Two Pass Assembler Documentation (Website)

Angelina K Joseph

40

Table Of Contents

Introduction

Welcome to the Two-Pass Assembler documentation. This application allows you to convert assembly language code into machine code using a two-pass assembly process. This guide will help you understand how to effectively use the assembler.

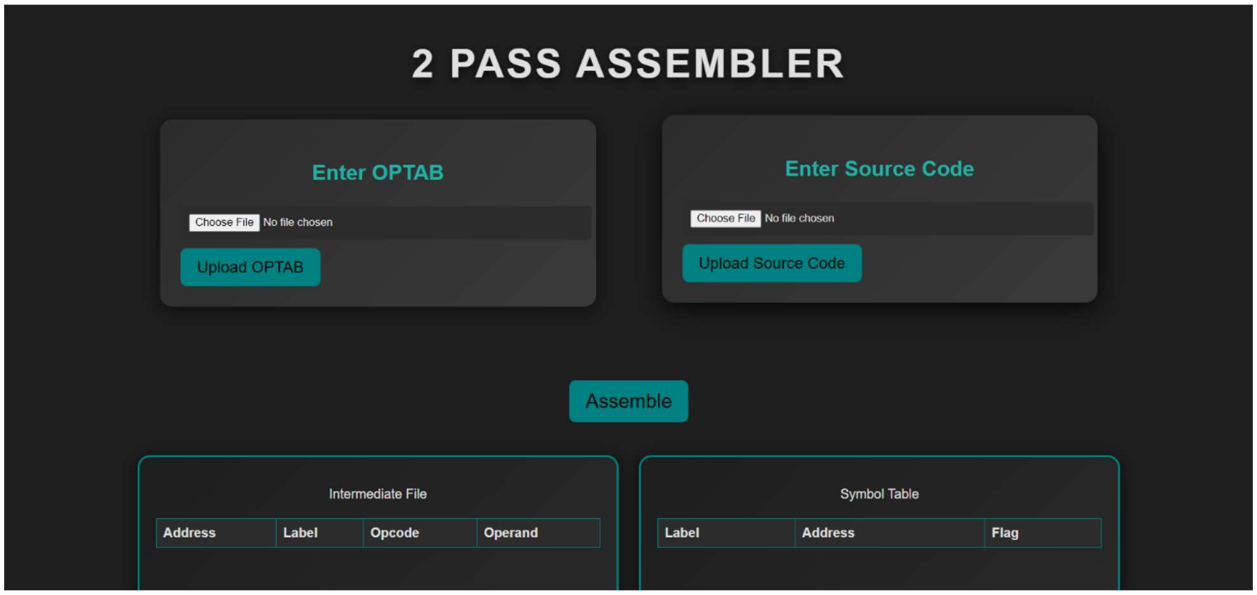
System Requirements

- A modern web browser (Chrome, Firefox, Safari, Edge).
- Access the assembler by opening the index.html file in your browser.

Opening the assembler

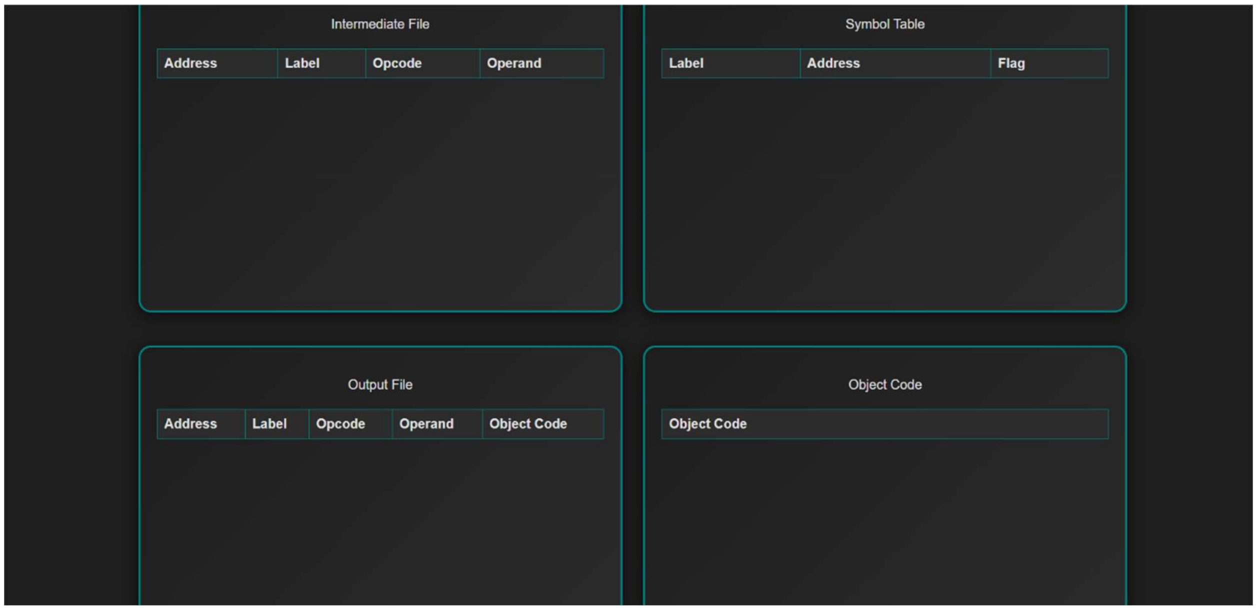
- Download the source code package
- Locate the index.html file in your downloaded folder
- Double-click the index.html file to open it in your browser

The website should look like this



Address	Label	Opcode	Operand

Label	Address	Flag



Address	Label	Opcode	Operand

Label	Address	Flag

Address	Label	Opcode	Operand	Object Code

Object Code

Working

The two-pass assembler processes the assembly language code in two stages:

Pass One

- Objective: Scan the source code for labels and create a symbol table.

- Process:
    - Read each line of code.
    - Identify labels and record their corresponding memory addresses.
    - Calculate the size of each instruction and data definition.

Pass Two

- Objective: Translate assembly instructions into machine code using the symbol table created in Pass One.

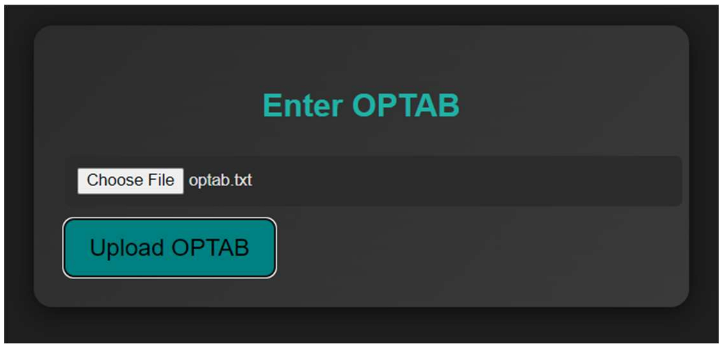
- Process:
    - Read the source code again.
    - Replace labels with their corresponding addresses from the symbol table.
    - Generate the final machine code output.

Using the Assembler

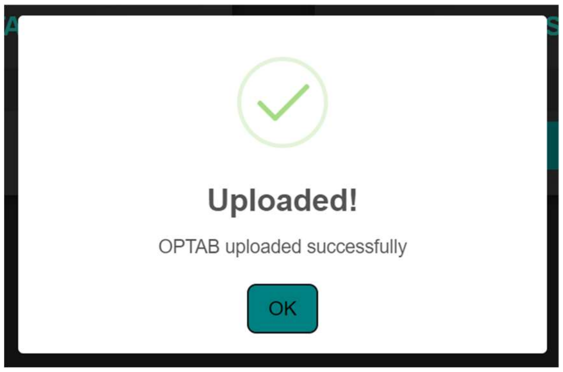
Step-By-Step Instructions

1. Upload Optab:

   - Click the “Choose File” button.
   - Select your text file containing the assembly code and upload it.

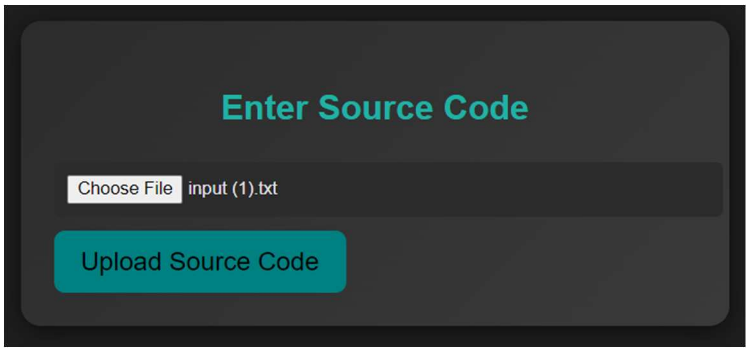


   - Click the “Upload OPTAB” button.

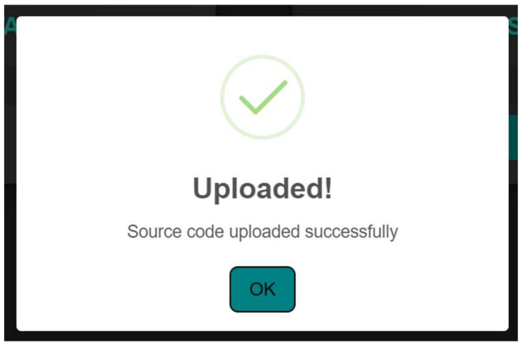


OPTAB uploaded successfully

2. Upload Source Code:
   - Click the “Choose File” button.
   - Select your text file containing the source code and upload it.



- Click the “Upload Source Code” button.

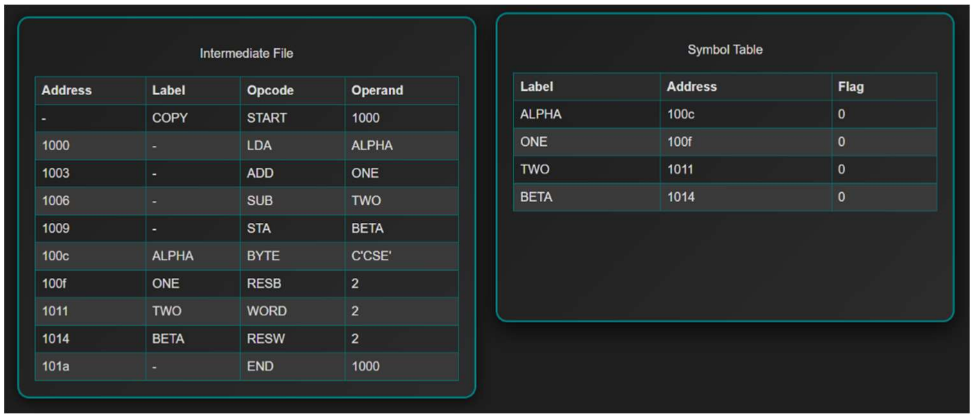


Uploaded!

Source code uploaded successfully

3. Run the Assembler:
    - Click the “Assemble” button to initiate the two-pass process.

4. View Output:
    - The different outputs will be displayed below the input area. Any error messages will also be shown for correction.

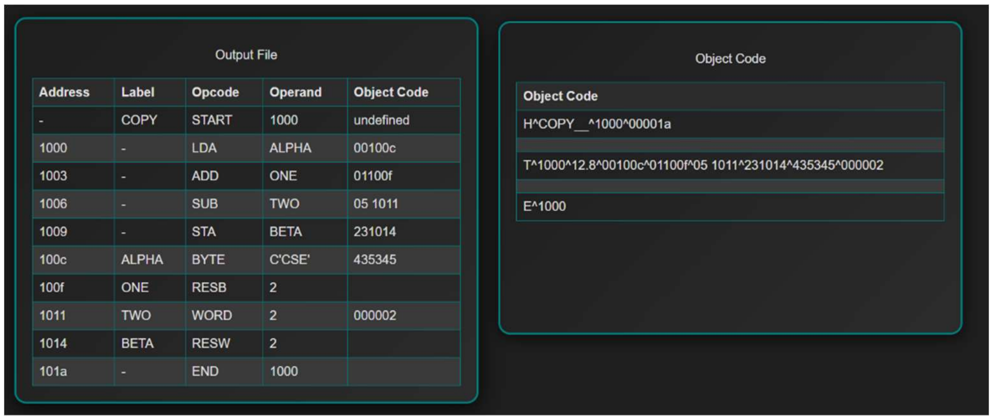


Intermediate File

Address	Label	Opcode	Operand
-	COPY	START	1000
1000	-	LDA	ALPHA
1003	-	ADD	ONE
1006	-	SUB	TWO
1009	-	STA	BETA
100c	ALPHA	BYTE	C'CSE'
100f	ONE	RESB	2
1011	TWO	WORD	2
1014	BETA	RESW	2
101a	-	END	1000

Symbol Table

Label	Address	Flag
ALPHA	100c	0
ONE	100f	0
TWO	1011	0
BETA	1014	0



Output File

Address	Label	Opcode	Operand	Object Code
-	COPY	START	1000	undefined
1000	-	LDA	ALPHA	00100c
1003	-	ADD	ONE	01100f
1006	-	SUB	TWO	05 1011
1009	-	STA	BETA	231014
100c	ALPHA	BYTE	C'CSE'	435345
100f	ONE	RESB	2	
1011	TWO	WORD	2	000002
1014	BETA	RESW	2	
101a	-	END	1000	

Object Code

Object Code
H^COPY__^1000^00001a
T^1000^12.8^00100c^01100f^05 1011^231014^435345^000002
E^1000

Features

- Error checking: The assembler provides error messages for any issues that might occur.
- Output display: The resulting machine Code is displayed for review after assembly.

Example

Sample Source Code

Suppose you have a source code file named input.txt with the following syntax and content:

```
COPY     START     1000
-        LDA       ALPHA
-        ADD       ONE
-        SUB       TWO
-        STA       BETA

ALPHA    BYTE      C'CSE'
ONE      RESB         2
TWO      WORD         2
BETA     RESW         2
-        END        1000
```

Sample Optab File

The optab file, optab.txt, should look like this:

```
SUB  05
CMP  03
LDA  00
STA  23
ADD  01
JNC  08
```

Intermediate File

After uploading both files and running the assembler, the intermediate file might look like this:

```
-       COPY       START        1000
1000    -               LDA          ALPHA
1003    -               ADD          ONE
1006    -               SUB          TWO
1009    -               STA          BETA
100C    ALPHA           BYTE         C'CSE'
100F    ONE             RESB         2
1011    TWO             WORD         2
1014    BETA            RESW         2
101A    -               END
```

Symbol Table

The symbol table should look like this:

ALPHA	100C	0
ONE	100F	0
TWO	1011	0
BETA	1014	0

Assembled Output File

The output file should look like this:

```
        COPY     START    1000
1000    -        LDA      ALPHA     00100c
1003    -        ADD      ONE       01100f
1006    -        SUB      TWO       051011
1009    -        STA      BETA      231014
100C    ALPHA    BYTE     C'CSE'    435345
100F    ONE      RESB       2
1011    TWO      WORD       2       000002
1014    BETA     RESW       2
101A    -        END
```

Object Code File

The final object code would look like this:

```
H^COPY__^1000^00001a
T^1000^12^00100c^01100f^051011^231014^435345^000002
E^1000
```

Troubleshooting

Common Issues

- **Error: Syntax Error**: Ensure that your assembly instructions follow the correct syntax. Refer to the syntax guide within this guide for formats.
- **Error: File Not Found**: Ensure that the uploaded files are in the correct format and accessible