Centro Federal de Educação Tecnológica de Minas Gerais
Departamento de Computação – Laboratório de arquitetura e organização de computadores II

Relatório da Prática 2

Ulisses Andrade Carvalho

01 de outubro de 2023

Decisões e detalhes do projeto

A seguir serão abordados em formato de tópicos alguns pontos relevantes do projeto.

1. A memória funciona na borda de descida do clock e processador na borda de subida.

O funcionamento em bordas contrárias, permite que a memória faça a leitura ou escrita de um dado no entre duas bordas de subido. Isso faz com que o processador não tenha que esperar nenhum ciclo a mais quando utiliza a memória. A Imagem 1 ilustra essa modificação.

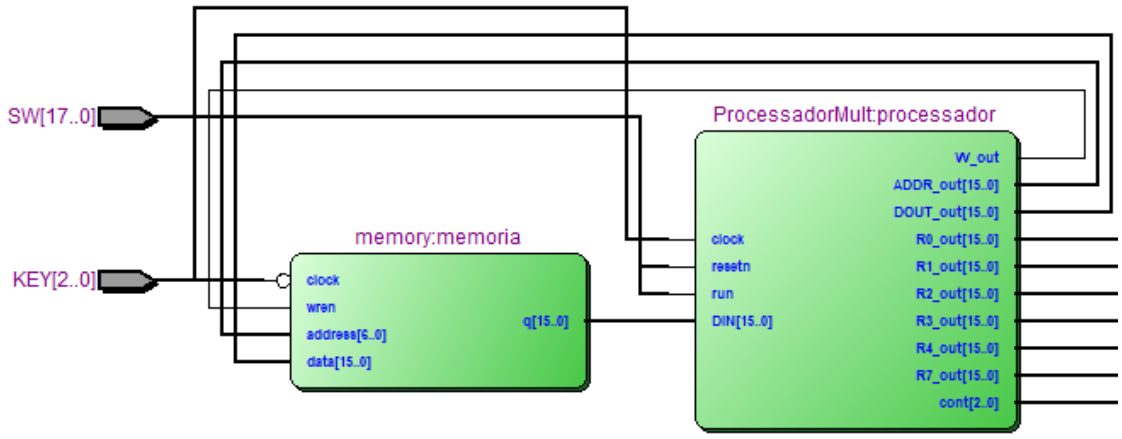

IMAGEM 1: Diagrama de módulos - Memória e processador

2. Aumento do número de bits do contador, de 2 bits para 3.

Essa mudança foi realizada, pois ao adicionar o registrador ADDR, para a passagem do endereço a memória, se tornou necessário acrescentar em todas instruções mais um estágio para a escrita nesse registrador. Sendo assim as instruções que usavam a ULA que antes gastavam 4 ciclos passaram a gastar 5 ciclos.

3. Passagem do valor do registrador G para o controlador.

Essa alteração surgiu como uma solução para a implementação das instruções MVNZ e SLT. Essas instruções tem uma condição para a realização de alguma ação, sendo que essa condicional depende do valor armazenado no registrador G.

4. Número de estágios para a execução de cada instrução:

Instruções que utilizam a ULA (SRL, SLL, SLT, OR, SUB e ADD) gastam 5 ciclos. Sendo 2 ciclos para a leitura da instrução na memória, 1 para passar o valor de um registrador (R0 a R7) para o registrador A, 1 para passar outro valor de um registrador (R0 a R7) para a ULA e escrever o resultado da operação da ULA no registrador G e por fim mais 1 ciclo para escrever o valor de G no primeiro registrador informado pela instrução.

Instruções que acessam a memória (LD e ST) gastam 4 ciclos, 2 para leitura da instrução na memória e mais 2 ciclos para acessar novamente a memória, tanto para escrita quando para leitura.

Instruções que movem valores de registradores (MV e MVNZ) gastam 3 ciclos. Sendo 2 ciclos para a leitura da instrução na memória, e 1 ciclos para passar o valor de um registrador para o outro. No caso do MVNZ, essa instrução verifica o valor do registrador G antes de realizar a movimentação dos valores.

Por fim a instrução MVI gasta 4 ciclos, sendo 2 para leitura da instrução na memória e depois mais 2 ciclos para ler novamente a memória no endereço passado pelo registrador PC que agora representa um valor inteiro, o qual será escrito em um registrador informado pela instrução lida anteriormente.

Simulações

- Em cada simulação o sinal KEY representa o clock, SW[17] representa o run e SW[1] representa o resetn;

- As barras em amarelo separam cada estágio do processador para a execução da devida instrução, sendo que o primeiro estágio está entre a primeira e a segunda barra, o segundo entre a segunda e a terceira barra e assim por diante;

- Para facilitar a escrita e leitura, seque uma convenção que será seguida nesse relatório:

  Rx representara o primeiro registrador informado pela instrução, sendo identificado pelos bits 5 a 3. E Ry representa o segundo registrador informado pela instrução, sendo identificado pelos bits 2 a 0.

- Para o entendimento dos sinais de controle mux_selector e regs_in, considere o seguinte:

Mux_selector:

0 -> 0000 -> R0

1 -> 0001 -> R1

2 -> 0010 -> R2

3 -> 0011 -> R3

4 -> 0100 -> R4

5 -> 0101 -> R5

6 -> 0110 -> R6

7 -> 0111 -> R7

8 -> 1000 -> DIN

9 -> 1001 -> G

10 -> 1010 -> 0

11 -> 1011 -> 1

Regs_in:

0 -> 0000000000000 -> R0_in

1 -> 0000000000001 -> R1_in

2 -> 0000000000010 -> R2_in

3 -> 0000000000100 -> R3_in

4 -> 0000000001000 -> R4_in

5 -> 0000000010000 -> R5_in

6 -> 0000000100000 -> R6_in

7 -> 0000001000000 -> R7_in

8 -> 0000010000000 -> A_in

9 -> 0000100000000 -> G_in

10 -> 0010000000000 -> ADDR_in

11 -> 0100000000000 -> DOUT_in

12 -> 1000000000000-> IR_in

1. Funcionamento dos dois primeiros estágios, que são idênticos em todas as instruções:

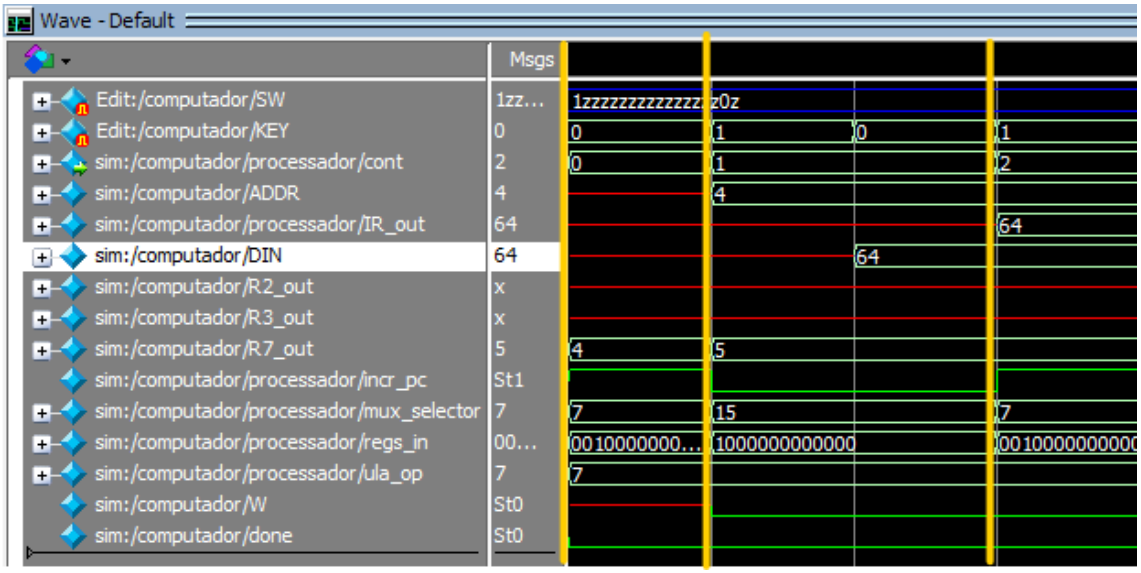

IMAGEM 2: Sinais de controle – busca da instrução

No primeiro estágio o controlador seleciona o registrador R7 no mux, através do sinal mux_selector, e coloca o valor contido nesse registrador no ADDR, usando o sinal regs_in. Ainda no primeiro estágio o sinal incr_pc é ativado para que no próximo estágio o valor de R7 seja atualizado. No próximo estágio, o valor do registrador ADDR é atualizado com o valor antigo de R7 e em paralelo os sinais de controle são modificados para que o valor lido pela memória, vindo através do fio DIN, seja escrito no registrador IR na próxima borda de subida.

2. MV:

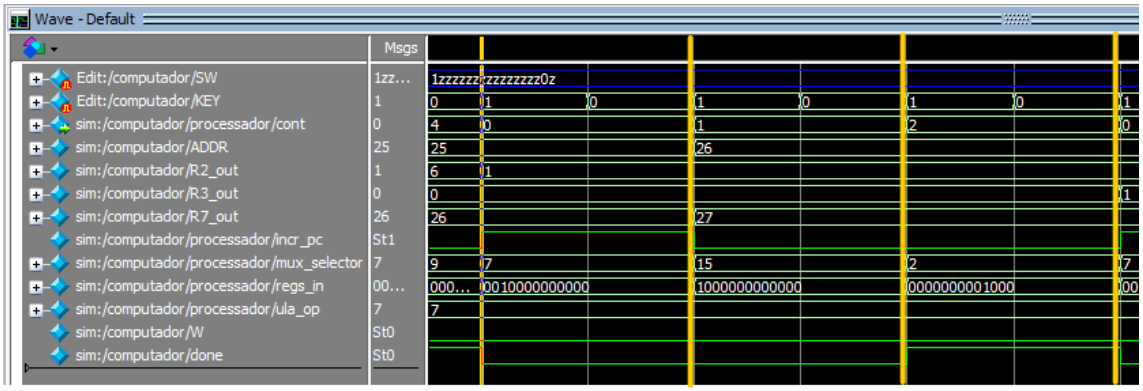

IMAGEM 3: Sinais de controle – MV

No terceiro e último estágio o controlador seleciona o valor Ry (que nesse caso representa o R2), através do mux, e o escreve no registrador Rx (que nesse caso representa o R3), o qual teve a escrita ativada através do sinal regs_selector. Como a instrução é finalizada nesse estágio o controlador ativo o sinal done, para que o sinal cont seja resetado.

3. MVI:

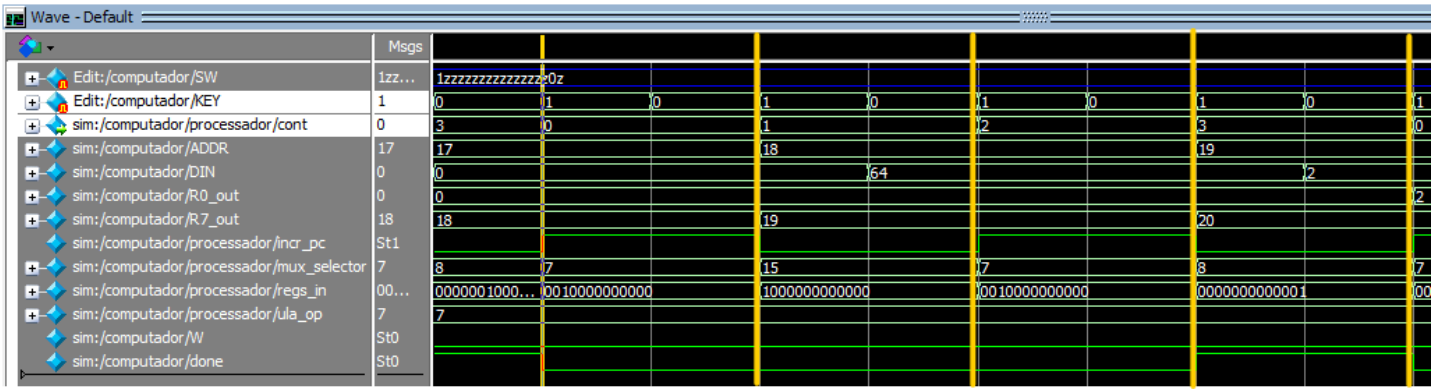

IMAGEM 4: Sinais de controle – MVI

No terceiro estágio o controlador seleciona o registrador R7 no mux, através do sinal mux_selector, e coloca o valor contido nesse registrador no ADDR, usando o sinal regs_in. Ainda no primeiro estágio o sinal incr_pc é ativado para que no próximo estágio o valor de R7 seja atualizado. No próximo estágio, o valor do registrador ADDR é atualizado com o valor antigo de R7 e em paralelo os sinais de controle são modificados para que o valor lido pela memória, vindo através do fio DIN, seja escrito no registrador Rx (nesse caso R0) na próxima borda de subida. Como a instrução finaliza nesse estágio o sinal done é ativado.

4. ADD:

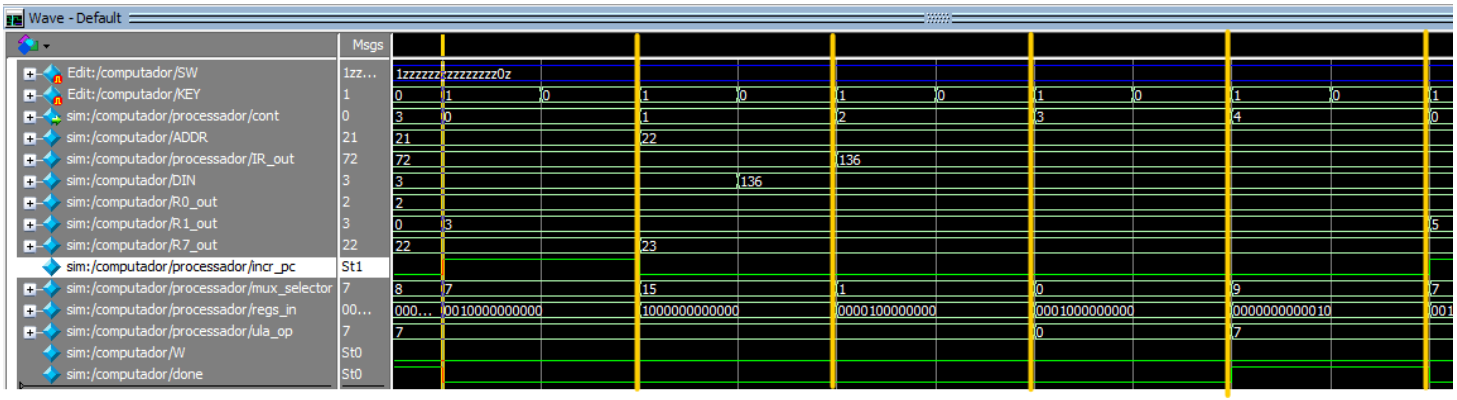

IMAGEM 5: Sinais de controle – ADD

No terceiro estágio o controlador seleciona o registrador Rx (nesse caso R1) no mux, através do sinal mux_selector, e coloca o valor contido nesse registrador no A, usando o sinal regs_in. No próximo estágio, o registrador Ry (nesse caso R0) é selecionado no mux e passado para a segunda entrada da ULA, para que ela realize a operação determinada pelo controlador através do sinal ula_op (nesse caso a operação selecionada é a soma). Ainda no quarto estagio, o controlador habilita a escrita no registrador G, para que o resultado da operação da ULA seja armazenado. Já no ultimo estagio, o controlador apenas faz a movimentação do valor do registrador G para o registrador Rx, ativando a escrita em Rx e selecionado o registrador G no mux.

5. SUB:

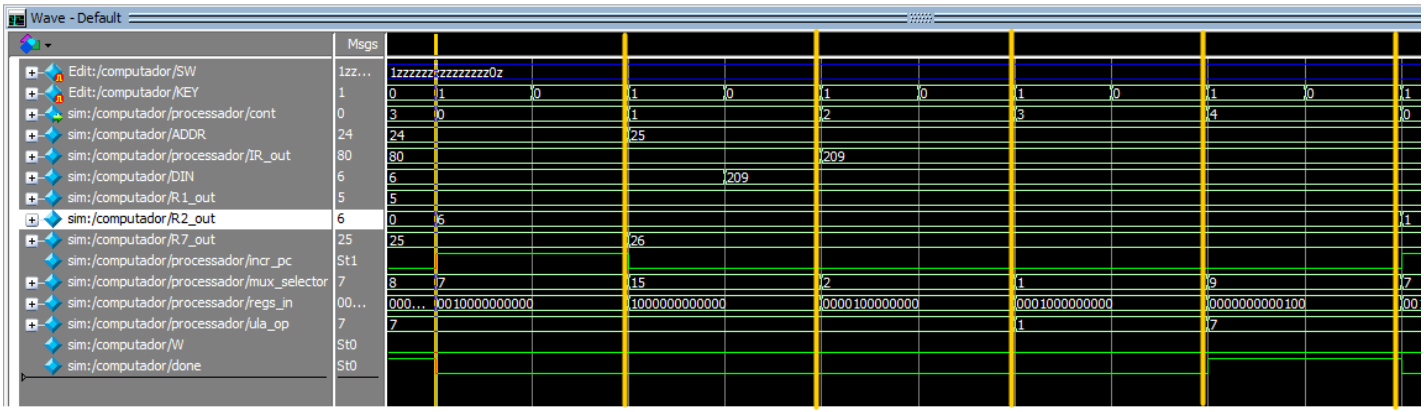

IMAGEM 6: Sinais de controle – SUB

A execução da instrução SUB é idêntica a instrução ADD, se diferenciando apenas na operação feita na ULA, que nesse caso é a subtração.

6. LD:

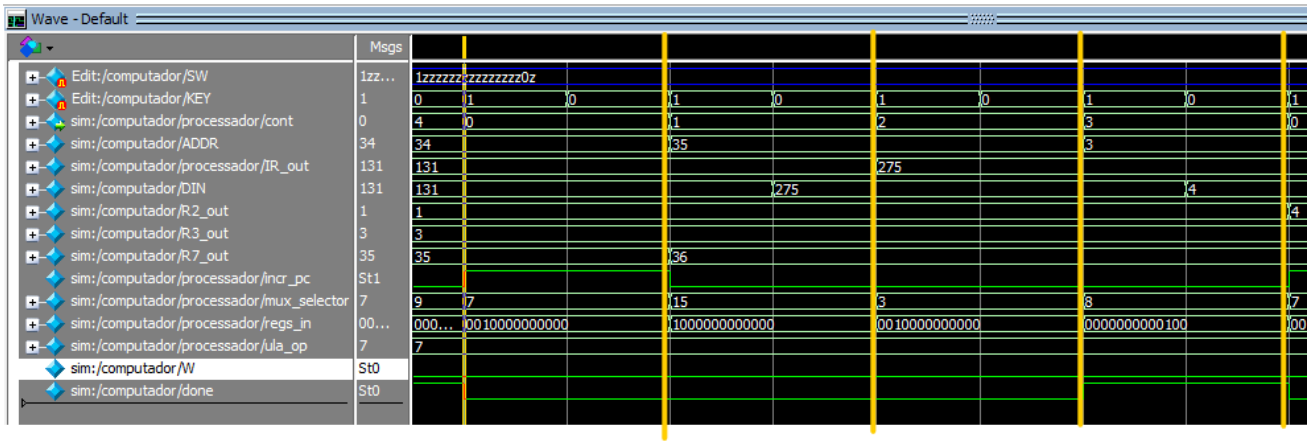

IMAGEM 7: Sinais de controle – LD

No terceiro estágio o controlador seleciona o registrador Ry (nesse caso R3) no mux, através do sinal mux_selector, e coloca o valor contido nesse registrador no ADDR, usando o sinal regs_in. No próximo ciclo os sinais de controle são modificados para que o valor lido pela memória, vindo através do fio DIN, seja escrito no registrador Rx (nesse caso R2) na próxima borda de subida. Como a instrução finaliza nesse estágio o sinal done é ativado.

7. ST:

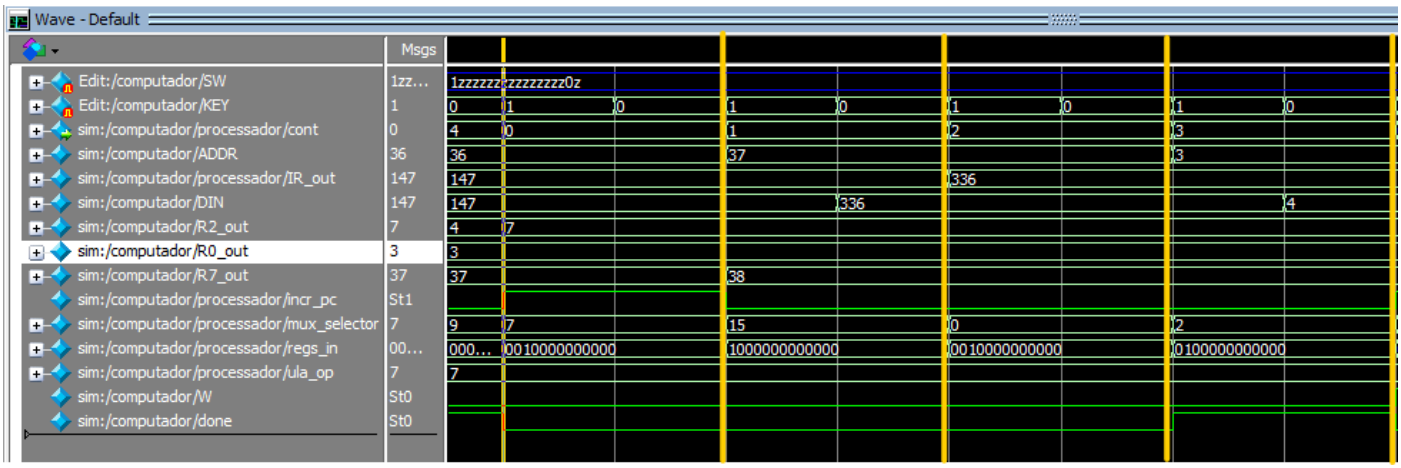

IMAGEM 8: Sinais de controle – ST

O terceiro estagio do ST é idêntico ao do LD. No quarto estagio o controlador seleciona o registrador Rx (nesse caso R2) no mux, através do sinal mux_selector, e coloca o valor contido nesse registrador no DOUT, usando o sinal regs_in. Além disso, como essa instrução escreve na memoria o sinal W é ativado. Dessa forma, na borda de descida do clock o valor presente em DOUT será escrito na memória na posição do valor de ADDR.

8. MVNZ:

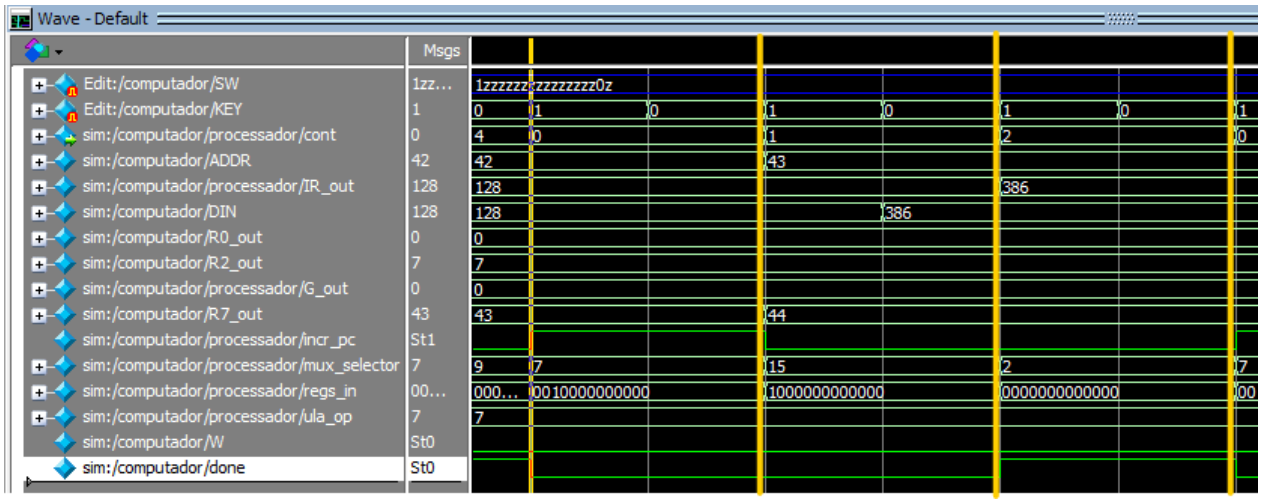

IMAGEM 9: Sinais de controle – MVNZ

Essa instrução funciona exatamente como o MV, porém antes de habilitar a escrita no registrador Rx (nesse caso R0) o controlador verifica se o valor contido em G é diferente de 0. No exemplo acima como o valor de G era 0 a instrução MVNZ não fez nada efetivamente.

9. OR:

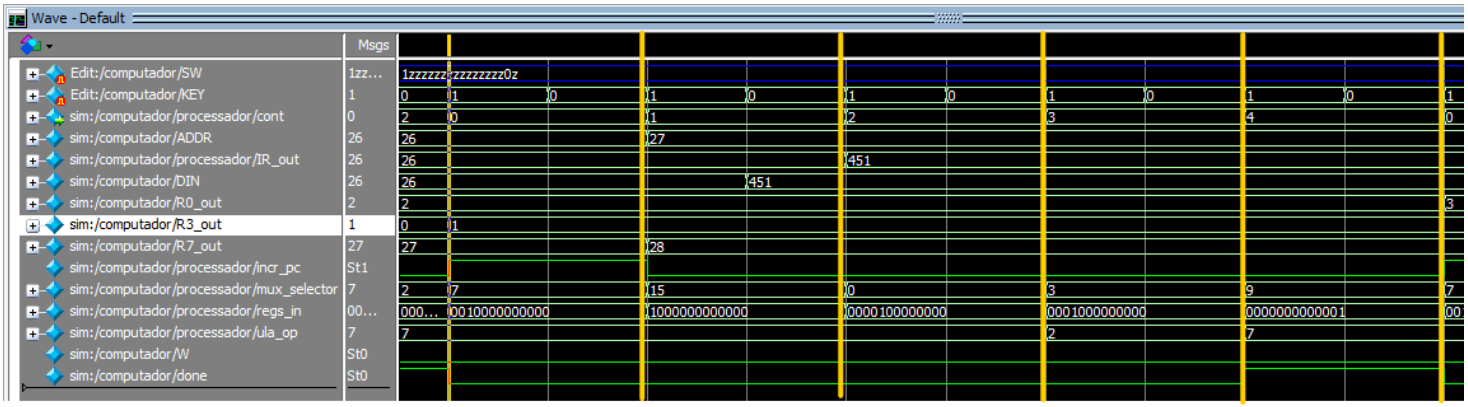

IMAGEM 10: Sinais de controle – OR

A execução da instrução OR é idêntica a instrução ADD, se diferenciando apenas na operação feita na ULA, que nesse caso é a operação OR.

10. SLT:

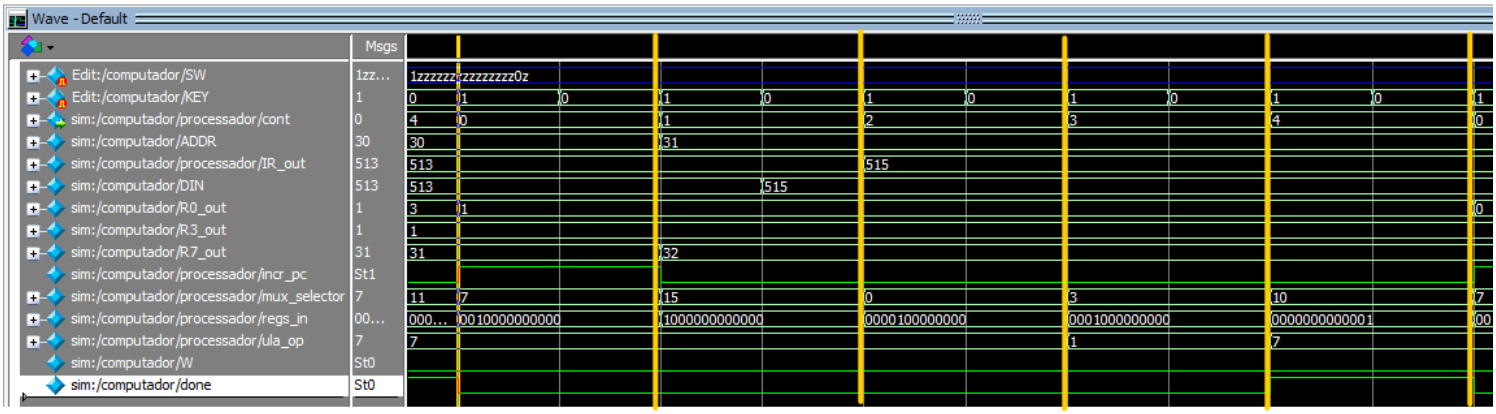

IMAGEM 11: Sinais de controle – SLT

A execução da instrução SLT é idêntica a instrução ADD até o quarto estagio, se diferenciando apenas na operação feita na ULA, que nesse caso é a operação de subtração. Já no quinto estagio o controlador verifica o valor do registrador G, se ele for negativo o mux seleciona o valor 1, caso contrário o mux seleciona o valor 0. Ainda no quinto estagio, o controlador habilita a escrita no registrador Rx (nesse caso R0) para que o valor vindo do mux possa ser gravado no seu devido lugar. Por fim, como esse é o último estágio o sinal done é ativado.

11. SLL:

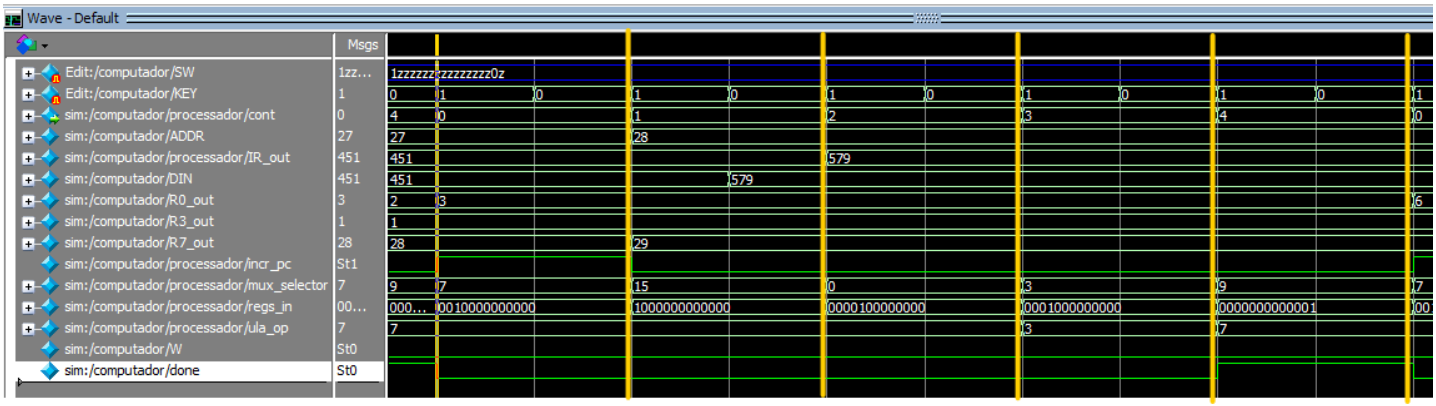

IMAGEM 12: Sinais de controle – SLL

A execução da instrução SLL é idêntica a instrução ADD, se diferenciando apenas na operação feita na ULA, que nesse caso é a operação SLL.

12. SRL:

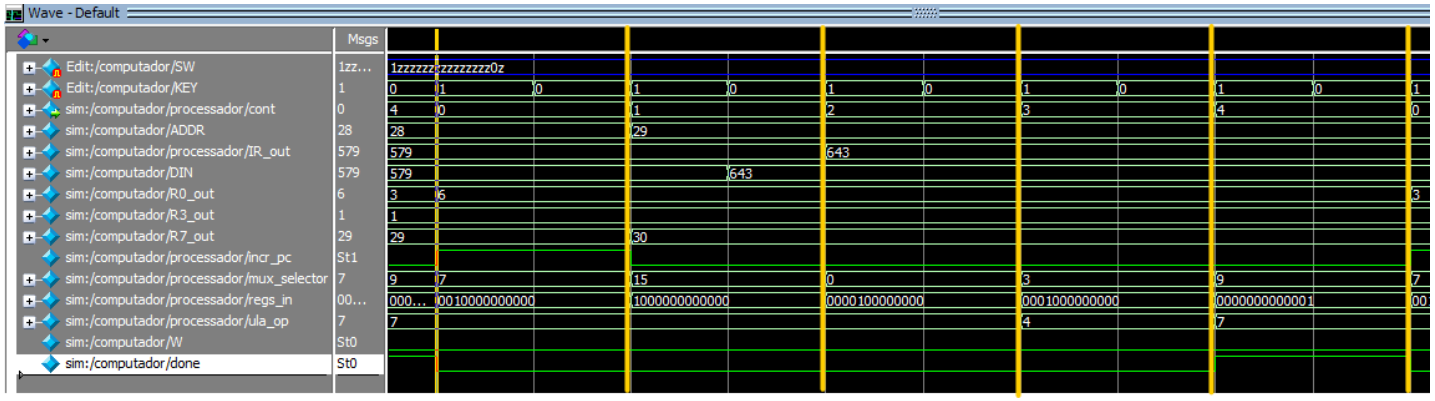

IMAGEM 13: Sinais de controle – SRL

A execução da instrução SRL é idêntica a instrução ADD, se diferenciando apenas na operação feita na ULA, que nesse caso é a operação SRL.