Report On Disassembler Lab 3

Name:Aditya Garg
Roll Number:CS22BTECH11002

October 9, 2023

1 Coding Approach

- I decided to divide my code into various functions which would make the code readable and follow 'DRY' principle.
- Since a machine code is interpreted by looking at opcode first in order to determine the format of instruction , based on the opcode present, i created functions for all format instructions .
- Implementation of Part1 - R , Part2 - I/S , Part -B/J/U were straightforward. Find the opcode , call the required function based on opcode, evaluate funct3 and funct7 , immediate using bit manipulations (whenever needed) . Most of the times funct3 plays the role to determine the instruction in the same format.
- Part4 was slightly different . I created an array of MAXSIZE initialized to 0 representing the label of the instructions. Every time a B/J type instruction was encountered the position denoted by immediate value was given a non-zero number if a label was not present previously.
- Each instruction is stored a string in a vector . For outputting the ans , for the ith instruction if value in the array is non-zero , output the label as 'L'+array[i] followed by the instruction .
- For negative immediate values, if the value after converting the hex input to integer results in an ans greater than 2047, I am subtracting it with 4096 to obtain the actual value . For B and U type the value was different which has been taken care of.

2 Testing Approach

- For generating test cases, I used RIPES and used some of the practice problem codes as input which also have been added in the submission.

- Multiple test cases have been added in TestCases.txt file along with the their output for verification.
- While testing , I have taken care of various inputs like having immediate as negative number, working with various labels and multiple format instructions .