**CSC390 (Computer Org. & Arch.)**

**Final Exam- Spring 2018**



**Name:**

**Student ID:**

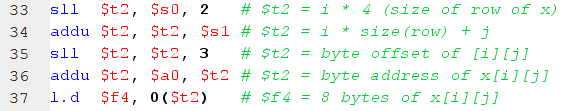
(

There are five questions and one bonus point question in the Exam.

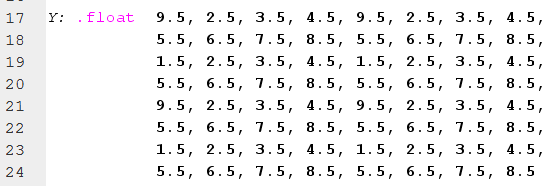
**Q1.** **(a)** What decimal number is represented by the following IEEE754 single precision floating point number? [8 pts]

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**(b)** The following lines of codes which calculate the address (index) of a 4x4 matrix X[i][j], where each element is considered as a double precision floating point number (64 bits, i.e. 8 bytes). The code also loads the indexed element in the register pair $f4. Note that $s0 and $s1 represent the i and j indexes, respectively. Register $a0 contains the base address of X. [8 pts]



What changes would you make in the code so that it can calculate the address (index) of the following 8x8 matrix Y[i][j], where each element in the matrix is a single precision floating point number (32 bits, i.e. 4 bytes). With the changed code and the initial values of $s0 = 3 and $s1 = 7, indicate which value of the matrix will be transferred to the register $f4. Show the details of you calculation.

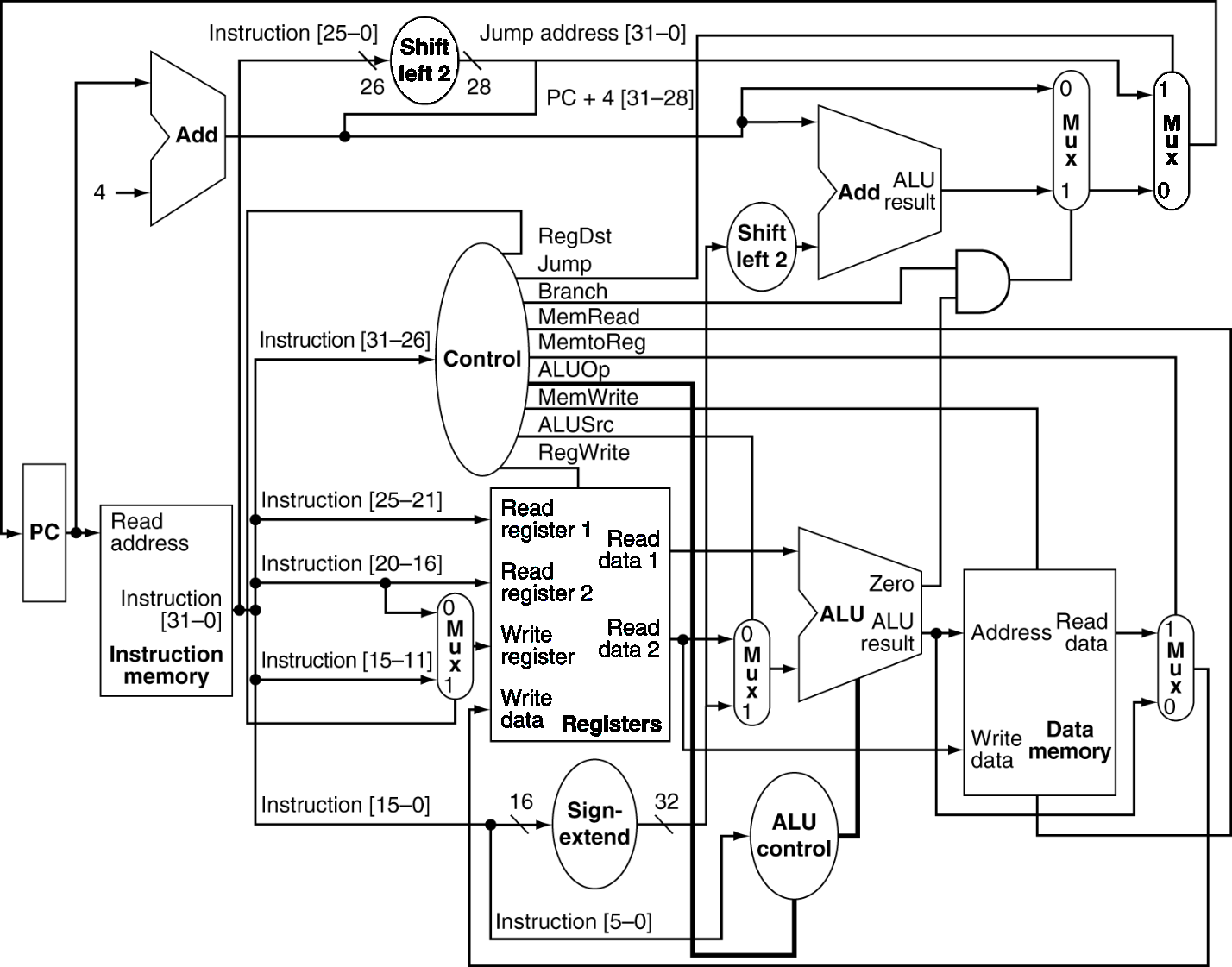


**Q2.** Consider the following R-type instruction, **or $s0, $a1, $t7**, and its corresponding machine code:

000000 00101 01111 10000 00000 100101

The machine code consists of mainly Opcode, Source registers, destination registers, and Function code etc.

Now suppose the instruction is executed in the MIPS processor as shown in the following figure. Highlight the different **registers,** **control signals** and **multiplexer inputs** that will be activated during the execution process of the above **or** instruction. Also **indicate the buses** where different portions of the machine code and the result of ($a1 or $t7) can be found. **[12 pts]**

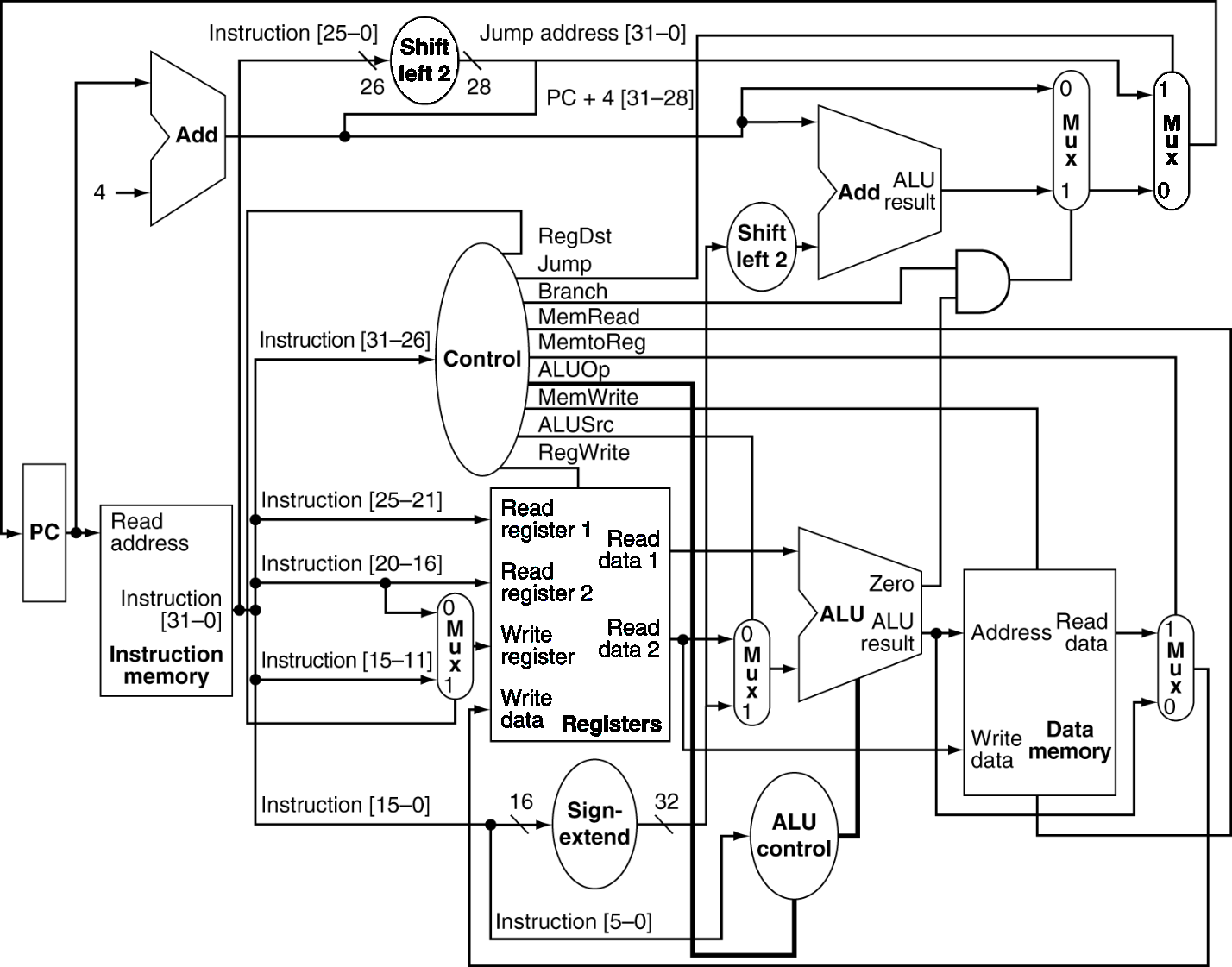


Q3. Consider the following load-type instruction, **sw $t2, 32($s0)**, and its corresponding machine code:

101011 01010 10000 0000000000100000

This machine code consists of Opcode, Source registers (rs), destination registers (rt), and 16 bits offset.

Now suppose the instruction is executed in the MIPS processor as shown in the following figure. Highlight the different **registers, control signals, and multiplexer inputs** that will be activated during the execution process of the above **sw** instruction. Also, **indicate the buses** where different portions of the machine code and the memory address **32($s0)** can be found. Clearly show the path how the content of the address **32($s0)** is updated by above store instruction.  **[12 pts]**



**Q3.** Pipelining is an implementation technique in which multiple instructions are overlapped in execution to improve the performance by increasing the instruction throughput of the system. However, there are situations in pipelining when the next instruction cannot execute in the following clock cycle. These events are called hazards. There are three different types of hazards, such as structural, data, and control hazards. Consider the following lines of codes and determine where you could have possible hazards. Also indicate the types of Hazard. **[4+12+8] pts**

**or $t2, $s1, $s2**

**and $s4, $t2, $s3**

**sub $s7, $s6, $t2**

**lw $t2, 100($t4)**

**add $s4, $t2, $s3**

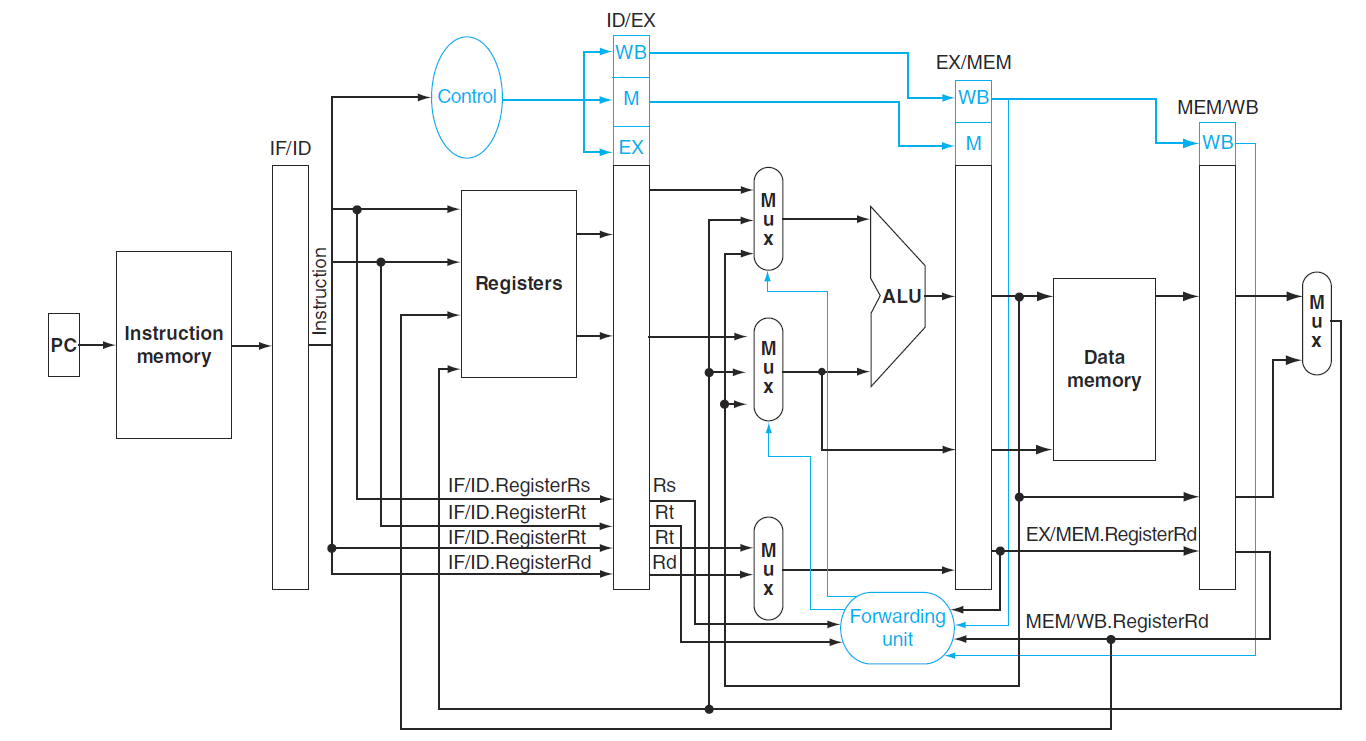
**beq $t2, $s8, exit**

**add $t7, $t2, $t5**

**exit: sw $s9, 100($t2)**

To support the pipelining operation, MIPS processors use **pipeline-register,** in between the pipeline stages.There are four pipeline registers (IF/ID, ID/EX, EX/MEM and MEM/WB registers) in the MIPS architecture (as shown in the following figure). Pipeline registers temporarily hold the control signals and register numbers to facilitate the data forwarding or bypassing, an efficient way of overcoming the data hazards.

1. Write a short notes on how you can detect the above (EX and MEM) data hazards by comparing the pipeline registers (such as EX/MEM.RegisterRd, ID/EX.RegisterRs and ID/EX.RegisterRt registers) and using the pipeline-register control signals (such as EX/MEM.RegWrite, MEM/WB.RegWrite). After detecting the (EX and MEM) data hazards what action would take to overcome the hazard? Explain it in terms of the forwarding unit. Clearly specify the ForwardA and ForwardB in the diagram.
2. Indicate where only forwarding will not be sufficient, you also need to insert stall to overcome the hazards. How do you decide when you need to stall the system? Explain it in terms of the pipeline registers and control signal.



**Q4.** One way to improve branch performance is to reduce the cost of the taken branch. Thus far, we have assumed the next PC for a branch is selected in the MEM stage, but if we move the branch execution earlier in the pipeline, then fewer instructions need be flushed and thereby pipelining would become more efficient. The designers observed that many branches rely only on simple tests (equality or sign, for example) and that such tests do not require a full ALU operation but can be done with at most a few gates. Figure 4 shows the datapath and control signals of such a MIPS processor.

**Show** what happens when the branch is taken in the following instruction sequence, assuming the pipeline is optimized for branches that are not taken and the branch execution is moved to the ID stage:

10 sub $10, $4, $6

14 beq $1, $7, 6

18 and $13, $2, $6

22 or $12, $2, $5

24 add $14, $4, $2

………..

38 lw $4, 50($7)

42 sw $5, 54($8)

Use **figure 4** (a & b) to show what happen when a branch is taken. Specifically, **indicate the input/output values of the PC, adders, pipeline register data, and control signals.** **[16 pts]**

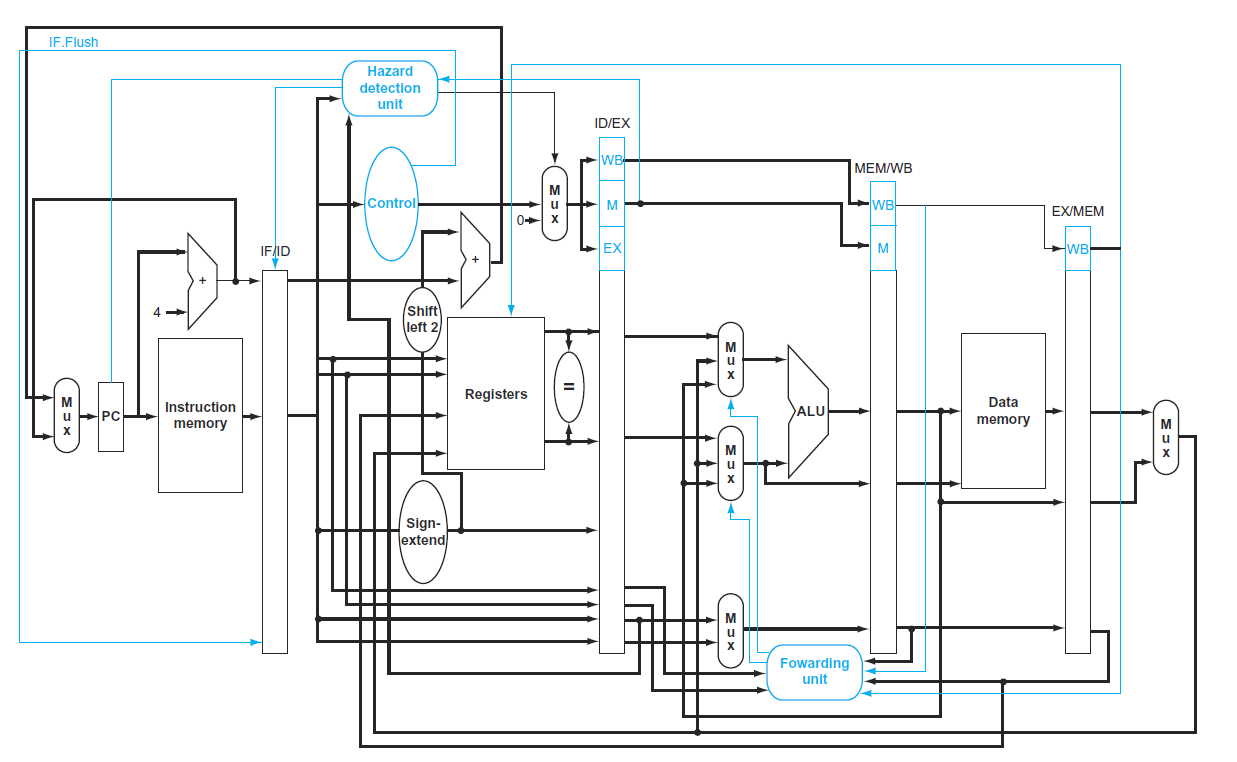


Figure 4 (a)

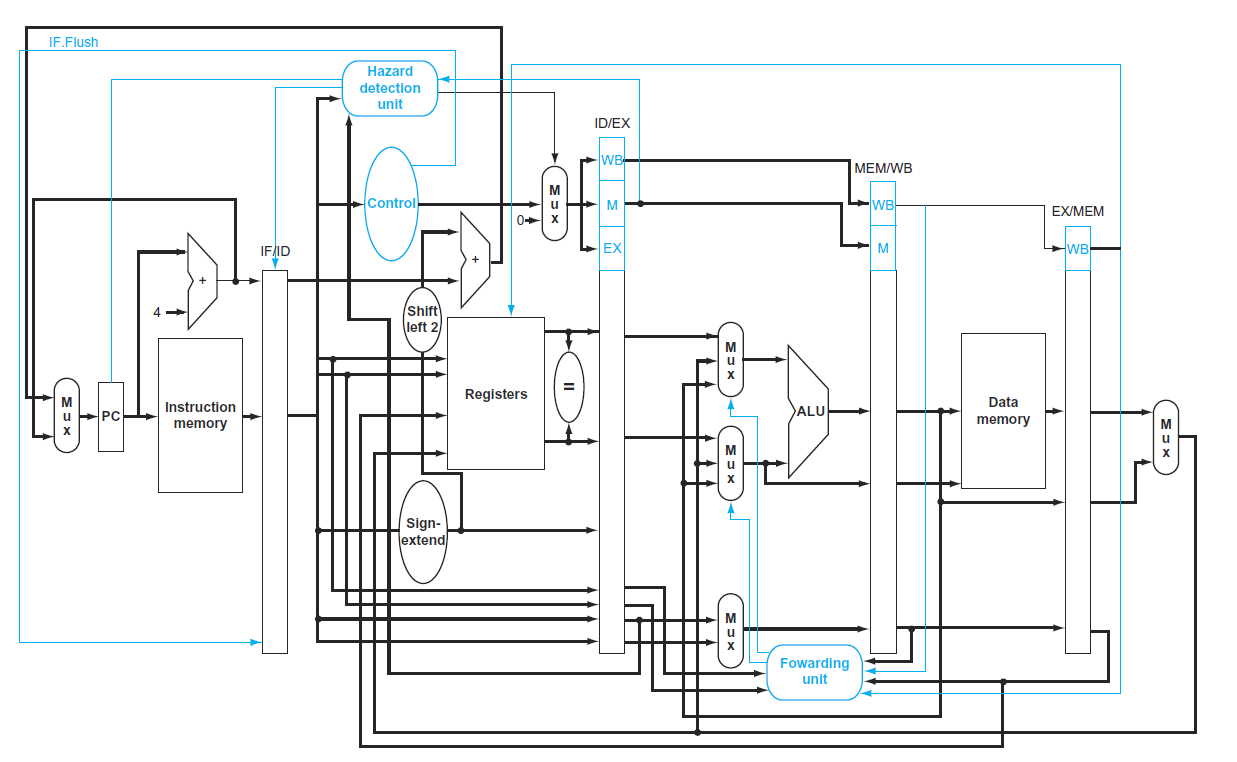


Figure 4(b)

**Q5.** **(a) (4+4 pts)**

(i) What are the tracks and sectors in a magnetic hard disk? Briefly discuss, **how data is accessed** in a hard disk.

(ii) Given, 4096B sector, 72,000rpm, 2ms average seek time, 200MB/s transfer rate, 0.1ms controller overhead, idle disk. Find the average read time.

**Q5. (b) (4+8 pts)**

(i) What is a cache memory? Briefly explain, **how a cache memory operates** in the memory hierarchy?

(ii) Consider a direct-mapped cache with 16 KiB of data and a block size of 16 bytes. Assume the 32-bit physical address is used to map the cache memory. Identify the memory bits that are used as index bits, tag bits and the offset bits of this direct-mapped cache memory. Also, **draw a block diagram** of the above cache memory clearly showing the data field, tag field, valid bit and how the hit/miss is detected using the valid bit and the tag bits. How many total bits are required for this direct-mapped cache memory?

**Bonus Points:**  Control is the most challenging aspect of processor design: it is both the hardest part to get right and the hardest part to make fast. One of the hardest part of control is implementing exceptions and interrupts- event other than branches and jumps that change the normal flow of instruction execution. A pipeline implementation treats exceptions as another form of control hazard. For example, suppose there is an arithmetic overflow in an **add** instruction. We must flush the instructions that follow the **add** instruction from the pipeline and began fetching instruction from new address.

Given the following sequence of instruction,

14hex slt $11, $2, $4

18hex or $12, $2, $5

2Chex and $13, $2, $5

30hex add $7, $3, $7

34hex lw $16, 50($7)

38hex slt $15, $6, $7

4Chex sw $1, 32($7)

……………

Assume the instruction to be invoked on an exception begin like this:

80000180hex lw $20, 1000($0)

80000184hex sw $21, 1004($0)

Show what happens in the pipeline if an overflow exception occurs in the **add** instruction. Specifically, **indicate the input/output values of the PC, adders, pipeline register data, bubble (no operation) and control signals.** On Figure 5 (a & b) clearly indicate your answer. (20 pts)

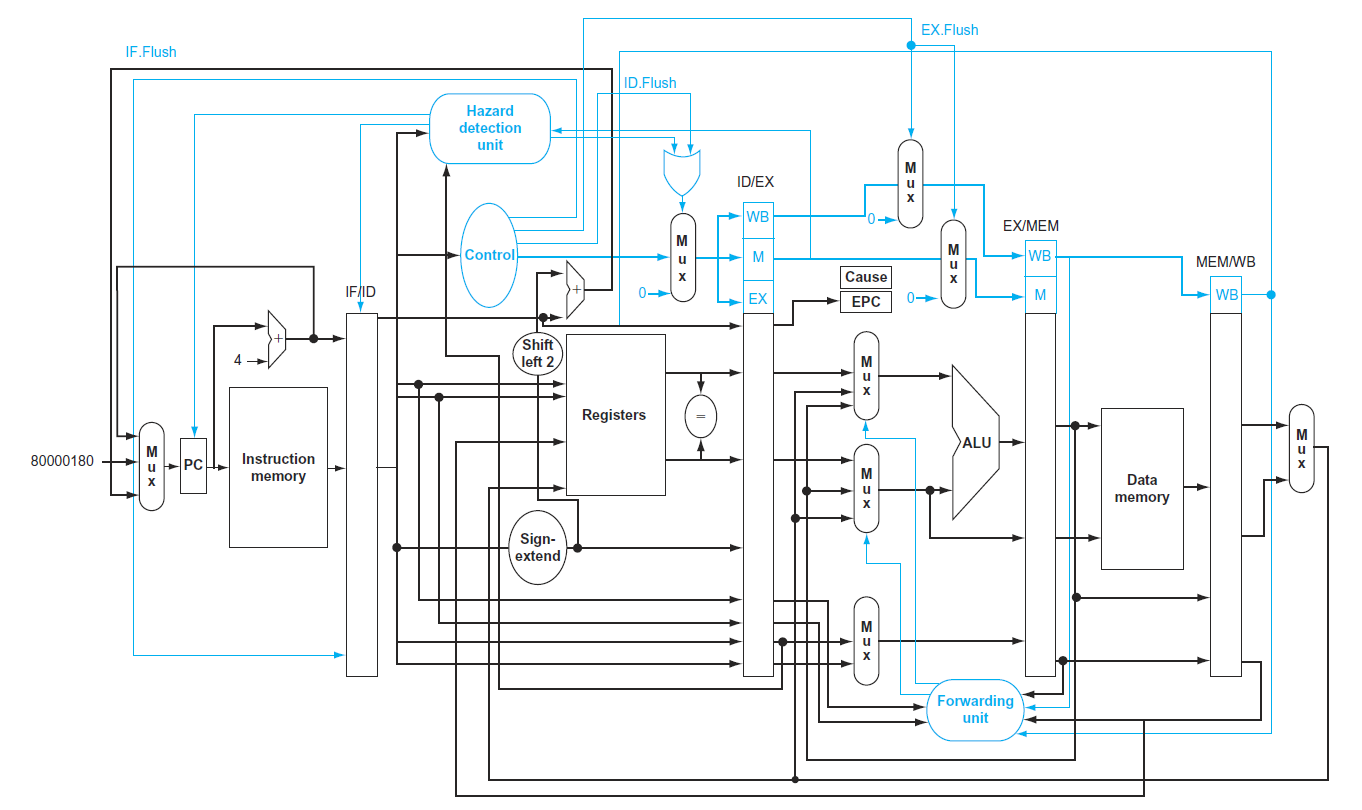


Figure 5(a)

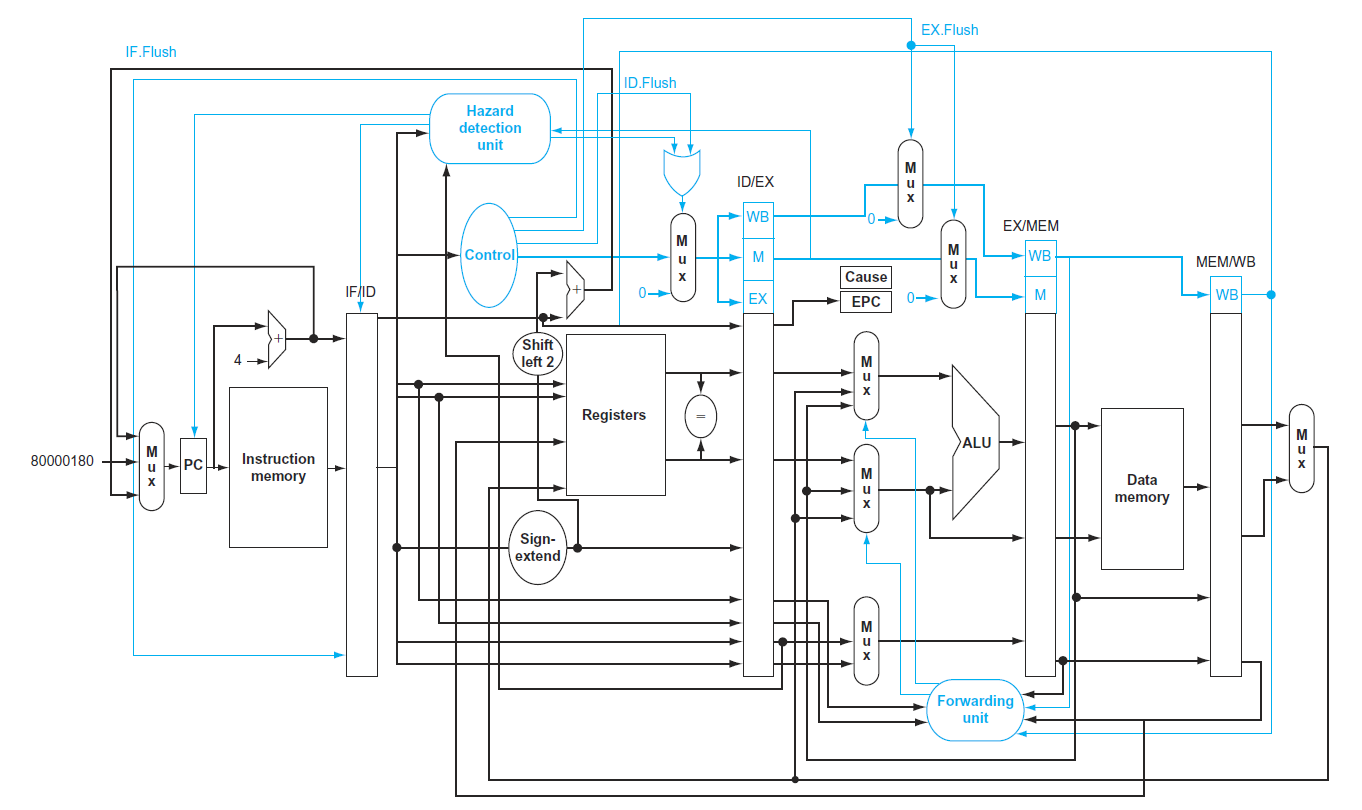


Figure 5(b)