Machine Architecture
Assignment 2

Casper B. Hansen
University of Copenhagen
Department of Computer Science
fvx507@alumni.ku.dk

Sine Vestergård Jensen
University of Copenhagen
Department of Computer Science
kms698@alumni.ku.dk

Nikolaj Høyer
University of Copenhagen
Department of Computer Science
ctl533@alumni.ku.dk

October 14, 2013

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Contents

1 Overview

...

1.1 Instruction set

Our pipeline is quite simple, its instruction set (see Figure 1) consists of only 14 instructions.

Arithmetic-logical

Mnemonic	Code	Type	Description
addu	0x21	R	Add unsigned
addiu	0x09	I	Add imm. unsigned
slt	0x2A	R	Set less than
slti	0x0A	I	Set imm. less than
subu	0x23	R	Subtract unsigned
and	0x24	R	Logical AND
andi	0x0C	I	Logical imm. AND
or	0x25	R	Logical OR
ori	0x0D	I	Logical imm. OR

Memory-reference

Mnemonic	Code	Type	Description
lw	0x23	I	Load word
sw	0x2B	I	Store word

Branching

Mnemonic	Code	Type	Description
beq	0x04	I	Branch on equal
jal	0x03	J	Jump and link
jr	0x08	R	Jump to register

Figure 1: Instruction set

1.2 Pipeline

...

2 Preliminary design

...

3 Tests

3.1 Arithmetic-logical operations

...

3.1.1 Forwarding

...

...

```
addiu $s0, $zero, 5
addiu $s1, $zero, 4
addu $s2, $s0, $s1
```

...

3.2 Memory-reference operations

...

...

```
```

...

3.3 Branching operations

...

3.3.1 Hazards

...

...

```
beq $zero, $zero, label
addiu $s0, $zero, 5
addiu $s1, $zero, 4

label:
addiu $s2, $zero, 3
```

...

References

[1] David A. Patterson, John L. Hennessy, *Computer Organization and Design.* Morgan Kaufmann, Revised 4th Edition, 2009.