UM PROCESSADOR SIMPLES

Versão 2015

RESUMO

Esta experiência tem como objetivo o desenvolvimento do projeto de um núcleo de um processador simples. Na parte experimental este projeto deverá ser sintetizado usando componentes MSI e SSI ou VHDL (fluxo de dados e unidade de controle) tendo como plataforma a placa de desenvolvimento FPGA DE2 da Altera.

OBJETIVOS

Após a conclusão desta experiência, os seguintes tópicos devem ser conhecidos pelos alunos:

- *Projeto monociclo de processadores;*
- *Projeto de circuitos digitais usando FPGA e VHDL;*
- *Projeto integrado em fluxo de dados e unidade de controle.*

1. PARTE TEÓRICA

1.1. Especificação do Processador Simples

O processador simples é um processador baseado no processador MIPS [Patterson & Hennessy, 2012] [Harris & Harris, 2013], simplificado para torná-lo adequado para um projeto didático. O conjunto de instruções a serem implementadas estão especificadas na tabela 1.

Tabela 1 – Conjunto de instruções do processador simples.

instr.	tipo	exemplo	significado
add	R	add $r5,$r1,$r2	R[rd] = R[rs] + R[rt]
sub	R	sub $r5,$r1,$r2	R[rd] = R[rs] – R[rt]
and	R	and $r5,$r1,$r2	R[rd] = R[rs] and R[rt]
or	R	or $r5,$r1,$r2	R[rd] = R[rs] or R[rt]
lw	I	lw $r5,8($r1)	R[rt] = Mem[R[rs]+extensão_com_sinal(imediato)]
sw	I	sw $r5,8(4r1)	Mem[R[rs]+extensão_com_sinal(imediato)] = R[rt]
beq	I	beq $r1,$r2,end	if (R[rs]=R[rt]) PC=PC+4+endereço_de_desvio
addi	I	addi $r3,$r7,4	R[rt] = R[rs] + extensão_com+sinal(imediato)
j	J	j end	PC = endereço de desvio

Este conjunto de instruções obedece ao formato padrão de instruções do **MIPS de 32 bits**, conforme detalhado na figura 1.1. Como no processador MIPS, temos 3 **tipos de instruções**: o primeiro tipo, chamado de **tipo R**, envolve operações sobre registradores do banco de registradores do processador. A operação a ser realizada é especificada pelo campo **função**. O segundo tipo (**tipo I**) envolve o acesso à memória para o acesso ao dado (lw e sw), operações com dados imediatos (addi) e para a especificação da próxima instrução a ser executada (beq). O cálculo deste **endereço de memória** para as instruções lw e sw é realizado pelo conteúdo do registrador rs mais um valor obtido pela extensão com sinal para 32 bits do valor do campo **imediato**. A instrução addi tem um dos operandos presente no registrador rs e o outro no campo imediato da instrução. No caso da instrução beq, o endereço é dado pelo valor atual do registrador PC mais o valor da extensão com sinal do campo imediato mais 4. O último tipo (**tipo J**) envolve o cálculo de um endereço de memória dado pelo campo **endereço de desvio** multiplicado por 4 (os 5 bits mais significativos do endereço são iguais ao do PC+4).

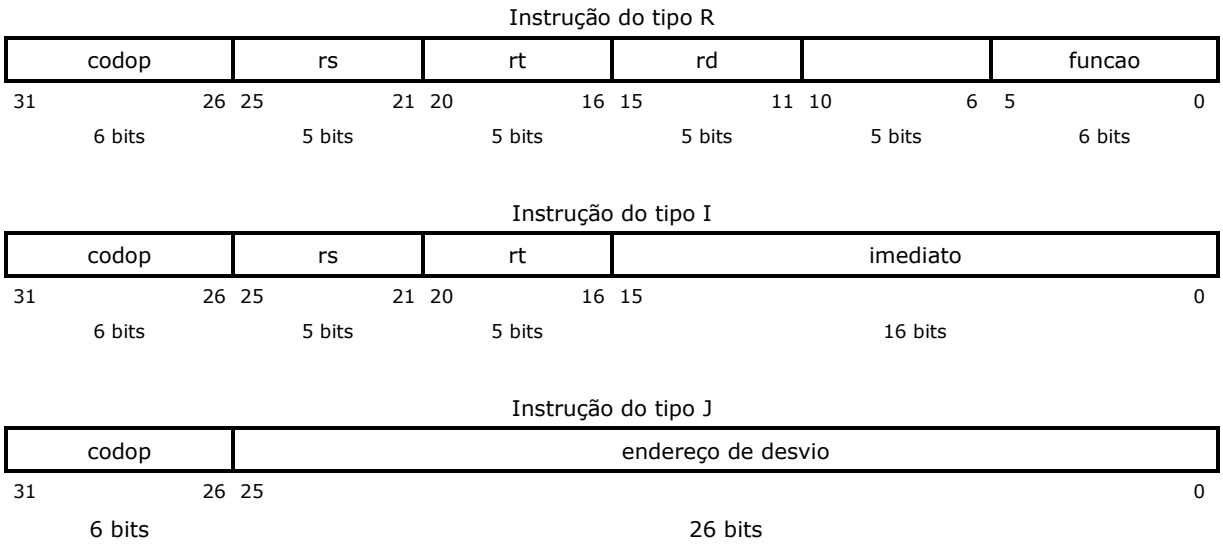


Figura 1.1 – Formatos das instruções do processador simples.

Uma implementação deste processador pode ser vista na figura 1.2. Neste diagrama temos representados os elementos principais do **fluxo de dados**. Este fluxo de dados é composto pelo registrador PC (contador de programa), memória de instruções, banco de registradores, ULA e memória de dados. Outros elementos menores do fluxo de dados incluem o circuito de extensão de valor com sinal para 32 bits, dois somadores e vários multiplexadores. Todos estes elementos têm palavras de 32 bits.

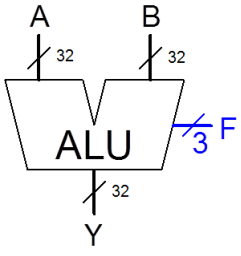
Temos também na figura 1.2 a **unidade de controle** que recebe a instrução a ser executada e gera os sinais de controle para os elementos do fluxo de dados. A tabela 2 apresenta os sinais de controle para cada tipo de instrução.

Tabela 2 – Sinais de controle para os elementos do fluxo de dados.

Instrução	codop	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	$ALUOp_{1:0}$	Jump
tipo R	000000	1	1	0	0	0	0	10	0
lw	100011	1	0	1	0	0	1	00	0
sw	101011	0	X	1	0	1	X	00	0
beq	000100	0	X	0	1	0	X	01	0
addi	001000	1	0	1	0	0	0	00	0
j	000010	0	X	X	X	0	X	XX	1

A operação que a ULA executa depende do sinal de controle ALUop e do campo função da instrução. A figura 1.3 detalha estes sinais de controle.

ALUOp[1:0]	descrição
00	Soma
01	Subtração
10	Depende de função
11	Não usado



F[2:0]	Função
000	A and B
001	A or B
010	A + B
011	não usado
100	não usado
101	não usado
110	A – B
111	não usado

Figura 1.3 – Sinais de controle da ULA.

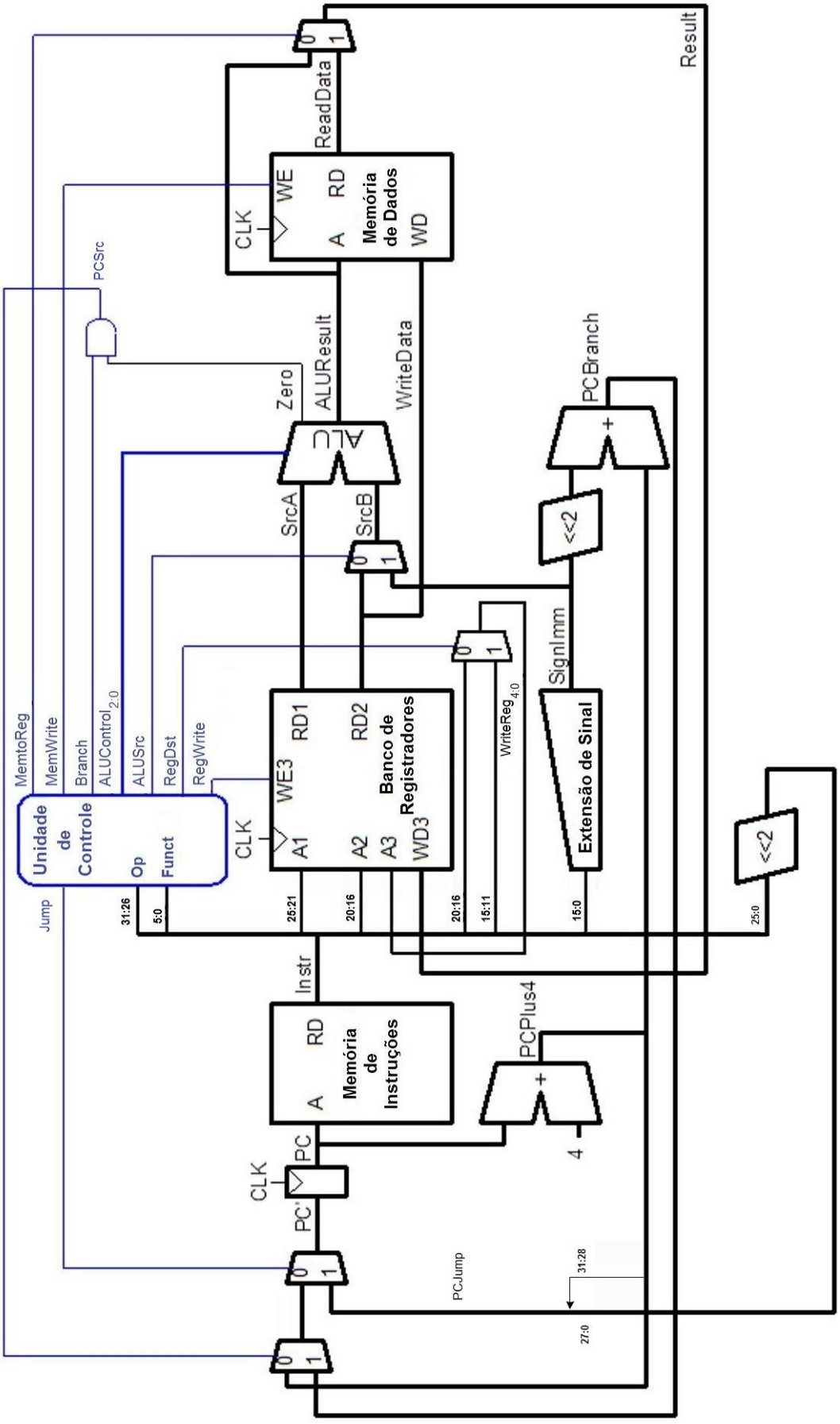


Figura 1.2 – Diagrama geral da microarquitetura do processador MIPS (fonte: Harris & Harris, 2013).

1.2. Algumas Simplificações do Projeto

De forma a simplificar alguns aspectos do projeto, iremos restringir certas características do circuito final do processador. A primeira simplificação diz respeito ao **tamanho das memórias** de instruções e de dados. Embora os endereços envolvidos no projeto tenham 32 bits, ambas as memórias terão apenas 64 endereços, ou seja, serão memórias de 64x32.

DICA: Embora a arquitetura do processador seja de 32 bits, iremos reduzir o espaço de endereçamento das memórias. Desta forma, os bits 2 a 7 do endereço de 32 bits devem ser ligados à entrada de endereços das memórias. A ligação destes bits fará com que as memórias sejam endereçadas como um conjunto de palavras de 32 bits ao invés de considerarmos endereços de byte.

Outro aspecto a ser considerado leva em conta o **teste do circuito**. Para isto os conteúdos iniciais das memórias deverão ser especificados pelos arquivos **mem_instrucoes.mif** e **mem_dados.mif**. Com isto, tanto o programa de teste como alguns dados necessários podem ser pré-estabelecidos no circuito. Um exemplo disto pode ser especificar valores na memória de dados necessários para o teste (figura 1.4).

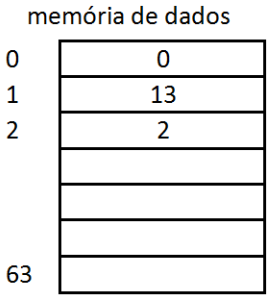


Figura 1.4 – Exemplo de inicialização da memória de dados.

O banco de registradores do processador MIPS é composto por 32 registradores de 32 bits cada. No projeto do processador simples, o **banco de registradores** será composto por 8 registradores de 32 bits. Ele é organizado, como no MIPS, em uma estrutura com três portas de acesso: duas portas de leitura RD1 e RD2, que mostram dados armazenados em registradores identificados pelas entradas A1 e A2. Esta leitura é feita de forma assíncrona, ou seja, assim que uma das entradas de endereço de leitura A1 ou A2 (ou ambas) for modificada, a respectiva saída mostrará o dado. A terceira porta WD3 é para gravar um novo dado no registrador identificado pela entrada A3. Esta operação de escrita é realizada de forma síncrona com o acionamento da entrada de relógio CLK e pela habilitação da operação através do sinal WE3. A figura 1.5 ilustra a estrutura do banco de registradores.

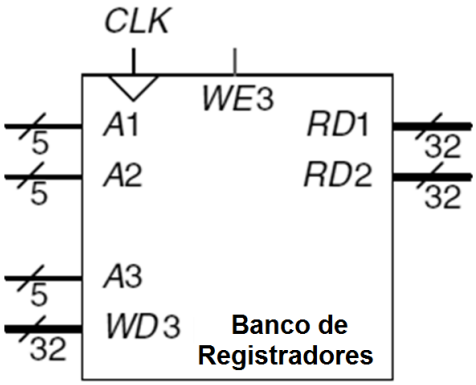


Figura 1.5 – Banco de registradores do processador MIPS.

Os registradores do banco de registradores serão referenciados com nomes de $r0 a $r7 e identificados com números de 0 a 7 no circuito. Os bits da interface do circuito do banco de registradores continuam com 5 bits, portanto os dois bits mais significativos devem ser ignorados pelo projeto.

1.3. A Unidade de Controle

Embora o fluxo de dados do projeto seja concebido para ser executado em um único ciclo de relógio (projeto monociclo), a unidade de controle do processador simples deverá seguir uma temporização diferente. Com base em um sinal de *clock* geral, os sinais de controle dos elementos deverão ser gerados com base no diagrama de transição de estados de alto nível da figura 1.6.

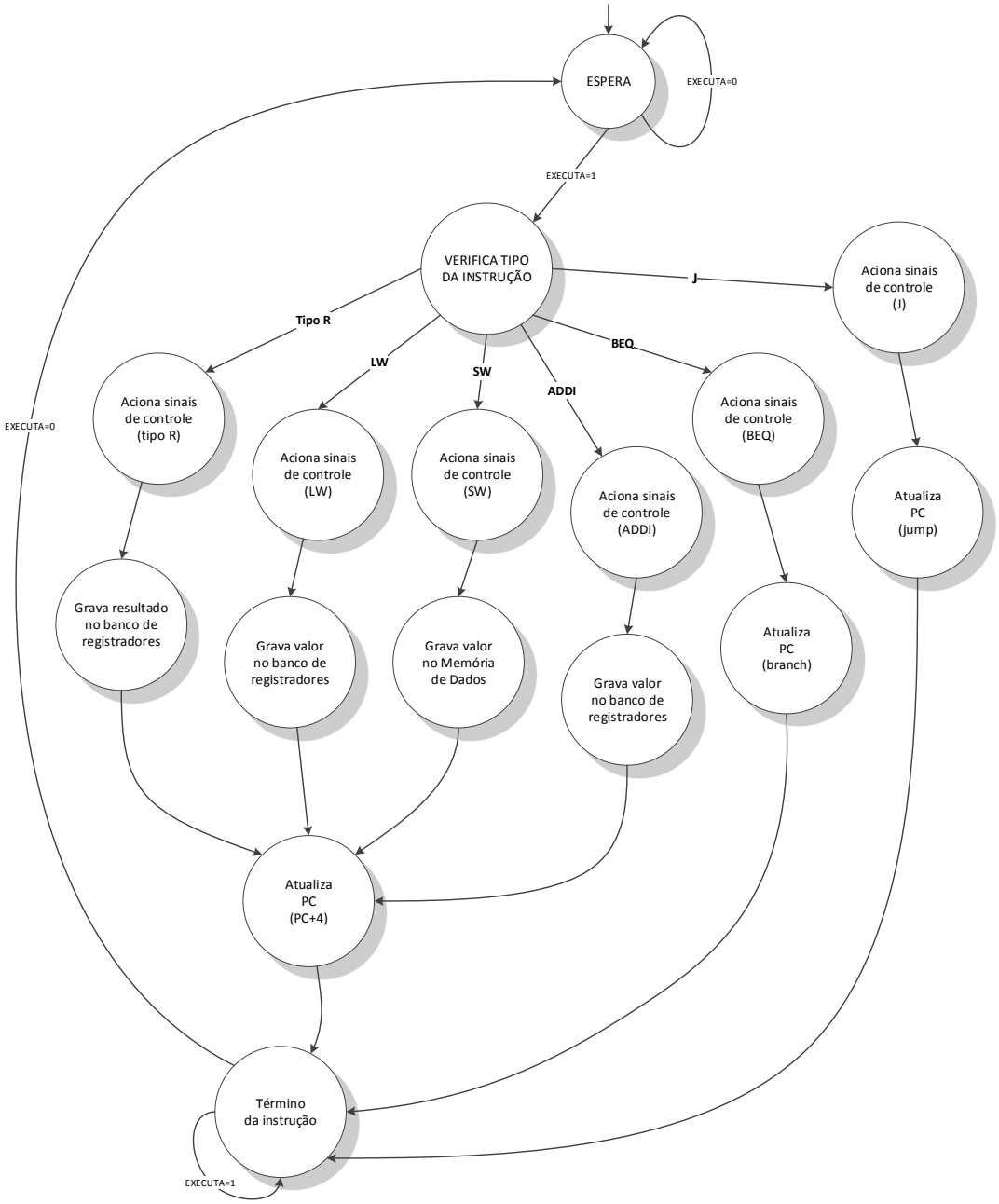


Figura 1.6 – Diagrama de Transição de Estados do Processador Simples.

O projeto da unidade de controle também precisa levar em conta o acionamento do sinal EXECUTA. A execução da instrução corrente inicia somente com o seu acionamento. Ao final da execução da instrução, o circuito deve aguardar o desacionamento do sinal antes de dar início à execução da próxima instrução. Elabore um **diagrama ASM** ou diagrama de transição de estados detalhando os sinais de controle acionados em cada estado. Se necessitar, novos estados podem ser criados.

2. PARTE EXPERIMENTAL

2.1. Atividades Pré-Laboratório

a) **Projeto do Processador Simples**. O projeto do processador simples deve ser realizado a partir da especificação apresentada a seguir.

A operação do projeto do processador simples deverá seguir os seguintes passos:

1. O circuito é programado na placa de desenvolvimento;
2. Os *displays* HEX7 e HEX6 devem apresentar os 8 bits menos significativos do conteúdo do registrador PC (iniciados em 00);
3. As instruções presentes na memória de instruções são executadas uma a uma com o acionamento do botão EXECUTA;
4. Para acompanhar a execução das instruções, resultados parciais do programa devem ser mostrados nos *displays* HEX1 e HEX0, que devem apresentar os 8 bits menos significativos do registrador $r1 do processador;
5. A qualquer momento, o processador pode ser reiniciado (execução volta para a primeira instrução da memória de programas) com o acionamento do sinal RESET. Porém, o conteúdo da memória de dados deve ser mantido inalterado, ou seja, os dados alterados até o momento ficam na memória.

O circuito do processador simples deverá ser desenvolvido para a placa de desenvolvimento FPGA DE2 da Altera. Para seu funcionamento a seguinte designação de pinos deverá ser adotada:

- EXECUTA : botão KEY0
- RESET : botão KEY3
- CLOCK : clock interno de 50 MHz (CLK_50)
- registrador PC : display HEX7 e HEX6
- saída de dados ($r1) : display HEX1 e HEX0

DICA: lembrem-se que os botões na placa DE2 são ativos em baixo.

Sinais adicionais de depuração devem ser especificados e podem usar outros recursos da placa. O planejamento deve conter uma tabela com a designação destes sinais.

DICA 1: Para os testes do circuito, sugere-se que sinais extras de teste e depuração sejam monitorados em *leds* e *displays* na placa de desenvolvimento. Estes sinais devem monitorar partes do fluxo de dados e também o **estado da máquina de estados** da unidade de controle.

DICA 2: Tragam **cópias impressas** do diagrama de blocos ou diagrama lógico do fluxo de dados e do diagrama ASM ou diagrama de transição de estados do projeto base do processador simples para arguição pelo professor e para serem usados na modificação do circuito.

2.2. Implementação na Placa DE2

b) Execute a programação do projeto do processador simples na placa DE2 e inicie os procedimentos iniciais de teste e depuração do circuito.

c) Anote no relatório os procedimentos executados e os resultados obtidos.

2.3. Testes do Processador Simples

d) Verifique o funcionamento do circuito para várias sequências de operações simples, variando também os dados de entrada. Explique os testes programados.

e) Estude o seguinte "programa" e explique o que acontece a cada instrução até o final de sua execução.

```
end     instrução           ; comentário (valores numéricos em hex)
00      lw  $r1, 4($r0)     ; R1 <= 5 (* saída no display deve mostrar 5)
04      lw  $r2, 8($r0)     ; R2 <= 1
08      add $r3, $r1, $r2   ; R3 <= 5+1=6
0C      sw  $r3, 10($r0)    ; MD[16] <= R3=6 (guarda 6 na memória de dados)
10      add $r1, $r3, $r0   ; R1 <= R3 (display mostra 6)
14      beq $r2, $r2, 5     ; testa R2=R2 e desvia para end.2c => 2c-14-4=14, 14/4=5
18      lw  $r4, c($r0)     ; carrega 6 em R4
1C      lw  $r5, 14($r0)    ; carrega 2 em R5
20      sub $r6, $r4, $r5   ; R6 recebe 6-2=4
24      add $r1, $r6, $r0   ; transfere 4 para R1 (display)
28      j   a               ; fica no endereço 28 (loop) => 28/4=a
2C      lw  $r2, 10($r0)    ; R2 <= M[16]=6
30      lw  $r1, c($r0)     ; R1 <= MD[12]=3
34      and $r3, $r1, $r2   ; R3 <=6and3=2
38      add $r1, $r3, $r0   ; transfere 2 para R1 (display)
40      j   6               ; desvia para o endereço 18 => 18/4=6
```

f) Considere o projeto da memória de programa com 64x32 bits. Traga um arquivo chamado **memoria_programa.mif** que contém as instruções acima codificadas em binário.

g) Considere que a memória de dados de 64x32 bits tem os valores iniciais abaixo. Crie um arquivo chamado **memoria_dados.mif** com este conteúdo.
DICA: os outros endereços não especificados devem ser iniciados com zeros.

endereço	conteúdo
0	00000000
1	00000005
2	00000001
3	00000003

h) Execute o "programa" acima e anote o que aconteceu após a execução de cada instrução.

i) Em seguida, escreva programas para a avaliação das seguintes expressões:

I. Y = (2×(A + B)) AND (C − D)

II. Z = (A OR B) AND (NOT(C) OR D)

j) Verifique os resultados obtidos executando os "programas", para diferentes valores de A, B, C e D. Não miminize ou modifique as expressões acima.

ATENÇÃO: O planejamento deve trazer a codificação binária destes programas para sua execução no Laboratório Digital.
DICA: use um dos programas de simulação do processador MIPS indicados.

2.4. Desafio

k) Implemente a **modificação** proposta pelo professor. Documente o circuito e apresente seu funcionamento de forma objetiva e convincente.

2.5. Atividades Pós-Laboratório

I) Após a conclusão das atividades programadas, responda as perguntas abaixo:

1. *Mostre outra forma de acompanhar o andamento da execução do programa com a visualização do conteúdo dos registradores.*
2. *Como o projeto da Unidade de Controle pode ser modificado para incluir uma nova instrução? Por exemplo, ilustre a implementação da instrução jr (jump register).*

3. BIBLIOGRAFIA

- ALTERA. **DE2 Development and education board user manual**. 2008. Version 1.42.
- ALTERA. **Design Debugging Using the SignalTap II Embedded Logic Analyzer**. Quartus II Handbook Version 9.1 Volume 3: Verification. November 2009.
- HARRIS, D.M. & HARRIS, S.L. **Digital Design and Computer Architecture**, 2nd ed., Morgan Kaufmann, 2013.
- MENEZES, M.P.; SATO, L.M.; MIDORIKAWA, E.T. **Projeto de Circuitos com Quartus II 9.1**. Apostila de Laboratório Digital. Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da USP. Edição de 2011.
- PATTERSON, D.A. & HENNESSY, J.L. **Computer Organization and Design: the hardware/software interface**. Revised 4th ed., Morgan Kaufmann, 2012.
- TOCCI, R. J., WIDMER, N. S., MOSS, G.L. **Digital Systems: principles and applications**. 11th ed., Prentice-Hall, 2011.
- WAKERLY, J. F. **Digital Design: principles and practice**. 4th ed., Prentice- Hall, 2006.

4. EQUIPAMENTOS NECESSÁRIOS

- 1 placa de desenvolvimento FPGA DE2 da Altera com o dispositivo Altera Cyclone II EP2C35F672C6.
- 1 computador PC com programa Altera Quartus II e interface USB.

Histórico de Revisões
E.T.M./2013 – versão inicial.
E.T.M./2014 – revisão.
E.T.M./2015 – revisão.