CISC 380 Architecture

Project 1 Overview

James Allender & Mitchell Nelson

Description:

For project 1 we developed a simple assembler to assemble the UST-3400 (Rip Van saWinkle) ISA into machine code. Our program uses a single procedural C program to read in a given file containing assembly code and assemble that into machine code and either output it to the screen or to a file specified by the user.

Overview:

Our program takes one or two arguments, an input file or an input file and an output file. Any deviation from this will lead to a failure and exit from the program.

One input example

$./assembler -i [assembler code file]

Two input example

$./assembler -i [assembler code file] -o [out file]

The program can also have its output redirected to a file with

$./assembler -i [assembler code file] > [out file]

The program requires that an input -i file is given or it will fail out and close. When an input file is provided the program checks that the input file exists in main, and if it exists it starts processing the file.

The file is read twice by the program. On the first pass the program iterates over each line and records any labels it finds in a hash map with the corresponding line number of the label. The program validates that the provided labels meet the requirement for a valid label.

On the second pass over the file we read in all of the arguments on a line into an array to be used to construct the machine code for an instruction. Each Instruction type has a separate builder to interpret the instruction into a decimal number depending on the unique field layout for that instruction.

In each of these machine code creators each parameter of an instruction is passed to a parameter handler function that validates it is a valid parameter and returns its number value if it is a string of a number, or the corresponding decimal number for a label.

For I type instructions the machine code builder also calculates the offset for labels by taking the corresponding decimal value to the label provided by the parameter handler function and either subtracts the current line number + 1 from that value if the label is for a location prior to the current line or subtracts the current line number if the label location is after the current location.

The machine code creators then takes the interpreted parameters and packs them into their appropriate position in a 32bit integer by shifting them an amount specified in the program and then bitwise or’ing all the pieces together. These decimal machine codes generated by the machine code creator are then printed to the screen or the specified output file.

Challenges: