The prerequisite quiz is used to determine if you have the background necessary to take CS252. This course assumes that you have a solid understanding of the material presented in CS152 or equivalent courses.

**Problem 1**

Describe the operation of a data cache. Your description should include discussion of the following:

a) Spatial and temporal locality.

b) Valid bits.

c) Direct mapped versus set-associative structures. Show how cache indexing and tag match works for both direct mapped and 2-way set-associative cache configurations assuming one word per cache line. What are the advantages and disadvantages of direct mapped versus set-associative structures?

d) Multiple-word cache lines. What are the advantages and disadvantages of multiple-word cache lines? Describe how they are implemented for a direct mapped cache.

e) LRU and random replacement policies. What are their relative advantages and disadvantages?
Problem 2

Ben Bitdiddle is designing a 32-bit MIPS processor that implements the ISA that has been covered in lecture. He starts off with the fully-bypassed 5-stage datapath shown in Lecture 2 as a baseline for his design, and then tries to optimize this pipeline.

a) Ben suggests replacing the ALU with a simple adder in the Execute stage and moving the original ALU from the Execute stage to the Memory stage.

In this new datapath, the 3rd stage (formerly Execute) now contains only an adder to do address calculations if it is a load/store instruction. Otherwise, the data is simply forwarded to the 4th stage.

The ALU will now run parallel to the data memory in the 4th stage of the pipeline (formerly Mem). During load/store instructions, the ALU is inactive. During ALU operations, the data memory is inactive. The datapath of the new pipeline is shown on the next page. The pipeline control logic signals and the program-counter logic are not shown.

With these changes, he hopes to decrease the number of bubbles in the instruction stream. What hazard was Ben trying to eliminate with his datapath change? Give an example of a set of MIPS instructions that would cause a pipeline bubble in the original datapath, but not in Ben’s modified datapath. Please use five or fewer instructions in your example.
<table>
<thead>
<tr>
<th>IF</th>
<th>ID</th>
<th>AC</th>
<th>EX/MEM</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instruction fetch</td>
<td>Instruction decode and register read</td>
<td>Address calculation</td>
<td>ALU execution and memory access</td>
<td>Writeback to register file</td>
</tr>
</tbody>
</table>

The diagram illustrates the pipeline stages of an instruction processing system, including:

- **Instruction Fetch (IF)**: Fetching the next instruction from memory and decoding it.
- **Instruction Decode (ID)**: Decoding the instruction and reading registers.
- **Address Calculation (AC)**: Calculating addresses for memory access.
- **Execute/Memory (EX/MEM)**: Executing the instruction and accessing memory.
- **Writeback (WB)**: Writing the results back to the register file.

The pipeline stages are interconnected, showing the flow of data and instructions through the system, from fetching to writeback.
b) Ben is extremely happy with his new implementation. He believes his new datapath is clearly better than the original. Is he right? What new hazard is introduced with Ben's modification? Do not consider jumps or branches for this part. Give an example of a set of instructions where a bubble is created in the new datapath that did not exist in the original. Please use five or fewer instructions in your example. Compare the advantages and disadvantages of Ben's modifications.

c) Suppose we had a MIPS ISA without a displacement (base+offset) addressing mode. This version of the MIPS ISA only supports register indirect addressing. Assuming the new machine only had to support this ISA, how could the datapath be improved? Draw the new datapath showing your design. (You do not have to show everything -- just the important features like pipeline registers, major components, major connections, etc.) Compare the hazards in this new datapath with Ben's modified datapath and the original datapath. Again, do not consider jumps and branches for this part. How is this datapath faster?

d) If the MIPS ISA did not have displacement addressing, what would programmers do? Could you still write the same programs as before?

e) Now we will consider jumps and branches for the pipeline shown in part A of this problem. Assume that the branch target calculation is performed in the Instruction Decode stage. In what pipeline stages can Ben put the logic to determine whether a conditional branch is taken? (don’t worry about duplicating logic) What are the advantages and disadvantages between the different choices? For each choice, consider the number of cycles for the branch delay, any additional stall conditions, and any potential changes in the clock period.
Problem 3

The questions below refer to the following circuit and its associated timing parameters. The flip-flops are positive-edge triggered, and FF0 has an enable input (Q only changes if En is high). Assume that all timing parameters are positive.

![Circuit Diagram]

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{xor}$</td>
<td>Delay through XOR gate</td>
</tr>
<tr>
<td>$t_{inv}$</td>
<td>Delay through inverter</td>
</tr>
</tbody>
</table>

Flip-flop parameters (for both flip-flops)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_{clkQ}$</td>
<td>Delay from clock to Q output</td>
</tr>
</tbody>
</table>

a) Fill in the following timing diagram for the circuit:

```
clk

q

d (glitch)
```

b) What is the maximum clock frequency for the circuit in terms of the given timing parameters (assume there is no setup time for the flip flops)?