1. Matching. Write the correct term from the list into each blank. (2 pts. each)

<table>
<thead>
<tr>
<th>硬接线</th>
<th>结构性危险</th>
<th>控制性危险</th>
<th>思想性执行</th>
</tr>
</thead>
<tbody>
<tr>
<td>微程序化</td>
<td>负载-使用数据危险</td>
<td>前向路由</td>
<td>精确异常</td>
</tr>
</tbody>
</table>

(a) ____________________ generation of control signals from random logic or a PLA
(b) ____________________ generation of control signals from microinstructions fetched from a control store
(c) ____________________ providing a data value to any unit where it is needed after the data value has been produced but before it is available in the register file
(d) ____________________ allowing an instruction that is control dependent on a branch to execute after the branch direction is predicted and before the branch is resolved

2. Consider the MIPS “load word” instruction as implemented on the datapath above (Figure 4.2 from textbook):


Circle the correct value 0 or 1 for the control signals (a-d) and circle whether each of the three muxes (e-g) selects its upper input, lower input, or don't care. For the ALU operation (h) circle one of the function names. (The Zero condition signal will be assumed to be 0.) (16 pts.)

(a) Branch = 0 1 (e) Mux1 (upper left; output to PC) = upper, lower, don't care
(b) MemRead = 0 1 (f) Mux2 (upper middle; output to Data port of Regs) = upper, lower, don't care
(c) MemWrite = 0 1 (g) Mux3 (lower middle; output to bottom leg of ALU) = upper, lower, don't care
(d) RegWrite = 0 1 (h) ALU operation = and, or, add, subtract, set-on-less-than, nor
3. Identify the five stages of the simple scalar pipeline we studied, and explain what each stage does when processing the load instruction from question 2 above. (10 pts.)

\[ \text{lw} \ R2, 8(R1) \ // \ \text{Reg}[2] \leftarrow \text{memory}[	ext{Reg}[1] + 8] \]

4. Associate each term or statement below with a type of dependency. Circle one or more of RAW, WAR, or WAW. (Destination registers are listed first for add and subtract instructions.) (3 pts. each)

(a) RAW / WAR / WAW  true data dependency
(b) RAW / WAR / WAW  false data dependency
(c) RAW / WAR / WAW  add R3,R1,R2 followed by sub R1,R3,R4
(d) RAW / WAR / WAW  add R3,R1,R2 followed by sub R5,R3,R4
(e) RAW / WAR / WAW  type of dependency that can cause a load-use penalty

5. Matching. Write the correct term from the list into each blank. (2 pts. each)

<table>
<thead>
<tr>
<th>EPIC</th>
<th>not an approach we studied</th>
<th>superscalar</th>
<th>VLIW</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dependency checking</td>
<td>function unit assignment</td>
<td>execution scheduling</td>
</tr>
<tr>
<td>(a)</td>
<td>_________________________</td>
<td>hardware</td>
<td>hardware</td>
</tr>
<tr>
<td>(b)</td>
<td>_________________________</td>
<td>hardware</td>
<td>software</td>
</tr>
<tr>
<td>(c)</td>
<td>_________________________</td>
<td>software</td>
<td>hardware</td>
</tr>
<tr>
<td>(d)</td>
<td>_________________________</td>
<td>software</td>
<td>software</td>
</tr>
</tbody>
</table>
6. Draw the dependency diagram for the following MIPS code. Destination registers are listed first except for the sw (store word) instructions; sw writes into memory rather than a register. (12 pts.)

```
lw    r2, 0(r1)
sw    r3, 0(r2)
add   r1, r2, r3
lw    r5, 4(r1)
add   r7, r5, r6
```

7. Give the pipeline cycle diagram (i.e., stairstep diagram) for the code segment given in question 6 above for the 5-stage pipeline with forwarding. (10 pts.)

```
lw    r2, 0(r1)
sw    r3, 0(r2)
add   r1, r2, r3
lw    r5, 4(r1)
add   r7, r5, r6
```

8. The branch CPI penalty is calculated as extra CPI = (branch freq.)*(misprediction freq.)*(mispredict penalty)

(a) Which one of the three terms in the penalty equation will techniques like loop unrolling and predication reduce? (3 pts.)

(b) Which one of the three terms does dynamic branch prediction attempt to reduce? (3 pts.)
9. Consider the following datapath. (Assume all registers are edge-triggered and thus immune from races.) Control signal identifiers are given for the in and out control points of the registers. Additional control signals include memory signals Mem, R (read), W (write), and 3-bit ALU function field F.

Complete the step-by-step RTL and the control signal sequence to fetch and execute a “store X” instruction. Assume that the instruction is composed of two memory words: a one-word opcode followed by a one-word address. Assume also that initially the address of the instruction is in the PC, and that the memory is word-addressable. The action of the instruction is memory[x] <- ACC, for the memory address X given in the second word of the instruction. (15 pts.)

// fetch opcode and place in IR
MAR <- PC
PC <- PC + 1
MBR <- memory[MAR]
IR <- MBR

// control signals
5 (A=PC), F=001 (C=A), 10 (MAR=C)
5 (A=PC), F=010 (C=A+1), 11 (PC=C)
Mem, R
1 (A=MBR), F=001 (C=A), 13 (IR=C)
Extra Credit.

XC-1. Explain what the term BTB means and what the BTB component does in the P6 pipeline (see below). (5 pts.)

XC-2. Explain the purpose of the reorder buffer in the P6 pipeline. (5 pts.)