Processor Architecture #2 **4190.308 Computer Architecture**

Name: _____

Due Date: Thursday, November 13, 2014, 23:59 **Student-Number:** _____

Submission: in paper form.
There will be a drop off box in class and in front of the CSAP Lab in building 301, room 419.

Question 1

SEQ Stage Implementations

Write HCL code for the signal `instr_valid` in the SEQ implementation

```
bool instr_valid = icode in
  { INOP, IHALT, IRRMOVL, IIRMOVL, IRMMOVL, IMRMOVL,
    IOPL, IJXX, ICALL, IRET, IPUSHL, IPOPL };
```

Question 2

SEQ Stage Implementations

The register signal `srcB` indicates which register should be read to generate the signal `valB`. The desired value is shown as the second step in the decode stage in Figures 4.18 to 4.21 in the textbook. Write HCL code for `srcB`.

```
## What register should be used as the B source?
int srcB = [
  icode in { IOPL, IRMMOVL, IMRMOVL  } : rB;
  icode in { IPUSHL, IPOPL, ICALL, IRET } : RESP;
  1 : RNONE;  # Don't need register
];
```

Question 3

SEQ Stage Implementations

Based on the first operand of the first step of the execute stage in Figures 4.18 to 4.21 in the textbook, write an HCL description for the signal `aluB` in SEQ.

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

Question 4

SEQ Stage Implementations

Looking at the memory operations for the different instructions shown in Figure 4.18 to 4.21 in the textbook, we can see that the data for memory writes is always either `valA` or `valP`. Write HCL code for the signal `mem_data` in SEQ.

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

Question 5

SEQ Stage Implementations

Write HCL code for `Stat`, generating the four status codes `SAOK`, `SADR`, `SINS`, and `SHLT` (see Figure 4.26 in the textbook).

```
## Determine instruction status
int Stat = [
  imem_error || dmem_error : SADR;
  !instr_valid: SINS;
  icode == IHALT : SHLT;
  1 : SAOK;
];
```