VE370 Review class (Week 2)

CPU Time

Equations

CPU Time = CPU Clock Cycle per program * Clock Cycle Time

= CPU Clock Cycles/ Clock Rate

Clock Cycles = Instruction Count (IC) * Clock Cycle per Instruction(CPI)

CPU Time = Instruction Count (IC) * CPI * Clock Cycle Time

= IC *CPI / Clock Rate

CPU Time = (Instruction / Program) * (Clock cycles / Instruction) * (Seconds/Clock cycle)

Exercise

Calculate their CPU time

PROCESSOR	CLOCK RATE	CPI	NO. INSTR
P1	4GHz	0.9	5.0*10^(6)
P2	3GHz	0.75	1.0*10^(6)

P1: 5*10^6 *0.9/(4 *10^9) =0.001125s

P1: 1*10^6 *0.75/(3 *10^9) =0.00025s

Registers

$zero

number: 0, Use: The constant value 0

$v0, $v1

number:2-3, Use: value for function results and Expression Evaluation

$a0-$a3

number: 4-7, Use: Arguments

$t0-$t7, $t8,$t9

number: 8-15, 24,25Use: temporaries

$s0-$s7

number: 16-23,Use: saved temporaries

$gp

number: 28,Use: global pointers

$sp

number: 29,Use: stack pointers

$ra

number: 31,Use: return address

MIPS Instruction

(refer to MIPS Instruction card)

Addition and subtraction

add/addi/addiu/addu/sub/subu

Example

add $t1, $t2, $t3

$t1= $t2+$t3

addi $t1, $t2, 10

$t1= $t2+10

Multiply and divide

sll rd, rt, shamt

sll $s0, s1, 2

$s0= s1*4

srl rd, rt, shamt

srl $s0, s1, 2

$s0= s1/4

Memory operands

Big Endian vs little Endian

32bit number 1020A0B0

Big Endian

Most significant byte at least address of a word

ADDRESS	0XFFFF0000	0XFFFF0001	0XFFFF0002	0XFFFF0003
Content	10	20	A0	B0

little Endian

ADDRESS	0XFFFF0000	0XFFFF0001	0XFFFF0002	0XFFFF0003
Content	B0	A0	20	10

lw/sw/lb/sb/lbu....

load word

lw $s1, 100($s2)

$s1 = Mem[100+$s2]

sw $s1, 100($s2)

Mem[100+$s2]= $s1

Exercise: load 32-bit constant

Combine use lui and ori

(lui: load the immediate to the upper 16bit of the register

lui $s1, 100

$s1 = 100*2^16

ori : or immediate

ori $t1, $t2, 10

$t1= $t2| ZeroExtImm

If we want to load 0x56781234 to register $s3

The MIPS assembly will be

lui $s3, 0x5678

ori $s3, $s3, 0x1234

Difference between lb and lbu

lb:load byte

R[rt] = SignExt(M[R[rs]+SignExtImm])

lbu: load byte unsigned

R[rt] = {24b'0,M[R[rs]+SignExtImm] (7:0)}

Exercise

	0	1	2	3
ox10007896	26	78	99	20

how to load byte 99 to $s2, load byte 26 to $s5?

Solution

lui $s1, 4096

ori $s1, $s1, 0x7896

lbu $s2, 2($s1)

lb $s5, 0($s1) [or lbu $s5, 0($s1)]

For the number stored in a byte if its value is larger than **(10000000) or(80)hex**. If we want to load its original value to a new register, we need to use lbu. If we use lb at this time, the value stored in the new register will be negative.

Branch/Jump operation

beq rs,rt, L1

if rs==rt branch to instruction labeled L1

bne rs,rt, L1

if rs !=rt branch to instruction labeled L1

j L1

unconditionally jump to instruction labeled L1

other instructions

blt, bgt,ble,bge

Take blt as an example

blt rs,rt, L1

if rs < rt, branch to instruction labeled L1

it equals to the following MIPS instruction

slt $s1, rs, rt

bne $s1, $0, L1

slt rd, rs, rt

if rs<rt, rd = 1, else rd =0

Tips about Project 1

1. add meaningless instruction like add $t0, $t0, $0 before/ after jump/jal/ jr instructions
2. Strictly follow the function calling convention