♢ CSCI 2500 — Computer Organization ♢
Fall 2019 Quiz 2 (October 2, 2019)

Xinhao Luo	luox6@rpi.edu Lab section: 3
Room: Sage 3303 Zone: BLUE Row: 6 Seat: 2	7:00 pm - 7:50 pm

Please silence and put away all laptops, notes, books, phones, electronic devices, etc. This quiz is designed to take 50 minutes; therefore, for 50% extra time, the expected time is 1 hour and 15 minutes and 100% extra time is 1 hour and 40 minutes. Questions will not be answered except when there is a glaring mistake or ambiguity in the statement of a question. Please do your best to interpret and answer each question.

Xinhao Luo luox6@rpi.edu

1. **(15 POINTS)** What is the **exact** terminal output of the C code below, assuming a 64-bit architecture? Note that this code compiles and executes without crashing; further, `#include` directives are not shown. Write your response next to each `printf()`.

```
int main(void)
{
  int q = 10;
  int * p = &q;
  char * s = calloc( q, sizeof( char ) );
  printf( "AA%d%lu-%lu%dBB\n", q, sizeof( int ), sizeof( int * ), *p );
  strcpy( s, "LLMMNNOO" );
  char * t = s + 5;
  *t = 'Z';
  printf( "%s\n", t );
  t -= 2;
  *t = '\0';
  printf( "%s\n", s );
  free( s );
  return EXIT_SUCCESS;
}
```

AA104-810BB
ZOO
LMM

2. **(15 POINTS)** Consider we have a function `filesize(FILE* fp)` that tells us the size of a file in bytes. What might the expected values of `x` and `y` from the code snippet below be if we assume one byte per character in ASCII text files?

```
char* a = "Engineers";
FILE* fp1 = fopen("file1", "w");
FILE* fp2 = fopen("file2", "w");
fprintf(fp1, "%s\n", a);
fwrite(a, 1, strlen(a), fp2);
int x = filesize(fp1);
int y = filesize(fp2);
```

x:

10

y:

9

3. **(20 POINTS)** The code below should have a single parent process read in a two dimensional matrix, formatted similarly to inputs for Homework 1. The parent should then spawn exactly one child for each column of the matrix. The children should only determine the maximum value for one single column and then execute `/bin/echo` with an argument of that maximum value. A child's column should be assigned in the order it was spawned by the parent, and the assignment should be bijective (one-to-one).

   The function listings from the `man` page for each library call are given below. Assume we have all necessary includes. Find and correct all the bugs. Hint: 4 points per bug found up to 20 total points; false positives will be penalized.

```
pid_t getpid(void);
void *malloc(size_t size);
int scanf(const char *format, ...);
pid_t fork(void);
pid_t wait(int *status);
int execl(const char *path, const char *arg, ..., (char  *) NULL);

int main()
{
  int max, stat, i, j;
  int rows = 8;
  int cols = 4;
  int pid = getpid();
  int** matrix = (int**)malloc(rows*sizeof(int));
  for (i = 0; i < rows; ++i)
    matrix[i] = (int*)malloc(rows*sizeof(int));

  for (i = 0; i < rows; ++i)
    for (j = 0; j < cols; ++j)
      scanf("%d", matrix[i][j]);

  for (i = 0; i < cols; ++i)
    if (!fork())
      for (j = 0; j < rows; ++j)
        if (matrix[i][j] > max)
          max = matrix[i][j];

  if (getpid() == pid)
    for (i = 0; i < cols; ++i)
      wait(&stat);
  else
    execl("/bin/echo", "echo", max, NULL);

  return EXIT_SUCCESS;
}
```

① sizeof(int*)

cols * sizeof(int)

② matrix[i][j]

max = 0 ③

④ => matrix[j][i]

⑤ {char output[20];
sprintf(output, "%d", max);

↳ output

4. **(15 POINTS)** Give the values stored in temporary registers `$t0`, `$t1`, `$t2`, and `$t3` after the following block of MIPS code executes. You can give decimal integer values or binary (you only need to show the least significant byte of each word).

```
li    $t0, 3
sll   $t1, $t0, 2
or    $t2, $t1, $t0
and   $t3, $t2, $t0
```

Register	Value
$t0	3
$t1	12
$t2	15
$t3	3

5. **(10 POINTS)**

Consider two processors P1 and P2. P1 has clock rate 3.0GHz and a CPI of 1.5. P2 has a clock rate of 2.5GHz and a CPI of 1.0.

Recall: $CPU\ Time = \frac{Instruction\ Count \times CPI}{Clock\ Rate}$.

Calculate the number of instructions per second that each processor executes.

P_1 : $\frac{3\times10^9}{1.5} = 2\times10^9$ instructions/s.

P_2 : $\frac{2.5\times10^9}{1.0} = 2.5\times10^9$ instructions/s.

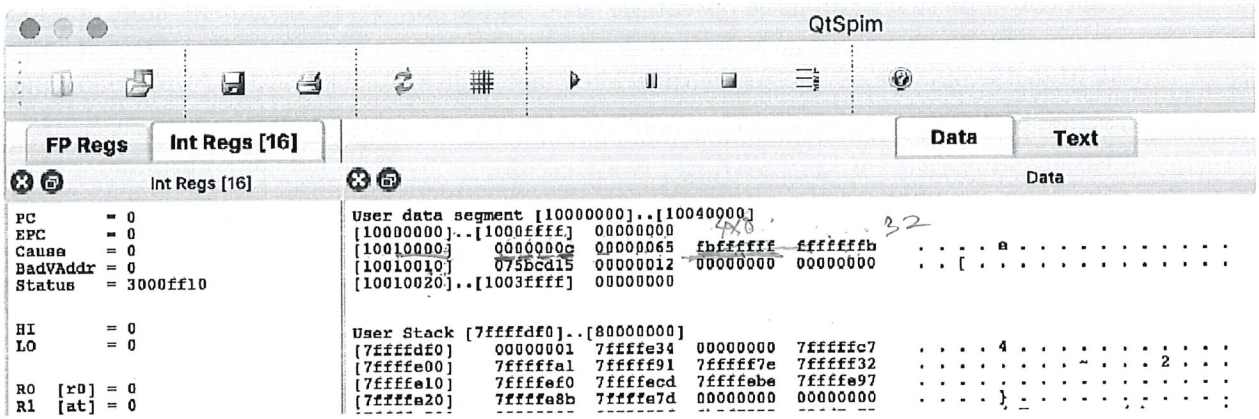
If each processor executes a program in 10 seconds, how many instructions does each processor execute?

P_1 : $2\times10^9 \times 10s = 2\times10^{10}$ inst.

P_2 : $2.5\times10^9 \times 10s = 2.5\times10^{10}$ inst.

Xinhao Luo luox6@rpi.edu

6. **(15 POINTS)** Consider the following memory dump of the data segment in QtSpim:



Part a: (2/15 points) Assuming that the data segment starts at address 0x10000000 and that this architecture is little-endian, give the hexadecimal numeric value of an integer from the 65539^{th} word of the data segment (the word at 0x10000000 would be counted as the first word). Recall that $0x10000 = 2^{16} = 65536$.

fffffffb

Part b: (4/15 points) Now, if this integer is unsigned, what would its decimal value be? Recall that the maximum 32-bit unsigned integer value is $4,294,967,295_{10}$.

4,294,967,291

Part c: (4/15 points) If this integer is signed and uses 2's complement notation, what would its decimal value be?

-5

Part d: (5/15 points) Finally, you need to change the value of this integer to represent -3 using 2's complement notation. Show the contents of the data segment after updating the value. For full credit, fill in both the Memory Address and the Value columns for all 4 rows.

Memory address	Value (byte)
10010008	ff
10010009	ff
1001000a	ff
1001000b	fd

7. **(10 POINTS)** Your MIPS program takes 60 seconds to execute. You analyze your code and conclude that arithmetic operations account for 50% of the execution time, and load and store operations account for 40% of the execution time. There is a potential of improving arithmetic operations execution time by a factor of 5, and load and store operations by a factor of 6.

Recall: $T_{improved} = \frac{T_{affected}}{improvement factor} + T_{unaffected}$

Part a: (5/10 points) Assuming you can only improve either arithmetic or load and store operations, given the improvement factors specified above, which would yield the best improvement of the total execution time of your program?

(a) arithmetic
(b) load and store
(c) improvement would be the same
(d) no improvement either way

Part b: (5/10 points) Now assume that you can improve both arithmetic and load and store operations, given the improvement factors specified above. In the box below, indicate the total execution time of your program after improvements.

$T_{total} = \frac{60 \times 50\%}{5} + \frac{60 \times 40\%}{6} + 60 \times 10\% = 16\ s$