ECE 3056: Architecture, Concurrency and Energy of Computation

Instruction Set Architecture (ISA): Sample Problems and Solutions
1) Consider the following block of SPIM code. The text segment starts at 0x00400000 and the data segment starts at 0x10010000.

.data

label: .word 8,16,32,64
L1: .byte 64, 32
.text
.globl main

main: la $a0, label  #pass array base address
      li $a1, 16   #pass array count

      jal func      #function call

      li $v0, 10   #terminate program
      syscall

func: move $v0, $a0  #get array base address
      move $v1, $a1  # get array count
      add $v1, $v1, $v0    #array ending address
      move $t1, $0    #initialize loop sum

loop: lw $s6, 0($v0)   #load first value
      add $t1, $t1, $s6 #update sum
      addi $v0, $v0, 4  #update pointer
      slt $t0, $v0, $v1 #check for termination
      bne $t0, $0, loop  #onto next element
      move $v0, $t1    #pass sum back to main
      jr $ra

(a) What are the values of main, func, and loop? Clearly state which instructions, if any, have to be translated into more than one native instruction.

i. main   _______________________

ii. func  _______________________

iii. loop _______________________
(b) The second time through the loop, what are the values in registers $ra$, and $t0$.

iv. $ra$ __________

v. $t0$ __________

The answer to i) depends on the answer to part a).

(c) If this is a Big Endian machine what value is returned in register $t0$ for the following instruction.

vi. lw $t0, L1($0) __________

(d) The following is encoded SPIM instructions. Decode the sequence to produce the original MIPS instructions.

0043402a __________

1500ffffc __________

(e) Imagine you have doubled the number of general-purpose registers in the processor but still need to encode instructions in 32-bits. What is the impact on the encoding of the `beq` and `bne` instructions in terms of the range of the target addresses?
2) Consider the execution of the following block of SPIM code. The text segment starts at 0x00400000 and the data segment starts at 0x10010000.

```
.data
start:  .word 0x32, 44, Top, 0x33
str:    .asciiz "Top"
        .align 8
end:    .space 16
.text
        .globl main
main:   li $t3, 4
        la $t0, start
        la $t1, end
Top:    lw $t5, 0($t0)
        sw $t5, 0($t1)
        addi $t0, $t0, 4
        addi $t1, $t1, 4
        addi $t3, $t3, -1
        bne $t3, $zero, Top
exit:   li $v0
        syscall
```

(a) Show the word addresses and corresponding contents of the locations in the data segment that are loaded by the above SPIM code. Clearly associate addresses and corresponding contents.

<table>
<thead>
<tr>
<th>Data Segment Addresses</th>
<th>Data Segment Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) What are the values of the following.

Exit. ________________

Top. ________________

addi $t3, $t3, -1 (encoding). ________________
(c) Provide a set of data directives that will perform the memory allocation implied by the following high level language declarations.

- character My_Char;
  integer Limit;

- integer array[10];

** Note there are many solutions *

(d) Given the following sequences, what is the range of feasible addresses for the label target in the MIPS architecture?

```assembly
start:   beq$1, $2, target
        add$4, $5, $6

...  ...

target: sub$6, $4, $8
```

(e) What is the difference between the j and jal instructions?
3) Consider the following MIPs code taken from some program. We wish to turn this piece of code into a procedure. Note that no procedure call conventions have been followed and the procedure must be re-written to follow the MIPs procedure call conventions.

```
procA:
 lw $12, 0($8)
 sw $12, 0($15) 
 addi $8, $8, 4
 addi $15, $15, 4
 addi $16, $16, -1
 bne $16, $0, procA
 jr $31
```

a) What registers will have to be saved on the stack by the procedure if the MIPS register procedure call convention is to be followed and you do not modify the code shown above?

b) Which registers are used to pass parameters to this procedure? How many registers do you think will be required for parameter passing?

c) If the procedure is stored in memory starting at location 0x00400020, what will be the encoding of the jal procA instruction that is in the calling program?
4) Consider the execution of the following block of SPIM code on a multicycle datapath. The text segment starts at 0x00400000 and that the data segment starts at 0x10010000. Assume immediate instructions take 4 cycles.

```
.data
start: .word 21, 22, 23,24
str: .asciiz "CmpE"
.align 4
.word 24, 0x77
.text
.globl main
main:   li $t3, 4
        lui $t0, 0x1001
        lui $t1, 0x1002
move:  lw $t5, 0($t0)
        sw $t5, 0($t1)
        addiu $t0, $t0, 4
        addiu $t1, $t1, 4
        addi $t3, $t3, -1
end:   bne $t3, $zero, move
done:
```

a) Show the word addresses and corresponding contents of the locations in the data segment that are loaded by the above SPIM code. Clearly associate addresses and corresponding contents.

<table>
<thead>
<tr>
<th>Data Segment Addresses</th>
<th>Data Segment Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
b) Show a sequence of SPIM instructions that will branch to the label loop if the contents of register $t0 is 13.

c) What are the values of end, done, start, and main?

d) What does the following bit pattern represent.
10101101 00101101 00000000 00000000. Provide your answer in the form indicated.

A MIPS instruction (show in symbolic code) ___________________________

5) Answer the following questions with respect to the MIPS program shown below. Assume that the data segment starts at 0x10010000 and the text segment starts at 0x00400000.

```
L1: .data
    .word 0x32, 104
    .asciiz "Test 1"
    .align 2
Blank: .word
L2: .space 64

.text
main:  li $t0, 4
       move $t2, $zero
       li $a0, 1
loop:  jal solo
       addi $t0, $t0, -1
       addi $a0, $a0, 1
       add $t2, $t2, $v0
       slt $t1, $t0, $zero
       bne $t1, $zero, loop
       li $v0, 10
       syscall
```

a) How much space in bytes does the program occupy in the data segment and text segment?

b) With respect to the above program,
   • what are the values of the following?

   L2 __________________________________________

   loop ________________________________________

   memory location 0x1001000c ___________________
• What information would be stored in the symbol table for the above program? Be specific about the above program. Do not provide a generic answer.

c) What is the encoding of the following instructions?

\[
\begin{align*}
\text{bne $t1, $zero, loop} & \quad \text{__________________________} \\
\text{add $t0, $t0, $v0} & \quad \text{__________________________}
\end{align*}
\]

d) Which of the instructions in the above program, if any, must be identified as requiring relocation information?

e) Suppose the procedure solo is independently compiled as a separate module. When it is linked with the main program the first instruction in solo is placed in memory at address 0x0040100c. Provide the hexadecimal encoding of the jal solo instruction.
6) Answer the following questions with respect to the MIPS program shown below. Assume that each instruction can be stored in one word, i.e., do not worry about pseudo instructions! Further, assume the data segment starts at 0x1001000 and that the text segment starts at 0x0400000.

```
.data
.word 24, 0x16
L2: .byte, 77,66,55,44
.text
move $t0,$0
loop: mul $t8, $t0, 4
     add $t1, $a0, $t8
     sw $0, 0($t1)
     addi $t0, $t0, 1
     slt $t7, $t0, $a1
     bne $t7, $0, loop
```

a) What are the values of the labels L2 and Loop?

```
L2 ____________________________
loop _________________________
```

b) Provide the hexadecimal encodings of the following instructions?

```
bne $t7, $0, loop ____________________________
add $t1, $a0, $t8 ____________________________
```

c) Show the contents of the data segment. Show both addresses and values at those addresses.

d) The instruction BLE (Branch on less than or equal too) is not a native instruction. Show the SPIM code that could be used to implement this test.

```
ble $t2, $t3, loop →
```
7) Answer the following questions with respect to the MIPS program shown below. Assume that the data segment starts at 0x10010000 and that the text segment starts at 0x00400000. Assume that the instruction `done` is encoded in one word.

```
.data
label: .word 24, 28
.byte 64, 32
.ascii "Example Program"
.text
main:   jal push
       jal pop
       done

pop:    lw $fp, 0($sp) lw $ra, 4($sp) addiu $sp, $sp, 32
ret1:   jr $ra
push:   subiu $sp, $sp, 32
        sw $fp, 0($sp)
        sw $ra, 4($sp)
ret2:   jr $ra
```

a) Write the values of the words stored at the following memory locations. Provide your answer in hexadecimal notation.

<table>
<thead>
<tr>
<th>Word Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x10010008</td>
<td></td>
</tr>
<tr>
<td>0x0040000c</td>
<td></td>
</tr>
<tr>
<td>0x00400000</td>
<td></td>
</tr>
</tbody>
</table>

b) Consider the process of assembly of the above program.
   - What would the symbol table contain if the main program was independently compiled?
• Assuming that the memory map is fixed, what relocation information would be recorded if any?

c) What are the values of the following labels?

   ret1 _______________________
   push _______________________

d) Assume that the procedures push and pop were assembled in a distinct file and linked with the main program. Further assume that the procedures were placed in memory starting at location 0x00400040. Provide the encoding of the ‘jal pop’ instruction.
8) Answer the following questions with respect to the MIPS program shown below. Assume that the data segment starts at 0x10010000 and the text segment starts at 0x00400000.

```
data
first: .word 0x21, 32
.byte 4, 3
.align 2
str: .ascii "Test"

.text
main: li $4, 4
loop: jal func
      slt $7, $0, $4
      bne $7, $0, loop
      li $v0, 10
end: syscall
func: addi $4, $4, -1
      jr $31
```

a) What function does the last two instructions in the main program realize?

b) With respect to the above program,
   - what are the values of the following symbols?

   end ________________
   loop ________________
   func ________________
   str ________________

   - What is the total number of bytes required for this program, both data segment and text segment storage?

   data segment -
   text segment -
c) What is the encoding of the following instructions?

```
 bne $7, $0, loop
 jal func
 addi $4, $4, -1
```

---

d) Using only native instructions, how can you implement the following test: branch-on-greater-than-or-equal-to-5?
9) Consider the execution of the following block of SPIM code on a multicycle datapath. Assume that the text segment starts at 0x00400000 and that the data segment starts at 0x10010000. Assume that registers $7, $8 and $9 have initial values 16, 0x10010020, and 0x10020020 respectively.

```assembly
.data
str: .asciiz “Start”
.align 2
.word 24, 0x6666
.text
move: lw $12, 0($8)
     sw $12, 0($9)
     addiu $8, $8, 4
     addiu $9, $9, 4
     addi $7, $7, -1
end:  bne $7, $0, move
```

a) Show the word addresses and corresponding contents of the locations in the data segment that are loaded by the above SPIM code. Clearly associate addresses and corresponding contents.

<table>
<thead>
<tr>
<th>Data Segment Addresses</th>
<th>Data Segment Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b) Assume the exception/interrupt handler is stored at location 0x80000020. Provide the SPIM instruction(s) for transferring program control to begin executing instructions at this address. Do not worry about setting the return address.
10) Consider the execution of the following block of SPIM code on a multicycle datapath. Assume that the text segment starts at 0x00400000 and that the data segment starts at 0x10010000. Assume that registers $7, $8 and $9 have initial values 16, 0x10010020, and 0x10020020 respectively.

```assembly
.data
str: .asciiz “Exams”
.align 4
.word 24, 0x6666
.text
move: lw $12, 0($8)
      sw $12, 0($9)
      addiu $8, $8, 4
      addiu $9, $9, 4
      addi $7, $7, -1
end:  bne $7, $0, move
```

a) Show the addresses and corresponding contents of the locations in the data segment that are loaded by the above SPIM code. Clearly associate addresses and corresponding contents.

<table>
<thead>
<tr>
<th>Data Segment Addresses</th>
<th>Data Segment Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

b) What is the difference between signed and unsigned instructions and what is the motivation for making this distinction?
c) What are the values of end and str?

d) What does the following bit pattern represent?
00101001 10111010 00000000 00000000. Provide your answer in the form indicated.

A MIPS instruction (show in symbolic code)  __________________________

11) Answer the following questions with respect to the MIPS program shown below. Note that this simple program does not use the register saving conventions followed in class. Assume that each instruction is a native instruction and can be stored in one word! Further, assume the data segment starts at 0x10001000 and that the text segment starts at 0x00400000.

```
.data
label:  .word 8,16,32,64
       .byte 64, 32
.text
.globl main
main:  la $4, label
       li $5, 16
       jal func
       done

func:  move $2, $4
       move $3, $5
       add $3, $3, $2
       move $9, $0
loop:  lw $22, 0($2)
       add $9, $9, $22
       addi $2, $2, 4
       slt $8, $2, $3
       bne $8, $0, loop
       move $2, $9
       jr $31
```

a) What does the above program do?

b) State the values of the labels loop and main?

   loop ________________________
   main _______________________

c) what are the hexadecimal encodings of the following instructions?

   bne $8, $0, loop ________________________
   add $9, $9, $22 _______________________
d) Identify the local and global labels in the program?
12) What is an unresolved reference? How and when does it become resolved?

13) Write the SPIM instruction sequence that you could use to implement the branch-on-greater-than-or-equal-to-test. For example, how would implement “bge $7, $6, loop”.
14) Consider the following isolated block of code. Note that some initialization code is missing.

```
.text
move: lw $12, 0($8)
    sw $12, 0($9)
    addiu $8, $8, 4
    addiu $9, $9, 4
    addi $7, $7, -1
end:   bne $7, $0, move
```

a) If the text segment starts at 0x04000000, what is the value of the label `end`?

b) Show the hexadecimal encoding of the last instruction.

(c) What does the following bit pattern represent?
00101100 01110000 11000000 00000000

A MIPS instruction (symbolic code!)
15) Consider the following isolated block of SPIM code. Ignore initialization code. The data segment starts at 0x1001000 and the text segment starts at 0x00400000. Note that this block of code does not use registers saving conventions on a procedure call.

<table>
<thead>
<tr>
<th>start:</th>
<th>addi $2, $0, 85</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>addi $3, $0, 1</td>
</tr>
<tr>
<td></td>
<td>jal mystery</td>
</tr>
<tr>
<td>end:</td>
<td>j exit</td>
</tr>
<tr>
<td>mystery:</td>
<td>add $4, $0, $0</td>
</tr>
<tr>
<td>loop:</td>
<td>andi $5, $2, 1</td>
</tr>
<tr>
<td></td>
<td>beq $5, $0, skip</td>
</tr>
<tr>
<td></td>
<td>add $4, $4, $3</td>
</tr>
<tr>
<td>skip:</td>
<td>srl $2, $2, 1</td>
</tr>
<tr>
<td></td>
<td>bne $2, $0, loop</td>
</tr>
<tr>
<td>exit:</td>
<td>jr $31</td>
</tr>
<tr>
<td></td>
<td>li $v0, 10</td>
</tr>
<tr>
<td></td>
<td>syscall</td>
</tr>
</tbody>
</table>

a) What are the values of the following labels?

loop ______________________

skip ______________________

b) What are the encodings of the following instructions?

addi $3, $0, 1 ______________________

beq $5, $2, skip ______________________

c) Let us suppose the procedure mystery is independently compiled and linked when the program is run. After linking the program is stored in memory and mystery has a value of 0x00400010. What is the encoding of the jal mystery instruction.
d) Which of the following are valid MIPS instructions? If they are invalid you must state the correct form or clearly state why it is incorrect.

- lw $t0, $t1($t3)
- sltiu $t1, $t2, 0x44
- bne $t1, label, loop

e) Suppose I want to store the following information in the data segment in the order shown. Show a sequence of SPIM data directives that will correctly do so. The text string, each reserved word and the byte array should be labeled so that programs may reference them easily.

- the text string “Enter a SPIM instruction”
- reserve space for 4 words
- store an array of 16 bytes (pick your own values)
f) In the preceding question (1(f)) what assembler directive would cause the byte array to start on a 64 byte boundary?

g) What does the program in question 1 do? You have to specific and state the function or operation performed.
16)(a) What is the difference between native instructions and pseudoinstructions?
17) Consider the function `func` from the SPIM program shown in question 1).

```
func:
    move $v0, $a0
    move $v1, $a1
    add $v1, $v1, $v0
    move $t1, $0

loop:    lw $s6, 0($v0)
         add $t1, $t1, $s6
         addi $v0, $v0, 4
         slt $t0, $v0, $v1
         bne $t0, $0, loop
    move $v0, $t1
    jr $ra
```

(a) Assume that `func` was independently compiled assuming a starting instruction address of 0x00000000. It is actually loaded into memory starting at location 0x00402000.

i. Which instructions in `func`, if any, have to be corrected? Explain.

ii. What would the symbol table contain after assembly is complete?
18) Consider the following block of SPIM code. The text segment starts at 0x00400000 and the data segment starts at 0x10010000.

.data
first: .word 1,0,1,0,1,0,1, 0
     .word 1,0,1,0,1,0,1, 0
     .word 1,0,1,0,1,0,1, 0
     .word 1,0,1,0,1,0,1, 0
     .word 1,0,1,0,1,0,1, 0
     .word 1,0,1,0,1,0,1, 0
     .word 1,0,1,0,1,0,1, 0
     .word 1,0,1,0,1,0,1, 0
last:   .word 0

.text
.globl main

main:   la $t0, first     #load start address of array
        la $t1, last      #load end address of the array
        addi $t1, $t1, 4 #point to first word after the array
        li $t2, 0        #initialize count using immediate
        add $t3, $zero, $zero #initialize sum using another approach
loop:   lw $t4, 0($t0)    #fetch array element
        add $t3, $t3, $t4 #update sum
        add $t0, $t0, 4 #point to next word
        addi $t2, $t2, 1 #increment count
        bne $t1, $t0, loop #if not done, start next iteration
li $v0, 10
syscall

a) Provide the hexadecimal encodings of the following

lw $t4, 0($t0)

bne $t1, $t0, loop
b. Is the above code relocatable? Explain.

c. The following is the binary representation of a block of assembled SPIM code. Disassemble the program producing the original SPIM instructions. Use the opcode map at the end of this exam.

<table>
<thead>
<tr>
<th>Assembled Binary</th>
<th>MIPS Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x21080004</td>
<td></td>
</tr>
<tr>
<td>0x2129ffff</td>
<td></td>
</tr>
<tr>
<td>0x1520ffffff</td>
<td></td>
</tr>
</tbody>
</table>

d. Consider a jal instruction stored at address 0x00400028. Determine whether each of the following addresses can be a target of this instruction, i.e., the starting location of the procedure. You must justify your answer to receive credit.

i. 0x20020040

ii. 0x04040044

e. What will be contained in the symbol table after assembly?
Consider the following piece of SPIM code. The text segment starts at 0x00400000 and the data segment starts at 0x10010000. Assume all registers and memory locations are initialized to 0x00000000.

```plaintext
.data
   .word 0x21, 22,
 .str:   .asciiz "CmpE"
   .align 3
   .word 24, 0x77

.text
.globl main

main:   li $t3, 4
        lui $t0, 0x1001
        lui $t1, 0x1002
move:   lw $t5, 0($t0)
        sw $t5, 0($t1)
        addi $t0, $t0, 4
        addi $t1, $t1, 4
        addi $t3, $t3, -1
end:    bne $t3, $zero, move
done:
```

(a) How many total bytes of storage are taken up by this program. If you need to make any assumptions about program assembly, state them explicitly.

(b) Show the addresses and corresponding contents of the first four word locations in the data segment that are loaded by the above SPIM code.

<table>
<thead>
<tr>
<th>Word Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
(c) What would the contents of register $t0 (in hexadecimal notation), after the execution of the following instruction, assuming Big Endian storage format?

\[ \text{lw $t0, str($0)} \]

The byte value at the word boundary is the most significant byte of the 32-bit word.

(d) Show the hexadecimal encoding of the following instruction

\[ \begin{align*}
\text{i. bne $3, $zero, move} \\
\text{ii. lw $t5, 0($t0)}
\end{align*} \]

(e) The following is the binary representation of a block of assembled SPIM code. Disassemble the program producing the original SPIM instructions. Use the opcode map at the end of this exam.

<table>
<thead>
<tr>
<th>Assembled Binary</th>
<th>MIPS Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xafbe003c</td>
<td></td>
</tr>
<tr>
<td>0x00082821</td>
<td></td>
</tr>
</tbody>
</table>

(f) Consider a program that references independently compiled procedures. What is the difference between static and dynamic linking? Be specific.