

CSC3420 Computer System Architectures
2nd Term, Year 2008-2009 Quiz 1
Suggested Solution

Q1) Performance

- a. The average CPI for:

$$\text{Wintel + Peter: } 6 * 50\% + 7 * 20\% + 15 * 30\% = 8.9$$

$$\text{Wintel + Ricky: } 6 * 40\% + 7 * 40\% + 15 * 20\% = 8.2$$

$$\text{CMD + Peter: } 4 * 50\% + 9 * 20\% + 17 * 30\% = 8.9$$

$$\text{CMD + Ricky: } 4 * 40\% + 9 * 40\% + 17 * 20\% = 8.6$$

[12 marks, 3%*4]

- b. Time for running experiment:

$$\text{Wintel + Peter: } 1900 * 10^6 \times 8.9 / 1800 * 10^6 = 9.394\text{s}$$

$$\text{Wintel + Ricky: } 1500 * 10^6 \times 8.2 / 1800 * 10^6 = 6.833\text{s}$$

$$\text{CMD + Peter: } 1900 * 10^6 \times 8.9 / 2100 * 10^6 = 8.052\text{s}$$

$$\text{CMD + Ricky: } 1500 * 10^6 \times 8.6 / 2100 * 10^6 = 6.143\text{s}$$

Because both programs always run faster on CMD machine, both programs should be run on machine CMD and the total time needed is 8.052s.

[16 marks, 3%*4+4%]

- c. Assume the new cycles per I/O instruction is y

$$1500 * 10^6 \times (6 \times 40\% + 7 \times 40\% + y \times 20\%) / 1800 * 10^6 < 6.143$$

$$y < 10.860$$

So, the cycles per I/O instruction can be at most 10.

[7 marks, 4%+3%]

- d. Assume the new cycles per integer instruction is y

$$1500 * 10^6 \times (y \times 40\% + 9 \times 40\% + 17 \times 20\%) / 2100 * 10^6 < 6.143 / 3$$

$$y < -10.333$$

As the CPI cannot be a negative number, therefore, it is impossible for CMD to make such improvement.

[5 marks, 3%+2%]

[Alternative]

Assume the new cycles per integer instruction is improved to 0

$$1500 * 10^6 \times (0 \times 40\% + 9 \times 40\% + 17 \times 20\%) / 2100 * 10^6 = 5.000\text{s}$$

Even CMD improves the cycles per Integer instruction to 0, the running time of Ricky's program still cannot be three times faster, $6.143\text{s} / 3 = 2.048\text{s}$. Therefore, it is impossible for CMD to make such improvement.

[5 marks, 3%+2%]

Q2) MIPS: Pseudorandom Generator**[30 marks]**

```
    addi    $t1, $zero, 1           # $t1 = 1;
    la     $t2, initial             # $t2 = initial
    lw     $t3, 0($t2)              # $t3 = modulus;
    lw     $t4, 4($t2)              # $t4 = multiplier;
    lw     $t5, 8($t2)              # $t5 = increment;
    lw     $t6, 12($t2)             # $t6 = initial_seed;
    la     $t2, initial             # $t2 = answer
    sub    $t9, $t9, $t1            # $t9 = $t9 - 1
    addi   $t8, $zero, 10           # $t8 = 10;
    add    $t7, $zero, $t6          # $t7 = initial_seed;
Loop:    beq    $t8, $zero, Done     # while ($t8 != 0) {
    mult   $t7, $t4                  #   $hi,$lo = Xn-1 * multiplier;
    add    $t7, $zero, $lo          #   $t7 = $lo;
    add    $t7, $t7, $t5            #   $t7 = $t7 + increment;
    and    $t7, $t7, $t9            #   $t7 = $t7 mod modulus;
    sw     $t7, 0($t2)              #   Mem[$t2] = $t7;
    addi   $t2, $t2, 4              #   $t2++;
    sub    $t8, $t8, $t3            #   $t8--;
    j      Loop                     # }
Done:
```

Simulation:

[30 marks]

Cycle	MIPS	Used Registers' Values	Comment
1	addi \$t1, \$zero, 1	\$t1 = 1	Initialize \$t1 to value 1
2	la \$t2, initial	\$t2 = initial	Initialize \$t2, array ptr, to value initial
3	lw \$t3, 0(\$t2)	\$t3 = 0x0000800	Save the modulus to \$t3
4	lw \$t4, 4(\$t2)	\$t4 = 0x00343FD	Save the multiplier to \$t4
5	lw \$t5, 8(\$t2)	\$t5 = 0x0269EC3	Save the increment to \$t5
6	lw \$t6, 12(\$t2)	\$t6 = 0x0000015	Save the initial_seed to \$t6
7	la \$t2, answer	\$t2 = 0	Initialize \$t2, array ptr, to value answer
8	sub \$t9, \$t9, \$t1	\$t9 = 0x00007FF	Set the and value for modulus
9	addi \$t8, \$zero, 10	\$t8 = 10	Initialize \$t8, iteration ptr, to value 10
10	add \$t7, \$zero, \$t6	\$t7 = 0x0000015	Save the X_0 to \$t7
11	beq \$t8, \$zero, Done		Jump to Done if \$t8 decrease to 0
12	mult \$t7, \$t4		Multiply \$t7 and \$t4 into \$hi, \$lo
13	add \$t7, \$zero, \$lo	\$t7 = 0x04493C1	Extract the \$lo to \$t7
14	add \$t7, \$t7, \$t5	\$t7 = 0x06B3284	Add the increment to \$t7
15	and \$t7, \$t7, \$t9	\$t7 = 0x0000284	Mod the \$t7 with modulus by and (modulus-1)
16	sw \$t7, 0(\$t2)		Store the \$t7 to answer[\$t2]
17	addi \$t2, \$t2, 4	\$t2 = 4	\$t2++
18	sub \$t8, \$t8, \$t3	\$t8 = 9	\$t8--
19	j Loop		Jump for another pseudorandom number calc.
20	beq \$t8, \$zero, Done		Jump to Done if \$t8 decrease to 0
21	mult \$t7, \$t4		Multiply \$t7 and \$t4 into \$hi, \$lo
22	add \$t7, \$zero, \$lo	\$t7 = 0x8370874	Extract the \$lo to \$t7
23	add \$t7, \$t7, \$t5	\$t7 = 0x85DA737	Add the increment to \$t7
24	and \$t7, \$t7, \$t9	\$t7 = 0x0000737	Mod the \$t7 with modulus by and (modulus-1)
25	sw \$t7, 0(\$t2)		Store the \$t7 to answer[\$t2]
26	addi \$t2, \$t2, 4	\$t2 = 8	\$t2++
27	sub \$t8, \$t8, \$t3	\$t8 = 8	\$t8--
28	j Loop		Jump for another pseudorandom number calc.

The first pseudorandom number is 644, second one is 1,847.