Relatório Trabalho Final - Linguagem e Descrição de Hardware

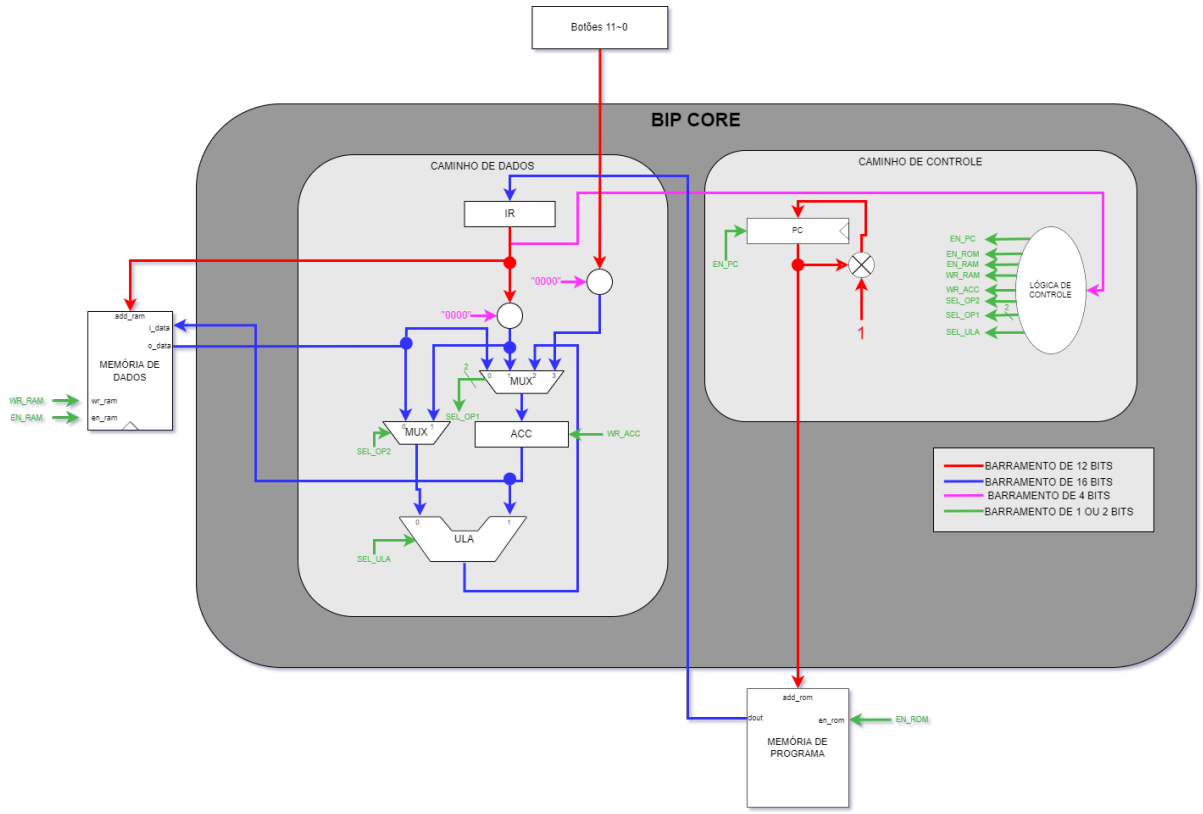
Nome: Rodrigo Ferraz Souza
Matrícula: 19103563

1 - Alterações no sistema

1.1 - Diagrama de Blocos

Ao notar que o multiplexador controlado pelo SEL_OP1 é um MUX-4x1 foi escolhido remover o Multiplexador controlado pelo EN_IN, fazendo com que, desta forma, houvesse uma variável de controle a menos na tabela opcode. Além disso as memórias foram mantidas fora dos caminhos pois, desta forma torna-se possível encomendar a construção do sistema sem que haja a necessidade de gastar silício construindo-as, havendo apenas a necessidade de anexar memórias externas ao sistema, além de tornar a elaboração do projeto mais simples.

Também separei os pinos de entrada e saída de dados da RAM para facilitar a compreensão.



Novo diagrama de Blocos

1.2 - Tabela OPCODE

- O comando **INP** agora é dado pelo **OPCODE 1000** invés de 1001
- Como um dos multiplexadores foi removido, não havia mais a necessidade do sinal **EN_IN**, então o removi e deixei o sinal dos switches entrar no pino 3 do MUX 4x1, logo **SEL_OP1** ficou **11** no comando **INP**
- **A ROM agora está sempre em 1**, mesmo no comando **STOP**, pois, caso contrário, o programa nunca se iniciaria já que os dados são inicializados em **0000** na rom durante o início da execução, o que **faz com que o program counter seja 0** e não avance nunca para a posição 001 da ROM, e também

faz com que o **EN_ROM fique em 0**, portanto o dado da posição 000 que deveria ser 3013 nunca aparece e o programa fica travado..

INSTRUÇÃO	OPCODE BINARIO (ABCD)	WR_RAM	EN_RAM	EN_PC	EN_ROM	WR_ACC	SEL_OP1		SEL_OP2	SEL_ULA
							BIT 1	BIT 0		
STOP	0000	0	0	0	1	0	0	0	0	0
STO	0001	1	1	1	1	0	0	0	0	0
LD	0010	0	1	1	1	1	0	0	0	0
LDI	0011	0	0	1	1	1	0	1	1	0
ADD	0100	0	1	1	1	1	1	0	0	0
ADDI	0101	0	0	1	1	1	1	0	1	0
SUB	0110	0	1	1	1	1	1	0	0	1
SUBI	0111	0	0	1	1	1	1	0	1	1
INP	1000	0	0	1	1	1	1	1	0	0
RESERVADO	1***	X	X	X	X	X	X	X	X	X
LOGIC EXP		B'C'D	CD' + B'C'D + BD'	C + D + A + B	1	C + B + A	B + A	A + B'CD	CD + BD = D(C+B)	BC

Nova tabela OPCODE

2 - Desenvolvimento

2.1 - Arquitetura dos componentes

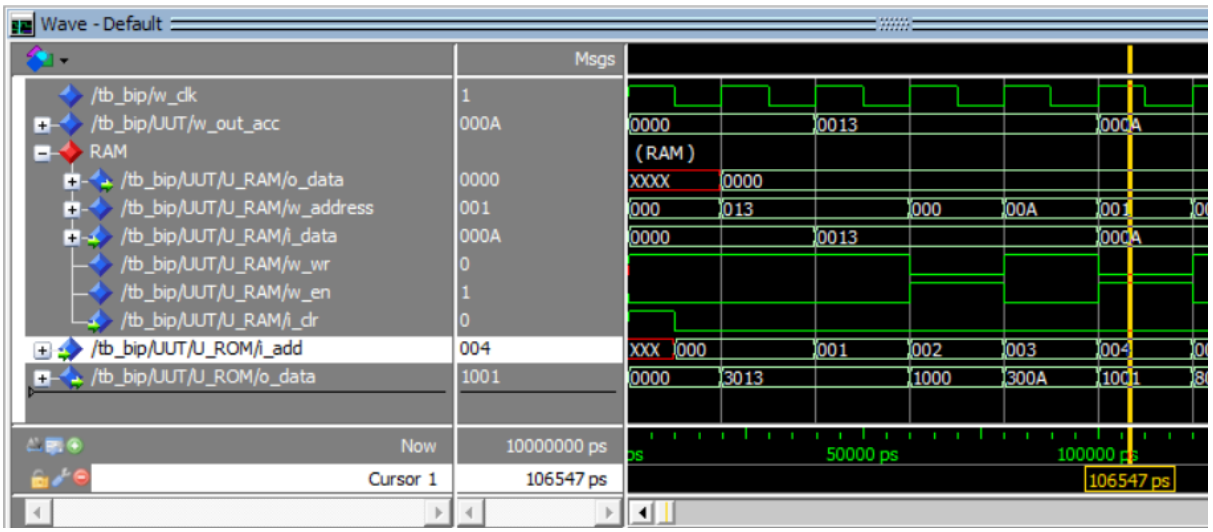
O projeto foi desenvolvendo uma hierarquia nos componentes VHDL que reflete na hierarquia dos arquivos que foram salvos em estrutura de árvore, separados por pastas.

- BIP
    - PLL
    - RAM
    - ROM
    - BIP CORE
        - Caminho de Controle
            - Lógica de Controle
            - Address Counter
        - Caminho de Dados
            - ALU
            - Registrador de Instrução
            - Acumulador

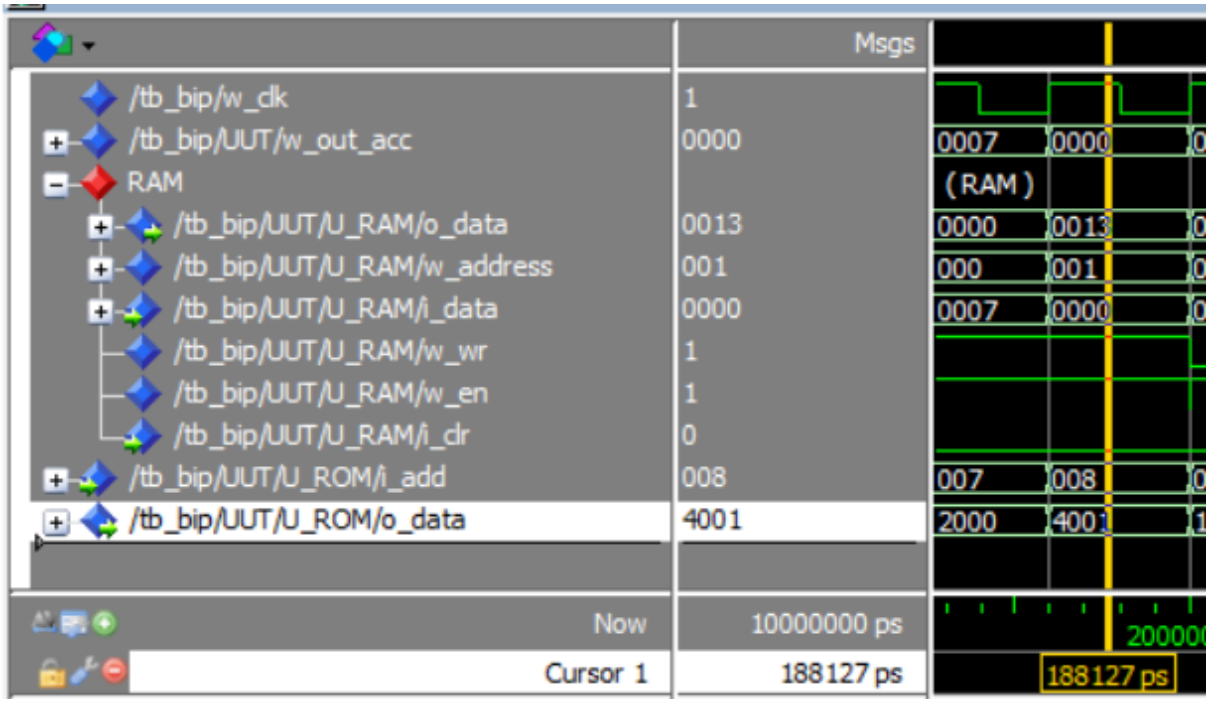
2.2 - Considerações

2.2.1 - RAM

- Não está sendo utilizado nem IP nem Process para criar a ram. Foi optado por utilizar-se de WHEN/ELSE pois quando era um dos dois outros modos a RAM utilizava os dados de uma unidade de tempo anterior à subida de clock, portanto foi removido o comportamento de apenas executar suas tarefas na borda de subida do clock. Exemplo do problema descrito:
    - Nesta posição ele deveria salvar 000A em 001



- Mas ao dar 4001 (soma o Acumulador com com o endereço 001) o que aparece é 0013, pois ao dar o sinal de clock e trocar da instrução 300A para 1001 a RAM ainda estava identificando os dados anteriores à subida de clock, o que levou ela a salvar o dado 0013 invés do 000A



A solução escolhida foi descrever a RAM da seguinte forma:

```
w_memoria_ram(conv_integer(w_address)) <=

     i_data when w_wr = '0' and falling_edge(i_clk) and w_en = '1'

     else  w_memoria_ram(conv_integer(w_address));

     o_data <= w_memoria_ram(conv_integer(w_address)) when

                    not rising_edge(i_clk)  and

                    w_en = '1' and i_clr = '0' and w_wr = '1'

               else (others => '0') when w_en = '0' and i_clr = '1'

               else w_memoria_ram(conv_integer(w_address));
```

Contudo este código não pode ser compilado no Quartus, apenas no ModelSIm, pois ele acusa o seguinte erro:
Error (10818): Can't infer register for "o_data[0]" at RAM.vhd(116) because it does not hold its value outside the clock edge

2.2.2 ROM

Não é possível sintetizar utilizando a ROM com blocos lógicos, pois, aparentemente 4096 endereços de dados de 2 bytes usam recursos demais do FPGA, o que impossibilita com que o BIP CORE seja sintetizado corretamente. Contudo na máquina utilizada para o

desenvolvimento este problema não era apresentado na síntese, apenas demonstrado na falha da simulação pois o Quartus não consegue abrir o chip planner corretamente.

Estas conclusões podem estar erradas, contudo foi a hipótese que melhor explicou o comportamento falho do sistema nestas condições.

2.2.3 - FPGA

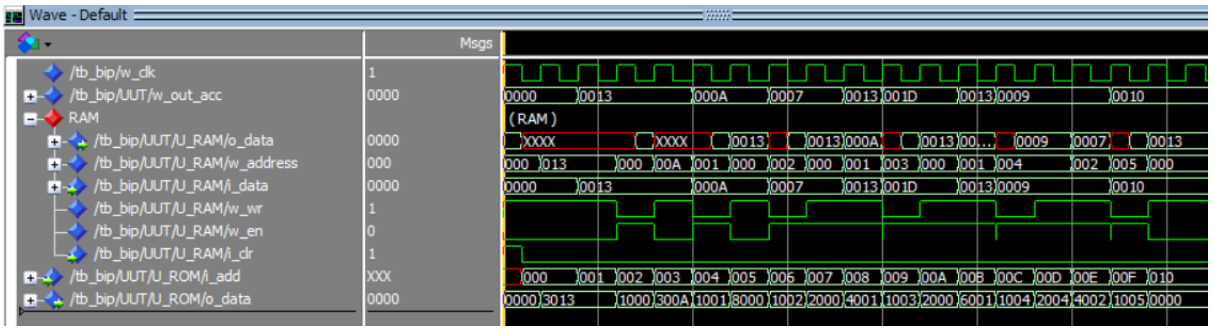
Por conta do erro citado em 2.2.1 no qual a RAM não sintetiza quando está na forma descrita no tópico não foi possível executar um teste no FPGA com o código que demonstra pleno funcionamento. Apesar disso, foi tentado um teste com a RAM da Intel, que não performa como esperado, mas que para o propósito de averiguar a escrita no FPGA é o suficiente.

Neste teste foi, primeiro, visto a simulação no modelsim, que executa as funções, contudo, de maneira completamente errada. Então foi passado o código para a placa utilizando um outro computador, usando um clock de 10 Hz para que fosse possível para um humano verificar a transição de estados do processador.

Absolutamente nada apresentou o comportamento que deveria. O reset não zerava o acumulador e a ROM não era lida

3 - Testes

3.1 - BIP

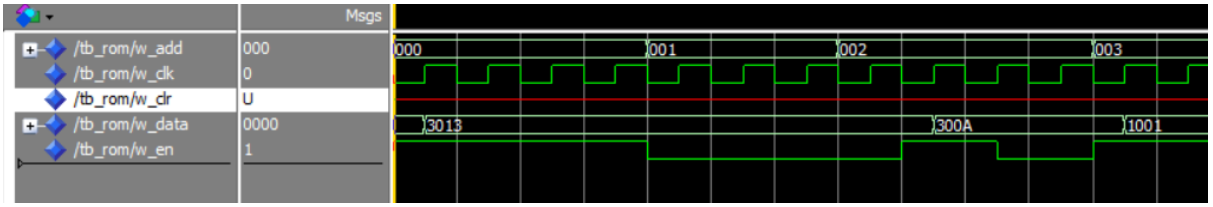


3.2 - PLL

Como o PLL é um circuito híbrido analógico/digital o ModelSim não consegue simulá-lo pois apenas simula circuitos digitais.

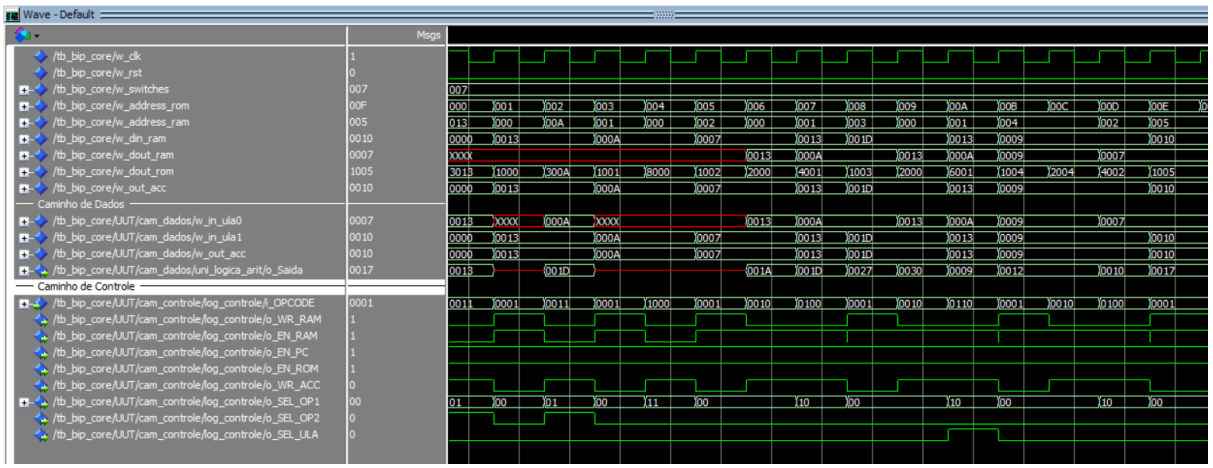
3.3 - RAM

3.4 - ROM

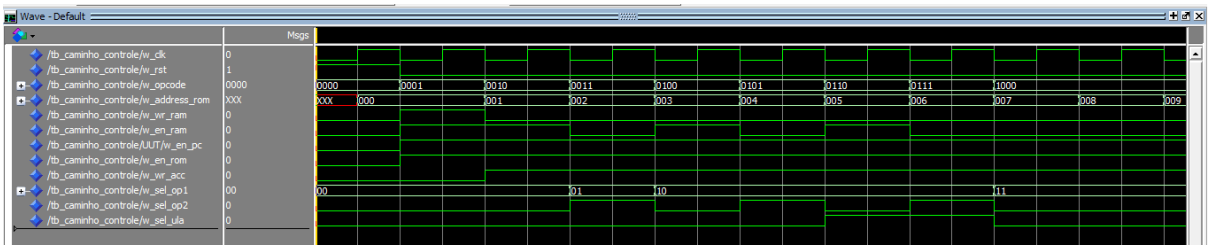


3.5 - BIP CORE

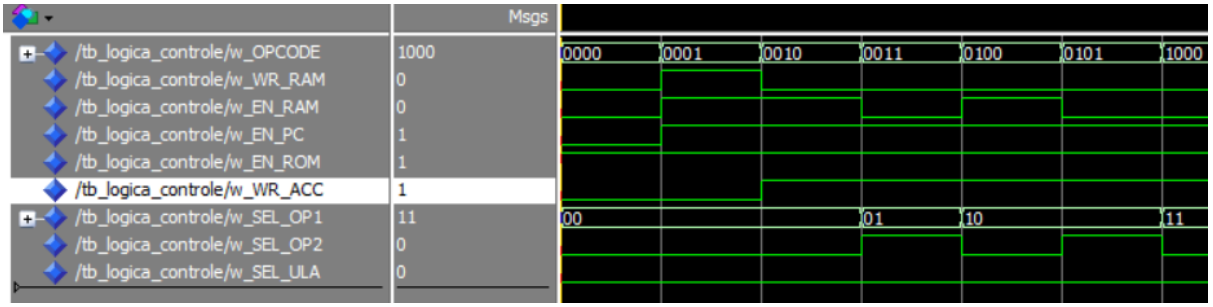
Aqui eu simulei os valores da RAM e ROM pelo TestBench



3.5.1 - Caminho de Controle

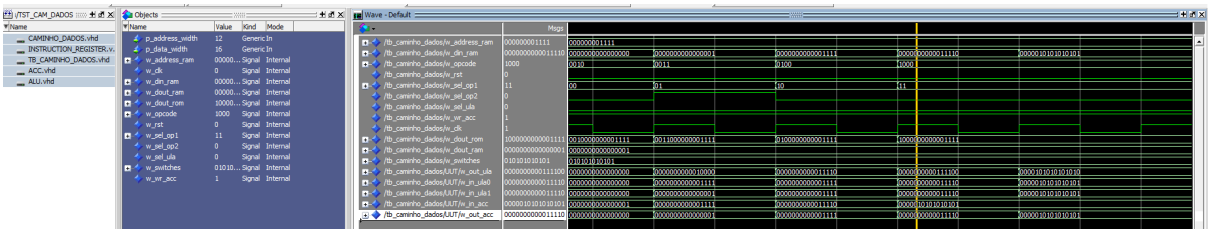


3.5.1.1 - Lógica de Controle



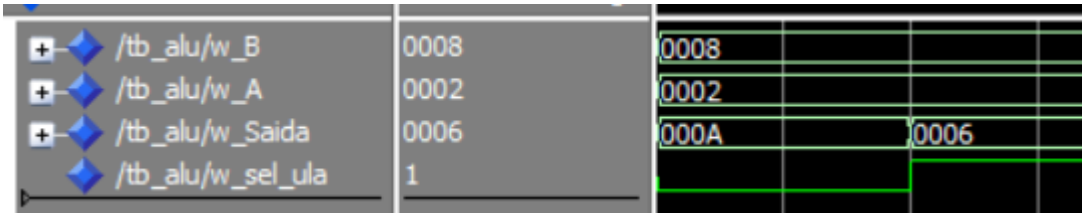
3.5.1.2 - Address Counter

3.5.2 - Caminho de Dados



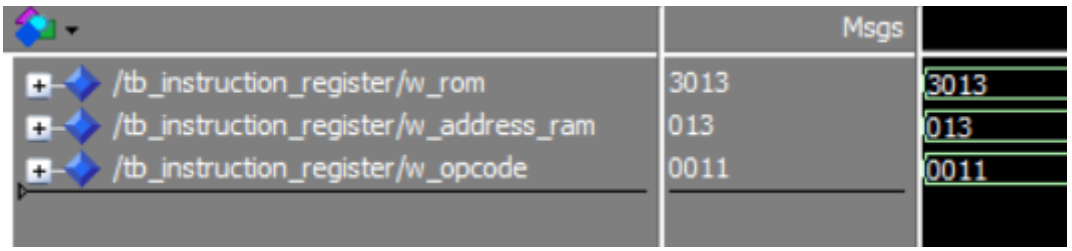
3.5.2.1 - ALU

B é onde entra o Acumulador, então como deve ser feito ACC - X, foi desenvolvido a ULA como B - A



3.5.2.2 - Registrador de Instrução

Não fiz tantos casos de testes pois é um componente extremamente simples que serve, única e exclusivamente, para separar os bits que vêm da rom para o opcode e para o endereço da ram.



3.5.2.3 - Acumulador