Homework 13

Problem 1

```
bool need_valC =
    icode in { IIRMOVL, IRMMOVL, IMRMOVL, IJXX, ICALL, IIADDL };

int srcB = [
    icode in { IOPL, IRMMOVL, IMRMOVL, IIADDL } : rB;
    icode in { IPUSHL, IPOPL, ICALL, IRET } : RESP;
    icode in { ILEAVE } : REBP;
    1 : RNONE;  # Don't need register
];

int dstM = [
    icode in { IMRMOVL, IPOPL } : rA;
    icode in { ILEAVE } : REBP;
    1 : RNONE;  # Don't write any register
];

int aluB = [
    icode in { IRMMOVL, IMRMOVL, IOPL, ICALL, IPUSHL, IRET, IPOPL, IIADDL,
ILEAVE } : valB;
    icode in { IRRMOVL, IIRMOVL} : 0
    # Other instructions don't need ALU
];

int mem_data = [
    # Value from register
    icode in { IRMMOVL, IPUSHL } : valA;
    # Return PC
    icode == ICALL : valP;
    # Default: Don't write anything
];
```

Problem 2

SEQ: 8 hours (480 minutes)
PIPE: 4.5 hours (270 minutes)
Speedup: 8/4.5 = 1.78

Problem 3

Data Hazard:

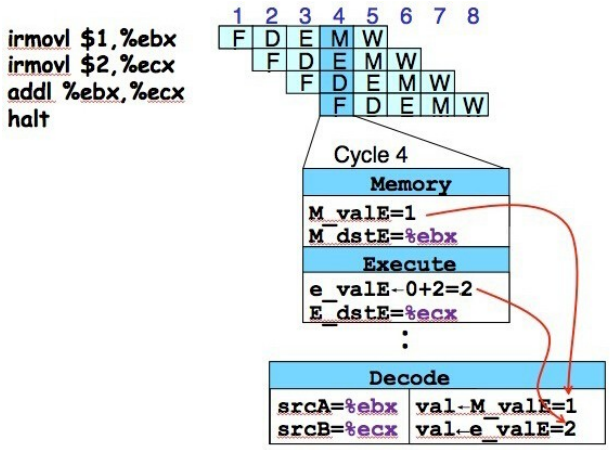
Instruction depends on result of prior instruction still in the pipeline

Control Hazard:

Caused by delay between the fetching of instructions and decisions about changes in control flow

Problem 4

1) In cycle 4, the decode stage logic detects a pending write to register %ebx in the memory stage. It also detects that a new value is being computed for register %ecx in the execute stage. It uses these as the values for valA and valB rather than the values read from the register file.



2）No, the read of %eax of the first instruction is in Stage 2, while the write of %eax of the next instruction is in Stage 5, there will be no data hazard.