Projeto T-FIVE: 5a. Entrega

PCS3722: Organização e Arquitetura de Computadores II

Por Wilson Ruggiero

5a. Entrega: Implementação do estágio de Execução : **estagio_ex**

- Baseado nas especificações que o professor forneceu faça a **implementação em VHDL (descrição comportamental) do estágio de execução de instruções – estagio_ex**

- **Os sinais de entrada e saída deste estágio devem seguir rigorosamente as especificações fornecidas pelo professor.**

- Os testes desta implementação serão feitos **integrando-se este estágio com os estágios de busca e de decodificação,** que foram implementados nas entregas anteriores.

- Os testes desses estágios devem ser feitos lendo-se o código da **rotina swap**, que faz parte do programa **sort**, substituindo-se a última instrução do swap – **jalr** de retorno de sub-rotina, por uma instrução de **Halt**

- Esse código encontra-se a seguir no próximo slide:

Programa de teste: estagio_if, estagio_id e estagio_ex

```
Main:
    jal     ra,     swap        # chama subrotina swap                          PC = 00
fim:
    jal     zero,   fim         # Pare                                          PC = 04
swap:
    addi    a0,     zero,   0   # carrega 0 em r10 e                            PC = 08
    addi    a1,     zero,   3   # carrega 3 em r11 e                            PC = 0C
    slli    t0,     a1,     2   # r5 = r11*4 = 3*4 = 12 (C em hexa) e           PC = 10
    add     t0,     a0,     t0  # r5 = r10 + r5 e                              PC = 0C
    lw      t1,     0(t0)       # r6 = M[0+12] = 9 e                            PC = 14
    lw      t2,     4(t0)       # r7 = M[4+12] = 7 e                            PC = 18
    sw      t2,     0(t0)       # M[0+12} = r7 = 7 e                            PC = 1C
    sw      t1,     4(t0)       # M{4+12} = r6 = 9 e                            PC - 20
    halt                        # Pare a execucao                               PC = 24
```

OBS.: Se o aluno rodar este código Assembler no montador do T-FIVE ele deve obter o código de máquina a ser carregado na imem para teste do estágio_if!!!!

Programa de teste dos estágios de Busca, Decodificação e Execução - código de máquina

- Código de máquina a ser carregado na imem:

```
00000000100000000000000011101111
00000000000000000000000001101111
00000000000000000000010100010011
00000000001100000000010110010011
00000000001001011001001010010011
00000000010101010000001010110011
00000000000000010101000110000011
00000000100001010101000111000011
00000000111001010100000000100011
00000000110001010100010000100011
00000000000000000000000001101111
```

Limitações da Bancada de teste

- As **decisões alternativas para realização deste projeto são inúmeras** e por isso as bancadas de teste não conseguem executar uma verificação completa do projeto realizado

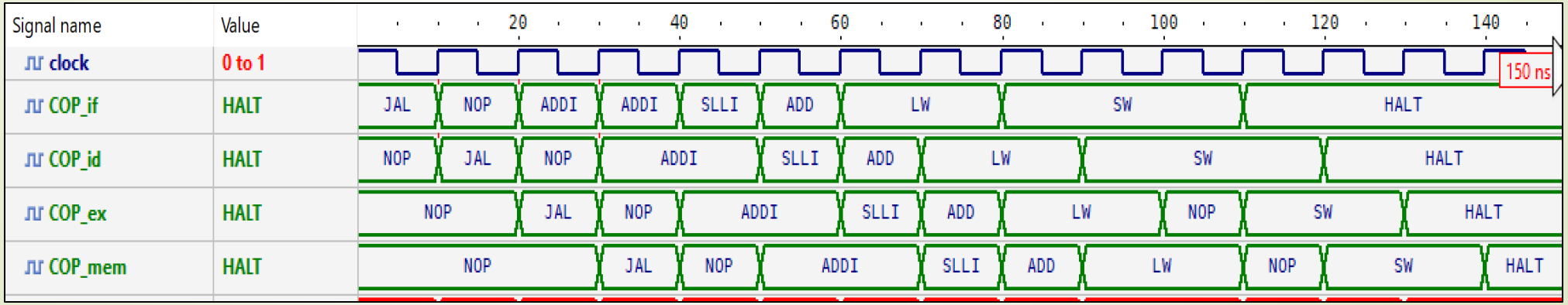
- Mesmo assim, em alguns casos particulares, **pequenas variações podem ser observadas** em seu conteúdo sem significar que elas estejam erradas

- A bancada de teste servirá para o professor fazer uma primeira análise no projeto de cada grupo, mas a **verificação final será feita em cima das forma de onda produzidas pela implementação do grupo** e pelo código VHDL dessa implementação.

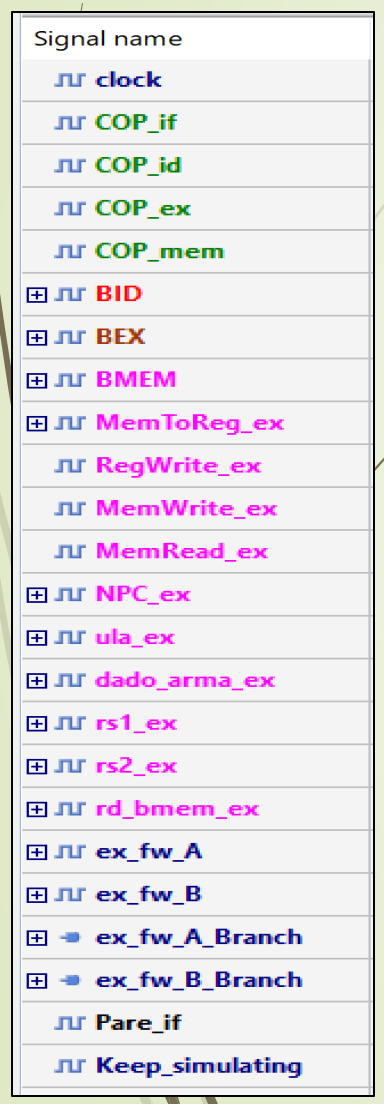
- Por estes motivos, **os valores impressos pela bancada de teste no console do simulador devem ser interpretadas somente pelo professor**, pois eles podem apresentar peculiaridades que devem ser devidamente interpretadas.

Ajuda para depuração

- A **utilização dos sinais mnemônicos** que identificam as instruções de máquina podem **facilitar a depuração** da implementação dos estágios do Pipeline do projeto T-FIVE
- Esses sinais são:
  - **COP_if, COP_id, COP_ex, COP_mem e COP_wb.**
- Todo conjunto de sinais nas formas de onda produzidas devem iniciar com os seguintes sinais:
  - **Clock**, **COP_if**, **COP-id**, ..., **até o último dos sinais COP's que estão sendo produzidos na corrente implementação**



Forma de onda que deve ser gerada:



- Na bancada de teste, os testes devem ser **executados por 150 ns** e devem produzir formas de onda com os **seguintes sinais (nessa ordem)**:
  - **clock**
  - **COP_if, COP_id, COP_ex, COP_mem,**
  - **BID, BEX,**
  - **BMEM, sinais que compõem o BMEM iniciando pelos bits mais significativos,**
  - **ex_fw_A, ex_fw_B,**
  - **ex_fw_A_Branch, ex_fw_B_Branch,**
  - **Pare_if**
  - **keep_simulating**

Relatório da 5a. Entrega

- O relatório desta entrega deve **descrever a implementação em VHDL** feita para o **estágio de execução destacando as suas principais características, tais como**:
  - **Quais conflitos demandam antecipação de dados para resolve-los?**
    - **Para cada um deles, explicar como ele é detectado, onde estão os dados necessários e como traze-los para o local demandado?**
  - **Como é feita a antecipação de valores no caso de uma instrução lw seguida de outra sw, ambas referenciando o mesmo registrador, como no exemplo:**

```
lw      t0, 0(t1)
sw      t0, 0(t0)
```

  - **Como é feito a antecipação de dados em conflitos envolvendo as instruções de desvio condicional?**
    - **Explique como detecta-los, onde estão os dados necessários e como resolve-los?**
  - **Demais características que o grupo achar necessário ou relevante descrever.**
- Este relatório deve iniciar com a **declaração do que cada membro do grupo fez nesta entrega**
- Pede-se para que **o relatório não seja assinado com os certificados digitais**, pois se assim for, ele impede que o professor possa colocar os comentários no próprio relatório. Os alunos podem assinar simplesmente com a imagem de suas assinaturas.

estagio_ex – Declaração de Entidade fornecida pelo Professor

```
----------------------------------------------------------------------------------
----------MODÚLO DE BUSCA - ID ----------------------------------------------------
----------------------------------------------------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_signed.all;


library work;
use work.tipos.all;
```

Estagio_id – Declaraçao de Entidade

```
entity estagio_ex is
   port(
            -- Entradas
            clock              : in   std_logic;                          -- Relógio do Sistema
            BEX                : in   std_logic_vector (151 downto 0);    -- Dados vindos do id
            COP_ex             : in   instruction_type;                   -- Mnemônico no estágio ex
            ula_mem            : in   std_logic_vector (031 downto 0);    -- ULA no estágio de Memória
            rs1_id_ex          : in   std_logic_vector (004 downto 0);    -- rs1 no estágio id para o ex
            rs2_id_ex          : in   std_logic_vector (004 downto 0);    -- rs2 no estágio id para o ex
            MemRead_mem        : in   std_logic;                          -- Leitura na memória no  mem
            RegWrite_mem       : in   std_logic;                          -- Escrita nos regs. no  mem
            rd_mem             : in   std_logic_vector (004 downto 0);    -- Destino nos regs. mem
            RegWrite_wb        : in   std_logic;                          -- Escrita nos regs no estágio wb
            rd_wb              : in   std_logic_vector (004 downto 0);    -- Destino no regs no estágio wb
            writedata_wb       : in   std_logic_vector (031 downto 0);    -- Dado a ser escrito no regs.
            Memval_mem         : in   std_logic_vector (031 downto 0);    -- Saída da memória no mem

            -- Saídas
            MemRead_ex         : out  std_logic;                          -- Leitura da memória no ex
            rd_ex              : out  std_logic_vector (004 downto 0);    -- Destino dos regs no ex
            ULA_ex             : out  std_logic_vector (031 downto 0);    -- ULA no estágio ex
            ex_fw_A_Branch     : out  std_logic_vector (001 downto 0);    -- Dado comparado em A no id
                                                                          -- em desvios com forward
            ex_fw_B_Branch     : out  std_logic_vector (001 downto 0);    -- Dado comparado em B no id
                                                                          -- em desvios com forward
            BMEM               : out  std_logic_vector (115 downto 0) := (others => '0');    -- dados para mem
            COP_mem            : out  instruction_type := NOP             -- Mnemônico no estágio mem

            );
end entity;
```

Definição dos sinais do estágio ex

Sinais de entrada do estágio ex:

- **clock**: sinal de **base de tempo** gerada na bancada de teste e serve a todos os estágios
- **BEX**: sinal contendo as informações vindas do estágio id;
- **COP_ex**: mnemônico da instrução presente no estágio ex;
- **ula_mem**: vinda do estágio memória contendo o valor da saída da ULA no estágio de Memória;
- **rs1_id_ex**: valor do campo rs1 no estágio id enviado para o estágio ex;
- **rs2_id_ex**: valor do campo rs2 no estágio id enviado para o estágio ex;
- **MemRead_mem**: indica presença de uma instrução de leitura na memória no estágio mem;
- **RegWrite_mem**: instrução de escrita nos regs. no estágio mem;
- **rd_mem**: endereço destino nos regs. da instrução que no estágio mem;
- **RegWrite_wb**: instrução de escrita nos regs no estágio wb;

Definição dos sinais do estágio ex (cont.)

- **rd_wb**: endereço destino nos regs da instrução que se encontra no estágio wb;
- **writedata_wb**: valor do dado a ser escrito nos regs. no estágio wb;
- **Memval_mem**: valor da saída da unidade de memória no estágio mem;

Sinais de saída do estágio ex:

- **MemRead_ex**: indica presença de uma instrução de leitura da memória no estágio ex ;
- **rd_ex**: valor do campo destino dos regs da instrução presente no estágio ex;
- **ula_ex**: valor da saída da ULA da instrução presente no estágio ex;
- **ex_fw_A_Branch**: valor do dado a ser comparado na posição A no estágio id nas instruções de desvio condicional com conflitos;
- **ex_fw_B_Branch**: valor do dado a ser comparado na posição B no estágio id nas instruções de desvio condicional com conflitos;
- **COP_mem**: mnemônico da instrução que se encontra no estágio mem;

Definição dos sinais do estágio ex (cont.)

- **BMEM**: sinal enviado ao estágio mem que possui **116 bits** alocados da seguinte forma:

```
BMEM(115 downto 114)  <= MemToReg_ex;
BMEM(113)             <= RegWrite_ex;
BMEM(112)             <= MemWrite_ex;
BMEM(111)             <= MemRead_ex;
BMEM(110 downto 079)  <= NPC_ex;
BMEM(078 downto 047)  <= ULA_ex;
BMEM(046 downto 015)  <= dado_arma_ex;
BMEM(014 downto 010)  <= rs1_ex;
BMEM(009 downto 005)  <= rs2_ex;
BMEM(004 downto 000)  <= rd_ex;
```

- Nas formas de onda que incluem o sinal **BMEM**, a ordem de seus sinais internos iniciando pelos de bits mais significativos é a mostrada neste slide.
- **É possível que nem todos sinais deste estágio sejam necessários**.

Funções executadas pelo estágio

- Neste estágio são **executadas as instruções do tipo RR** e **calculado os endereços de memória** referenciados pelas instruções de **load (lw) e store (sw).**
- O módulo que implementa a **antecipação de valores (Forwarding - fw)** é feita neste estágio num módulo separado dentro do estágio ex.
- A **unidade lógica e aritmética - ULA** - fica neste estágio.
- Os **multiplexadores de estrada da ULA** que selecionam os valores corretos dependendo da antecipação de valor desejado ficam neste estágio.
- A definição dos sinais de entrada e saída do estágio ex encontram-se definidos também na declaração da entidade estágio ex e são passados pelo registrador **BEX** vindo do estágio id e pelo **BMEM** indo para o estágio mem.

Informações e Módulos do estágio ex

- As informações passadas deste estágio para o estágio mem devem ser feitas por meio de um **registrador (BMEM)**. Para identificar clara e textualmente cada campo desse registrador pode-se utilizar o mecanismo do VHDL de **definição de apelidos ("alias")**

- Foi adicionado um sinal, para fins de depuração, chamado **COP_ex** que identifica a instrução (mnemônico) sendo processada pelo estágio

- Neste estágio deve ser implementado também o **módulo de antecipação de valores – forwarding - fw**

- Devem existir **diversos sinais vindos de outros módulos** que são necessários para a realização das funções alocadas ao estágio ex

- A definição dos sinais vindos de outros módulos encontra-se também nos comentários da declaração da entidade do estágio ex

Bancada de testes para o estágio ex

- A bancada de testes que os alunos devem utilizar é a fornecida pelo professor

- Essa bancada de testes realiza um **teste parcial deste estágio**, pois a maioria das funcionalidades das instruções de máquina ainda não são possíveis de serem conseguidas (faltam os demais estágios)

- A bancada de testes determina a **leitura do código de máquina que se encontra na memória de instruções – imem,** sendo enviada para o estágio id e os seguintes, para ali ter a continuação de sua execução

- **Os arquivos (.vhd) da especificação do estágio if, id e ex e sua bancada de teste encontram-se disponíveis para download na ferramenta "Atividades", ou na página que detalha as informações para a 5ª. entrega do projeto T-FIVE no site da disciplina no sistema Tidia Ae**

Informações sobre as bancadas de teste

- Como teste adicional, os alunos podem também utilizar o seguinte programa para testar este estágio ex:

```
Main:
    addi    sp,  zero,   4     # carrega sp = r2 = 4                                  PC = 00
    addi    a5,  zero,   15    # a5 = r15 = 15                                        PC = 04
    addi    ra,  zero,   10    # ra = r1 = 10                                         PC = 08
    addi    gp,  zero,   6     # gp = r3 = 6                                          PC = 0C
    add     sp,  ra,     gp    # Registrador sp escrito gp = r3, sp = r2 = 16         PC = 10
    slt     a2,  sp,     ra    # 1º  operando(sp) depende de ra = r1, a2 = r12 = 0    PC = 14
    slt     a3,  t1,     sp    # 2º  operando(sp) depende de t1 = r6, a3 = r13 = 1    PC = 18
    add     a4,  sp,     sp    # 1º (sp) & 2º (sp) depende de a4 = r14 = 32           PC = 1C
    sw      a5,  100(sp)       # A base (sp) depende de a5 = r15,M[116] = 15          PC = 20
fim:
    halt                       # Pare                                                 PC = 24
```

Informações sobre as bancadas de teste

- O código de máquina deste novo programa de teste a ser utilizada em conjunto com a Bancada de teste é o seguinte:

00000000100000000000100010011
00000000111100000000111110010011
00000000101000000000000010010011
00000000110000000000000110010011
00000000011000010000000100110011
00000000000100010010011000110011
00000000001000110010011010110011
00000000010000100000011100110011
00000110111100010010001000100011
00000000000000000000000001101111

- Se o programa assembler do slide anterior for submetido ao montador do T-FIVE, este código de máquina será obtido.

Perguntas e Dúvidas ?????

5a. Entrega: Implementação do estágio de Execução de instruções

- A **data limite** para esta entrega é: **15/07/2024 - 17:00 horas**

- Procurem fazer um **relatório que possa ser incrementado** `a medida que as próximas implementações forem sendo realizadas

Bom Trabalho a todos!!!!!!