**10520 CS410001 - Computer Architecture 2017**

**Appendix A for Projects**

Datasheet for the Reduced MIPS R3000 ISA

Table 1: R-Type Instructions

| R | 31 26 opcode(6) | 25 21 rs(5) | 20 16 rt(5) | 15 11 rd(5) | 10 6 C(shamt)(5) | 5 0 funct(6) | Syntax | Semantic |
|---|---|---|---|---|---|---|---|---|
| add | 0x00 | | | | x | 0x20 | add $d,$s,$t | $d = $s + $t |
| addu | 0x00 | | | | x | 0x21 | add $d,$s,$t | $d = $s + $t(unsigned, no overflow exception) |
| sub | 0x00 | | | | x | 0x22 | sub $d,$s,$t | $d = $s - $t |
| and | 0x00 | | | | x | 0x24 | and $d,$s,$t | $d = $s & $t |
| or | 0x00 | | | | x | 0x25 | or $d,$s,$t | $d = $s \| $t |
| xor | 0x00 | | | | x | 0x26 | xor $d,$s,$t | $d = $s ^ $t |
| nor | 0x00 | | | | x | 0x27 | nor $d,$s,$t | $d = ~ ($s \| $t) |
| nand | 0x00 | | | | x | 0x28 | nand $d,$s,$t | $d = ~($s & $t) |
| slt | 0x00 | | | | x | 0x2A | slt $d,$s,$t | $d = ($s < $t), signed comparison |
| sll | 0x00 | x | | | | 0x00 | sll $d,$t,C | $d = $t << C |
| srl | 0x00 | x | | | | 0x02 | srl $d,$t,C | $d = $t >> C |
| sra | 0x00 | x | | | | 0x03 | sra $d,$t,C | $d = $t >> C, with sign bit shifted in |
| jr | 0x00 | | x | x | x | 0x08 | jr $s | PC=$s |
| mult | 0x00 | | | x | x | 0x18 | mult $s $t | {Hi \|\| Lo} = $s * $t |
| multu | 0x00 | | | x | x | 0x19 | multu $s $t | {Hi \|\| Lo} = $s * $t (unsigned, no overflow exception) |
| mfhi | 0x00 | x | x | | x | 0x10 | mfhi $d | $d = Hi |
| mflo | 0x00 | x | x | | x | 0x12 | mflo $d | $d = Lo |

Table 2: I-Type Instructions

| I | 31 26 opcode(6) | 25 21 rs(5) | 20 16 rt(5) | 15 0 C(immediate)(16) | Syntax | Semantic |
|---|---|---|---|---|---|---|

| I | 31 26 opcode(6) | 25 21 rs(5) | 20 16 rt(5) | 15 0 C(immediate)(16) | Syntax | Semantic |
|---|---|---|---|---|---|---|
| addi | 0x08 | | | | addi $t,$s,C | $t = $s + C(signed) |
| addiu | 0x09 | | | | addi $t,$s,C | $t = $s + C(unsigned, no overflow exception) |
| lw | 0x23 | | | | lw $t,C($s) | $t = 4 bytes from Memory[$s + C(signed)] |
| lh | 0x21 | | | | lh $t,C($s) | $t = 2 bytes from Memory[$s + C(signed)], signed |
| lhu | 0x25 | | | | lhu $t,C($s) | $t = 2 bytes from Memory[$s + C(signed)], unsigned |
| lb | 0x20 | | | | lb $t,C($s) | $t = Memory[$s + C(signed)], signed |
| lbu | 0x24 | | | | lbu $t,C($s) | $t = Memory[$s + C(signed)], unsigned |
| sw | 0x2B | | | | sw $t,C($s) | 4 bytes from Memory[$s + C(signed)] = $t |
| sh | 0x29 | | | | sh $t,C($s) | 2 bytes from Memory[$s + C(signed)] = $t & 0x0000FFFF |
| sb | 0x28 | | | | sb $t,C($s) | Memory[$s + C(signed)] = $t & 0x000000FF |
| lui | 0x0F | x | | | lui $t,C | $t = C << 16 |
| andi | 0x0C | | | | andi $t,$s,C | $t = $s & C(unsigned) |
| ori | 0x0D | | | | ori $t,$s,C | $t = $s \| C(unsigned) |
| nori | 0x0E | | | | nori $t,$s,C | $t = ~($s \| C(unsigned)) |
| slti | 0x0A | | | | slti $t,$s,C | $t = ($s < C(signed) ), signed comparison |
| beq | 0x04 | | | | beq $s,$t,C | if ($s == $t) go to PC+4+4*C(signed) |
| bne | 0x05 | | | | bne $s,$t,C | if ($s != $t) go to PC+4+4*C(signed) |
| bgtz | 0x07 | | x | | bgtz $s,C | if ($s > 0) go to PC+4+4*C(signed) |

Table 3: J-Type Instructions

| J | 31 26 opcode(6) | 25 0 C(address)(26) | Syntax | Semantic |
|---|---|---|---|---|
| j | 0x02 | | j C | PC = (PC+4)[31:28] \| 4*C(unsigned) |
| jal | 0x03 | | jal C | $31 = PC + 4;PC = (PC+4)[31:28] \| 4*C(unsigned) |

Table 4: Specialized Instruction (to Terminate Simulation)

| S | 31 26 | 25 0 | Syntax | Semantic |
|---|---|---|---|---|

| | **opcode(6)** | **C(address)(26)** | | |
|---|---|---|---|---|
| halt | 0x3F | x | halt | halt the simulation |

In our simulator, all the comparison operation should be implemented using “==” operation instead of hardware subtraction comparison.