CFS2101: y86 Disassembler Starting Source Code

Dr Minsi Chen

1 Introduction

This document describes the starting source code provided for the y86 disassembler coursework. Please read through the document carefully so that you understand the following:

- the content contained in each task package
- how to compile the given source code
- how to run the disassembler with the given test files

2 Source Code Package Content

You are given three source code package in ZIP format. Each serves as a starting point for the tasks specified in the Assignment 02 brief.

The contents for each package after unpacking are listed below.

- **y86DisasmTask1.zip** and **y86DisasmTask2.zip** both contain identical starting files for Task 1 and Task 2 respectively:
  - **y86disasm.c** - this is the source file containing a partially implemented y86 disassembler.
  - **test1.txt** - this is a text file containing a simpler set of y86 instruction bytecode for testing your disassembler.
  - **test2.txt** - this is a text file containing a slightly more complex set of y86 instruction bytecode for testing your disassembler.
  - **test1-output.txt** and **test2-output.txt** contain the y86 assembly code corresponding to the bytecode listed in test1.txt and test2.txt. You can use this two files to check if your disassembler produces expected results.
- **y86DisasmTask3.zip** contains the starting files for Task 3
  - **y86disasm.c** - this is the source file contain an incomplete y86disassembler.
  - **prog1.o, prog2.o and prog3.o** are the three test files containing y86 bytecode store in binary format.
  - **prog1.ys, prog2.ys and prog3.ys** are the three corresponding y86 programs written in assembly. You can use these files to check if your disassembler is able to decode the bytecode correctly.

3 How to Use the Provided Source and Test Files

Please note: the following description assumes you are working on this assignment using `repl.it`.

When working on each task, you must ensure all files provided in a package are uploaded to repl.it.

Compiling source code : As a reminder you can compile your source code using the following command in the repl.it terminal.

```
clang y86disasm.c -o y86disasm
```

This generates an executable `y86disasm`.

Using the test files : The test files are provided for you to test the completeness of your disassembler. To use a test file with your disassembler, you need to pass a test file as an input when running the executable. See below as an example:

```
./y86disasm test1.txt
```

The y86diasm will read the content of `test1.txt` and let the disassembler to decode each instruction.

Please note: the starting source code only dissasembles three of the simplest instructions, i.e. nop, halt and ret. The rest will be displayed as `TODO:`. As you work through each given task, those TODO should be replaced by correctly disassembled instruction.

4 Tips on Completing the Assignment Tasks

Here are few pointers on what you need to know to succeed on completing this coursework.

- You only need to code inside the main function. Ignore everything that are provided to you as helper utility. The TODO comments in the source code signpost the part you need to work on.
- The core C programming knowledge required for this coursework includes bitwise operations, use of arrays, if statements and printf.
- For Task 1 and task 2, you only need to know the y86 ISA format and its assembly mnemonics. There is a cheatsheet provided as a quick reference on Brightspace.
- For Task 3, you also need to know the concept of program counter and use it to fetch instructions from the "memory".
- You should use the provided test files to check your implementation, and compare your disassembler outputs with the provided output files, e.g. test1-output.txt and test2-output.txt. This gives you an idea on how complete your implementation is for each task.