Documentation for Two-Pass Assembler Web Application

Overview

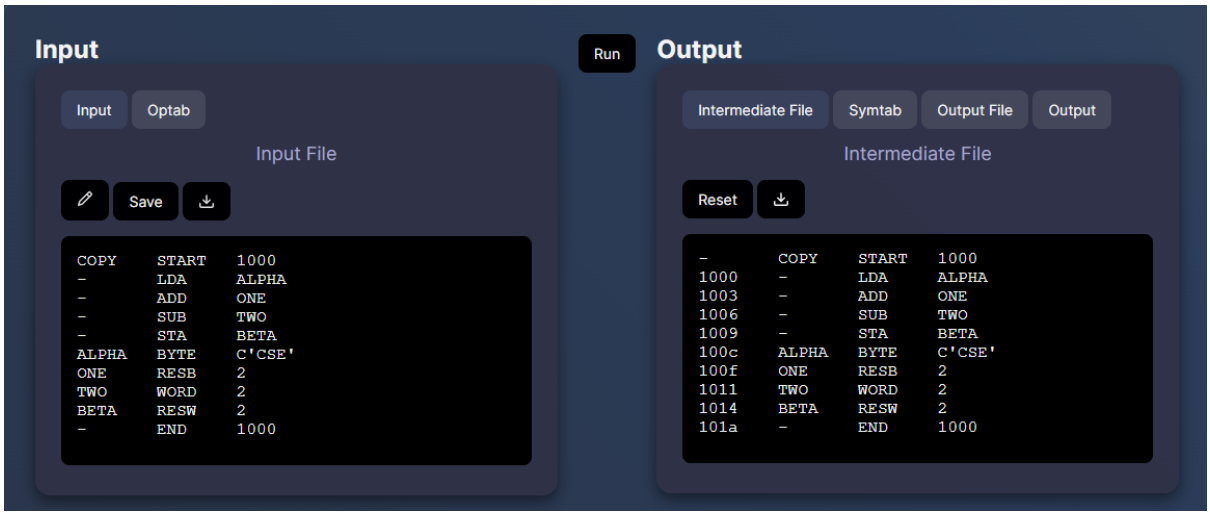
This web-based assembler tool provides users with an interface to input assembly code,

run a two-pass assembler, and view intermediate files, symbol tables, and the final

output. The application is designed with an intuitive user interface and offers

functionalities such as editing, saving, downloading, and resetting files.

Key Features



Two-Pane Display:

Left Pane: Displays the input file (assembly code) or the operation table (optab).

Right Pane: Displays the intermediate file, symbol table (symtab), or output files

generated after running the assembler.

Editing and Saving:

Users can edit the assembly code or optab file and save changes in local storage or

download the file to their computer.

Run Assembler:

A "Run" button allows users to run the assembler, which processes the input file and generates intermediate, symbol table, and final output files.

File Management:

Users can download any file (input, intermediate, symbol table, output) for offline use. A reset button allows users to clear all data and start fresh.

Detailed Functionality

1. File Viewing

Left Pane:

The user can toggle between viewing the input file or the optab by clicking buttons. For example, selecting the "Input" button displays the current assembly code, while selecting the "Optab" button shows the operation table.

Right Pane:

The user can switch between viewing intermediate files, symbol tables, and final outputs.

Each button in the right pane loads the respective file into the view.

2. File Editing and Saving

Editing:

The input file (assembly code) and the optab file can be edited directly in the text area of the left pane. Users can click the edit button, make changes, and then save or download

the updated file.

Saving:

After editing, users can save the file to local storage, ensuring that any changes they make

are stored and can be accessed later during their session.

Download:

The system allows users to download the contents of either pane. Once downloaded, users can keep a local copy of their input, optab, or output files.

3. Running the Assembler

The application includes a two-pass assembler simulation:

First Pass: Processes the input assembly code and generates an intermediate file and symbol table.

Second Pass: Using the intermediate file and symbol table, the final machine code or equivalent output is generated.

Once the user clicks the "Run" button, the input and optab are processed, and the results

(intermediate file, symbol table, output file) are displayed in the right pane.

4. Resetting and Clearing Files

Users can reset the system by clicking the reset button, which clears all files from local storage and empties the display areas.

This allows for a fresh start, ensuring that previous data does not interfere with new input.

5. Downloading Files

Each pane has a "Download" button, enabling users to download the currently displayed

file (input, optab, intermediate, symtab, or output).

This functionality uses the browser's File System API to create a downloadable file containing the current content.

User Interaction Workflow

Loading the Page:

By default, the input file (assembly code) is displayed in the left pane, and the intermediate file is shown in the right pane.

Editing Files:

Users can edit the assembly code or optab by clicking the edit button. After making changes, they can save the file to local storage or download it.

Running the Assembler:

Clicking the "Run" button initiates the assembler, generating intermediate and final output files.

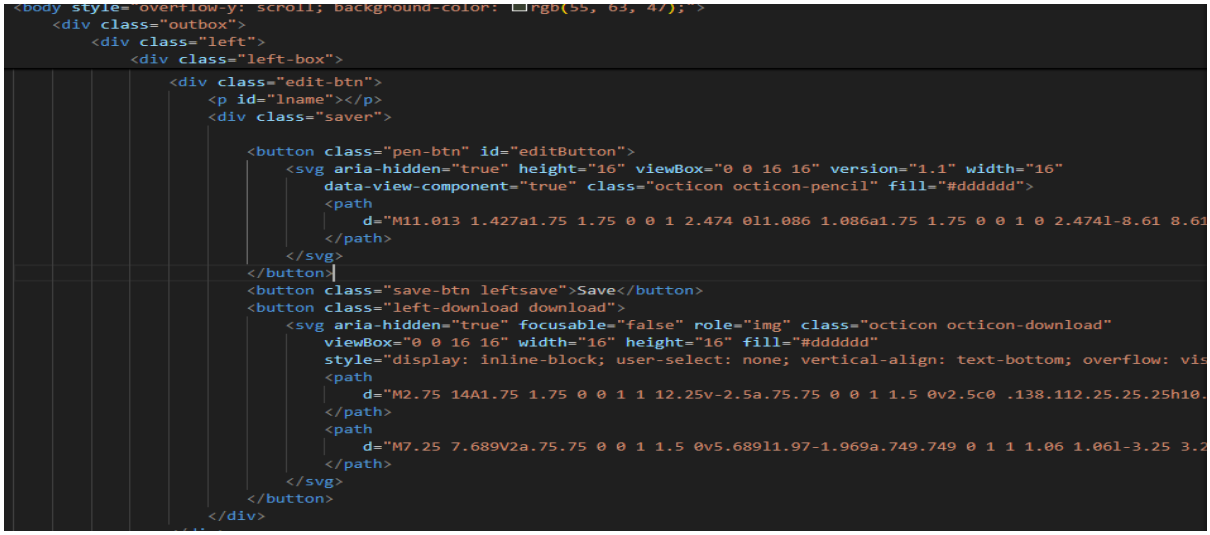
Viewing and Downloading Output:

Users can view intermediate files, symbol tables, or output files in the right pane and download them for further use.

Resetting the System:

The reset button clears all data, resetting the application to its initial state.

The html code for this page is



The css code for this is

```css
/* Reset default browser styles */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

body {
    margin: 0;
    padding: 0;
    font-family: 'Inter', sans-serif;
    background-color: #1e1e2e;
    color: #f5f5f7;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    min-height: 100vh;
    overflow: hidden; /* Hide overflow for background animations */
    position: relative; /* Positioning for background animations */
    padding: 20px;
    animation: fadeIn 0.7s ease-in-out;
}

/* Animation for fade-in effect */
@keyframes fadeIn {
    from {
        opacity: 0;
    }
    to {
        opacity: 1;
    }
```

And the script for this is

```javascript
document.querySelector('.intermediate-btn').addEventListener('click', async () => {
    try {
        document.querySelector('#rname').innerHTML = "Intermediate File";
        rightselected = "intermediate";
        document.querySelector('.intermediate-btn').style.backgroundColor = "#38405c";
        document.querySelectorAll('.symtab-btn, .output-btn, .output-btn2').forEach((btn) => {
            btn.style.backgroundColor = "#44475a";
        });

        const text = localStorage.getItem('intermediate.txt');
        if (text) {
            document.querySelector('.right-box textarea').value = text;
        } else {
            document.querySelector('.right-box textarea').value = "";
            document.querySelector('.right-box textarea').placeholder = "Run the assembler to generate intermediate file";
        }
    } catch (error) {
        console.error("Error in intermediate button click:", error);
    }
});

document.querySelector('.symtab-btn').addEventListener('click', async () => {
    try {
        document.querySelector('#rname').innerHTML = "Symtab File";
        rightselected = "symtab";
        document.querySelector('.symtab-btn').style.backgroundColor = "#38405c";
        document.querySelectorAll('.intermediate-btn, .output-btn, .output-btn2').forEach((btn) => {
            btn.style.backgroundColor = "#44475a";
        });
```

Conclusion

This assembler tool provides a browser-based solution to run a two-pass assembler with

the ability to edit, save, download, and reset files. Its intuitive interface makes it easy for

users to input assembly code, view the resulting intermediate and output files, and

manage these files efficiently.

You can refer to this link to understand the concept of pass assembler
https://pass-assembler-zpgv.vercel.app/