------|-------------|----------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|---------|
| 0000 | MOV G, A | 0010 0101 | 0011 0111 | 0 0010 0101 | 37 |
| 0001 | INC G, A | 0101 1010 | - | 0 0101 1011 | 91 |
| 0010 | DEC G, A | 0101 1010 | - | 1 0101 1001 | 89 |
| 0011 | ADD G, A, B | 0111 1111 | 1000 1000 | 1 0000 0111 | 7 |
| 0100 | MOV G, B | - | 0011 0111 | 0 0011 0111 | 55 |
| 0101 | INC G, B | - | 0101 0010 | 0 0101 0011 | 83 |
| 0110 | DEC G, B | - | 1011 0001 | 1 1011 0000 | -80 |
| 0111 | SUB G, A, B | 0001 1001 | 0101 1010 | 0 1011 1111 | -65 |



Tabela 5: Resultados da simulação dos testes de validação da UA

Abaixo estão as imagens referentes aos testes da UA:

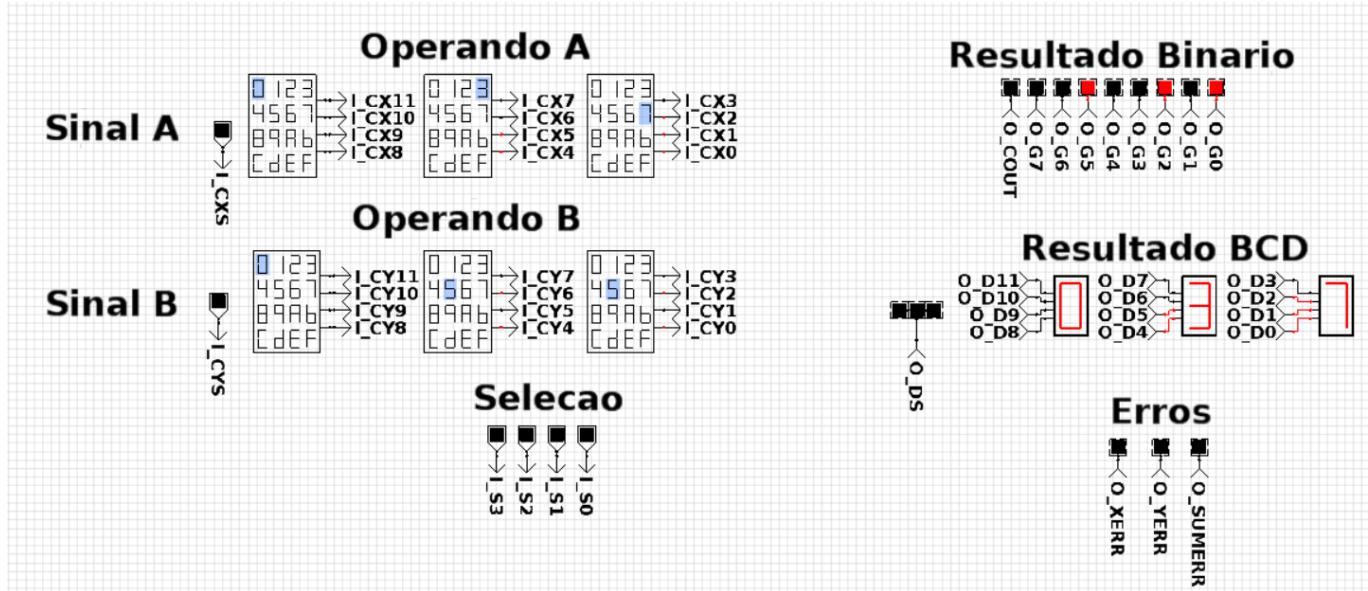


Figura 11 - Teste da operação MOV G, A

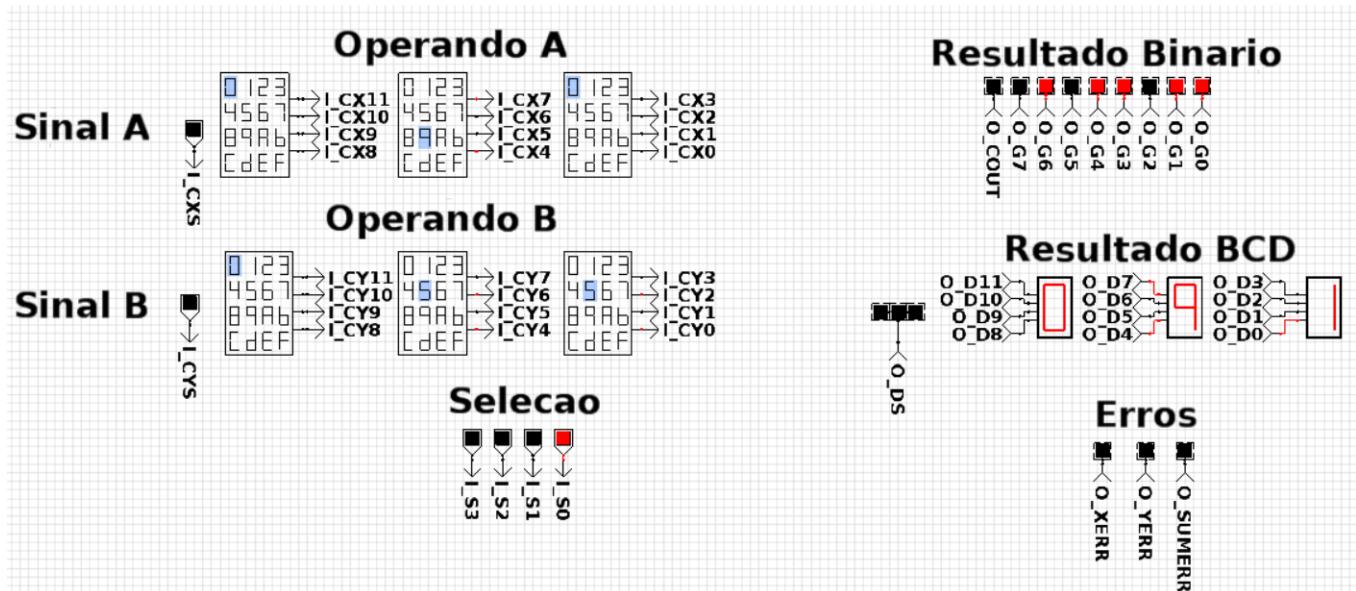


Figura 12 - Teste da operação INC G, A

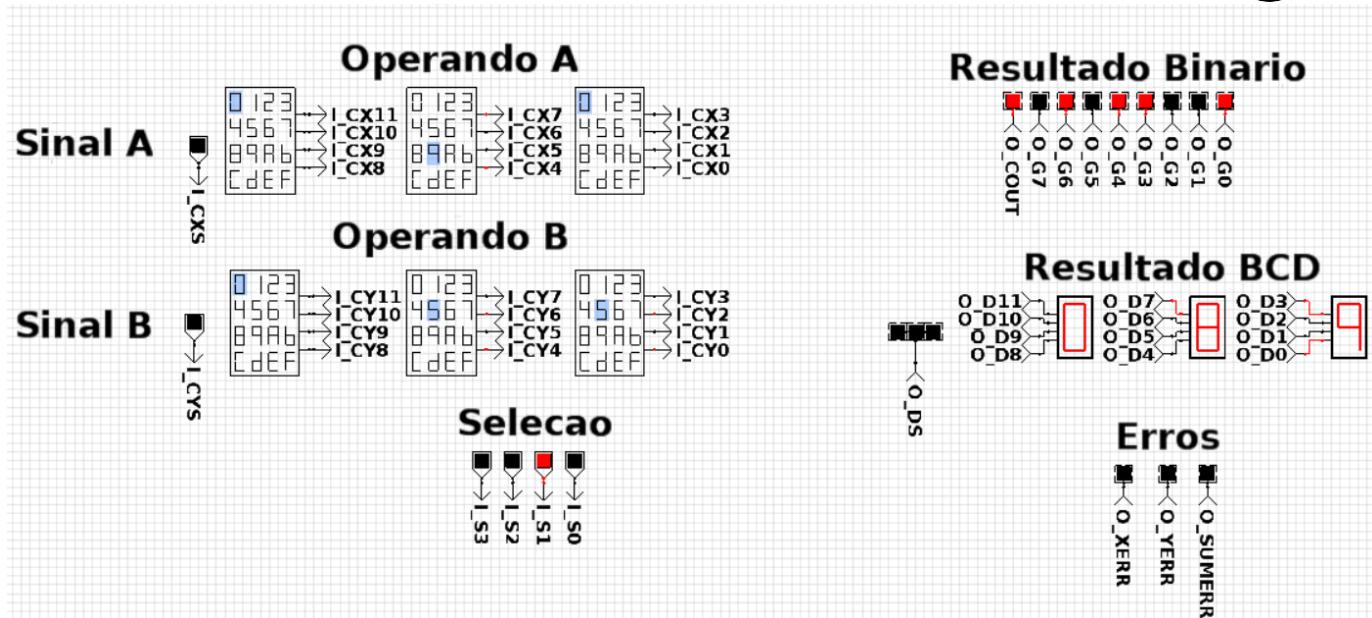


Figura 13 - Teste da operação DEC G, A

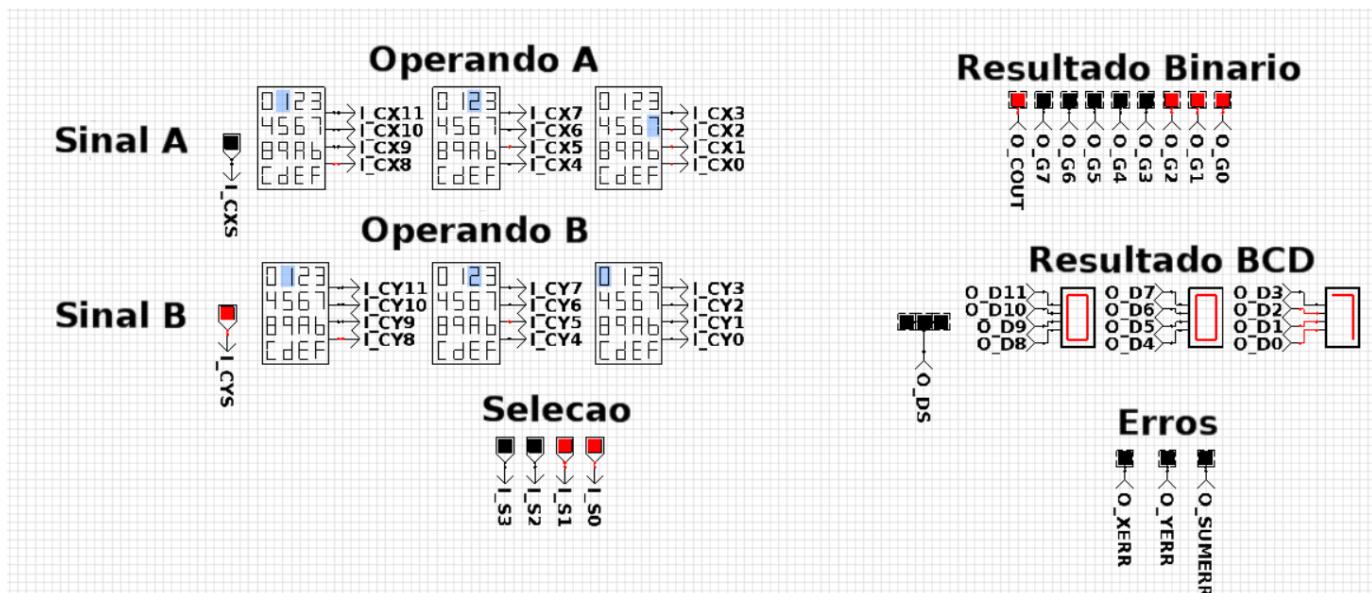


Figura 14 - Teste da operação ADD G, A, B

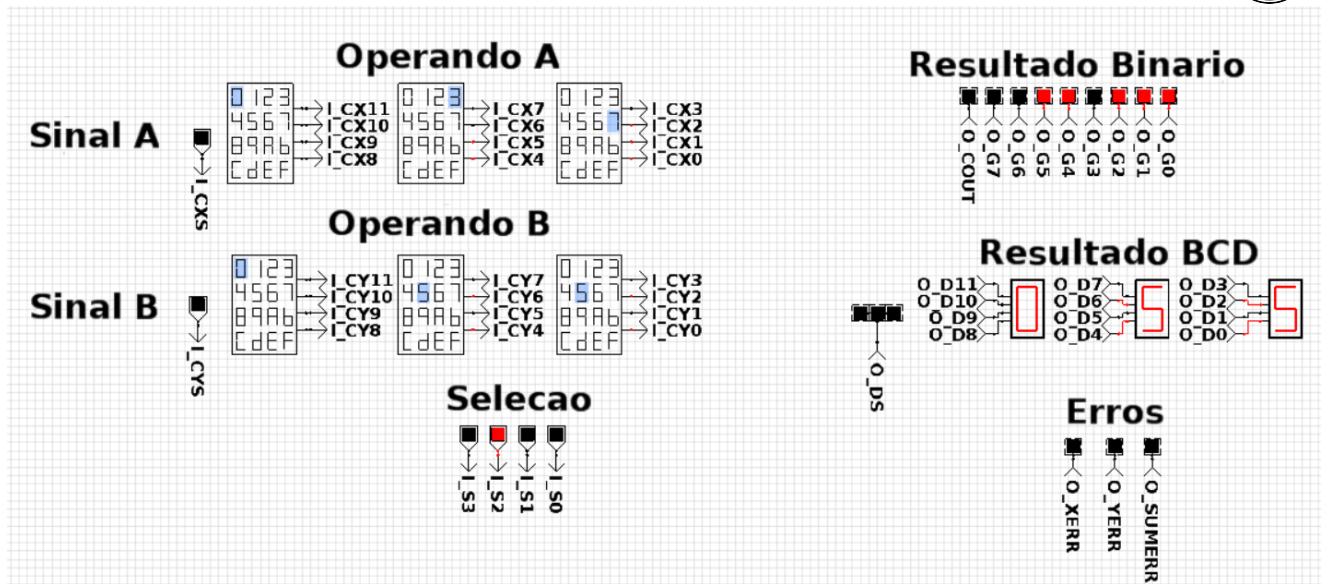


Figura 15 - Teste da operação MOV G, B

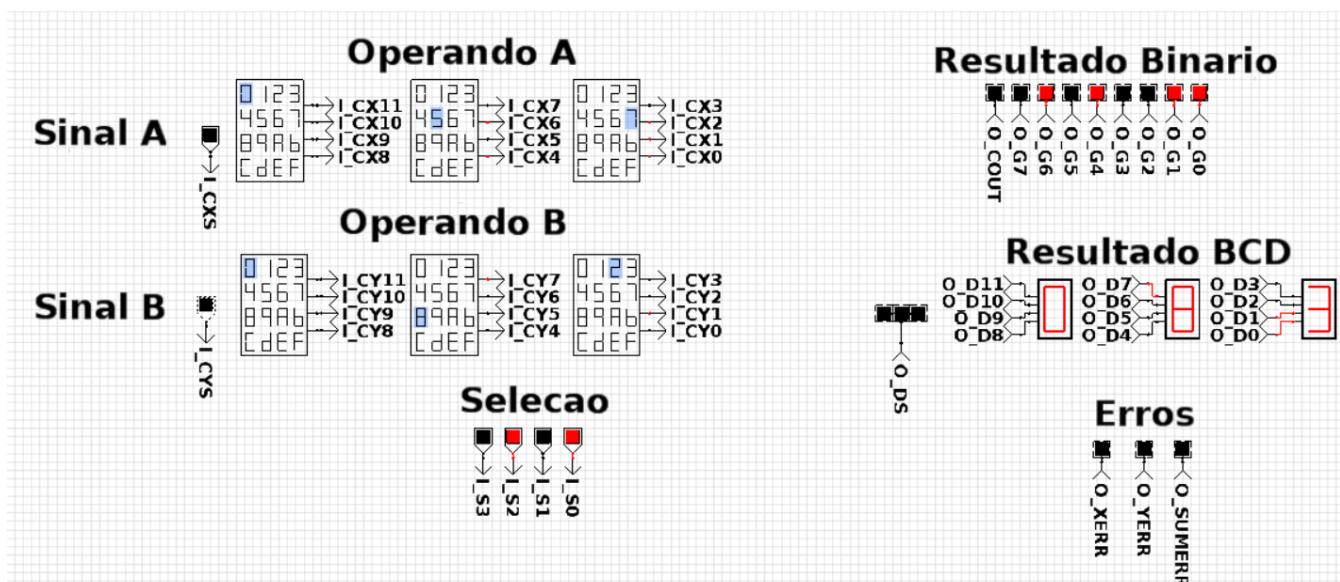


Figura 16 - Teste da operação INC G, B

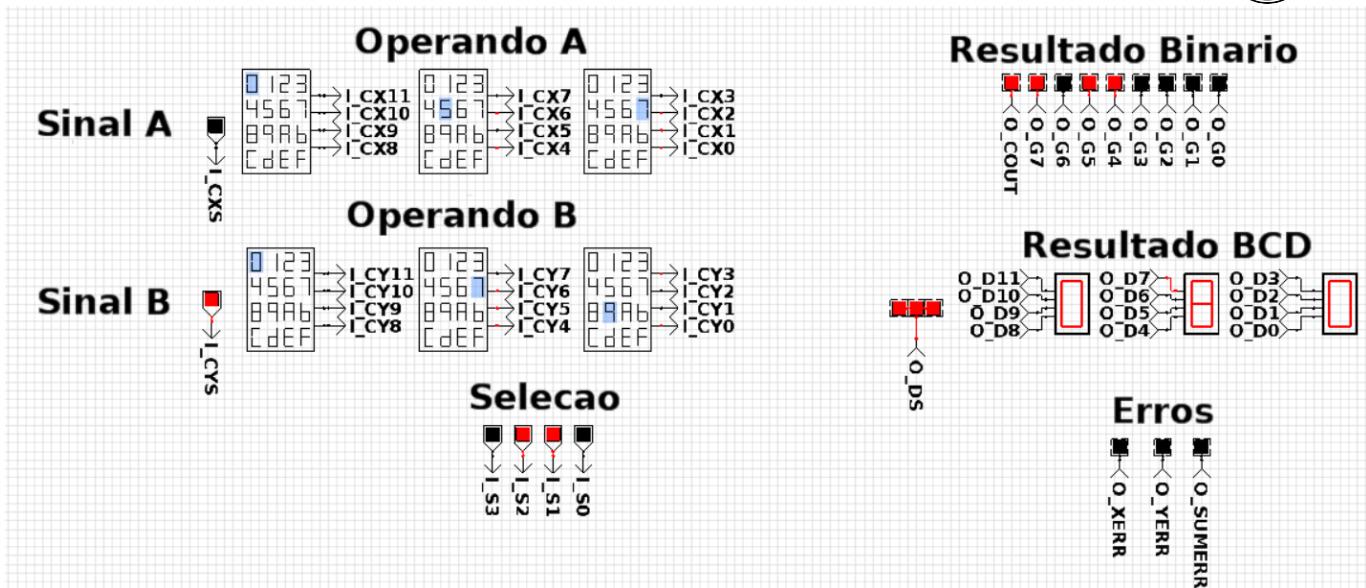


Figura 17 - Teste da operação DEC G, B

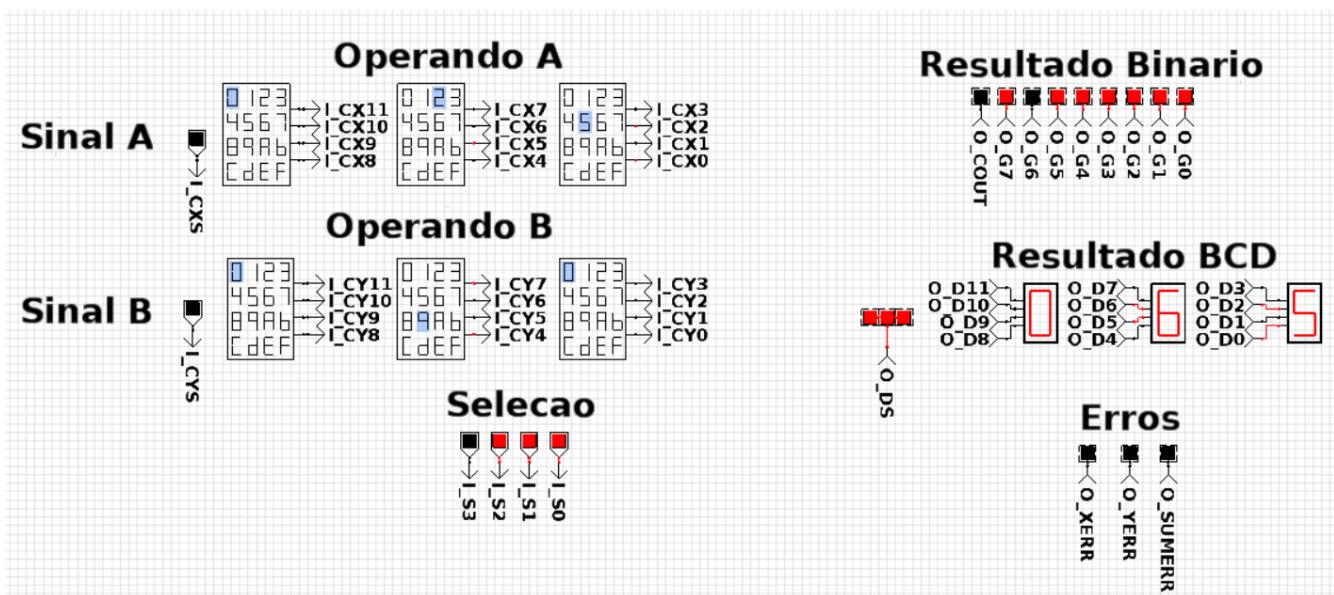


Figura 18 - Teste da operação SUB G, A, B

É possível ver que todos os testes executam perfeitamente, funcionando adequadamente os sinais de entrada e de saída. Valores inválidos resultam nas flags de erros. Como não houve erros, os LEDs não foram ativados.

Abaixo estão alguns exemplos de erros, sinalizados usando os LEDs inferiores, com as entradas X e Y inválidas (F7 e AB respectivamente) e o resultado da soma inválido (99 + 89 resulta em overflow). A flag O_SUMERROR é ativada caso qualquer operação resulte em erro de overflow.

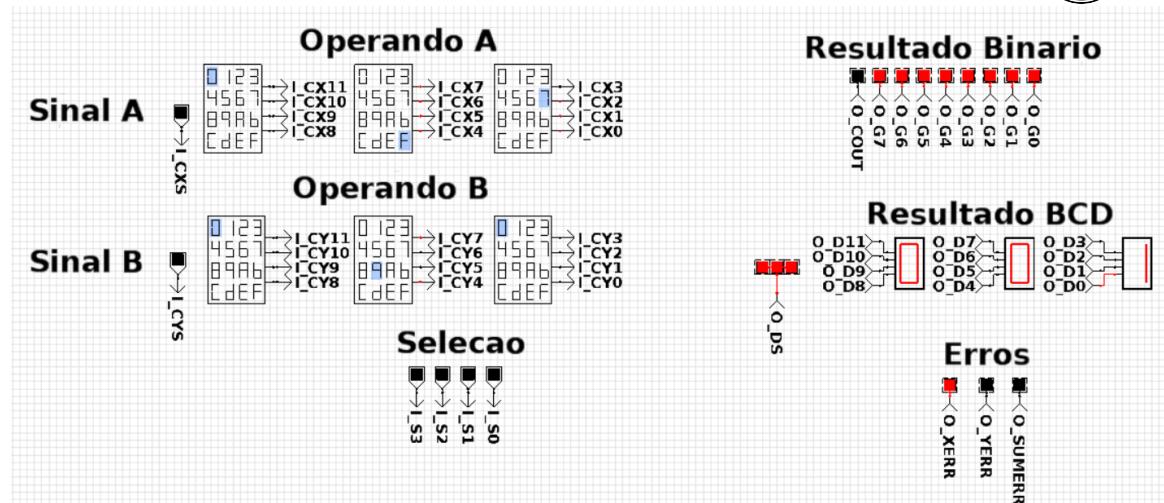


Figura 19 - Exemplo de entrada A inválida

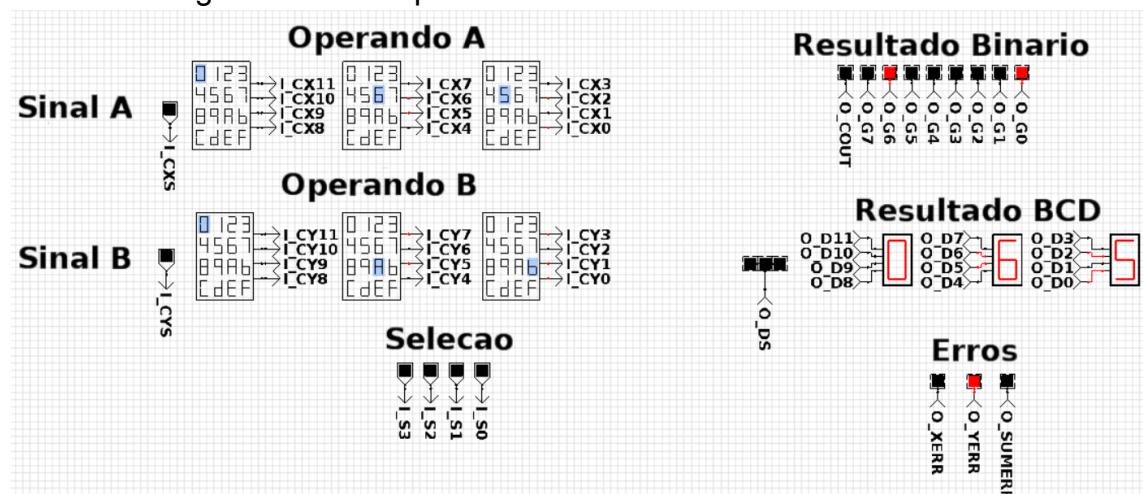


Figura 20 - Exemplo de entrada B inválida

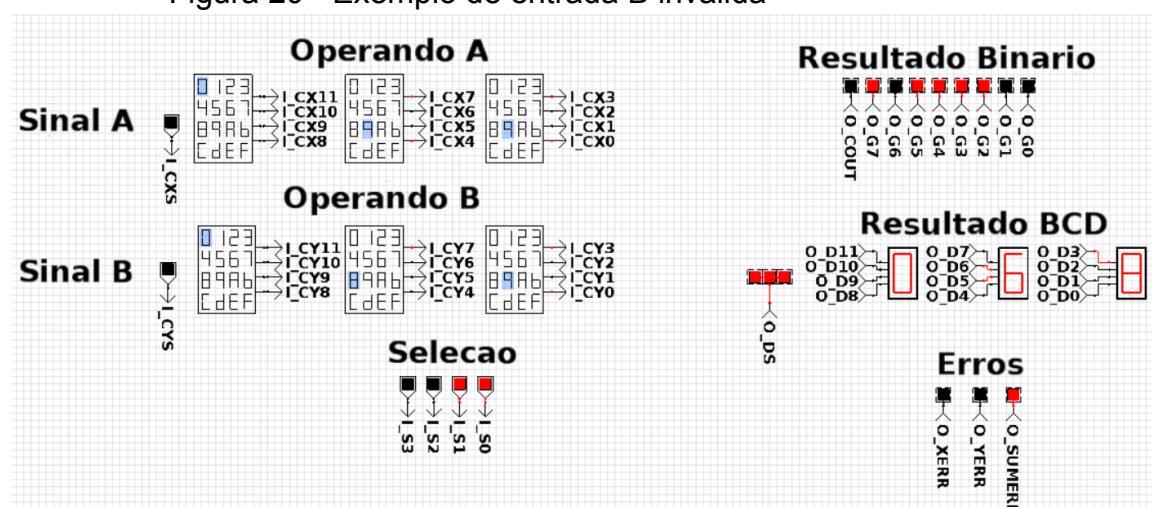


Figura 21 - Exemplo de soma inválida



4.2 Testes da Unidade Lógica

Abaixo estão os testes da unidade lógica (UL). Todos os testes realizados estão alinhados com os resultados esperados, conforme mostra a tabela abaixo.

| $S_3 S_2 S_1 S_0$ | Operação | $X_i(X_7 X_6 X_5 X_4 X_3 X_2 X_0)$ | $Y_i(Y_7 Y_6 Y_5 Y_4 Y_3 Y_2 Y_1 Y_0)$ | LEDs ($C_{out} G_7 G_6 G_5 G_4 G_3 G_2 G_1 G_0$) | DISPLAY |
|-------------------|-------------|------------------------------------|----------------------------------------|-------------------------------------------------------|---------|
| 1X00 | AND G, A, B | 0001 0001 | 0011 0011 | 0 0001 0001 | - |
| 1X01 | OR G, A, B | 0001 0001 | 0011 0011 | 0011 0011 | - |
| 1X10 | XOR G, A, B | 0001 0001 | 0011 0011 | 0010 0010 | - |
| 1X11 | XNR G, A, B | 0001 0001 | 0011 0011 | 1101 1101 | - |

Tabela 6: Resultados da simulação dos testes de validação da UL

Abaixo estão as imagens referentes aos testes da UL:

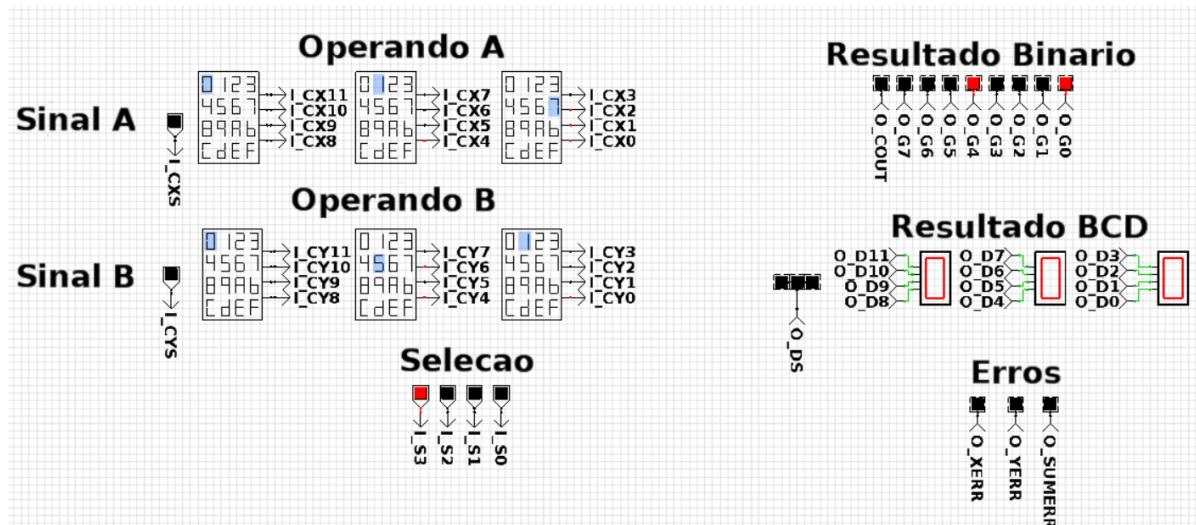


Figura 22 - Teste da operação AND G, A, B

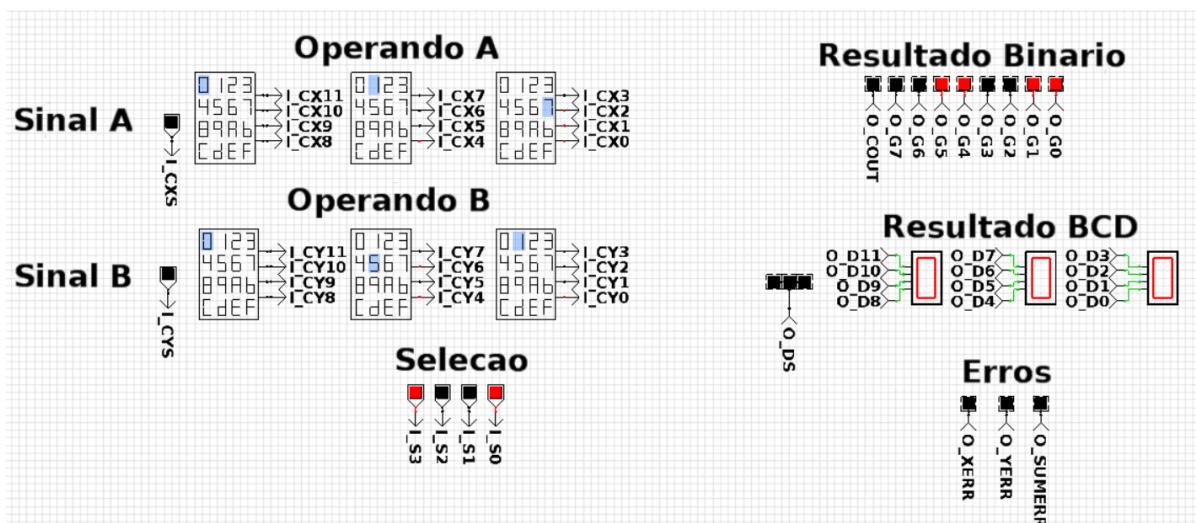




Figura 23 - Teste da operação OR G, A, B

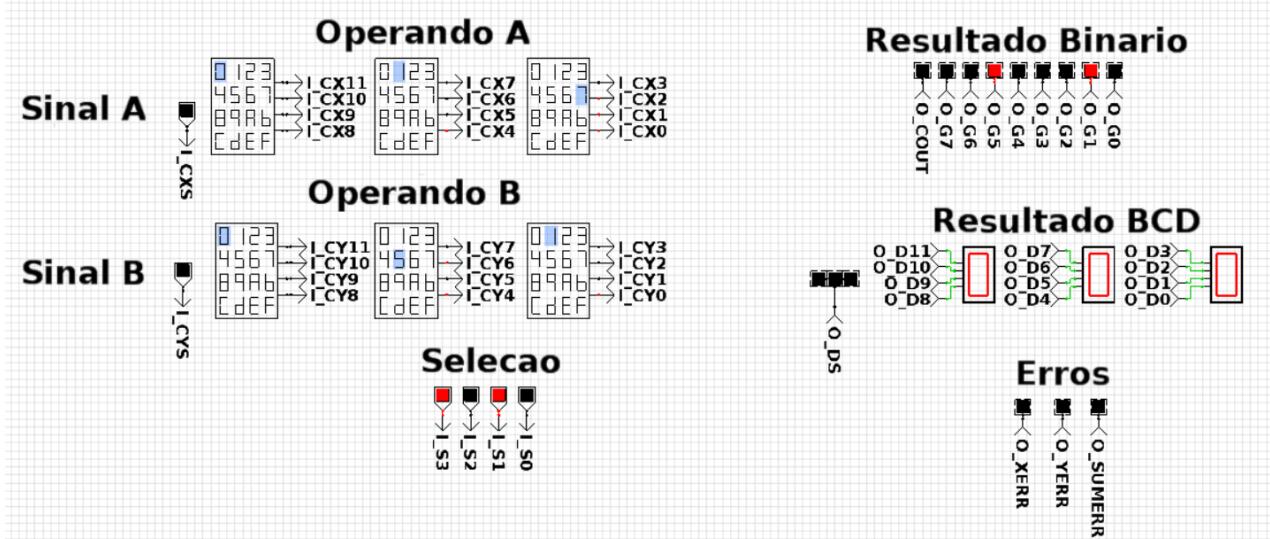


Figura 24 - Teste da operação XOR G, A, B

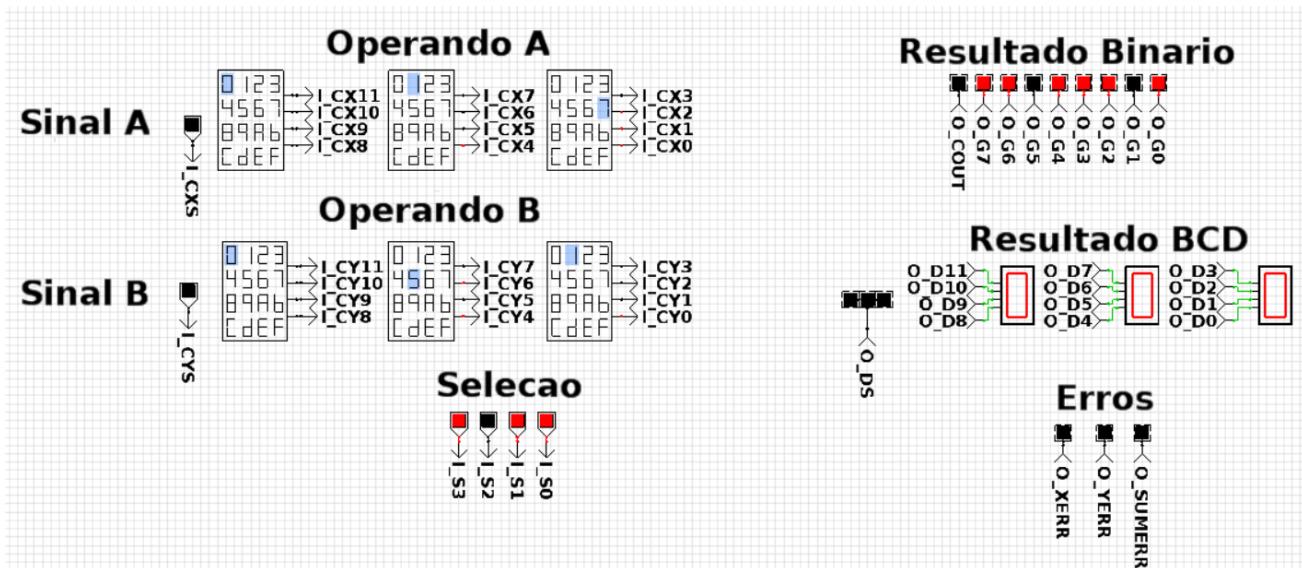


Figura 25 - Teste da operação XNR G, A, B

É possível ver que todos os testes executam perfeitamente. Algo que poderia ser melhorado é conseguir inserir os dados em outra representação, já que operações bit-a-bit são melhor utilizadas com valores sem sinal. Dessa forma, uma opção de colocar os valores dos operandos em hexadecimal ou binário seria uma boa adição (sem BCD).

Fora isso, a execução dos comandos está ótima, omitindo perfeitamente o uso dos displays ao usar a UL.



5. Conclusões finais

Ao longo deste projeto, aprendi muito sobre o funcionamento e na implementação de uma ULA de 8 bits enquanto a implementava. Comecei com uma compreensão básica dos princípios de operações lógicas e aritméticas, e avancei para a concepção e implementação prática de uma unidade que pode executar essas operações em dados de 8 bits.

Durante esse processo, aprendi sobre a importância da organização e da sincronização desses componentes para garantir o funcionamento adequado da unidade como um todo.

Uma das partes mais desafiadoras e gratificantes deste projeto foi a fase de implementação. Ao longo deste projeto, desenvolvi minhas habilidades de resolução de problemas e pensamento crítico, encontrando e superando desafios à medida que surgiam. O projeto ficou muito bom e acredito que uma unidade de controle seria uma grande adição ao projeto.

Tive algumas dificuldades, como usar uma memória ROM como conversor, algo que nunca tinha feito antes, o que gerou um grande aprendizado. O fato do projeto ter uma codificação BCD na entrada gerou uma limitação: só pode ter valores de -99 a 99, já que não conseguia colocar valores maiores com 99, pois demandaria verificações mais complexas e.g. testar se o valor inserido era válido. Mas ainda assim acredito que no contexto do projeto dois dígitos sejam o suficiente!

Em resumo, este projeto não apenas me proporcionou uma compreensão mais profunda dos princípios fundamentais da computação, mas também me equipou com habilidades práticas de design e implementação que serão inestimáveis em minha jornada contínua no campo da engenharia de computação. Estou ansioso para aplicar o que aprendi aqui em projetos futuros e continuar explorando as complexidades fascinantes do mundo da computação digital.