

Homework 1

Due: ~~Sept 20th~~22nd, 2025 @11:55pm on Gradescope

Name: _____ Uniquname: _____

1. Submit a pdf of your typed or handwritten homework on Gradescope.
2. Your answers should be neat, clearly marked, and concise. Typed work is recommended, but not required unless otherwise stated. Show all your work **where requested**, and state any special or non-obvious assumptions you make.
3. You may discuss your solution methods with other students, but the solutions you submit must be your own.
4. **Late Homework Policy:** Submissions turned in by 1:00am the following day will be accepted but with a 5% penalty. Assignments turned in between 1:00am and 11:55pm (about 24 hours after the due date) will get a 30% penalty, and any submissions made after this time will not be accepted.
5. When submitting your answers to Gradescope you need to indicate what page(s) each problem is on to receive credit. The grader may choose not to grade the homework if answer locations are not correctly indicated.
6. After each question (or in some cases question part), we've indicated which lecture number we expect to cover the relevant material. So "(L7)" indicates that we expect to cover the material in lecture 7.

Problem 1 (15 points): Transistor Performance (L1)

You must **type** the answer to these questions. Read about Moore's Law and Dennard Scaling on Wikipedia (or some other source if you wish) then answer the following questions.

1. Define Moore's Law in your own words. Your answer should be between 15 and 50 words. **[3]**
2. Define Dennard Scaling in your own words. Your answer should be between 30 and 60 words. **[5]**
3. Explain how "Moore's law" and "Dennard scaling" have historically impacted the advancement of computers. Explain how these things are now changing. Your answer should be between 100-200 words. **[7]**

Problem 2 (13 points): Numbers and stuff (L2)

1. Convert the following numbers to 8-bit binary two's complement representation. [5]

a. -14 (decimal)

b. 110 (decimal)

2. Convert the following 12-bit binary 2's complement numbers to decimal. [5]

a. 1011 0011 1101

b. 0101 0101 1011

3. Convert the following 16-bit binary numbers to hex. [3]

a. 1011 1011 0111 1110

b. 1101 1010 1110 1000

Problem 3 (14 points): Multiple choice and the like

1. Executing which of the following lines of LC2K assembly sets every bit of register 4 to 1, assuming register 0 is a zero? Register 1 has an initial value of 32, and memory address 0 has a value of 0x7. [5] (L3)

- A. nor 0 0 4
- B. nor 0 4 1
- C. lw 0 4 0
- D. .fill 4
- E. add 4 0 0

_____ <----- Answer(s) go here

2. If each activation record for “recurrence” (stack frame for the function) requires 80 bytes of storage, and the maximum stack size possible on a machine is 1000 bytes, identify *all* of the following recursive function calls that will be able to execute on the machine. [5] (L6)

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

- A. recurrence(2)
- B. recurrence(9)
- C. recurrence(13)
- D. recurrence(24)
- E. recurrence(38)

_____ <----- Answer(s) go here

3. In the blank provided write the letter which best defines the associated term [4] (L7)

2's complement _____

Hexadecimal _____

IEEE 754 _____

Double precision _____

- A) enables binary numbers to be represented more compactly
- B) a format for representing unsigned numbers
- C) a well-known standard for representing real numbers
- D) a signed integer representation supporting efficient hardware
- E) represents numbers using 64 bits
- F) a special format for representing negative numbers

Problem 4 (15 Points): Moving data between memory and registers (L5)

For the following LEGv8 assembly code, indicate the final state of the registers and memory locations listed once this code has finished. Assume any memory address outside of that displayed in the initial memory state contains 0.

```
MOVZ      X0, #0x1004
LDURH     X2, [X4, #2]
STURB     X3, [X0, #-1]
LDURSW    X5, [X4, #0]
STURH     X4, [X0, #-2]
```

Initial Register State		Initial Memory State	
Register	Value	Address	Value
X0	0xD	0x00001000	0x44
X1	0x1001	0x00001001	0x12
X2	0x6E	0x00001002	0x44
X3	0x3CB1	0x00001003	0x2E
X4	0x1000	0x00001004	0xA1
X5	0x5	0x00001005	0x72

Fill in the following table. Express your final answers in hexadecimal. Assume memory is organized in a **little endian** manner.

Final Register State		Final Memory State	
Register	Value	Address	Value
X0		0x00001000	
X1		0x00001001	
X2		0x00001002	
X3		0x00001003	
X4		0x00001004	
X5		0x00001005	

Problem 5 (12 points): Arrays (L5)

Convert the following C code to *4 lines* of LC2K assembly. Use r5 to hold the value of Gold[3], r6 to hold the value of the NOR of ~~b~~ a and Gold[3], and r7 to hold any results needed to compute the address of Go[b].

```
Go[b] = ~(a | Gold[3]);
```

All variables and arrays are initialized as shown. You may not modify other registers.

Variable to register mappings

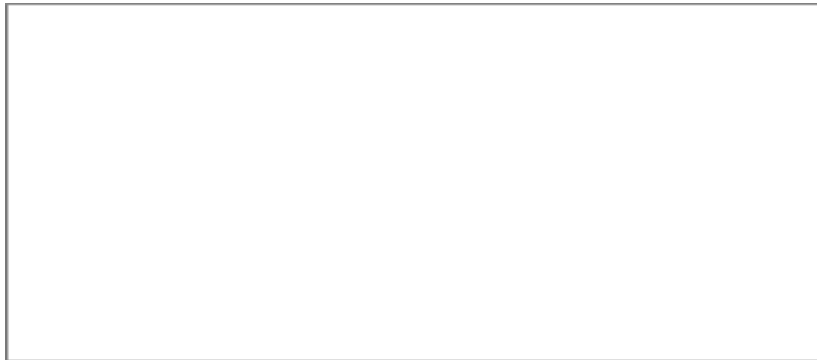
Starting addresses

int a : r1

int b : r2

Go : r3

Gold : r4



Problem 6 (15 points): Reading LC2K code (L6)

A now ex-coworker of yours wrote a function Find() and left a few comments, but very limited documentation. You know that the function takes two arguments. The first is a pointer to an array (passed in r1) and the second is the number of elements in the array (passed in r2). The return value is found in r5.

Answer the questions below. You may make use of the LC2K simulator (<https://eecs370.github.io/simulators/lc2k/>) on the EECS 370 website if you wish. However, remember that you should develop the ability to decipher what LC2K code is doing without the simulator, since you will not have access to the simulator on exams.

```
lw      0      1      ArrayS
lw      0      2      Num
lw      0      3      Fcall
jalr    3      7
halt

Find    lw      0      6      NegOne    // r6=-1
        add     0      2      5          // r5=Num
Top     add     2      6      2          // Decrement Num
        add     1      2      4          // 4 is address
        lw      4      3      0          // ld array element
        beq     3      0      skip       // is element 0?
        add     5      6      5          // if not sub 1
skip    beq     0      2      Done       // if Num=0 we are done
        beq     0      0      Top        // next iteration
Done    jalr    7      3              // return
Zero    .fill   0
NegOne  .fill   -1
Fcall   .fill   Find
ArrayS  .fill   ArrayS
Num     .fill   8
Array   .fill   0
        .fill   1
        .fill   2
        .fill   2121
        .fill   0
        .fill   787
        .fill   0
        .fill   1
        .fill   -5
```

- a) How many times will the line with the label “Done” get executed during this program? [3]

- b) What is the value of r3 when the program halts? **[3]** _____
- c) What is the value in r5 when the program halts? **[3]** _____
- d) *In 10 words or less*, describe what the return value of the function is in terms of the array.
Your answer should be something like “returns the maximum value in the array”. **[6]**

Problem 7 (16 points): Floating point--mind the gap (L7)

With integers, the “gap” between representations is always 1. This means that rounding a decimal to an integer can introduce an error of up to ± 0.5 .

- 1) In a 32-bit IEEE floating point number, what is the smallest number greater than 24 that can be exactly represented? What is the gap between that number and 24? Briefly justify your answers. **[5]**

- 2) Assuming we are always rounding to the nearest value we can exactly represent (rounding down in the case of ties), what is the largest value we can add to 24 and get a result that is still 24? **[3]**

- 3) Redo part 1) but for -12.625 rather than 24? **[5]**

- 4) Briefly explain how the gap between floating point numbers might cause a programmer difficulties. In particular, consider a for loop where small values are being added to a larger number many thousands of times. **[3]**