User Manual for Two-Pass Assembler GUI Application

Table of Contents

---

1. Introduction

The Two-Pass Assembler GUI application is a Java-based program designed to simulate a two-pass assembler for SIC/XE (Simplified Instructional Computer/Extended) assembly language. It reads an input file containing assembly code, processes it with an opcode table (OPTAB), and produces intermediate files, symbol tables, and final object code in HTE (Header, Text, End) format.

The application includes a graphical user interface (GUI) for easy file handling, execution, and output viewing, making it ideal for students, instructors, and hobbyists interested in learning how assemblers work.

---

2. Installation and Setup

1. **Download and Install Java (if not installed):**
   - Download the appropriate JDK for your platform.
   - Install it by following the on-screen instructions.
2. **Download the Source Code:**
   - Download or clone the source code of the Two-Pass Assembler GUI application to your local machine.

3. Compile and Run the Application:

- Open a command prompt or terminal. o Navigate to the directory where the source code is saved.
- Compile the program:

Copy code

```
javac TwoPassAssemblerGUI.java
```

- Run the program:

Copy code

```
java TwoPassAssemblerGUI
```

---

3. Running the Application

After launching the program, you will see a graphical user interface (GUI) consisting of several panels and buttons.

Loading Input Files

1. **What is the Input File?**
   - The input file is a text file (.txt) that contains SIC/XE assembly code.
   - Example:

sql

Copy code

```
**  START 2000
**  LDA FIVE
**  STA ALPHA
**  LDCH CHARZ
**  STCH C1
ALPHA RESW 2
FIVE  WORD 5
CHARZ BYTE C'Z'
C1    RESB 1
**  END
```

2. **Steps to Load the Input File:**
   - Click the Load Input File button on the GUI.

- A file explorer window will open. Navigate to the .txt file containing your assembly code and select it.
- After loading, the file is stored in the application and will be used when you run the assembler.

Loading OPTAB Files

2. **What is the OPTAB File?**
   - The OPTAB (opcode table) file is a text file that contains opcode mappings (operation codes and their corresponding hexadecimal values).
   - Example:

r

Copy code

LDA 03

STA 0F

LDCH 53

STCH 57

3. **Steps to Load the OPTAB File:**
   - Click the Load Optab File button on the GUI.
   - A file explorer window will open. Navigate to the .txt file containing your opcode mappings and select it.

Running the Assembler

1. Once both the input and OPTAB files are loaded, click the Run Assembler button.
2. The assembler will process the input file in two passes:
   - **Pass 1:** The assembler builds the intermediate file, calculates addresses, and constructs the symbol table.
   - **Pass 2:** The assembler generates the object code in the HTE format.
4. Once the assembler completes, the results will be displayed in the text areas of the GUI.

---

5. **Understanding the Output**

The GUI contains several panels that display important information during the assembly process.

Intermediate File:

- This panel shows the intermediate representation of the program, including memory addresses and instruction information.

- Each line consists of the address, label, opcode, and operand.

Symbol Table (SYMTAB):

- This panel displays the symbol table, which contains the label names and their corresponding memory addresses.
- Example: yaml

Copy code

```
ALPHA   2003
FIVE    2006
CHARZ   2009
C1      200A
```

Program Length:

- This panel displays the total program length (in hexadecimal), calculated by the assembler during **Pass 1**.

Output:

- The output panel shows the processed assembly code alongside the generated object code (without HTE formatting).
- Each line consists of the address, label, opcode, operand, and object code.

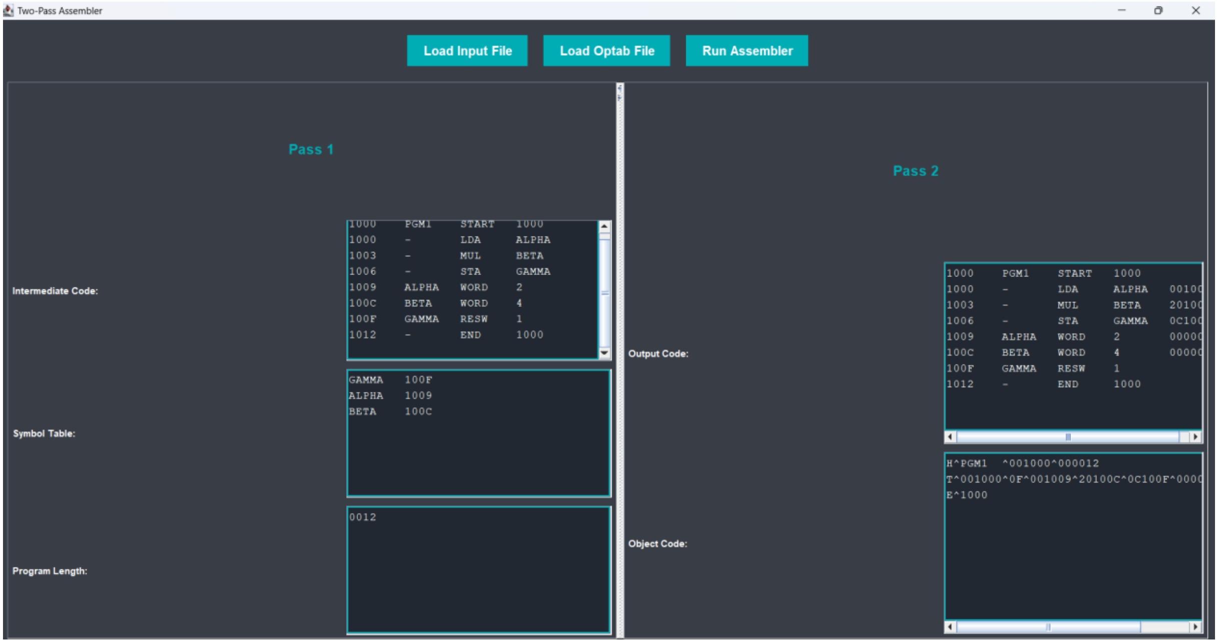
Object Code (HTE Format):

- This panel shows the final object code in HTE (Header, Text, End) format.
- Example:

Copy code

```
H^PROG^002000^00000A
T^002000^03^030005^0F2003^532009^57200A
E^002000
```

This user manual provides an overview of the Two-Pass Assembler GUI application, guiding users through setup, operation, and understanding the results of the assembly process. The tool is an educational resource that demonstrates how a two-pass assembler functions and how SIC/XE assembly programs are processed.



Pass 1

Intermediate Code:

```
1000    PGM1    START   1000
1000    -       LDA     ALPHA
1003    -       MUL     BETA
1006    -       STA     GAMMA
1009    ALPHA   WORD    2
100C    BETA    WORD    4
100F    GAMMA   RESW    1
1012    -       END     1000
```

Symbol Table:

```
GAMMA   100F
ALPHA   1009
BETA    100C
```

Program Length:

```
0012
```

Pass 2

Output Code:

```
1000    PGM1    START   1000
1000    -       LDA     ALPHA   00100
1003    -       MUL     BETA    20100
1006    -       STA     GAMMA   0C100
1009    ALPHA   WORD    2       00000
100C    BETA    WORD    4       00000
100F    GAMMA   RESW    1
1012    -       END     1000
```

Object Code:

```
H^PGM1  ^001000^000012
T^001000^0F^001009^20100C^0C100F^0000
E^1000
```