Universidade Federal do Rio Grande do Sul
Instituto de Informática

INF01113
Organização de Computadores B

Trabalho Prático 1 - Tarefa 09

Implementação de Instruções no MIPS

Bruno Alexandre Hofstetter Bourscheid (00550177)
Fernando Longhi de Andrade (00580366)
Luiz Augusto Ponzoni Schmidt (00580108)
Miguel Dutra Fontes Guerra (00342573)
Pedro Lubaszewski Lima (00341810)

Turma B

19 de abril de 2025

Sumário

1.1 MIPS Singlecycle

1.1.1 Modificações no Bloco Operativo

1.1.2 Modificações no Bloco de Controle

1.1.3 Testes de Implementação

2.1 MIPS Multicycle

2.1.1 Modificações no Bloco Operativo

2.1.2 Modificações no Bloco de Controle

2.1.3 Estados de Controle

2.1.4 Testes de Implementação

3.1 MIPS Pipeline

3.1.1 Modificações no Bloco Operativo

Primeiramente, segue uma visão de como estava a implementação do MIPS com pipeline antes do acrescimo das novas instruções:

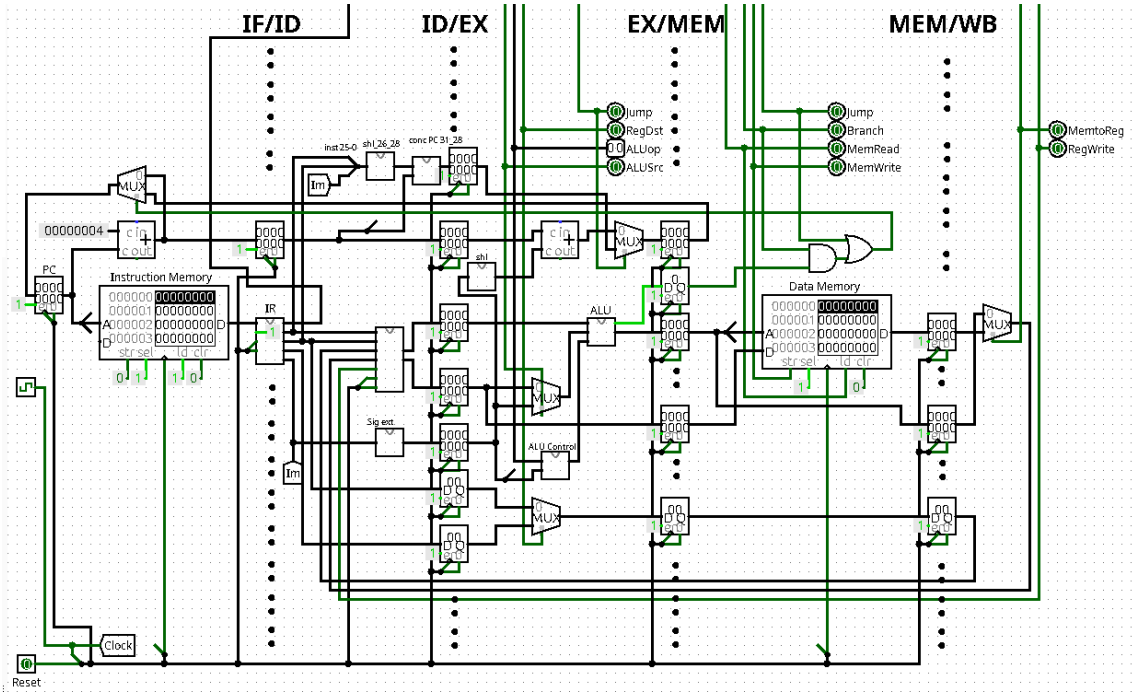

Figura 1: MIPS Pipeline - Bloco Operativo Antes

Sobre essa versão do MIPS com pipeline, foram feitas as modificações listadas à seguir:

- Adicionada operação multiplicação na ALU;
- Adicionada saída dos bits superiores da multiplicação na ALU;
- Adicionada barreira temporal para capturar valor de RS;
- Adicionado MUX para o JR na entrada do PC;
- Corrigidos os timings entre barreiras temporais para o sinal de IsMult;
- Utilizado o sinal IsMult para não habilitar a escrita nos registradores;
- Propagado o sinal de imediato com 32 bits para a etapa de MEM;
- Colocado comparador de menor que 0 ou menor que RS para instruções de BLTZ e SLTIU em MEM;
- Conectado o BLTZ ao mux de branches;
- Propagado o bit de comparação RS ¡ Imed para o estágio de MEM;
- Adicionado bit extender para o resultado da comparação acima;
- Conectado o resultado desse bit extender na entrada de escrita dos GPRs;
- Adicionado splitter para pegar os 8 bits menos significativos de um dado da memória;
- Extendido esses 8 bits acima;
- Conectado esse resultado controlado pelo sinal LB nos GPRs.

Com essas modificações listadas acima, obteve-se a seguinte nova representação do bloco operativo do MIPS:

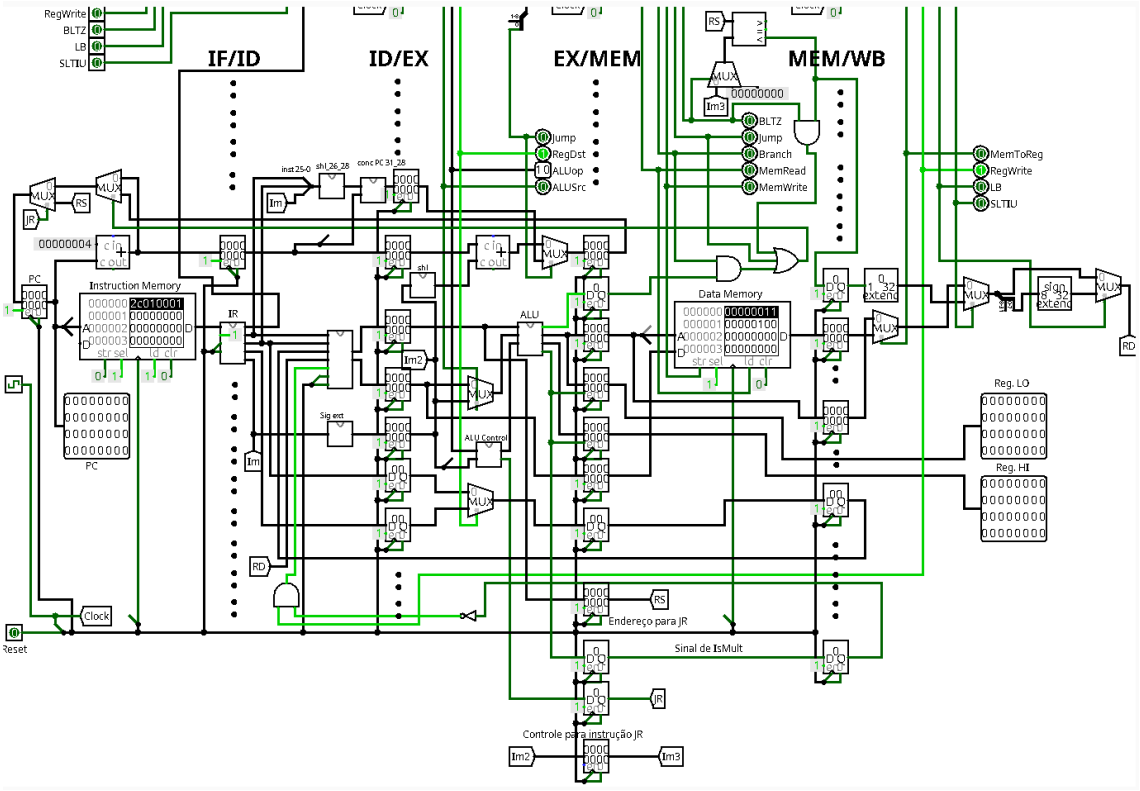

Figura 2: MIPS Pipeline - Bloco Operativo Depois

3.1.2 Modificações no Bloco de Controle

Já na parte de controle, anteriormente, tinha-se a seguinte representação:

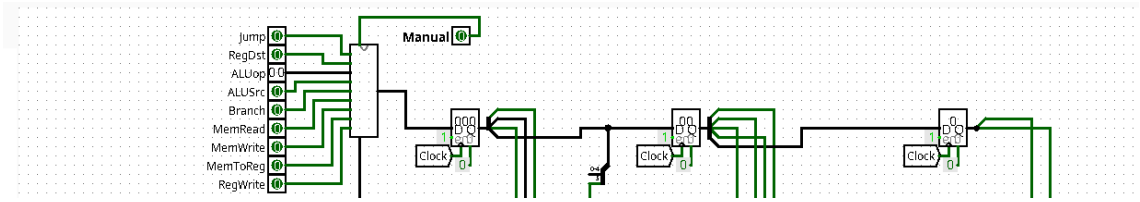

Figura 3: MIPS Pipeline - Bloco Controle Antes

A partir dessa parte de controle, é possível listar as seguintes modificações:

- Criados os sinais de controle LB, SLTIU, BLTZ, JR e IsMult;
- Atualizado o ALU Control para as instruções de MULT e JR;
- Adicionada saída IsMULT da ALU;
- Adicionada barreira temporal para o estágio de EX/MEM do JR;
- Propagados todos os sinais de controle para as respectivas etapas do circuito:

  - BLTZ em MEM;

  - LB e SLTIU em WB.

Além dessas modificações, foram reordenados os sinais de controle para serem equivalentes aos sinais de controle do MIPS Singlecycle. Assim, as modificações da ROM de controle foram as mesmas listadas na seção do MIPS Singlecycle:

- ROM para JR: 0x0241 end 0x00;
- ROM para BLTZ: 0x0404 end 0x01;
- ROM para LB: 0x0B18 end 0x20;

- ROM para MULT: 0x0241 end 0x00;
- ROM para SLTIU: 0x1300 end 0x0B.

Com tudo isso, obteve-se o seguinte novo esquemático externo da parte de controle do MIPS com pipeline:

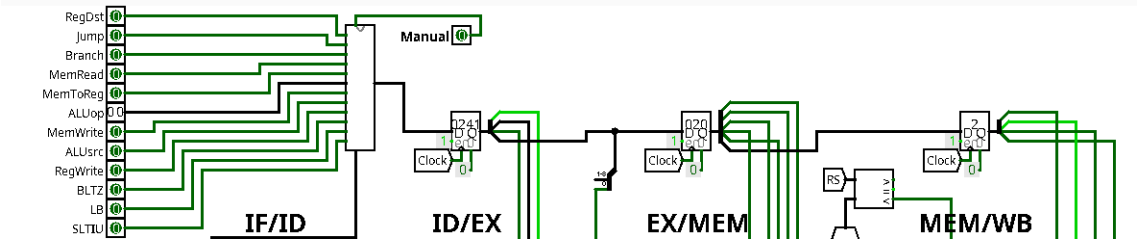

Figura 4: MIPS Pipeline - Bloco Controle Depois

3.1.3 Testes de Implementação

Para comprovar que essas mudanças acima trouxeram a correta implementação das instruções especificados neste trabalho, abaixo seguem os testes realizados com imagens e a respectiva sequência de instruções:

É importante notar, em todos os testes, que há uma grande quantidade de NOPs espalhados entre as instruções. Isso é devido a não terem sido implementadas soluções para lidar com dependências de dados e de controle.

- Teste da Instrução JR:

  0x8C010000; LW: Reg1 ← Mem[0] (64 ou 0x40)

  0x00000000; NOP

  0x00000000; NOP

  0x00000000; NOP

  0x00000000; NOP

  0x00200008; JR: PC ← Reg1 (64 ou 0x40)

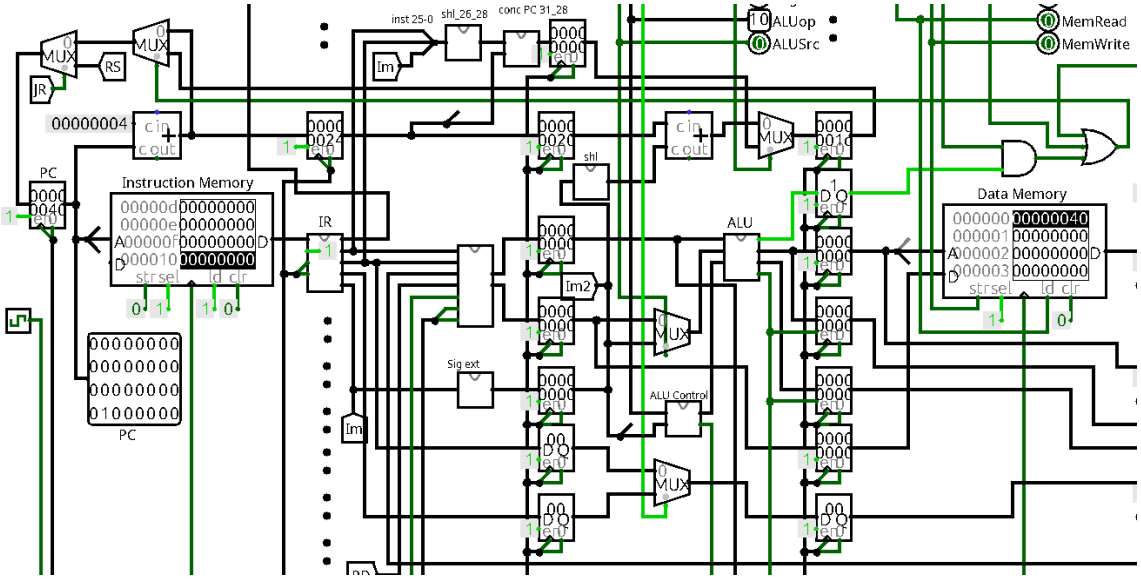

Figura 5: MIPS Pipeline - Teste JR

- Teste da Instrução MULT:

  MULT: HI, LO ← Reg1 · Reg2

0x8C010000; Reg1 ← Mem[0] (17 ou 0x11)

0x8C020001; Reg2 ← Mem[1] (256 ou 0x100)

0x00000000; NOP

0x00000000; NOP

0x00000000; NOP

0x00000000; NOP

0x00220018; HI, LO ← Reg1 · Reg2 (HI = 0, LO = 4352 ou 0x10400)

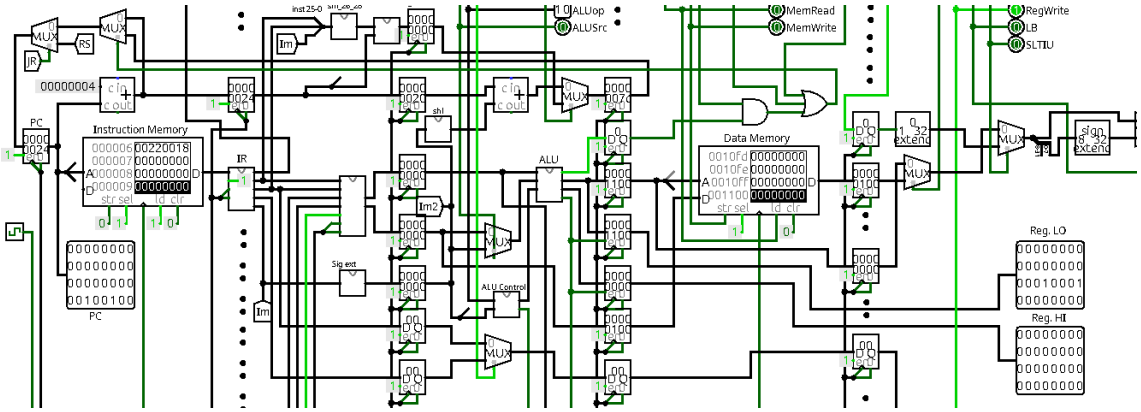

Figura 6: MIPS Pipeline - Teste MULT

- Teste da Instrução BLTZ:

  BLTZ: if Reg1 < 0 then PC += IMED·4 + 4 (IMED = 10)

  0x8C010000; Reg1 ← Mem[0] (-1 (0xFFFFFFFF))

  0x00000000; NOP

  0x00000000; NOP

  0x00000000; NOP

  0x00000000; NOP

  0x0420000A; PC += IMED·4 + 4 (PC = 64 ou 0x40)

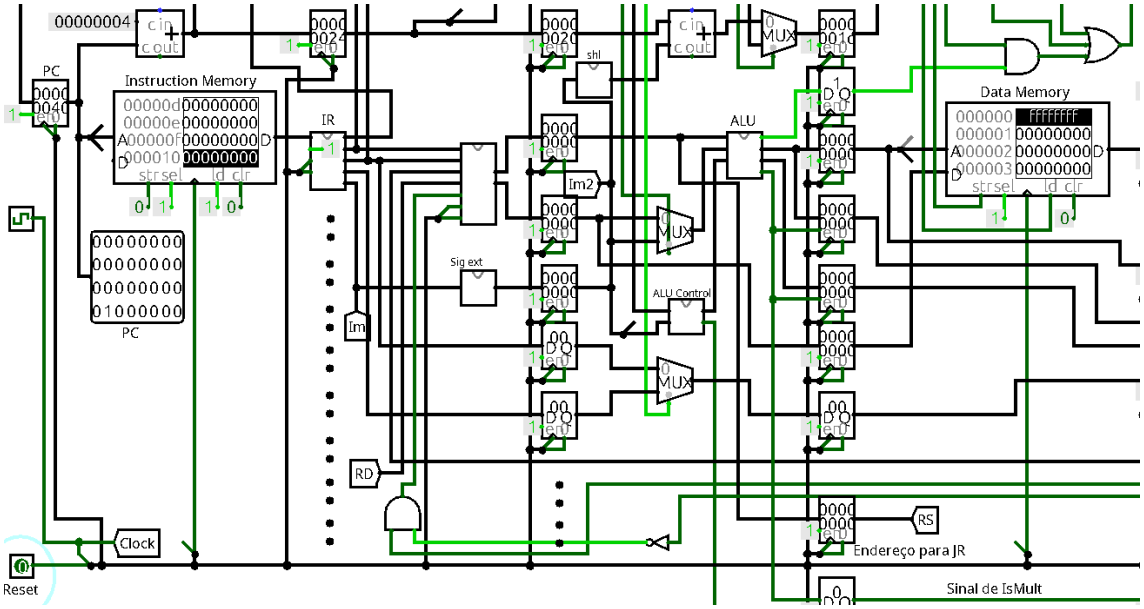

Figura 7: MIPS Pipeline - Teste BLTZ

- Teste da Instrução LB:

  LB: Reg1 ← byte(Mem[0]) (0x00000FFF)

  0x80010000; Reg1 ← byte(Mem[0]) (0xFFFFFFFF)

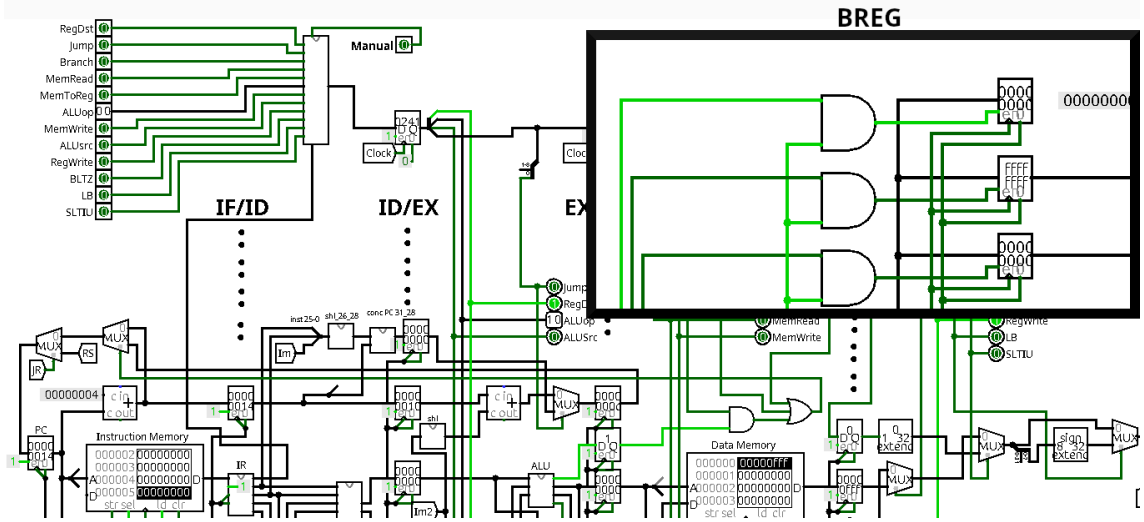

Figura 8: MIPS Pipeline - Teste LB

- Teste da Instrução SLTIU:

  SLTIU: Reg1 ← Reg0 < IMED (1)

  0x2C010001; Reg1 ← Reg0 < 1 (Reg1 = 1)

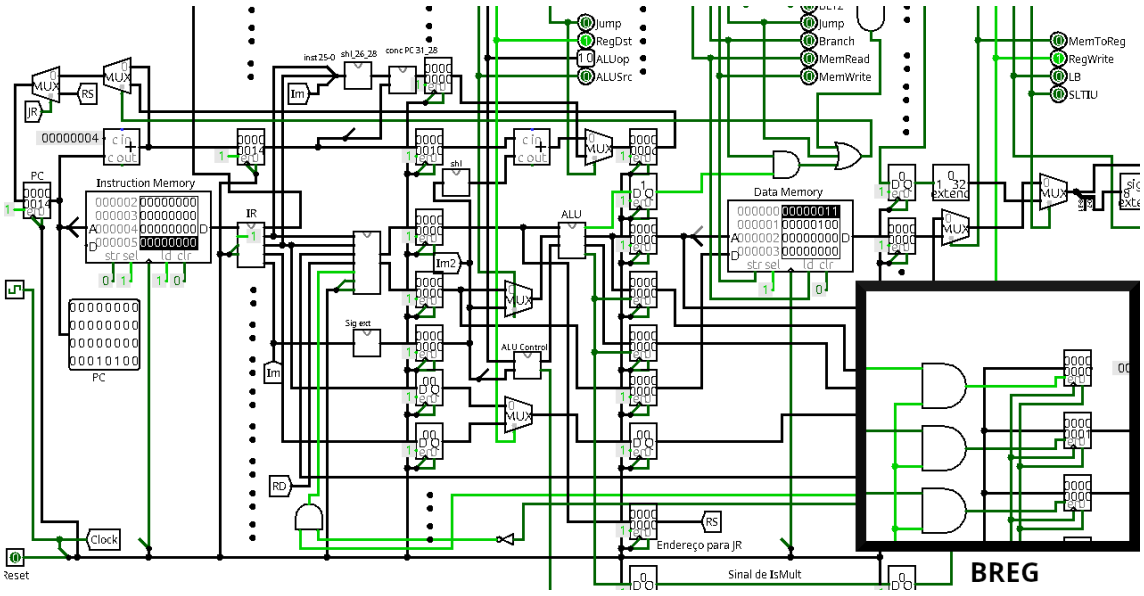

Figura 9: MIPS Pipeline - Teste SLTIU