Software Requirements Specification

for

Intel 8085 Online Simulator

Version 0.1

Prepared by Ayush Kashyap

MNIT Jaipur

15/1/2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the Intel 8085 online simulator software. It will explain the purpose and features of the software, the interfaces of the software, what the software will do and the constraints under which it must operate. This document is intended for users of the software and developers.

1.2 Document Conventions

This Document was created based on the IEEE template for System Requirement Specification Documents.

1.3 Intended Audience and Reading Suggestions

- *Typical Users, such as students, who want to use Simulator to understand the working of 8085 microprocessor and gain some experience in writing the assembly language.*
- *Programmers who are interested in working on the project by further developing it or fix existing bugs.*

1.4 Product Scope

This simulator is a tool that people can use to study Intel 8085 MP and learn assembly. Users can use it write the scripts against the simulator and run it. They can view the exact changes for the memory, registers and flags. They can debug it using step by step.

1.5 References

- *Simulator Source: https://github.com/ayushk7/simsim85*
- *Wikipedia: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiAn6Gk1tH1AhXP7XMBHQNiAP8QFnoECAMQAQ&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FIntel_8085&usg=AOvVaw0LDJdNeNu0kCgwtGv1VQ4l*
- *Tutorials Point: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiRmp3Z1tH1AhVD7XMBHQxrDY4QFnoECCYQAQ&url=https%3A%2F%2Fwww.tutorialspoint.com%2Fmicroprocessor%2Fmicroprocessor_8085_architecture.htm&usg=AOvVaw1cY5kkYY9TkfX_Lm-IyOzT*

2. Overall Description

2.1 Product Perspective

Intel 8085 online simulator was developed for everyone who wants to better understand the working of the 8085 microprocessor, or wants to test their codes or debug them or just wants to experiment.
It was developed to run on all major web browsers.

2.2 Product Functions

- *Assemble: Assembles the assembly program inputted in the editor to the machine instructions.*
- *Run: Executes the machine instructions.*
- *Start Debugging: Executes the machine instructions in one by one and reflects the changes in the register and memory views.*
- *Next: When in debug mode triggers the execution of the next instruction.*
- *Prev: When in debug mode goes used to trace back one instruction.*
- *Reset: Corresponding Reset buttons are used to reset the memory, register and flag views.*
- *Editor: User writes the code, do syntax highlighting and displays the errors.*
- *Memory View: Used to edit the memory stored values at any time and syncs with actual memory*
- *Register View: Used to edit the register stored values at any time and syncs with actual registers*
- *Flag View: Used to edit the flag values at any time and syncs with the actual flag values.*
- *GitHub Link: Directs users to the source code on the GitHub.*

2.3 User Classes and Characteristics

- *Typical users, such as students, who want to use Simulator to understand the working of 8085 microprocessor and gain some experience in writing the assembly language.*

- *Programmers who are interested in working on the project by further developing it or fix existing bugs.*

2.4 Operating Environment

Any major web browser (Firefox, Edge, Chrome etc).

2.5 Design and Implementation Constraints

The Simulator is developed in HTML, CSS and Vanilla JavaScript. It uses Model, View, Controller (MVC) Pattern where processing takes place in model and synced with view using the controller.

2.6 User Documentation

There is a quick start guide available on the GitHub: https://github.com/ayushk7/simsim85

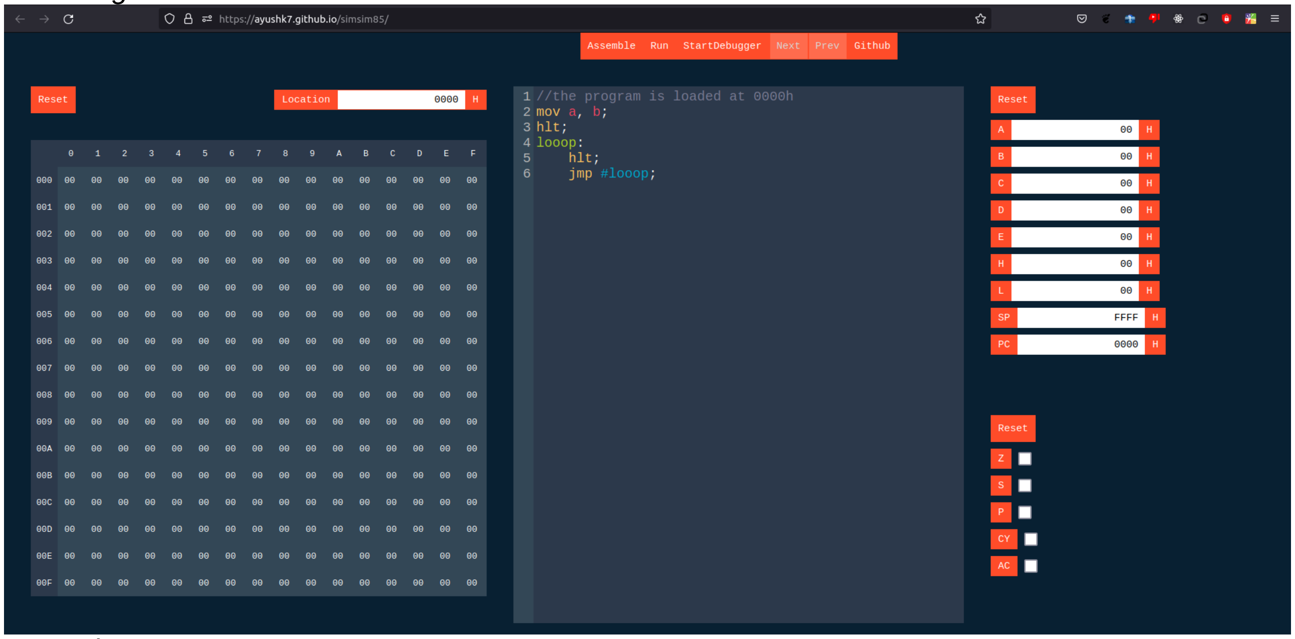
2.7 Assumptions and Dependencies

The Simulator is developed in JavaScript and therefore requires JavaScript to be enabled on the user's web browser. It requries ECMA version 6 enabled browser.

3. External Interface Requirements

3.1 User Interfaces

Main Page:



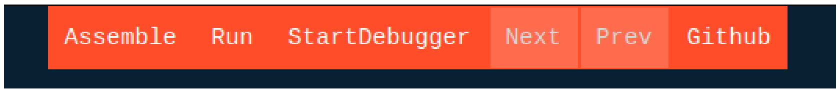
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
001	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
002	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
003	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
004	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
005	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
006	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
007	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
008	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
009	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

```
//the program is loaded at 0000h
mov a, b;
hlt;
looop:
    hlt;
    jmp #looop;
```

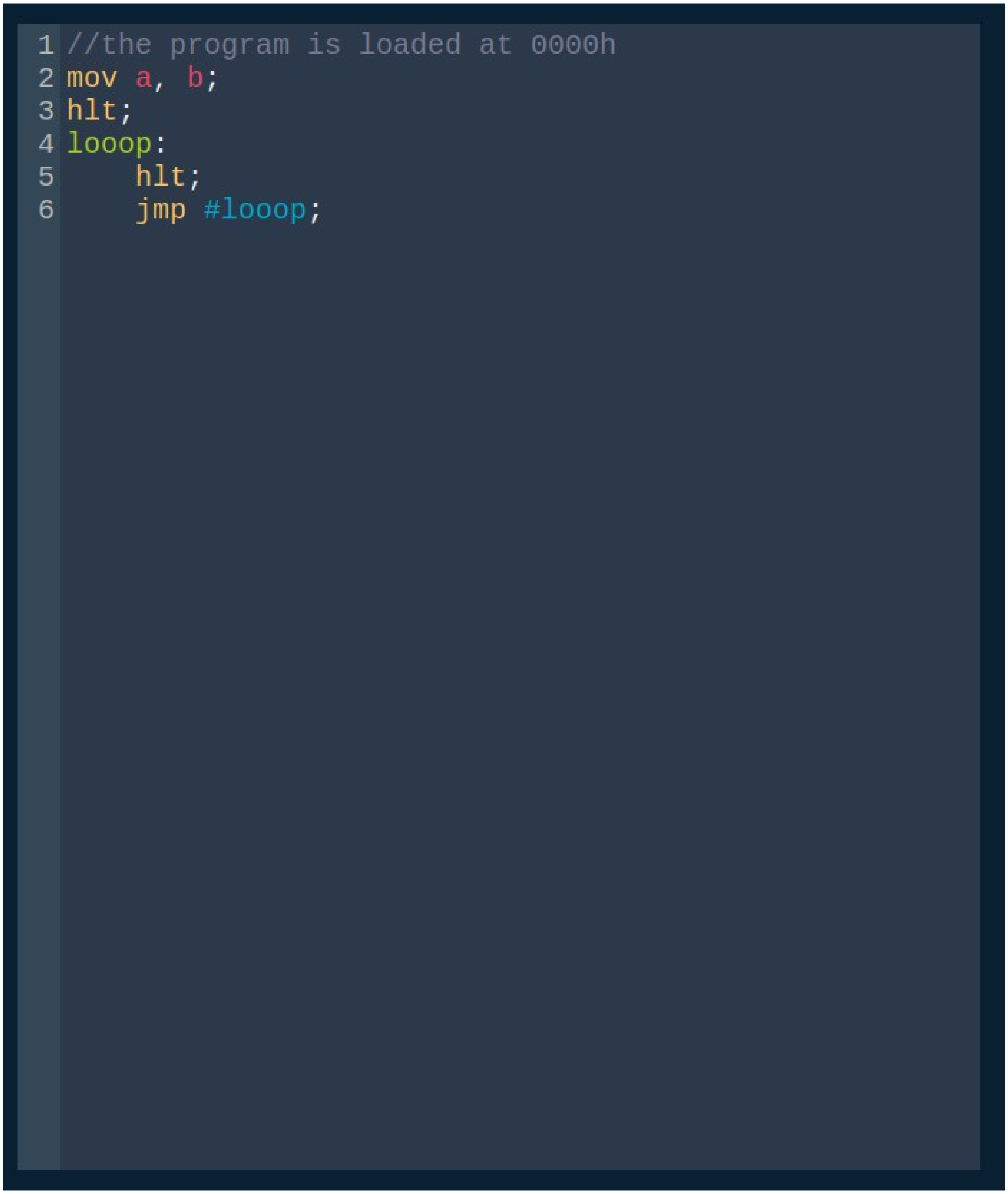
A	00	H
B	00	H
C	00	H
D	00	H
E	00	H
H	00	H
L	00	H
SP	FFFF	H
PC	0000	H

Z	☐
S	☐
P	☐
CY	☐
AC	☐

Control Bar:

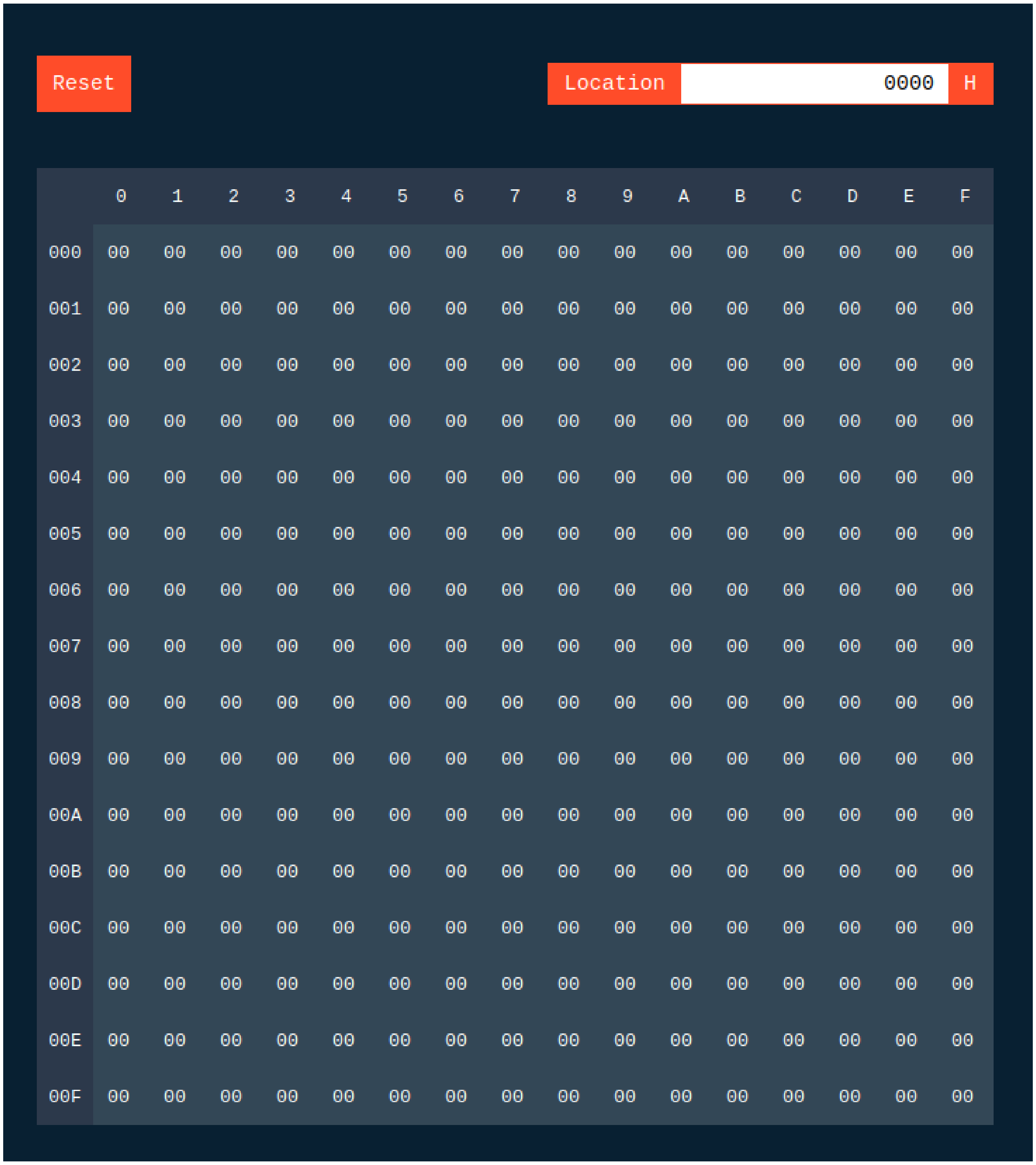


Text Editor:



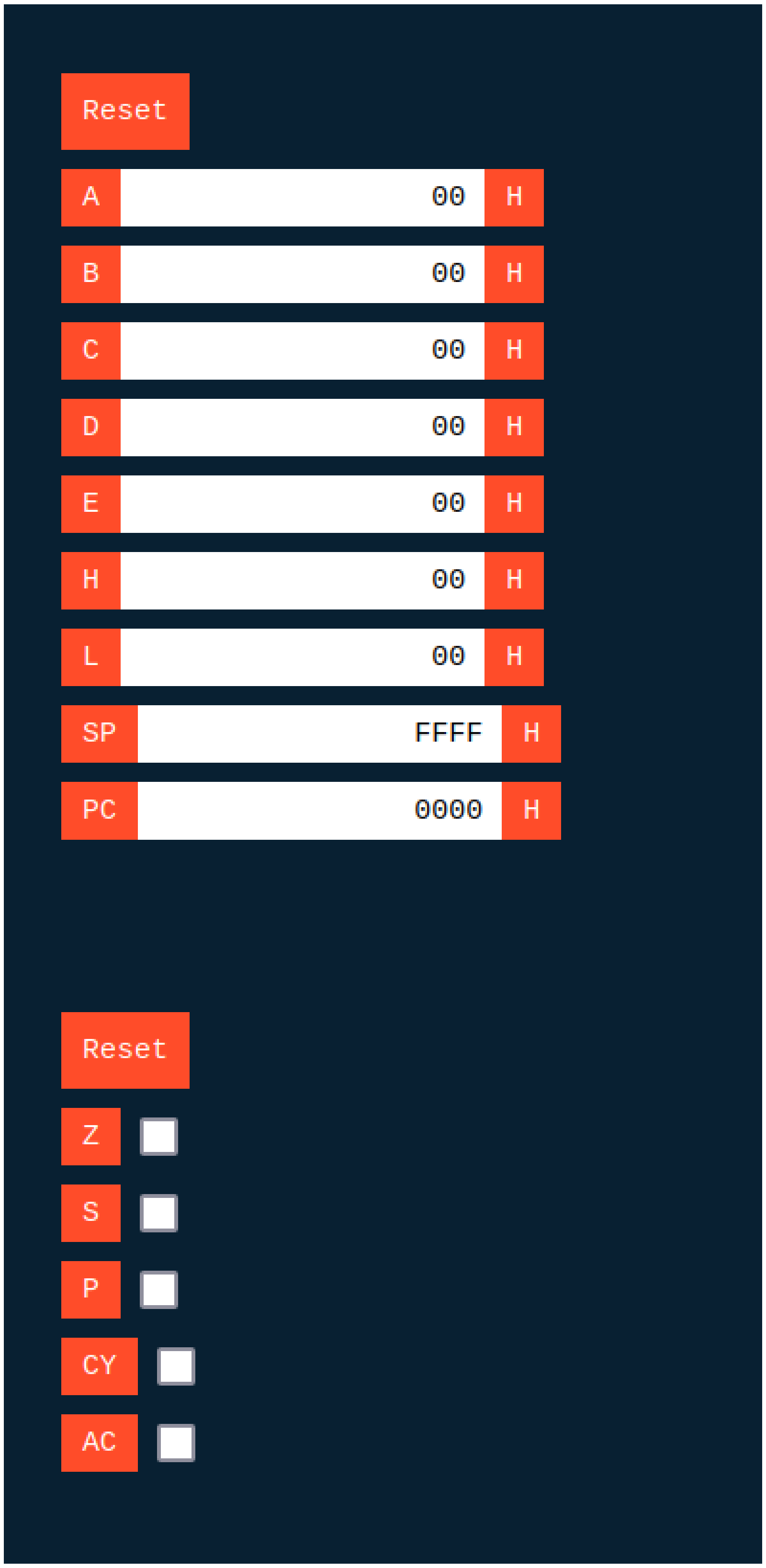
```
//the program is loaded at 0000h
mov a, b;
hlt;
looop:
    hlt;
    jmp #looop;
```

Memory Editor:

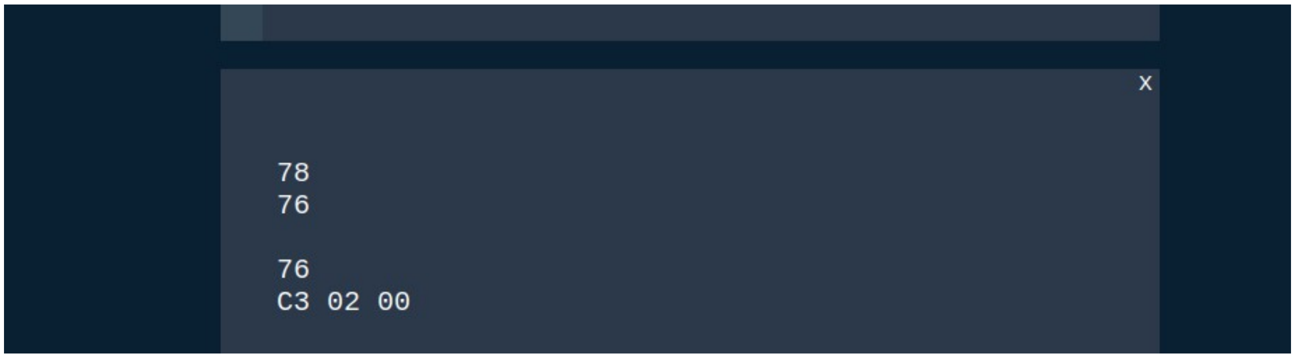


	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
000	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
001	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
002	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
003	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
004	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
005	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
006	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
007	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
008	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
009	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00A	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00B	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00C	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00F	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Register And Flags Editor:

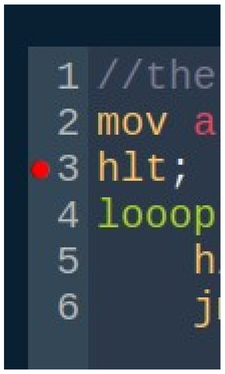

Reset
A 00 H
B 00 H
C 00 H
D 00 H
E 00 H
H 00 H
L 00 H
SP FFFF H
PC 0000 H
Reset
Z
S
P
CY
AC

Assembled Code:



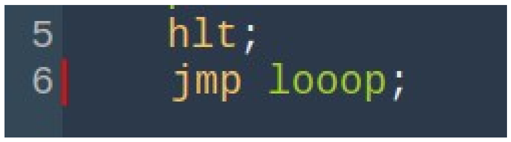
```
x

78
76

76
C3 02 00
```

Debug Signaling:



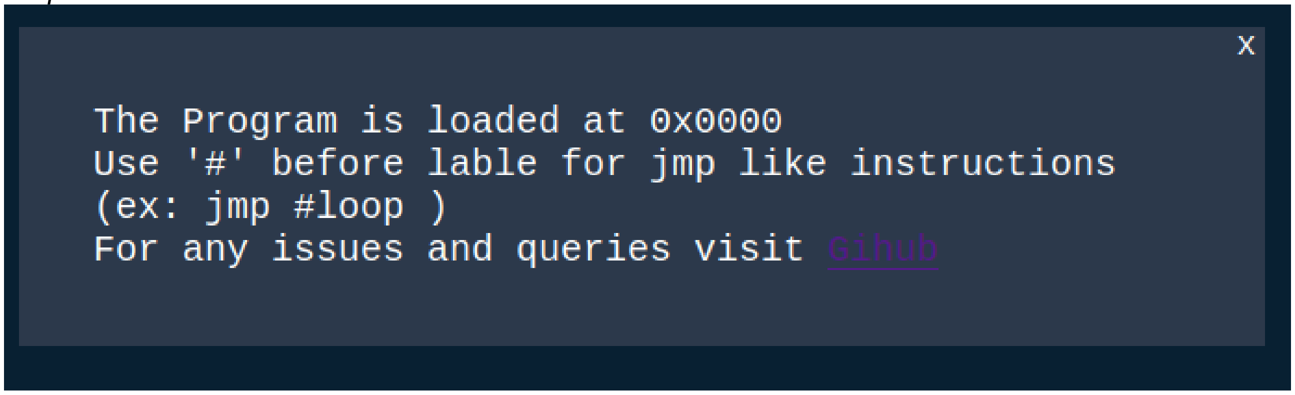
```
//the
mov a
hlt;
looop
    h
    j
```

Error Signaling:



```
    hlt;
    jmp looop;
```

Help:



```
X
The Program is loaded at 0x0000
Use '#' before lable for jmp like instructions
(ex: jmp #loop )
For any issues and queries visit Gihub
```

3.2 Hardware Interfaces

A PC which could run JavaScript enabled web browser is enough.

3.3 Software Interfaces

Requires JavaScript to be enabled on the user’s web browser. It requires ECMA version 6 enabled browser.

3.4 Communications Interfaces

It requires an internet connection to access the online simulator.

4. System Features

This section lists some of the key features of the simulator.

4.1 Modern Text Editor

1. Syntax Highlighting: Custom colors for instructions, numbers, opcodes, labels, jump labels, comments
2. Auto Indentation: Auto indents for label bodies

4.2 Inspect Memory and Register Status

1. Inspect the status of memory , registers and the flags.
2. Edit the status of memory, registers and the flags.

4.3 Assembler

1. Assembles the assembly code.

4.4 Runner

1. Runs the assembly output
2. Updates the status of registers, memory and the flags.

4.5 Debugger

1. Debug the program by running it step by step.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Requires modern web browser which can support ECMA script version 6.

5.2 Safety Requirements

Users can raise/report any issues/bugs they have encountered on the GitHub so that the developer can fix it.

5.3 Security Requirements

Requires modern web browser which can support ECMA script version 6.

5.4 Software Quality Attributes

Simulator can be used by both teachers and students.
However, users must already have a basic knowledge of Intel 8085 and assembly language before using it.

Appendix A: Glossary

- *ECMA:* European Computer Manufacturer's Association.
- ES6: ECMA script version 6 (JavaScript)
- Intel 8085: *The Intel 8085 is an 8-bit microprocessor produced by Intel and introduced in March 1976.*
- Registers: A processor register is a quickly accessible location available to a computer's processor.
- Flags: The FLAGS register is **the status register that contains the current state of a CPU**.
- RAM: Random Access Memory.
- Assembler: It is a program which converts assembly code to the binary.
- Debugger: It used to detect the logical errors in the program while executing it step by step.