ARCHITECTURE (25 points)

ıave a CPU that takes on the average 3 cycles for handling a branch (1 cycle for ic
and 2 cycles of branch penalty) and 1 cycle for handling any other instruction.
onsidering the use of the branch target buffer (BTB) to reduce the branch pena
3TB will not use branch folding.

) (6 pts) List all the fields in BTB and explain how they are filled and how they a
used during branch handling. Give the new **CPI** for taken branches found in BT
taken branches not found in BTB, not taken branches found in BTB and not tak
branches not found in BTB.

) (6 pts) If on the average 70% of branches are taken, 90% of taken branches
found in BTB and 5% of not taken branches are found in BTB, what is the n
CPI for a branch?

) (6 pts) The CPU spends 60% of **time** handling branches, before the improveme
How big is the overall speedup after the improvement? What percentage of ti
does the CPU spend handling branches **after** the improvement?

) (7 pts) What is the percentage of branch instructions in the overall instruct
mix? To calculate this use the knowledge of CPI for branches and non-branc
and the amount of time that CPU spends to handle branches.

. ARCHITECTURE (25 points)

For the following loop:

```
Loop:   L.D F0, 100(R1)
        L.D F2, 50(R1)
        ADD.D F4, F0, F2
        S.D F4, 100(R1)
        DADDUI R1, R1, #4
        DADDUI R2, R2, #-4
        BNEZ R2, Loop
```

For all instructions but S.D and BNEZ the first register specified in the instruction is the destination register. For S.D and BNEZ the first register specified in the instruction is the source register.

a) (8 pts) How many cycles does it take to execute one iteration of this loc without any instruction shuffling? Assume the standard 5-stage MIP pipeline. The branches are completely resolved in ID stage, the ADD.D take 4 cycles of execution, and all other instructions take 1 cycle of executio Assume that there is no structural hazard during the execution (pipelir resources are duplicated as needed), the processor is performing a sing instruction issue per clock cycle and the only slow-down factor is th existence of RAW and control hazards. Do not forget to account for branc delay in total cycles needed to execute one loop iteration. Count all the cycle before IF of the next iteration as belonging to the current iteration.

b) (8 pts) Shuffle instructions around to get rid of as many stalls as you ca (including the branch delay). You may change offsets if necessary. Show th transformed loop. How many cycles does it take to execute one iteration now Count all the cycles before IF of the next iteration as belonging to the curren iteration.

c) (9 pts) Unroll this loop twice (so that there are total of three iterations rearrange the code so that there are no remaining stalls and show th transformed code. Don't forget to rename the registers and adjust the offset How many cycles does one iteration of the original loop take now?

RCHITECTURE (25 points)

Assume the following fragment of code:

```
1. Loop:   L.D F2, 100(R1)
2.         MUL.D F6, F2, F4
3.         ADD.D F8, F6, F2
4.         ADD.D F10, F8, F4
5.         ADD.D F6, F2, F2
6.         S.D F10, 100(R1)
7.         DADDUI R1, R1, #4
8.         DADDUI R2, R2, #-1
9.         BNZ R2, Loop
```

Let time 1 be the time when this fragment starts being scheduled and let MU take 7 cycles of execution while ADD.D and SUB.D take 4 cycles of execu L.D and S.D take 2 cycles of execution each, one for address calculation and for memory access.

a) (9 pts) List all the data, output, anti and control dependencies. For dependence you must clearly state what type it is, between which instructions (use instruction number for this purpose) it occurs, and w register causes the dependence.

b) (8 pts) For one loop iteration, list when would each instruction be is complete execution and write CDB with Tomasulo's algorithm. Assume CDB and infinite number of reservation stations.

c) (8 pts) Using the code segment from the problem statement to illustrate discussion, discuss how Tomasulo's algorithm handles RAW (read-a write), WAW (write-after-write), and WAR (write-after-read) hazards compare this with a way in which scoreboard handles these same haz What would introduce larger slowdown in case of scoreboarding than in of Tomasulo's algorithm? What limits the speedup we can achieve Tomasulo's algorithm?

You are given a system with 2-level unified cache. L1 cache is write-allocate, write-through cache, with size 64KB and a block size of 32B. L2 is write-allocate, write-back cache with size 1MB and a block size of 64B. On the average 20% of blocks in L2 are dirty. For every 1000 cache system accesses, there are 100 misses in L1 cache and 20 misses in L2 cache. L1 access time is 1ns, L2 access time is 10ns and memory access time is 100ns. CPU reads and writes 16B words. The following table gives the cost of transferring 16B of data between CPU, L1, L2 and memory.

16B between CPU and L1	16B between L1 and L2	16B between L2 and MEM
0.5ns	5ns	10ns

a) (3 pts) Give L2 read miss and write miss penalties.
b) (2 pts) How much do we pay for L2 read hit?
c) (2 pts) How much do we pay for L2 write hit?
d) (3 pts) Give L1 read miss and write miss penalties.
e) (2 pts) How much do we pay for L1 read hit?
f) (2 pts) How much do we pay for L1 write hit?
g) (3 pts) What values (a—f) would change if L1 were no-write-allocate? List all the new values. Assume all the other settings for L1 and L2 remain as in the problem statement.
h) (4 pts) What values (a—f) would change if L1 were write-back with 10% dirty blocks? List all the new values. Assume all the other settings for L1 and L2 remain as in the problem statement.
i) (4 pts) What values (a—f) would change if L2 were write-through? List all the new values. Assume all the other settings for L1 and L2 remain as in the problem statement.