DOCUMENTATION

CSE-112 COMPUTER ORGANIZATION

PROJECT-MID SEMESTER [WINTER 2020]

ROLL NO.: 2019327_2019336

GROUP NO.: 06

PROJECT DESCRIPTION

The project requires us to design an Assembler which takes a text file as an input, which contains the Assembly code. The assembler must be able to translate the assembly code into machine code. The machine code generated must also be saved in a separate text file.

ASSEMBLER WORKING

The ASSEMBLER which we made works in TWO PASSES.

In Ist PASS it scans the Assembly Code and store it in the form of table.

In the 2nd PASS it checks for errors and print the Machine Code.

1st PASS

- Comment Handling: We are assuming that the comment will start by this symbol "//" and we will ignore the part written after that symbol.
- Literal Table: Assuming the format of literal to be "=value" .
- Label: For the Label we are assuming that the label is of the syntax ("LabelName:").
- Operands: The operands will be in the form of variable, literal or addresses can also be given on which the specific operation will be performed.
- We are saving the output in a txt file. And we are printing the literal table, symbol table, operands table and opcode table on terminal/console.
- We were given the valid Opcodes and their Machine Opcode as well for the project which is given below:

```
opcodes =
["CLA","LAC","SAC","ADD","SUB",'BRZ','BRN','BRP','INP','DSP','MUL','DIV','STP']
b_opcode= ['0000','0001','0010','0011','0100','0101','0110','0111','1000','1001','1010','1011','1100']
```

Note: We are assuming that the code will start only if there is "START" in the code at the beginning and End only if there is "END" in the end of code. We are not giving any Machine Opcode to the "START" & "END". Also here along with START position of location counter can also be given if you want to start from specific position else it will be 0.

2nd PASS

We will generate the final machine code in our second pass using the symbols table and literal table. We will use the address of the respective symbols, literals and operands.

And after the second pass, if there are any error then we have printed it on the line of the output file where the error is occurring, and then we will not get the machine code of the line on which there was error.

Else if there is no error, then we have printed the machine code on the line of the output file with the literal table, symbols table, operands table and opcode table on the terminal.

ERRORS

- “START” missing in the start of code.
- “END” missing in the end of code.
- Invalid Label.
- Same label defined more than once.
- Operands are given to “CLA” & “STP”.
- Symbol is used but not defined.
- Number of operands given is not correct.
- Division done by 0. (DIV 0 is written)
- Variable which cannot be used (Ex. “SUB” is used as variable but it is an opcode).
- No operand is given to certain opcode.
- Variable is not defined. (We have assumed that variable will only be defined if it is initialized by “INP” or “SAC”).
- Address (Instruction, Location Counter) has more than 8 bits.
- Wrong opcode is used.

HOW TO USE THE ASSEMBLER

We have written the assembler program in the python programming language.

To run the Assembler just run the file with extension ".py" given with this documentation.

Before running the Assembler provide the name of the input txt file to the assembler program code file which we have made in python.

To get the output file with the Machine code written in it in txt file just write the name of the output file and it will generate a txt file with machine code (If the file is existing it will make changes in that file only else it will create the file with that name which is given to it).