Amit Nijjar

A11489111

CSE141 SP 2016

4/12/16

CSE141 HW 1

**Chapter 1:** 1.9

1.9) Assume for arithmetic, load/store, and branch instructions, a processor has CPIs of 1, 12, and 5, respectively. Also assume that on a single processor a program requires the execution of 2.56E9 arithmetic instructions, 1.28E9 load/store instructions, and 256 million branch instructions. Assume that each processor has a 2 GHz clock frequency.

Assume that, as the program is parallelized to run over multiple cores, the number of arithmetic and load/store instructions per processor is divided by 0.7 x *p* (where *p* is the number of processors) but the number of branch instructions per processor remains the same.

**1.9.1** [5] <§1.7> Find the total execution time for this program on 1, 2, 4, and 8

processors, and show the relative speedup of the 2, 4, and 8 processor result relative

to the single processor result.

ET = IC \* CPI \* Cycle Time

1 processor:

ET(1) = ((1\*2.56x10^9)+(12\*1.28x10^9)+(5\*256,000,000))\*1/(2x10^9)

ET(1) = 9.6 seconds

2 processor:

ET(2) = ((1\*2.56x10^9\*1/1.4)+(12\*1.28x10^9\*1/1.4)+(5\*256,000,000))\*1/(2x10^9)

ET(2) = 7.04 seconds

4 processor:

ET(4) = ((1\*2.56x10^9\*1/2.8)+(12\*1.28x10^9\*1/2.8)+(5\*256,000,000))\*1/(2x10^9)

ET(4) = 3.84 seconds

8 processor:

ET(8) = ((1\*2.56x10^9\*1/5.6)+(12\*1.28x10^9\*1/5.6)+(5\*256,000,000))\*1/(2x10^9)

ET(8) = 2.24 seconds

**1.9.2** [10] <§§1.6, 1.8> If the CPI of the arithmetic instructions was doubled,

what would the impact be on the execution time of the program on 1, 2, 4, or 8

processors?

1 processor:

ET(1) = ((2\*2.56x10^9)+(12\*1.28x10^9)+(5\*256,000,000))\*1/(2x10^9)

ET(1) = 10.88 seconds

ET Difference = 10.88 – 9.6 = 1.28 seconds

2 processor:

ET(2) = ((2\*2.56x10^9\*1/1.4)+(12\*1.28x10^9\*1/1.4)+(5\*256,000,000))\*1/(2x10^9)

ET(2) = 7.95 seconds

ET Difference = 7.95 – 7.04 = 0.91 seconds

4 processor:

ET(4) = ((2\*2.56x10^9\*1/2.8)+(12\*1.28x10^9\*1/2.8)+(5\*256,000,000))\*1/(2x10^9)

ET(4) = 4.30 seconds

ET Difference = 4.30 – 3.84 = 0.46 seconds

8 processor:

ET(8) = ((2\*2.56x10^9\*1/5.6)+(12\*1.28x10^9\*1/5.6)+(5\*256,000,000))\*1/(2x10^9)

ET(8) = 2.47 seconds

ET Difference = 2.47 – 2.24 = 0.23 seconds

**1.9.3** [10] <§§1.6, 1.8> To what should the CPI of load/store instructions be

reduced in order for a single processor to match the performance of four processors

using the original CPI values?

Solve for Y

3.84 seconds = ((1\*2.56x10^9)+(12\*1.28x10^9)+((Y)\*256,000,000))\*1/(2x10^9)

7.68 x 10^9 = ((1\*2.56x10^9)+(12\*1.28x10^9)+((Y)\*0.256x10^9))

7.68 x 10^9 = (1\*2.56+12\*1.28+(Y)\*0.256) x10^9

7.68 = 1\*2.56+12\*1.28+(Y)\*0.256

5.12 = 12\*1.28+(Y)\*0.256

-10.24 = (Y)\*0.256

-40 = y

No Solution

**Chapter 2:** 2.27, 2.28, 2.29, 2.30 and 2.34

2.27) Translate the following C code to MIPS assembly code. Use a minimum number of instructions. Assume that the values of a, b, i, and j are in registers $s0, $s1, $t0, and $t1, respectively. Also, assume that register $s2 holds the base address of the array D.

for(i=0; i<a; i++)

for(j=0; j<b; j++)

D[4\*j] = i + j;

addi $t0, $0, 0

beq $0, $0, TESTA

LOOPA:addi $t1, $0, 0

beq $0, $0, TESTB

LOOPB:add $t3, $t0, $t1

sll $t2, $t1, 4

add $t2, $t2, $s2

sw $t3, $t2

addi $t1, $t1, 1

TESTB:slt $t2, $t1, $s1

bne $t2, $0, LOOPB

addi $t0, $t0, 1

TESTA:slt $t2, $t0, $s0

bne $t2, $0, LOOPA

2.28) How many MIPS instructions does it take to implement the C code from Exercise 2.27? If the variables a and b are initialized to 10 and 1 and all elements of D are initially 0, what is the total number of MIPS instructions that is executed to complete the loop?

It takes 14 instructions to implement and 158 instructions to execute

2.29) Translate the following loop into C. Assume that the C-level integer i is held in register $t1, $s2 holds the C-level integer called result, and $s0 holds the base address of the integer MemArray.

addi $t1, $0, $0

LOOP: lw $s1, 0($s0)

add $s2, $s2, $s1

addi $s0, $s0, 4

addi $t1, $t1, 1

slti $t2, $t1, 100

bne $t2, $s0, LOOP

for (i=0; i<100; i++) {

result += MemArray[s0];

s0 = s0 + 4;

}

2.30) Rewrite the loop from Exercise 2.29 to reduce the number of MIPS instructions executed.

addi $t1, $s0, 400

LOOP: lw $s1, 0($t1)

add $s2, $s2, $s1

addi $t1, $t1, −4

bne $t1, $s0, LOOP

2.34) Translate function f into MIPS assembly language. If you need to use registers $t0 through $t7, use the lower-numbered registers fi rst. Assume the function declaration for func is “int f(int a, int b);”. Th e code for function f is as follows:

int f(int a, int b, int c, int d){

return func(func(a,b),c+d);

}

f: addi $sp,$sp,−12

sw $ra,8($sp)

sw $s1,4($sp)

sw $s0,0($sp)

move $s1,$a2

move $s0,$a3

jal func

move $a0,$v0

add $a1,$s0,$s1

jal func

lw $ra,8($sp)

lw $s1,4($sp)

lw $s0,0($sp)

addi $sp,$sp,12

jr $ra