Dalhousie University
Department of Electrical and Computer Engineering

ECED 3403 – Computer Architecture

Lab 2: Decoding instructions

1 Objectives

In this lab you are to design, implement, test, and demonstrate a program that fetches instructions from your XM23p's IMEM and decodes the instruction. Only instructions required for Assignment 2 need to be decoded. However, all other valid instructions should be displayed without decoding.

2 Specifications

In this lab, you are to:

1. Load IMEM and DMEM with records from an S-Record file you have written.
2. Read each instruction from IMEM, starting from the starting address specified in the S-Record file. You can either read from the array directly or do a proper fetch using your own version of the Instruction Memory Controller if you have completed it.
3. Extract the opcode from each fetched instruction, display the name of the opcode and the instruction. For example, instruction `400A` fetched from location `2000` would decode into:

   ```
   2000: ADD RC: 0 WB: 0 SRC: R1 DST: R2
   ```

   The instruction `47A2` fetched from location `2006` would decode into:

   ```
   2006: AND RC: 1 WB: 0 CON: 8 DST: R2
   ```

4. The program should stop when the instruction has the value `0000`. Note that `0000` is a valid instruction (what is it?) and all memory has a default value of `0000`.

Only the instructions used in Assignment 2 should be fully decoded. All other instructions should be displayed along with their address; for example, `0000` at location `432A` would be displayed as:

```
432A: 0000
```

3 Suggestions

Do the design first. It will make life much easier.

Look for patterns in the XM23 instruction set.

Structures and unions allow individual bits and groups of bits to be extracted painlessly.

The instructions to decoded are the opcodes for `ADD` through `SXT` and `MOVL` through `MOVH`. Any other opcodes should be printed along with its address in memory.

4 Submission and Marking

4.1 Submission

The assignment will be marked using the following marking scheme:

Design: A brief design is required, consisting of a data dictionary of the major structures and algorithms for decoding and storing.

Software: A fully commented, indented, magic-numberless, tidy piece of software that meets the requirements described above and follows the design description.

Testing: Sufficient test results must be supplied to demonstrate that your implementation meets the specifications. For example, it will be necessary to demonstrate that the different opcodes and operands are recognized.

Your solution must be demonstrated in the lab for it to be marked.

Your design, software, and tests must be submitted electronically to the course website.

4.2 Marking

Marking is as follows:

3: All requirements (design, software, and testing) met.

2: One requirement missing or coding problem.

1: Two or three requirements missing or coding problems.

0: Not submitted or more than three requirements missing or coding problems.

Undemonstrated solutions will not be marked.

5 Important Dates

Available: 29 May 2024 at noon, Atlantic.

Submission due no later than: 30 May 2024 at 9am, Atlantic.

6 Miscellaneous

This lab is to be completed individually.

If you are having *any* difficulty with this or any part of the course, *please* contact Dr. Hughes or Emad as soon as possible.