CENG3420 Homework 1

Due: Feb. 17, 2019

Solutions

All solutions should be submitted to the blackboard in the format of **PDF/MS Word**.

Q1 (10%) Figure 1 shows a simple multiplication algorithm in ALU design. Write down the step by step procedure to calculate 5×4 or 00000101×0100. Multiplier0 indicates the least significant bit of the multiplier.

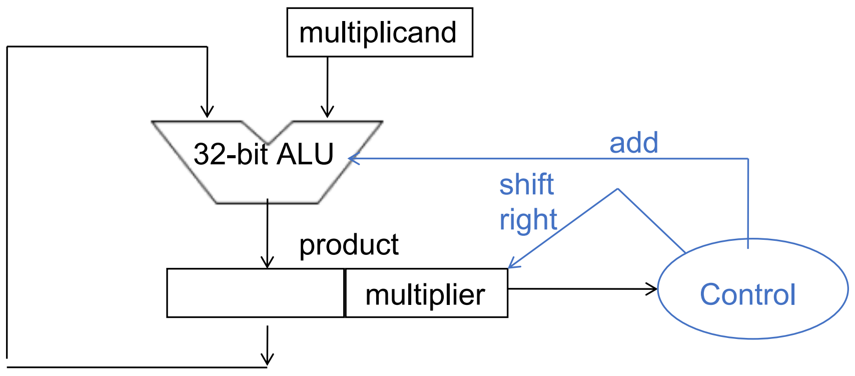


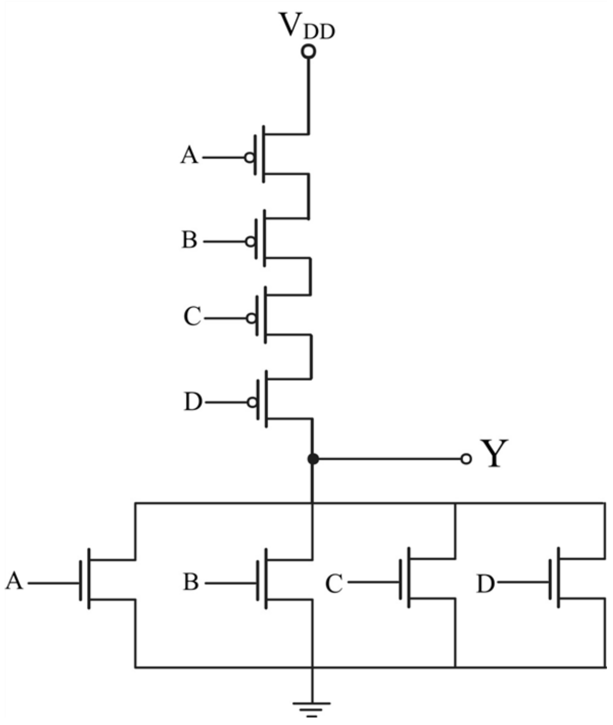
Figure 1: A simple multiplication algorithm.

A1 The operations can be found as follows.

1. 0000 0100 // 0000 0010
2. 0000 0010 // 0000 0001
3. 0101 0001 // 0010 1000
4. 0010 1000 // **0001 0100**

Q2 (10%) Draw the schematic view of four-input NOR gate.

A2 As shown below.



Q3 (15%) Assume `$t0=0xAAAAAAAA, $t1=0x12345678`. Find the value of `$t2` after the following instructions, respectively.

1.
```
sll $t2, $t0, 4
or  $t2, $t2, $t1
```

2.
```
sll  $t2, $t0, 4
andi $t2, $t2, 1
```

3.
```
srl  $t2, $t0, 3
andi $t2, $t2, 0xFFEF
```

A3 As shown below,

1. 0xBABEFEF8
2. 0x00000000
3. 0x00005545

Q4 (15%) Translate the following C function into MIPS assembly (assume that \$a0=n and \$a1=rst). **Please include comments for each instruction in your solution.**

```
int sum(int n, int rst){
    if (n<10)
        return sum(n+1, rst+n);
    else
        return rst;
}
```

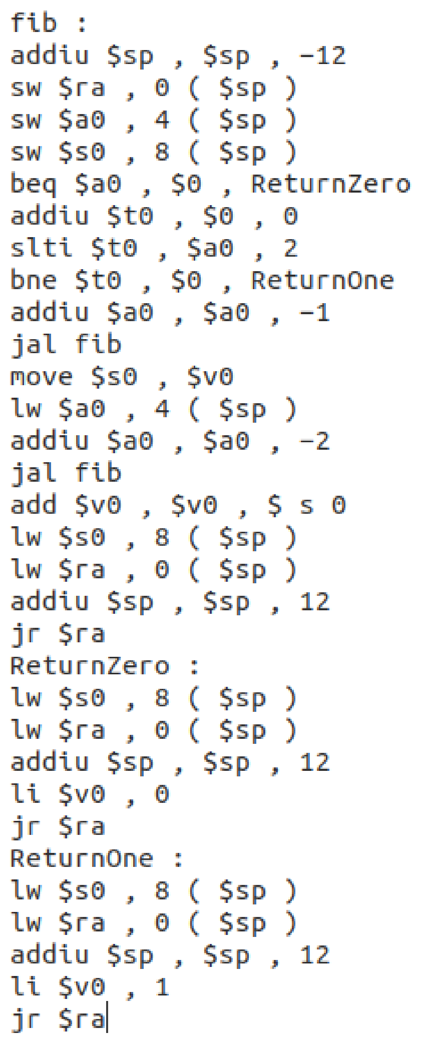
A4 As shown below (assume that \$a0=n and \$a1=rst):

```
sum:
slti $t0, $a0, 10
beq $t0, $zero, sum_exit
add $a1, $a1, $a0
addi $a0, $a0, 1
j sum
sum_exit:
add $v0, $a1, $zero
jr $ra
```

Q5 (15%) Translate the following C implementation of Fibonacci function into MIPS assembly (assume \$a0=n). **Please include comments for each instruction in your solution.**

```
int fib(int n){
    if (n==0){
        return 0;
    } else if (n==1){
        return 1;
    }
    return fib(n-1)+fib(n-2);
}
```

A5 As shown below (assume that \$a0=n):



```
fib :
addiu $sp , $sp , -12
sw $ra , 0 ( $sp )
sw $a0 , 4 ( $sp )
sw $s0 , 8 ( $sp )
beq $a0 , $0 , ReturnZero
addiu $t0 , $0 , 0
slti $t0 , $a0 , 2
bne $t0 , $0 , ReturnOne
addiu $a0 , $a0 , -1
jal fib
move $s0 , $v0
lw $a0 , 4 ( $sp )
addiu $a0 , $a0 , -2
jal fib
add $v0 , $v0 , $ s 0
lw $s0 , 8 ( $sp )
lw $ra , 0 ( $sp )
addiu $sp , $sp , 12
jr $ra
ReturnZero :
lw $s0 , 8 ( $sp )
lw $ra , 0 ( $sp )
addiu $sp , $sp , 12
li $v0 , 0
jr $ra
ReturnOne :
lw $s0 , 8 ( $sp )
lw $ra , 0 ( $sp )
addiu $sp , $sp , 12
li $v0 , 1
jr $ra
```

Q6 (10%) Amdahl's Law defines the speedup of a processor that can be gained by using a specific feature, which is given as follows

$$S_{overall} = \frac{ExecutionTime_{old}}{ExecutionTime_{new}} = \frac{1}{(1 - Fraction_{enhanced}) + \frac{Fraction_{enhanced}}{Speedup_{enhanced}}} \quad (1)$$

Suppose that we want to enhance the processor used for Web serving. The new processor is 10 times faster on computation in the Web serving application than the original processor. Assuming that the original processor is busy with computation 40% of the time and is waiting for I/O 60% of the time, what is the overall speedup gained by incorporating the enhancement?

A6

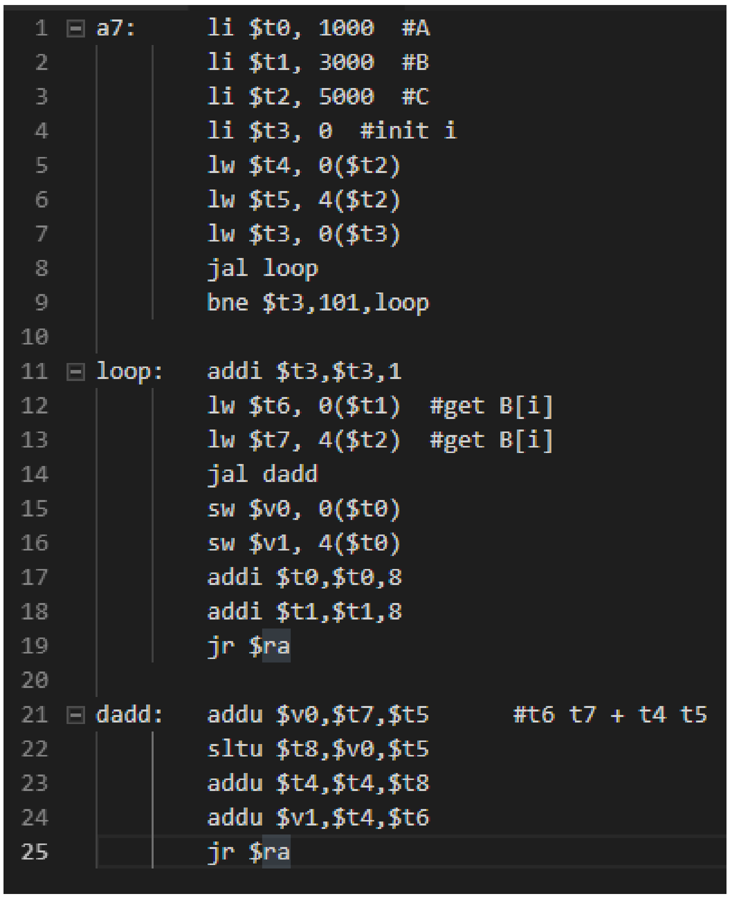
$$F_e = 0.4, S_e = 10, S_o = \frac{1}{0.6 + \frac{0.4}{10}} = 1.56. \quad (2)$$

Q7 (15%) Translate the following C function into MIPS assembly.

```
for (i=0; i<=100; i++)
{A[i]=B[i]+C;}
```

Assume that A and B are arrays of 64-bit integers, and C and i are 64-bit integers. Assume that all data values and their addresses are kept in memory (at addresses 1000, 3000, 5000, and 7000 for A, B, C, and i, respectively) except when they are operated on. Assume that values in registers are lost between iterations of the loop. **Please include comments for each instruction in your solution.**

A7 As shown below, it should be noted that it takes 2 mips32 registers to store a 64bit integer.



```
a7:     li $t0, 1000  #A
        li $t1, 3000  #B
        li $t2, 5000  #C
        li $t3, 0  #init i
        lw $t4, 0($t2)
        lw $t5, 4($t2)
        lw $t3, 0($t3)
        jal loop
        bne $t3,101,loop

loop:   addi $t3,$t3,1
        lw $t6, 0($t1)  #get B[i]
        lw $t7, 4($t2)  #get B[i]
        jal dadd
        sw $v0, 0($t0)
        sw $v1, 4($t0)
        addi $t0,$t0,8
        addi $t1,$t1,8
        jr $ra

dadd:   addu $v0,$t7,$t5      #t6 t7 + t4 t5
        sltu $t8,$v0,$t5
        addu $t4,$t4,$t8
        addu $v1,$t4,$t6
        jr $ra
```

Q8 (10%) A program runs in $10s$ on computer A with 2GHz clock. If we want to design a computer B such that the same program can be finished in $7s$, determine the clock frequency of computer B. Assume it requires only 0.7X clock cycles to execute the program on computer B due to different CPU design.

A8 CPU clock cycle of the program on computer A is,

$$cycle_A = 10s \times 2GHz = 2 \times 10^{10}\text{cycles}. \tag{3}$$

CPU clock cycle of the program on computer B is,

$$cycle_B = 0.7 \times cycle_A = 1.4 \times 10^{10}\text{cycles}. \tag{4}$$

Clock frequency of computer B will be,

$$cycle_B/7s = 2GHz. \tag{5}$$