

DAYANANDA SAGAR COLLEGE OF ENGINEERING

COMPUTER SCIENCE & ENGINEERING

Minor Project- Report

Apr 2023-Jul 2023

Course Faculty: Prof. Aparna

Course Name & Code: SYSTEM SOFTWARE LAB WITH MINI-PROJECT
(19CS6DCSSW)

Semester: VI

Date: 14/06/2023

TITLE OF THE PROJECT	TWO PASS ASSEMBLER			
STUDENT NAME	PRUTHA VASISHT	PUNITH KUMAR P R	SUHAS REDDY	RACHANA K
USN	1DS20CS157	1DS20CS158	1DS20CS159	1DS20CS160
INDIVIDUAL CONTRIBUTION	Pass 2 Code in the Assembler	Pass 1 Code in the Assembler	Pass 1 Code in the Assembler	Pass 2 Code in the Assembler
GUIDE	Prof. Aparna			
PROJECT ABSTRACT:	<p>A two-pass assembler works by performing the following steps in two passes:</p> <ul style="list-style-type: none">1. In Pass 1, addresses are assigned to all the statements in the program. Then the values assigned to the labels and symbols are saved for use in Pass 2 using SYMTAB and OPTAB. It also processes pseudo-operations. An intermediate file is generated.2. In Pass 2, instructions are assembled using values in SYMTAB and OPTAB. Data values defined by BYTE and WORD are generated and the assembler directives which were not processed in Pass 1 are processed. The object program and assembly listing are written. <p>We will be focusing on generating machine codes from a set of assembly language codes.</p>			
PLATFORM USED (H/W & S/W TOOLS TO BE USED)	<u>Development Environment:</u> Visual Studio Code, Windows OS <u>Programming Language:</u> C++			

DAYANANDA SAGAR COLLEGE OF ENGINEERING

COMPUTER SCIENCE & ENGINEERING

INTRODUCTION

An assembler is a program for converting instructions written in low-level assembly code into relocatable machine code and generating along information for the loader. If the assembler does all this work in one pass, then it is called a single pass assembler and otherwise if it does it in multiple passes then it is called a multiple pass assembler. When it does it in two passes, it is called a 2 Pass Assembler. The work done in both passes can be described as follows -

Pass - 1:

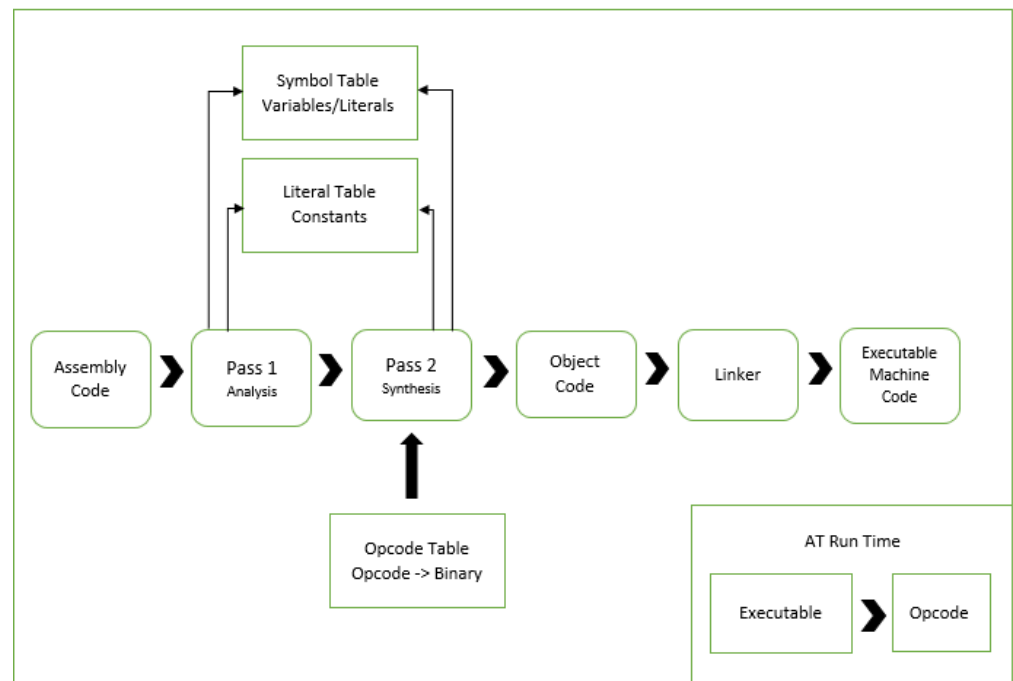
- Define symbols and literals and remember them in symbol table (SYMTAB) and literal table (OPTAB) respectively.
- Keep track of the location counter.
- Process pseudo-operations like macros and directives.

Pass - 2:

- Generate object code by converting symbolic opcode into respective numeric opcode.
- Generate data for literals and look for values of symbols in OPTAB and SYMTAB.

DESIGN

The architectural design of a two-pass assembler can be shown as follows -



DAYANANDA SAGAR COLLEGE OF

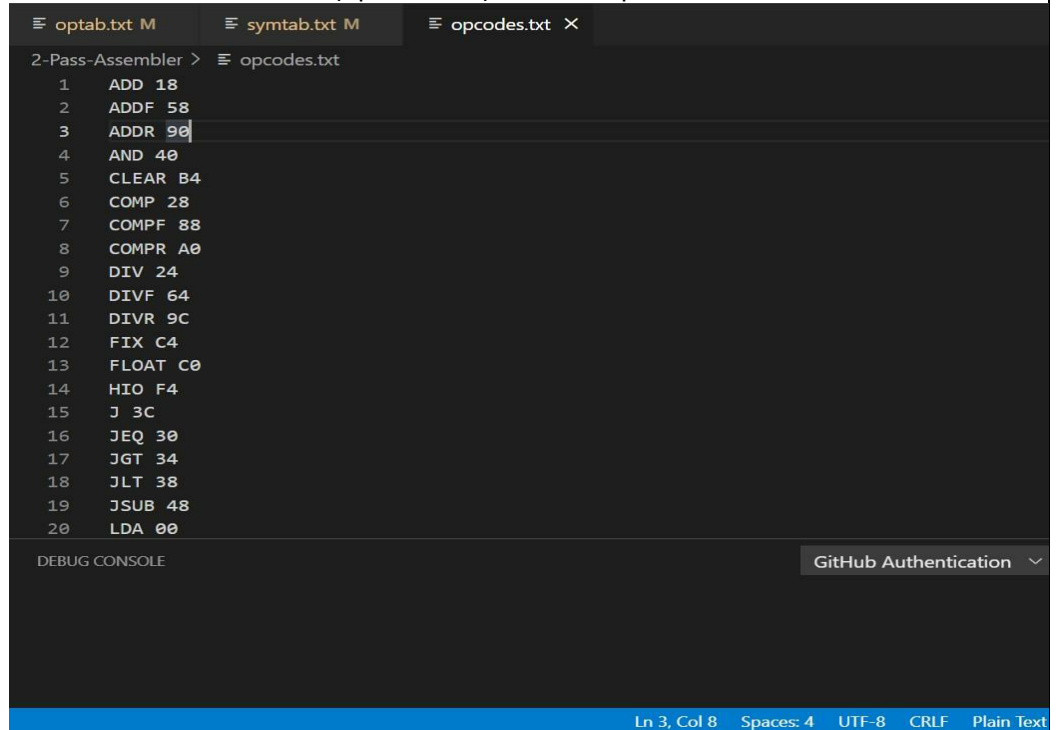


PROJECT SOURCE CODE LINK (GITHUB/ GOOGLE DRIVE)	https://github.com/punith-kumar-pr/2-pass-assembler
CONCLUSION /FUTURE ENHANCEMENT	The program implements the working of a 2 Pass Assembler for SIC machine. Further, the program could be written for SIC/XE machine where the instructions are written in 4 different formats.

DAYANANDA SAGAR COLLEGE OF

INPUT:

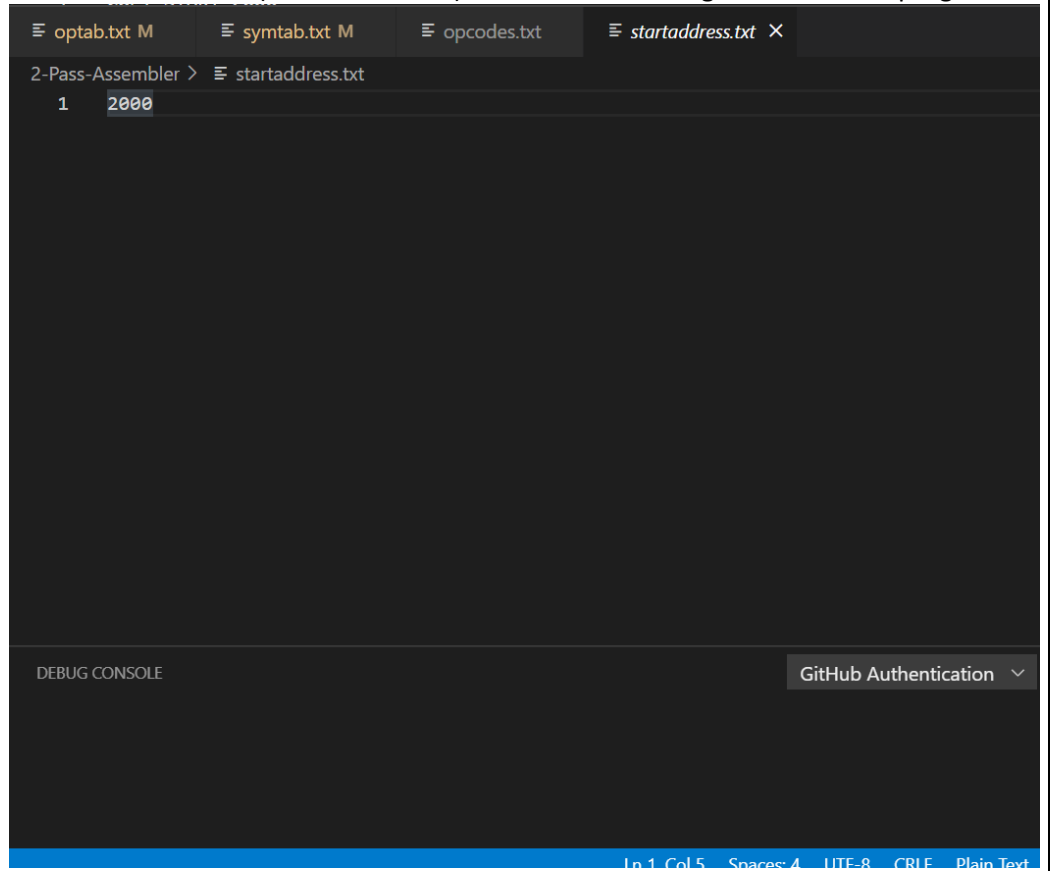
1. The below text file (opcodes.txt) contains opcodes for all mnemonics.



The screenshot shows a text editor window titled "2-Pass-Assembler" with a tab for "opcodes.txt". The file contains 20 lines of assembly opcodes. The status bar at the bottom indicates "Ln 3, Col 8", "Spaces: 4", "UTF-8", "CRLF", and "Plain Text".

```
1  ADD 18
2  ADDF 58
3  ADDR 90
4  AND 40
5  CLEAR B4
6  COMP 28
7  COMPF 88
8  COMPR A0
9  DIV 24
10 DIVF 64
11 DIVR 9C
12 FIX C4
13 FLOAT C0
14 HIO F4
15 J 3C
16 JEQ 30
17 JGT 34
18 JLT 38
19 JSUB 48
20 LDA 00
```

2. This text file (startaddress.txt) contains the starting address of the program.



The screenshot shows a text editor window titled "2-Pass-Assembler" with a tab for "startaddress.txt". The file contains a single line with the starting address "2000". The status bar at the bottom indicates "Ln 1, Col 5", "Spaces: 4", "UTF-8", "CRLF", and "Plain Text".

```
1  2000
```

UI SCREENSHOTS

3. This text file (input.txt) contains the source code that needs to be converted into object code.

```

input.txt M X
2-Pass-Assembler > input.txt
1 COPY START 2000
2 **** LDX ZERO
3 MOVECH LDCH STR1,X
4 **** STCH STR2,X
5 **** TIX ELEVEN
6 **** JLT MOVECH
7 **** RSUB ****
8 STR1 BYTE C'EOF'
9 STR2 RESB 1
10 ZERO WORD 0
11 ELEVEN WORD 11
12 **** END ****

optab.txt M symtab.txt M opcodes.txt startaddress.txt X
2-Pass-Assembler > startaddress.txt
1 2000

DEBUG CONSOLE
GitHub Authentication
Ln 12, Col 15 Spaces: 4 UTF-8 CRLF Plain Text

```

OUTPUT:

The final output program and object program is displayed.

```

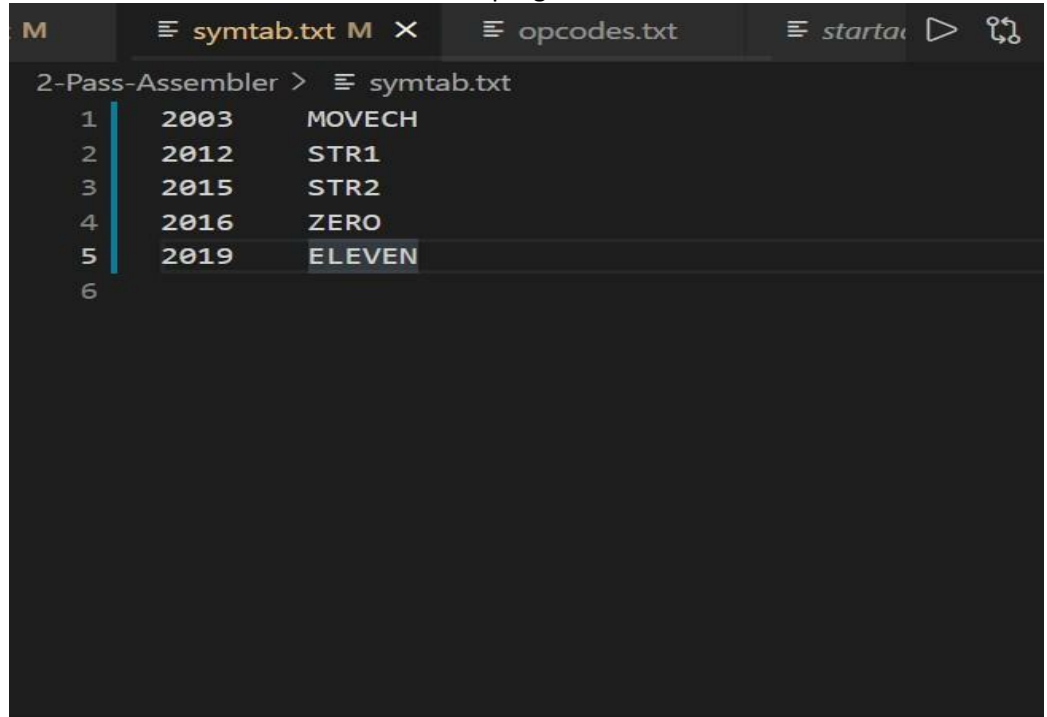
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL Cod
[Running] cd "d:\College\VI Sem\Two Pass Assembler Mini Project\" && g++ index.cpp -o index && "d:\College\VI Sem\Two Pass Assembler\index.cpp:16:14: warning: non-static data member initializers only available with -std=c++11 or -std=gnu++11 [enabled by default]"
THE FULL PROGRAM
=====
2000 COPY START 2000
2000 **** STCH Char2,X 54a021
2003 **** LDA FIVE 002018
2006 **** STA ALPHA 0c2012
2009 **** LDCH CHARZ 50201b
200c **** STCH Char1 54201f
200f **** RSUB **** 4c0000
2012 ALPHA RESW 2
2018 FIVE WORD 5 000005
201b CHARZ BYTE C'EOF' 454f46
201e CHARX BYTE X'f1' f1
201f Char1 RESB 2
2021 Char2 RESB 4
2025 **** END ****
=====

THE OBJECT PROGRAM
H^COPY^002000^000025
T^002000^12^54a021^002018^0c2012^50201b^54201f^4c0000
T^002018^07^000005^454f46^f1
E^002000
=====

[Done] exited with code=0 in 9.735 seconds

```

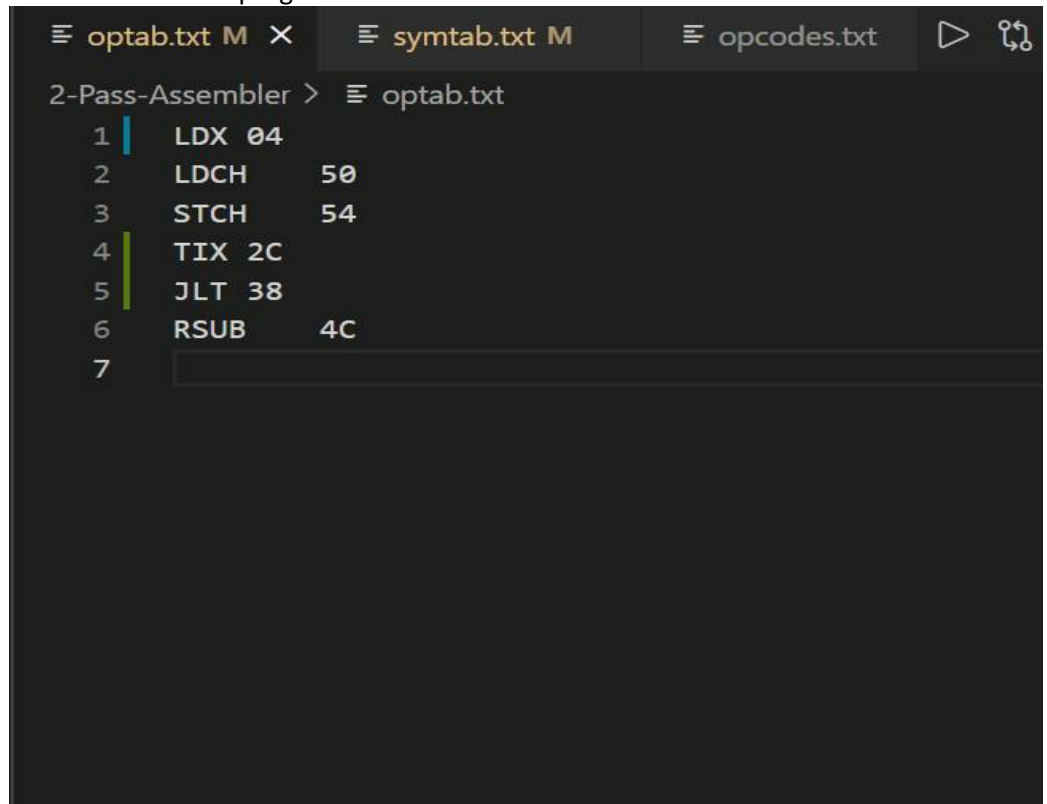
1. The following text file (symtab.txt) will contain the symbols and their addresses from the source program.



```

M      symtab.txt M X      opcodes.txt      starta
2-Pass-Assembler > symtab.txt
1      2003      MOVECH
2      2012      STR1
3      2015      STR2
4      2016      ZERO
5      2019      ELEVEN
6
  
```

2. This text file (optab.txt) contains the opcodes and mnemonics from the source program.



```

optab.txt M X      symtab.txt M      opcodes.txt      >
2-Pass-Assembler > optab.txt
1      LDX 04
2      LDCH 50
3      STCH 54
4      TIX 2C
5      JLT 38
6      RSUB 4C
7
  
```



References

- [1] "Two Pass Assemblers," www.entcengg.com. [Online].
Available: <https://www.entcengg.com/two-pass-assemblers/>. [Accessed: June 12, 2023].
- [2] Prithi Mishra, System Software. Bengaluru: Subhas Publications, 2015.