NAME:_____*

CS 355 - Computer Organization and Architecture II

Practice Questions for Final

No answers will be provided. (I will comment on your solution of being correct or not and how close you are, but I will not give hints - I want you to learn to think, having me telling you want the solution is will compromise my objective).

These questions are focussing on the latter half of the material.

Question 1.

In the multiprogramming operating system UNIX, programs are assigned priorities. A program with higher priority value will get the CPU for a longer time when it is its turn to be executed. This can be implemented by having two counters inside the PCB:

- PriorityCount
- RemainderCount

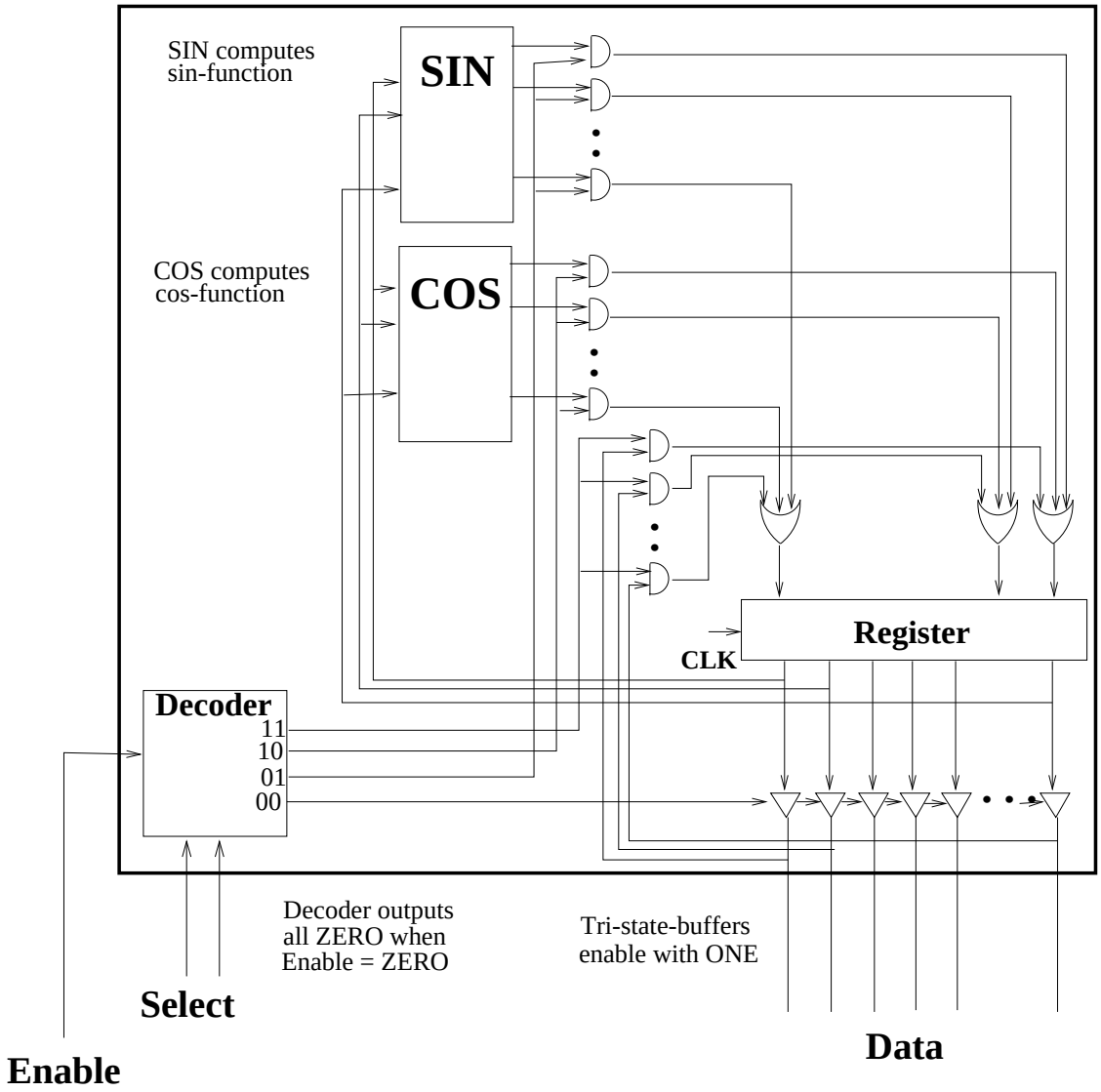
The PriorityCount counter is initialized by UNIX when the program is run: every UNIX user is assigned a priority by the UNIX system and when a user runs a program, the UNIX system will create a new PCB for the program and initialize the PriorityCount with the user's priority.

The RemainderCount is initialzed with the value PriorityCount when the program receives a turn to run. At each timer interrupt, the RemainderCount is decremented by 1 and when the RemainderCount becomes 0, the next program in the ReadyQueue is run.

Write the timer interrupt handler that implements the UNIX priority scheme. Use "Enqueue(ReadyQueue)" and "Dequeue(ReadyQueue)" to indicate the enqueue and dequeue operation of the ReadyQueue.

Question 2.

Many commercial CPU's have an accompanying math co-processor. A math co-processor is a specialized chip that can do mathematical operations and floating point operations at a much higher speed than the general purpose CPUs (such as SPARC or M68000).



The above figure shows the circuitry of a math co-processor. The math. co-processor has a decoder that can be in 5 states:

- if **Enable = ZERO**, then all outputs are ZERO.
- if **Enable = ONE**, then exactly one of the outputs is ONE and all others are ZERO. The output that becomes ONE depends on the **Select** inputs and the labeling of the outputs in the above figure. shows output that turns ONE for a given set of **Select** inputs.

The co-processor has two special (combinatorial) circuits built in named **SIN** and **COS**. These circuits compute the *sin* and *cos* functions of the input number, respectively. The

co-processor also has a 32 bit working register made of D-flipflops. The signal **CLK** is the clock signal and is tied to the clock input of all D-flipflops of the register.

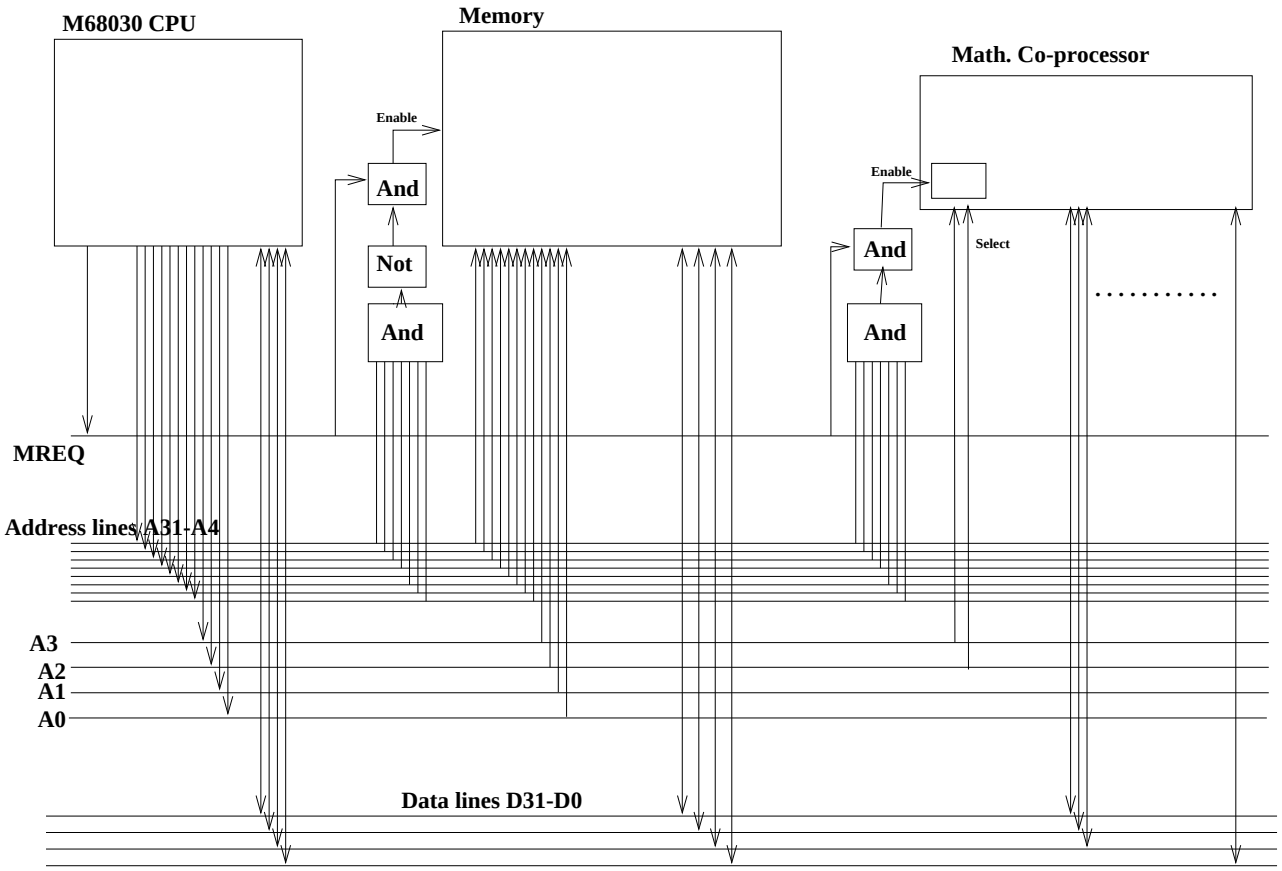
When the math. processor is **enabled**:

- What function will **Select** = 00 perform ?
- What function will **Select** = 01 perform ?
- What function will **Select** = 10 perform ?
- What function will **Select** = 11 perform ?

In the figure above, the clock signal of the math. processor is not yet connected. Note that the use of the clock signal is to update the register in the math. processor.

- Using the answer to the above 4 questions as hints, complete the connection for the clock signal that will enable the math. processor to update its register correctly. (6 pts)

The M68030 is a successor CPU of M680000 and can execute all M68000 instructions. The main difference is that the M68030 is a 32 bit processes (the number of lines on databus is 32). The following figure shows the way that the math. co-processor is connected on the system bus along with a M68030 CPU:



The memory will only enables if its **Enable** signal is equal to ONE.

Question:

1. Write a segment of M68000 assembler code that computes:

$$\texttt{Z} = \texttt{sin}(\texttt{X}) + \texttt{cos}(\texttt{Y});$$

Note: You may use binary numbers in your answer for convenience.
Note: You may assumed the following variables have already been defined:

```
X: ds.l 1              ; X is a 32 bit variable
Y: ds.l 1              ; Y is also a 32 bit variable
Z: ds.l 1              ; Z is also a 32 bit variable
```

Question 3. (The Belady's Anomaly)

A program is placed in a vritual memory system for execution. The summary of the sequence of pages used by the program is:

$$1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5$$

Initially, the entire memory is empty.

- Suppose the computer has 3 page frames and uses the FIFO replacement policy.

  Show the sequence of pages that are in the memory each time a new page is fetched from the disk.

  How many pages total have been yanked in ?

- Suppose we increase the amount of memory to 4 page frames.

  Repeat the same question (for a computer that has 4 page frames).

- We more memory pages available, one would expect to have less page faults....

  Does this example conform to the expected result ?

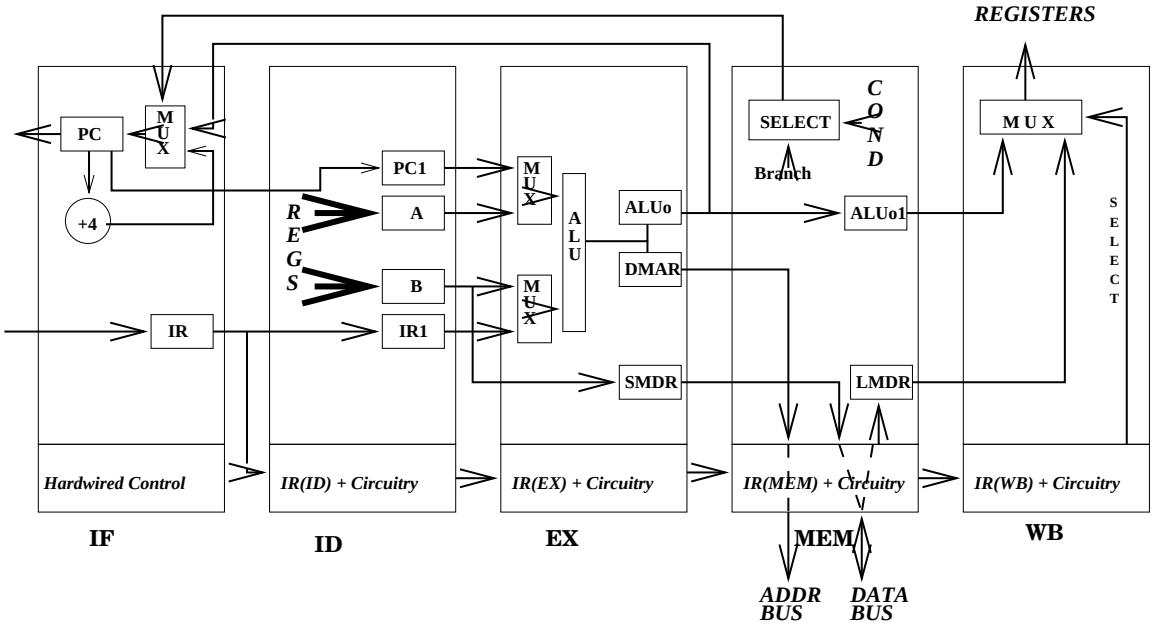
Question 4.

- Repeat Question 3 with the LRU replacement policy. Is the number of page fault with 4 memory pages less than 3 memory pages ?

- Repeat Question 3 with the second chance replacement policy. Is the number of page fault with 4 memory pages less than 3 memory pages ?

Question 5.

Consider the **basic** pipeline architecture discussed in class:



The following program is executed using the **basic** pipeline architecture:

```
LABEL0: mov 0, %r0              (* Store 0 to reg 0 *)
        ba  LABEL1              (* Branch (always) to label LABEL1 *)
LABEL5: ba  LABEL2
        ba  LABEL3
        ba  LABEL4
LABEL1: add %r0, 1, %r0         (* add 1 to reg r0 *)
        ba  LABEL5
LABEL2: add %r0, 2, %r0
        add %r0, 3, %r0
LABEL3: add %r0, 4, %r0
        add %r0, 5, %r0
        ba  LABEL5
LABEL4: add %r0, 6, %r0
        add %r0, 7, %r0
        add %r0, 8, %r0
        add %r0, 9, %r0
        ba  LABEL0
        add %r0, 10, %r0
        add %r0, 11, %r0
        add %r0, 12, %r0
```

Question:

- Starting at the instruction at the label `LABEL0`, give the **first 20** instructions that will be fetched by the IF stage.

Question 6

The following figure shows the partial content of a cache memory:

Row	Valid Bit	Block Number	Value (32 bits)
0	1	2	5
1	1	1	7
2	1	0	1
3	0	1	1
	⋮	⋮	⋮

The values are in decimal for your convenience. The cache memory has 64 rows and the value is a row is 32 bits. The computer address bus has 32 bits.

Part 1:

Assuming the cache memory is an **associative** cache, answer the following questions:

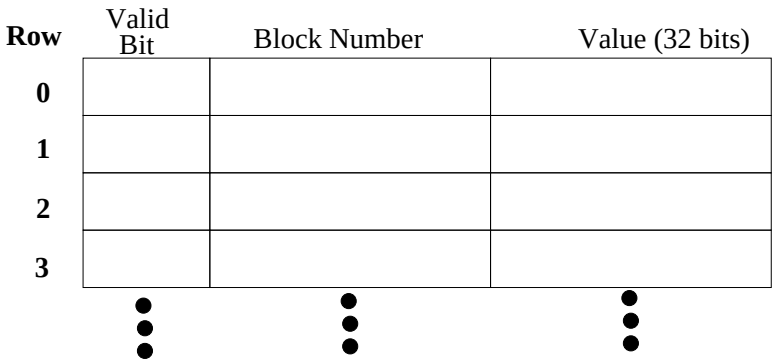
- How many bits do you need for the block number ? (3 pts)
- Using only information from the figure above, give memory locations that are being cached and the value at the memory location in the following table: (3 pts)

Row	Valid Bit	Block Number	Value (32 bits)
0	1	2	5
1	1	1	7
2	1	0	1
3	0	1	1

Memory Location	Value

- The CPU now reads the word at memory location 2056 which value is not currently cached[1]. Memory location 2056 contains 44. Give the new content of the cache when the value has been retrieved: (4 pts)

[1]For your convenience, 2056 ($= 2^{11} + 2^2$) decimal is equal to 00000000 00000000 00001000 00001000 binary.



Part 2:

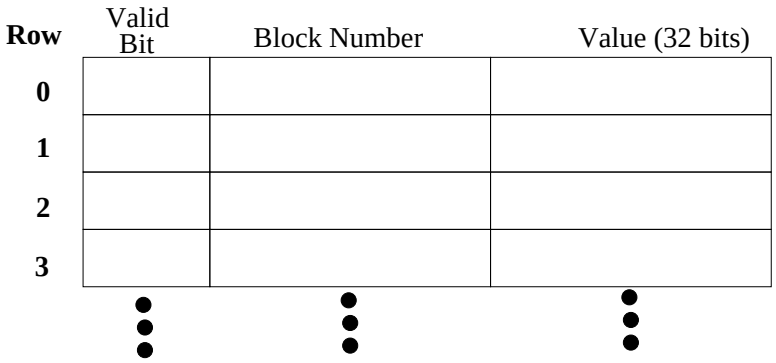
Assuming the cache memory is a **direct mapped** cache, answer the following questions:

- How many bits do you need for the block number ? (3 pts)
- Using only information from the figure above, give memory locations that are being cached and the value at the memory location in the following table: (3 pts)

Row	Valid Bit	Block Number	Value (32 bits)
0	1	2	5
1	1	1	7
2	1	0	1
3	0	1	1

Memory Location	Value

- The CPU now reads the word at memory location 2056 which value is not currently cached. Memory location 2056 contains 44. Give the new content of the cache when the value has been retrieved: (4 pts)



Question 7.

Write a parallel merge sort algorithm in CUDA.

Question 8.

In houses that have more than one floors, there are usually pairs of switches that can be used to turn the same light on and off. One switch on a floor will be able to switch the light on or off, regardless of the setting of the other switch. Design a (combinatorial) circuit with 2 inputs (switches) and one output (light) so that the light can be turned on and off by any one switch, regardless of the setting of the other switch.