

PROJETO

ZEPTOPROCESSADOR-III de 16 bits

Aquila Macedo Costa, 20/2021800
Eduardo Ferreira Marques Cavalcante, 20/2006368
Grupo G18

¹Dep. Ciência da Computação – Universidade de Brasília (UnB)
CIC0231 - Laboratório de Circuitos Lógicos

202021800@aluno.unb.br, 202006368@aluno.unb.br

Abstract. *This report will show the implementation of a 16 bit processor, made on in the software Deeds, with some programs that calculate certain basic operations.*

Resumo. *Nesse relatório será mostrado a implementação de um processador de 16 bits, feito no software Deeds, com alguns programas que calculam certas operações básicas.*

1. Introdução

Esse projeto se trata de um processador de 16 bits, ou seja, um sistema que faz leitura, decodificação e execução de um instrução em uma memória, e volta para as instruções seguintes. Partindo desse princípio foi desenvolvido um processador programável simples, o ZeptoProcessador-III, com apenas 13 instruções, mas capaz de executar programas com até 65536 instruções.

1.1. Objetivos

Objetiva-se apresentar um ZeptoProcessador-III que execute, implementados com dos conhecimentos adquiridos durante o semestre da matéria de Circuitos Lógicos e Laboratório de Circuitos lógicos, a seguintes instruções:

- 1) addi: Adição com imediato
- 2) subi: Subtração com Imediato
- 3) andi: AND bitwise com Imediato
- 4) ori: OR bitwise com Imediato
- 5) xori: XOR bitwise com Imediato
- 6) beq: Salto Condicional se igual
- 7) bne: Salto Condicional se diferente
- 8) ble: Salto Condicional menor ou igual (Signed)
- 9) bleu: Salto Condicional menor ou igual (Unsigned)
- 10) bgt: Salto Condicional maior que (Signed)

- 11) bgtu: Salto Condicional maior que (Unsigned)
- 12) jal: Salto incondicional ao endereço
- 13) jalr: Salto incondicional ao registrador

1.2. Materiais

Neste experimento foram utilizados os seguintes materiais e equipamentos:

- Software Deeds.

2. Metodologia

2.1. Registrador PC

O registrador PC (*program counter*) tem como onjetivo armazenar o endereço da instrução a ser executada, podendo ser somado o imediato ou 1. É conectado com o sinal de *Clock* e o sinal de RESET, com seu objetivo ser de calcular o próximo endereço para ser executado.

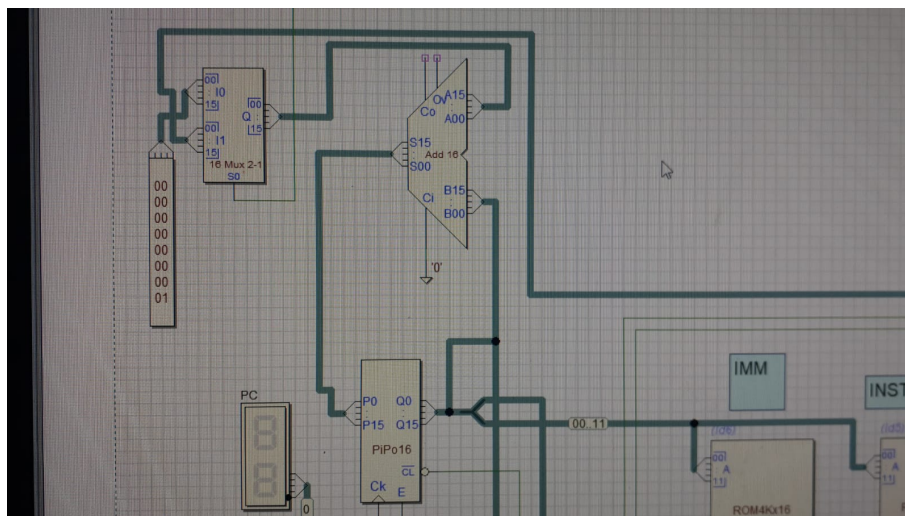


Figura 1. Registrador PC.

2.2. Memória de Instruções

Para execução das intruções, foi criada uma memória ROM (*Read Only Memory*), a qual é programável, sendo possível utilizar a técnica LUT (*Look Up Table*), permintindo a executar programas utilizando apenas os resultados de tabelas verdades.

Tal memória foi implementada utilizando módulos da memória ROM de 4K x 16 bits já encontradas no Deeds, com 16 bits para o imediato e 16 bits para a instrução, devido a instrução do ZeptoProcessador-III armazena instruções de 32 bits, assim uma ROM recebe os 16 bits menos significativos(instrução) e e outra ROM recebe os 16 bits mais significativos(imediato), como se pode ver a seguir:

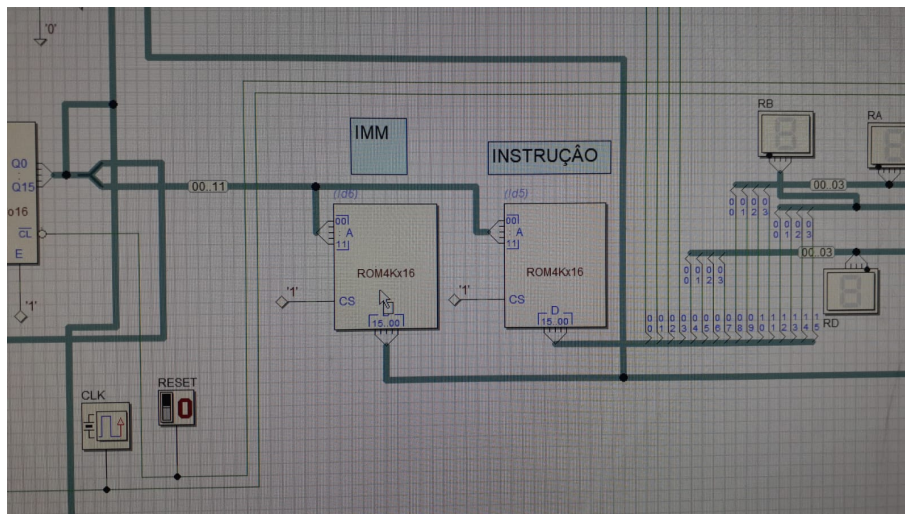


Figura 2. Memórias ROMs.

2.3. Banco de registradores

O banco de registradores implementado em bloco se trata de 16 registradores de 16 bits cada, permitindo armenar os dados utilizados nas intruções de 16 bits, com o Ra e Rb sendo registradores de leitura e o Rd como registrador de escrita, sendo possível a escrita somente com o $WE = 1$ e a borda de subida de clock, além do RESET reiniciar os registradores caso venha a subida de clock. Seguindo, em abstratamente, o seguinte esquemático:

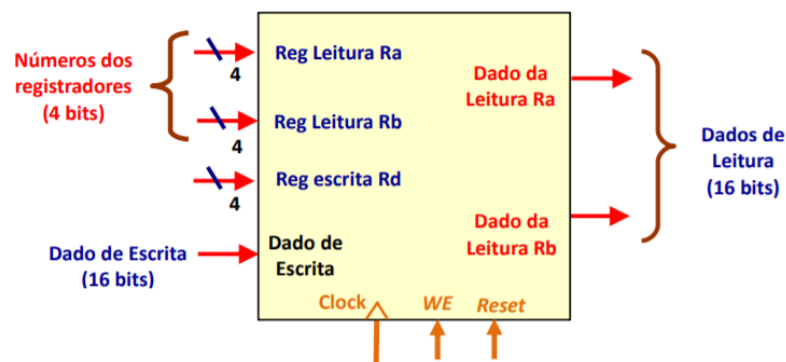


Figura 3. Banco de registradores. [Lamar 2021]

Com isso, chegou-se ao seguinte bloco que representa o banco de registradores:

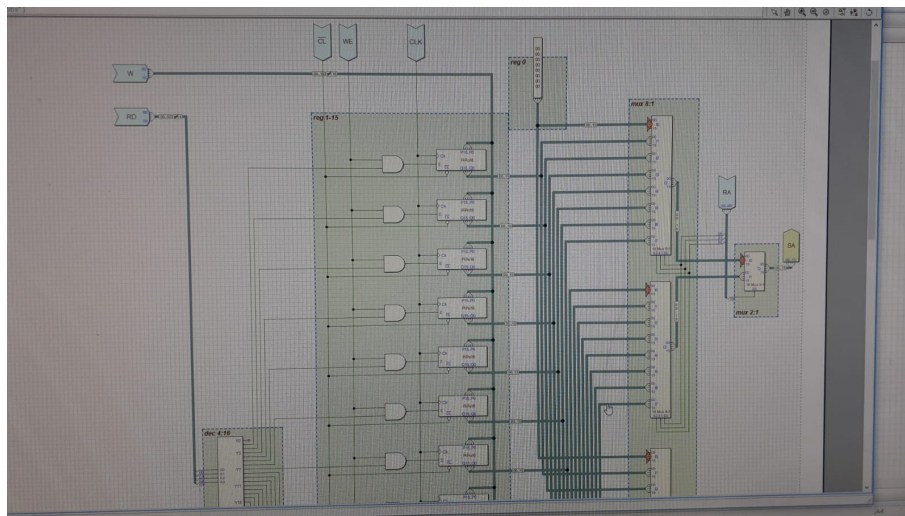


Figura 4. Banco de registradores parte 1.

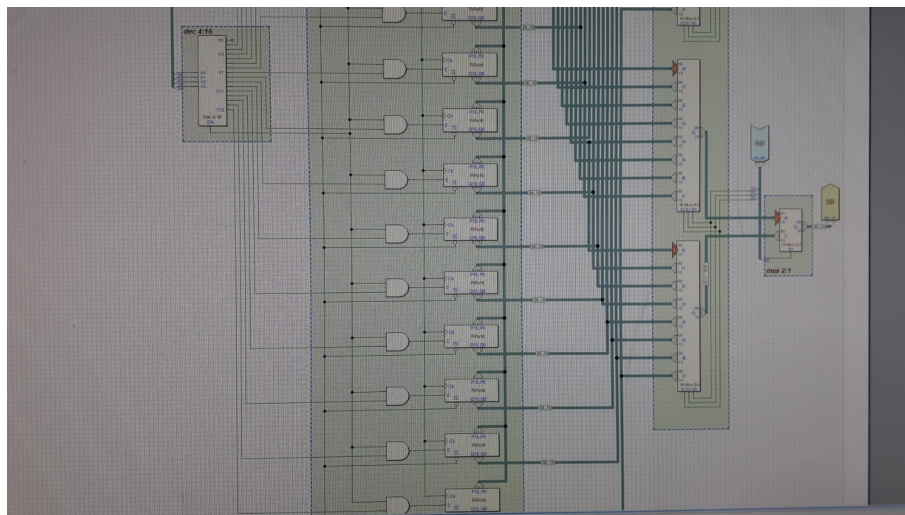


Figura 5. Banco de registradores parte 2.

É possível ver que o BR (Banco de Registradores) possui para a leitura 2 multiplexadores 8x1 e um multiplexador 2x1, para selecionar um dos endereços; e para a escrita possui um decodificador 4x16, recebendo o registrador Rd e o dado de escrita. O RESET limpa caso seja $RESET = 1$, e a escrita e leitura são comandadas pelo WE (*Write Enable*), caso seja $WE=0$, somente leitura ocorrerá, caso seja $WE=1$, somente escrita ocorrerá, com armazenamento no Rd.

2.4. Unidade Lógico-Aritmética (ULA)

A ULA permite as operações de soma e subtração de dois números, com o nosso bloco recebendo o opcode de 4 bits, mas sendo necessário utilizar somente 3 bits, que em seguida é transformado para 5 bits.

A seguir a ULA implementada (utilizada a do Deeds), que também recebe os dois números (A e B) de 16 bits, com a saída em 16 que logo após realizar a operação aritmética sua saída entra em outra ULA para realizar os cálculos com os imediatos.

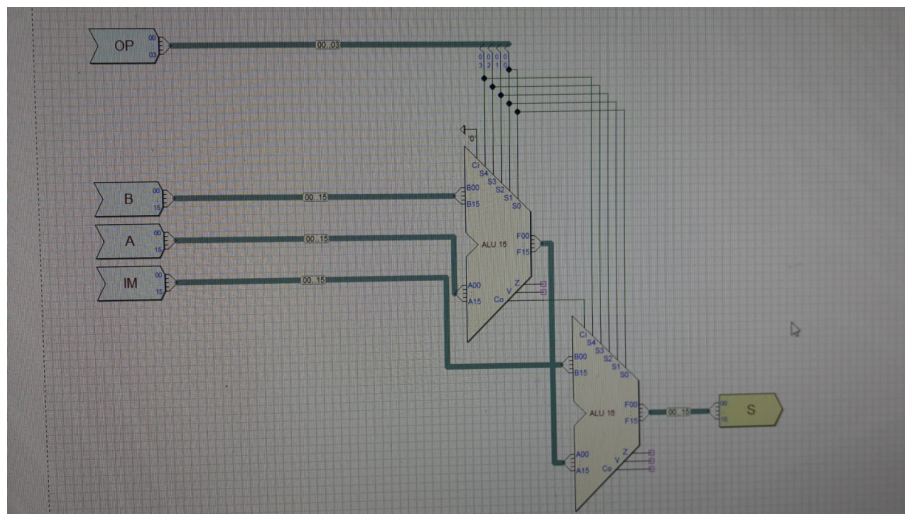


Figura 6. ULA.

2.5. Controlador da ULA

No controlador da ULA, existe um tipo de adaptação na qual realizamos uma operação lógica para realizar os calculos usando 3 bits, para que possamos usá-lo no controle geral do programa.

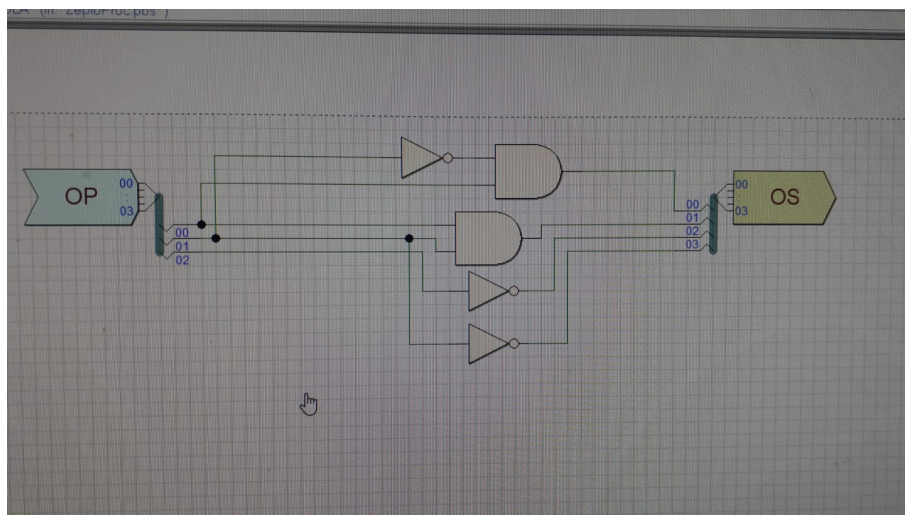


Figura 7. Controle da ULA.

2.6. Comparador

Este bloco recebe dois numeros de 16 bits da saída do Banco de Registrador, o SA e SB. Temos o SU, que indica se é uma comparação com sinal (SU=1) ou sem sinal (SU=0) e o opcode da instrução, e na saída indica se a condição é verdadeira, caso seja uma instrução de salto condicional. No final queremos verificar de o valor é igual ou menor igual, para isso usamos dois comparadores e uma ULA para realizar as operações.

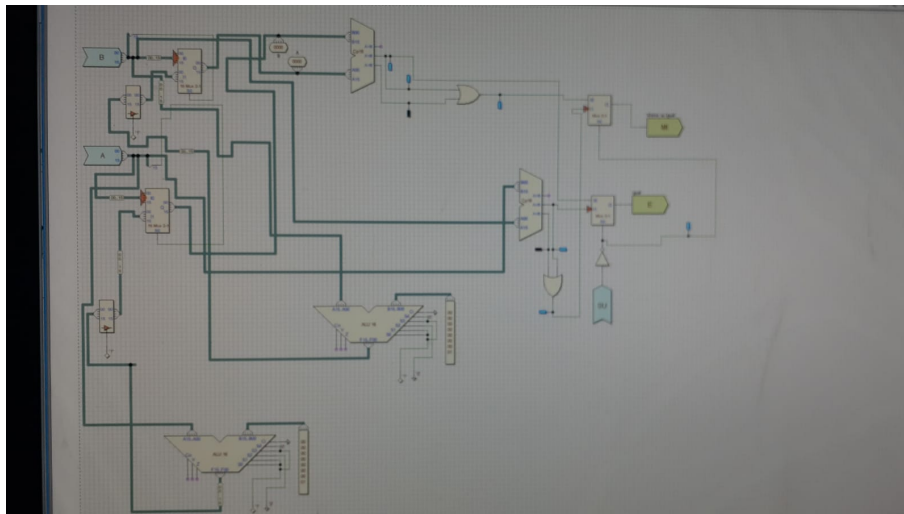


Figura 8. Comparador.

2.7. Bloco de Controle

O bloco de controle gera os sinais WE do BR, Op da ULA, SU do comparador e seleção de multiplexadores (selecionar o menor ou igual do comparador, por exemplo)

Se aprofundando um pouco, digamos que é o cérebro do projeto, onde organizamos quase toda a lógica dos blocos, pois é onde tratamos o OPCODE que por sua vez irá indicar qual instrução iremos utilizar para os cálculos, a saída SU indica se a operação deve considerar números com (SU=1) ou sem sinal (SU=0).

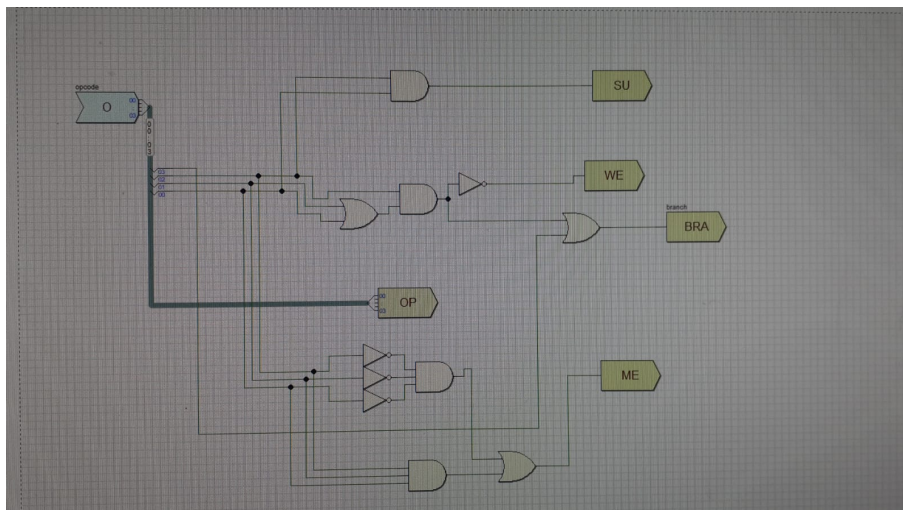


Figura 9. Bloco de controle.

3. Resultados obtidos

3.1. Multiplicação de dois números sem sinal

Para a multiplicação de dois números sem sinal, os seguintes números hexadecimais precisam ser implementados na ROM:

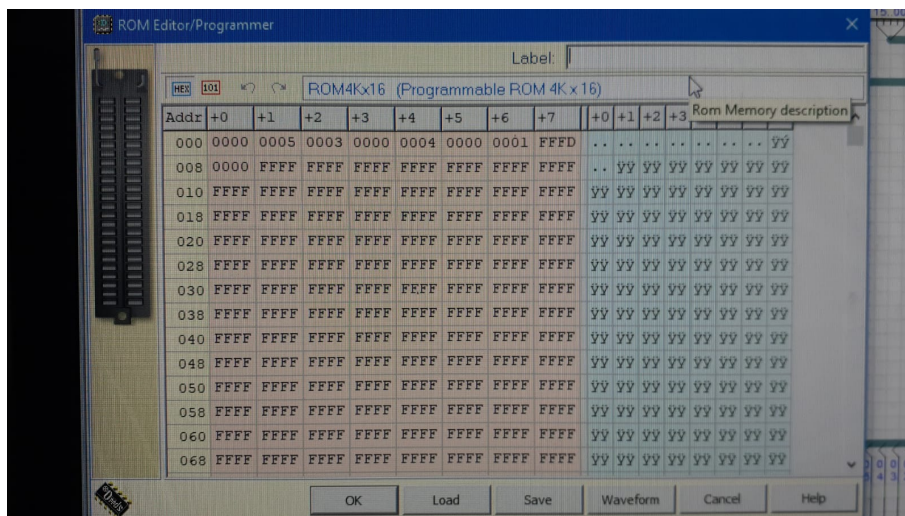


Figura 10. ROM imediato.

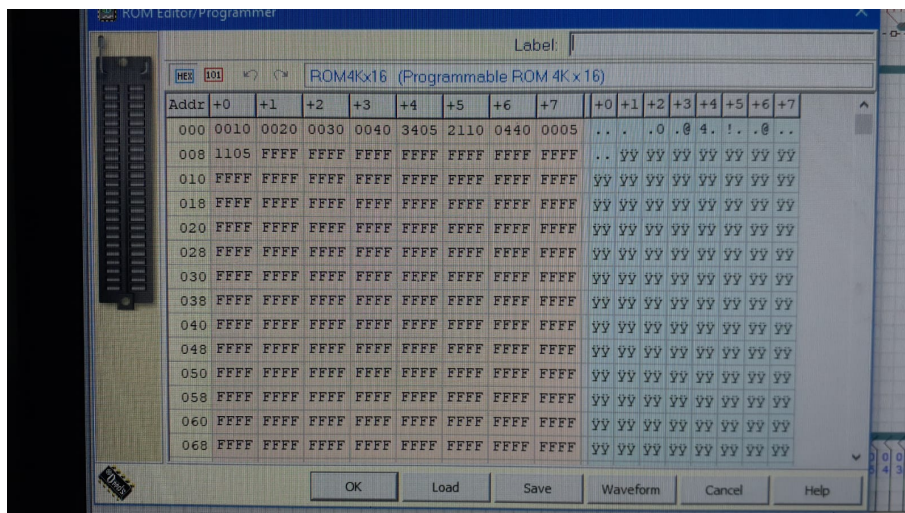
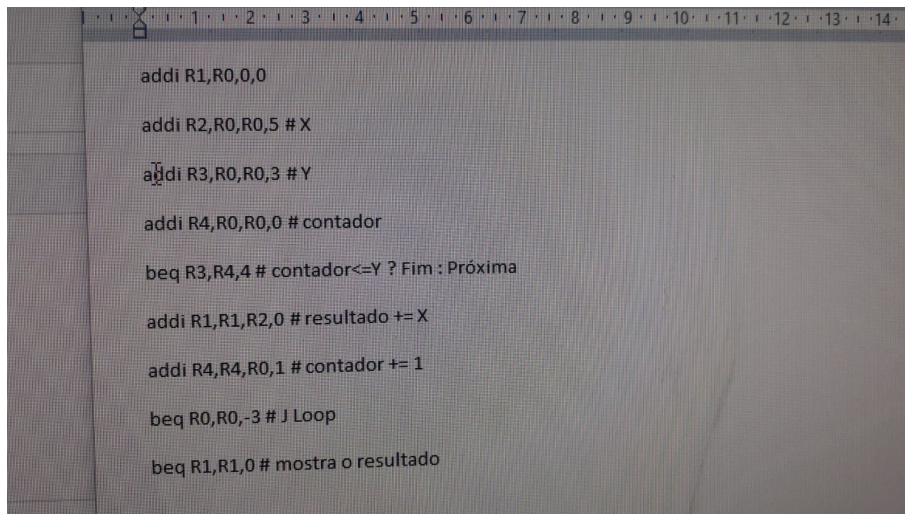


Figura 11. ROM instrução.

As duas memórias representam a seguinte lógica de instruções:



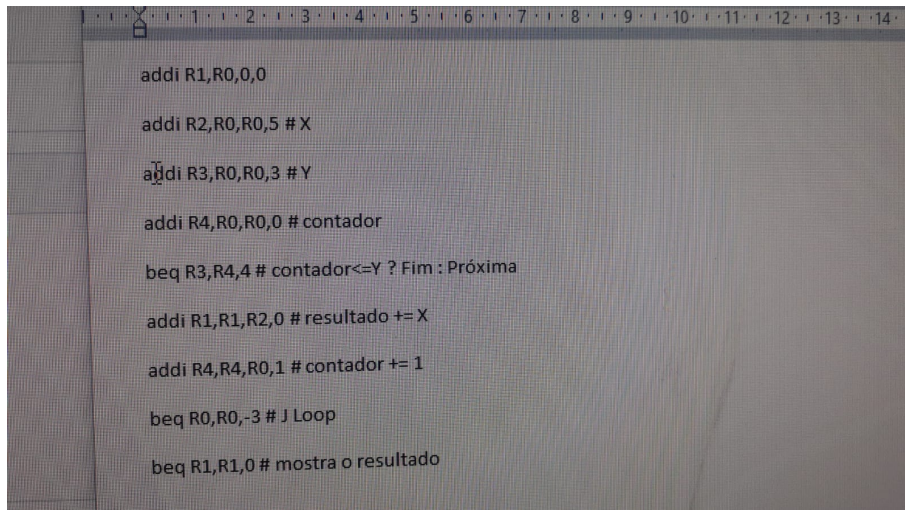
```

1 2 3 4 5 6 7 8 9 10 11 12 13 14
addi R1,R0,0,0
addi R2,R0,R0,5 # X
addi R3,R0,R0,3 # Y
addi R4,R0,R0,0 # contador
beq R3,R4,4 # contador<=Y ? Fim : Próxima
addi R1,R1,R2,0 # resultado += X
addi R4,R4,R0,1 # contador += 1
beq R0,R0,-3 # J Loop
beq R1,R1,0 # mostra o resultado

```

Figura 12. Lógica de operações.

<https://www.youtube.com/watch?v=1QZeQkZY6Ug>. aqui o link do vídeo mostrando sua execução com a multiplicação de 3×5 , o qual deve resultar em F na base hexadecimal (15 em decimal), com a maior frequência possível sendo de 6.25 MHz, caso seja maior, o procedimento de repetição acaba não ocorrendo e sinais de entrada se tornam discrepantes. A seguir a simulação temporal, com resultado sendo o Ra mais abaixo:



```

1 2 3 4 5 6 7 8 9 10 11 12 13 14
addi R1,R0,0,0
addi R2,R0,R0,5 # X
addi R3,R0,R0,3 # Y
addi R4,R0,R0,0 # contador
beq R3,R4,4 # contador<=Y ? Fim : Próxima
addi R1,R1,R2,0 # resultado += X
addi R4,R4,R0,1 # contador += 1
beq R0,R0,-3 # J Loop
beq R1,R1,0 # mostra o resultado

```

Figura 13. Simulação de ondas da multiplicação de dois números sem sinal na maior frequência (6.25 MHz).

4. Conclusão

Vimos na parte 3.1 que o processador tem limites do quão rápido fazer suas operações, limitado tanto pelas portas lógicas aplicadas, as quais possuem atrasos, quanto pelos circuitos utilizados do próprio Deeds, como a ROM e os registradores.

Acabamos por não implementar tudo que queríamos no nosso processador, como a multiplicação de números com sinal e outros programas, além de fazermos cada bloco

sem muito teste individual. No entanto, nos deparamos com problemas os quais foram interessantes de serem analisados e esperamos uma performance melhor no futuro, além de implementarmos corretamente grande parte das instruções.

Referências

[Lamar 2021] Lamar, V. (2021). Roteiro do trabalho. *Laboratório de Circuitos Lógicos - Roteiro do Trabalho*, pages 1–9.

[Mandelli 2021] Mandelli, M. G. (2021). Slides. *Aulas e slides apresentados pelo professor para auxiliar no trabalho final de LCL da matéria Circuitos Lógicos*.

[Lamar 2021] [Mandelli 2021] [?]