Electrical Engineering and Computer Science 2021.03

Midterm Test
Mon. Feb. 26 2018

Answer all questions in the space provided

Student Last Name: _____

Student Given Name: _____

Student Id. No: _____

Question	Value	Score
1	50	
2	15	
3	30	
4	20	
Total	100	

Question 1. [30 points; Assembly]

1. [3 points] Whose responsibility is to save the temporary registers when a procedure is called?

Caller's

2. [3 points] Where are the extra arguments for a procedure kept if the procedure needs more than eight arguments?

Stack

3. [3 points] Instructions like `bne` have a 12-bit field for the target address of the branch which is too small to fit an address. How does RISC-V (and most other CPUs) solve this problem?

PC-relative addressing

4. [3 points] What is the main use of *load-linked* and *store-conditional*?

building atomic operation primitives (for synchronization)

5. [3 points] Where do we store the return address (`ra`) register for safekeeping when entering a non-leaf procedure?

stack

6. [3 points] RISC-V has no instruction to copy the contents of one register onto another. How do we perform this copy?

```
addi   rd, rs, 0
```

7. [3 points] When we access an element of an array of integers in RISC-V, we always shift left (`sll`) the index by three places, then add to it the pointer to the beginning of the array and then access the element. Why do we need the shift left (`sll`)?

The index to an array refers to double words (8 bytes) and memory is addressed in bytes. `sll` *multiplies by eight.*

8. [9 points] Consider the following assembly code taken from a function

```
slli        t1,a1,2
add         t1,a0,t1
lw          a0,0(t1)
jalr        x0, 0(ra)
```

8.1 [3 points] What is the type of `a0` (char, short, int, long long int, or pointer to char, short, int or long long int)

pointer to 32-bit int

8.2 [3 points] What is the type of the return value

32-bit int

8.3 [3 points] If we call the first argument to this function `A` and the second `B` what operation do the first three lines do (in C or in Java)

A[B]

9. [10 points] Write a function called `srch`. The function takes three arguments: A vector (really the address of the first element) as the first argument, the number of elements in the vector as the second argument and a query as the third argument. Function `srch` searches the vector linearly with a loop and returns the index if the query is found and -1 if it is not. The first element of the vector has index 0.

```
srch:
          addi      x5, x0, 0           ; x5 = 0
loop:     bge       x5, x11, FAIL       ; Not found
          slli      x6, x5, 3           ; mult. by 8
          add       x6, x6, x10         ; V[x5] address
          ld        x7, 0(x6)           ; x7 = V[x5]
          beq       x7, x12, SUCC       ;
          addi      x5,x5,1             ; x5++
          jal       x0, loop

FAIL:     addi      x10, x0, -1         ; Return -1
          jalr      x0, 0(ra)

SUCC:     addi      x10, x5, 0          ; Return index
          jalr      x0, 0(ra)
```

10. [10 points] Translate the following procedure to RISC-V assembly

```
long long int FUN(long long int n)
{
  if (n==0) return 0;
    else return n + FUN(n-1);
}
```

This is a recursive procedure and you need to implement it as a recursive procedure. Pay attention to what is saved on the stack. You do not need to write comments.

```
FUN:      beq       a0, x0, FINISH        ; check base case
          addi      sp, sp, -16           ; save ra and a0 to stack
          sd        a0, 0(sp)
          sd        ra, 8(sp)
          addi      a0, a0, -1
          jal       ra, FUN               ; recursive call
          ld        t1, 0(sp)
          add       a0, a0, t1            ; n + FUN(n-1)
          ld        ra, 8(sp)
          addi      sp, sp, 16
FINISH:   jalr      x0, 0(ra)
```

Question 2.

[15 points, Representation]

1. [3 points] What is the binary representation of -3 (assume a 4 bit word)

```
0011    ; 3
1100    ; complement of 3
  +1
_____
1101    ; -3
```

2. [3 points] Show the result of the following addition (make sure you show both the sum and the carry)

```
 1 1 1 1
+0 0 1 1
_________
```

```
 1 1 1          ; caries
 1 1 1 1
+0 0 1 1
_________
 0 0 1 0        ; the leftmost 1 drops off
```

3. [4 points] What is the binary representation of the hex `0xFACE`.

1111 1010 1100 1110

4. [5 points] Name three advantages of dynamically linked libraries.

1. Library updates take effect immediately
2. No need to have a copy of the library in each executable file
3. Allows sharing

Question 3.
[30 points, Hardware]

1. [10 points] Simplify the following function symbolically:

$$B' + A'B$$

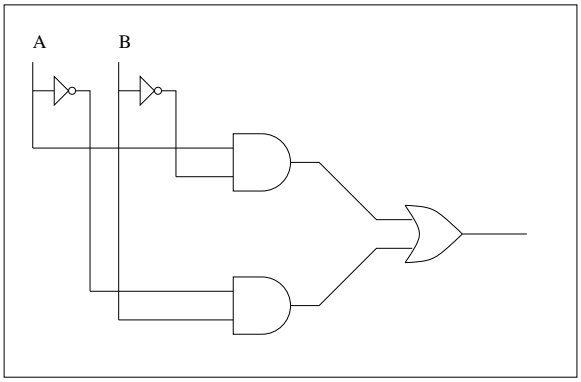
```
B' + A'B =
B'(A+A') + A'B =
AB' + A'B' + A'B + A'B' =
B'(A+A') + A'(B+B') =
B'1 + A'1 =
B' + A'

alternatively

B' + A'B =
(B'+A')(B'+B) =
(B'+A') 1 =
B' + A'
```

2. [10 points] Draw the gate level wiring diagram of the following function

$$AB' + A'B$$



3. [10 points] Write a simple Verilog module with two inputs and one output that implements the function below. You do not need to write comments.

$$AB' + A'B$$

using the `assign` keyword.

```
module xor2(Z, A, B);
   input A, B;
   output Z;

   assign Z = ~A&B | A&~B;
endmodule // xor2
```

Question 4.
[20 points, Performance]

1. [10 points] A processor has a clock cycle of 1 nsec. For a particular set of programs half the instructions executing have a CPI (Clock cycles Per Instruction) of 1 and the rest a CPI of 3. How long does it take to execute a program with 1000 instructions.

$$(\frac{1000}{2} \times 1 + \frac{1000}{2} \times 3) \times 1 =$$
$$500 + 1500 = 2000 nsec$$

2. [10 points] An engineer working for a company is tasked to improve the performance of an application. Upon examination the engineer realizes that half of the execution time is taken by the graphics of the application. A graphics card would speed up these graphics by a factor of 10 and would not affect the rest. If in the original configuration the application took 10 seconds to execute, how long will it take with the new graphics card.

Original execution time: 5sec + 5sec = 10 sec
*New execution time: 5sec + 5*0.1sec = 5.5sec*