

### ELEC 5200/6200 – CPU Design Project #3

You are to design an instruction set architecture (ISA) for a new microprocessor (uP). Your ISA is to be designed using RISC design principles, with primary design goals being low cost and a minimal number of clock cycles per instruction. Following are the requirements for your ISA.

1. The ISA may contain no more than 15 unique instructions. However, you may have multiple formats for a given instruction, if necessary.
2. The ISA is to support 14-bit data words only. (No byte operands.)
  - a. All operands are to be 14-bit signed integers (2's complement).
  - b. Each instruction must be encoded using one 12-bit word.
3. The ISA is to support linear addressing of 16K, 14-bit words memory. The memory is to be word-addressable only - not byte-addressable.
4. The ISA should contain appropriate numbers and types of user-programmable registers to support it.
5. The ISA must “support” the following C Programming Language constructs:
  - \* Assignment operator: `variable = expression;`  
Expressions must support the two arithmetic operators:  
add (+) and subtract (-)  
Data are limited to:
    - 14-bit two's-complement integers (Example: `int a;`)
    - One-dimensional integer arrays (Example: `int a[10];`)
  - \* Control flow structures: “if-else” structures, “while” loops, “for” loops  
These should support the six standard relational operators:  
`== != > <= < >=`
  - \* Functions (call and return), with parameters passed by value or by reference.

#### **Provide the following information about your ISA:**

- List and describe the user-programmable registers.
- For each instruction in your instruction set, list the following:
  - o Assembly language for each form of the instruction - mnemonic and operands
  - o Machine language for each form of the instruction:
    - instruction code format, op-code, and operand encoding
  - o Justification for including each form of the instruction in your ISA
- To verify the completeness of your ISA, “compile” the C program on the next page into the assembly language of your ISA, i.e. write an assembly language program that would implement the given C program.

```

/* ELEC 5200/6200 CPU Design Project - Test Program */
/* Function f fills array n using passed arguments*/

int f(g,n)
    int g,*n;           -- passed value g and address n
{
    int i,k;           -- local variables
    for (i=0; i<5; i++) -- exercise a for loop
    {
        k = g + i;     -- create value for k
        if (i >= 2)    -- test if-then-else
            n[i] = k + 0x0f0f; -- test add
        else
            n[i] = k - 0x10f0; -- test subtract
    }
    while (i < 9)      -- test while loop
    {
        i = i + 1;     -- loop control variable
    }
    return(i);         -- return value to main
}

/* Main Program - initialize a variable & call f */

void main()
{
    int a,b[8];        // define variables
    a = f(0x56e,b);    // call f and return b
}

```