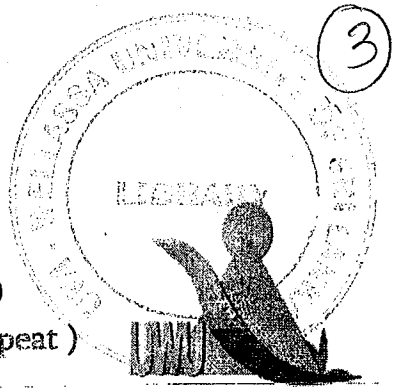

Uva Wellassa University, Sri Lanka
End Semester Examination – January 2010
CST303-2 Computer Systems Architecture (Repeat)

Time: Two (02) hours

Total 04 Questions. Answer all questions.
MIPS instruction formats are given in the Page No. 4.

01.

a.) Consider adding support for the following pseudo-instruction to the MIPS assembler:

```
asum    $rd, $rl, $rr  # place sum of |$rl| and |$rr| in $rd;
                       # the contents of $rl and $rr are not changed
```

This will require that the assembler replace an occurrence of the pseudo-instruction above with a sequence of one or more native MIPS instructions. What instruction(s) might the assembler use for the replacement? You may use any of the temp registers ($tx) as needed.

(10 marks)

b.) Consider the following C language while loop and corresponding MIPS assembly translation.

```
while ( save[i] == k )
        i += 1;

 Loop: sll $t1,$s3,2          #$t1 = 4 *i
       add $t1,$t1,$s6        #$t1=address of save[i]
       lw  $t0,0($t1)               #$t0=save[i]
       bne $t0,$s5, Exit #go to exit if save[i]
                                    #not equals k
add $s3.$s3,1           #i=i+1
       j   Loop
Exit:
```

However, the given translation uses both a conditional branch and an unconditional branch operation each time through the loop. Only poor compilers, and poor MIPS assembly programmers, would write something that inefficient.

Rewrite the assembly code so that it uses at most one branch or jump operation each time through the loop.

(15 marks)

02) Consider the following short MIPS assembly program:

```
                .data
x:              .word 12
y:              .word 17
z:              .word 0
                .text
main:
                lw      $s0, x
                lw      $s1, y
                blt     $s0, $s1, less
                add     $s0, $s0, $s1
                sw      $s0, z
                j       finish
less:           sub     $s1, $s1, $s0
                sw      $s1, z
finish:
                li      $v0, 10     # exits program
                syscall
```

Write an equivalent C/C++ program. "Equivalent" means the program will produce the same results given the same starting values for x, y and z, and do so in the same logical manner.

(25 marks)

03.

a) Explain the hierarchical memory design used in a modern processor.

(5 marks)

b) Explain why a Intel Core Solo processor running at 2GHz is faster than Intel Celeron (single core) Processor running also at 2Ghz. (Assume both processors have the same instruction set and same CPI for each instruction type)

(10 marks)

c) How many total bits are required for a direct mapped cache with 16KB of data and four words blocks, assuming a 32bit address?

(10 marks)

04.

a) Explain the functionality of a download accelerator?

(5 marks)

b) An internet user having CDMA dial-up connection experience an average download speed of 12KBytes/S at particular time of a day. When she tries to speed up the process by using a download accelerator, she finds that there's no apparent change in the download speed. Another user connected to the same ISP, but using an ADSL connection at the same time of the day as the previous user, finds that he can increase the download speed from 20KBytes/S to 60KBytes/S by using the same download accelerator. The ISP has claimed download speeds of 120Kbps and 512Kbps for the CDMA and ADSL connections respectively. Explain the possible causes for above scenario.

(10 marks)

c) A network administrator, on a vacation trip in Yala National Park, gets a call from his Colombo office to his mobile phone asking for help to solve a particular issue. Having trying to solve the problem and failed with a phone conversation he tries to remotely solve the problem by connecting to the Colombo office using his PDA. He successfully connects to internet using the 3G UMTS connection in his PDA. Then he tries to remotely control a server in his office using Remote Desktop feature, but finds that user interface is very sluggish to a point where he can't virtually do anything. So he does a quick speed test to find out the speed of the connection by downloading a file from the same server via ftp and finds out the speed is very good. (He can download a 1MiB file in 40Secs). At last he switched to a telnet connection (command line) and does his task successfully. Explain the possible courses for above scenario.

(10 marks)

Data:

Minimum speed to have a good RDP Remote Desktop session 56Kbps

Minimum speed to have a good telnet session is 12Kbps

Maximum speed of a typical 3G UMTS connection is 384Kbps

Category	Instruction	Example	Meaning	Comments
Arithmetic	add	add $s1,$s2,$s3	$s1 = $s2 + $s3	Three register operands
	subtract	sub $s1,$s2,$s3	$s1 = $s2 - $s3	Three register operands
	add immediate	addi $s1,$s2,100	$s1 = $s2 + 100	Used to add constants
Data transfer	load word	lw $s1,100($s2)	$s1 = Memory[$s2 + 100]	Word from memory to register
	store word	sw $s1,100($s2)	Memory[$s2 + 100] = $s1	Word from register to memory
	load half	lh $s1,100($s2)	$s1 = Memory[$s2 + 100]	Halfword memory to register
	store half	sh $s1,100($s2)	Memory[$s2 + 100] = $s1	Halfword register to memory
	load byte	lb $s1,100($s2)	$s1 = Memory[$s2 + 100]	Byte from memory to register
	store byte	sb $s1,100($s2)	Memory[$s2 + 100] = $s1	Byte from register to memory
	load upper immed.	lui $s1,100	$s1 = 100 × 2^{16}	Loads constant in upper 16 bits
Logical	and	and $s1,$s2,$s3	$s1 = $s2 & $s3	Three reg. operands; bit-by-bit AND
	or	or $s1,$s2,$s3	$s1 = $s2 | $s3	Three reg. operands; bit-by-bit OR
	nor	nor $s1,$s2,$s3	$s1 = ~ ($s2 |$s3)	Three reg. operands; bit-by-bit NOR
	and immediate	andi $s1,$s2,100	$s1 = $s2 & 100	Bit-by-bit AND reg with constant
	or immediate	ori $s1,$s2,100	$s1 = $s2 | 100	Bit-by-bit OR reg with constant
	shift left logical	sll $s1,$s2,10	$s1 = $s2 << 10	Shift left by constant
	shift right logical	srl $s1,$s2,10	$s1 = $s2 >> 10	Shift right by constant
Conditional branch	branch on equal	beq $s1,$s2,25	if ($s1 == $s2) go to PC + 4 + 100	Equal test; PC-relative branch
	branch on not equal	bne $s1,$s2,25	if ($s1 != $s2) go to PC + 4 + 100	Not equal test; PC-relative
	set on less than	slt $s1,$s2,$s3	if ($s2 < $s3) $s1 = 1; else $s1 = 0	Compare less than; for beq, bne
	set less than immediate	slti $s1,$s2,100	if ($s2 < 100) $s1 = 1; else $s1 = 0	Compare less than constant
Unconditional jump	jump	j 2500	go to 10000	Jump to target address
	jump register	jr $ra	go to $ra	For switch, procedure return
	jump and link	jal 2500	$ra = PC + 4; go to 10000	For procedure call