

CISCO SIMULATOR

Manual

V 1.0

Group 8

Md Shahjalal
Tianyou Bao
Xuzheng Lu

Contents

1 Introduction	1
1.1 Debugging Panel	1
1.1.1 Register Indicators Area	1
1.1.2 Memory Area	2
1.1.3 Controller Area	2
2.2 Classic Panel	3
2 Operation	3
2.1 Writing Values to Registers	3
2.2 Writing Values to Memory	3
2.2.1 Using Memory Address Register and Memory Buffer Register	3
2.2.2 Modifying the Memory Area	4
2.3 Executing Instructions	5
2.3.1 Executing Instructions Step-by-Step	5
2.3.1 Executing Instructions Automatically	5
3 Instructions Reference	6
3.1 Load/Store Instructions	6
3.1.1 LDR Instruction	6
3.1.2 STR Instruction	6
3.1.3 LDA Instruction	7
3.1.4 LDX Instruction	7
3.1.5 STX Instruction	7
3.2 Other Instructions	7
3.2.1 Halt Instruction	7

1 Introduction

This simulator is a simulation of a Complex Instruction Set Computer (CISC). Two panels are designed for the simulator.

1.1 Debugging Panel

Debugging Panel displays all the information about the Registers, Indicators, and Memory in the computer and can be written manually.

The screenshot shows the 'CISC Machine Simulator' window. It is divided into several sections:

- Indicators**:
 - Registers**: R0, R1, R2, R3, each with a value field (0000000000000000), a base (x0000), and a write button (W).
 - Index Register**: IX1, IX2, IX3, each with a value field (0000000000000000), a base (x0000), and a write button (W).
 - Memory Register**: MAR, MBR, each with a value field (0000000000000000), a base (x0000), and a write button (W).
 - Basic Indicators**: PC, IR, CC, MFR, each with a value field (000000000000, 0000000000000000, 0000, 0000), a base (x000, x0000, 0, 0), and a write button (W).
- Memory**: A table with columns: Address, Value, Hex Value, and Assemble Code. It shows memory locations from x0000 to x000C.
- Controller**: Contains an 'Instruction' field (0000000000000000) and buttons: Reload, Save, IPL, Program 1, Program 2, Floating Test, Vector Test, Auto Run, Single Run, Pause, Stop, and Restart.

The panel is divided into three parts:

1.1.1 Register Indicators Area

This close-up shows the 'Indicators' section of the simulator:

- Registers**: R0, R1, R2, R3. Each has a value field (0000000000000000), a base (x0000), and a write button (W).
- Index Register**: IX1, IX2, IX3. Each has a value field (0000000000000000), a base (x0000), and a write button (W).
- Memory Register**: MAR, MBR. Each has a value field (0000000000000000), a base (x0000), and a write button (W).
- Basic Indicators**: PC, IR, CC, MFR. Each has a value field (000000000000, 0000000000000000, 0000, 0000), a base (x000, x0000, 0, 0), and a write button (W).

The Register Indicators display the values of all kinds of registers.

- Click the 'W' button to manually modify the value of a register.
- Hexadecimal values are shown on the right.

Type	Size(bits)	Number	Description
R0...R3	16	4	General-Purpose Register
IX1...IX3	16	3	Index Register
MAR	16	1	Memory Address Register
MBR	16	1	Memory Buffer Register
PC	12	1	Program Counter
IR	16	1	Instruction Register
CC	4	1	Condition Code
MFR	4	1	Machine Fault Register

1.1.2 Memory Area

Address	Value	Hex Value	Assemble Code
x0000	0000000000000000	x0000	null
x0001	0000000001100100	x0064	null
x0002	0000000000000000	x0000	null
x0003	0000000000000000	x0000	null
x0004	0000000000100010	x0022	null
x0005	0000000000000000	x0000	null
x0006	0000000001100100	x0064	null
x0007	0000000000000000	x0000	null
x0008	0000010001000010	x0442	LDR 0,1,2
x0009	0000000000000000	x0000	null
x000A	0000000000000000	x0000	null
x000B	0000000001001101	x004D	null
x000C	0000000000000000	x0000	null

The Memory Area shows the address, the value, the Hexadecimal value, and the Assemble Code of each line on memory.

- The memory address pointed by the Program Counter will be highlighted.
- Double click to manually modify the binary value of a memory row.

1.1.3 Controller Area

Controller					
Instruction	0000000000000000	Reload	Save		
IPL	Program 1	Program 2	Floating Test	Vector Test	
Auto Run	Single Run	Pause	Stop	Restart	

The Controller Area integrates all function buttons and the instruction input box.

Button	Description
Reload	Initialize the values
Save	Save inputs
IPL	Pre-load a program
Auto Run	Run instructions automatically
Single Run	Run instructions step by step
Pause	Pause the machine
Stop	Stop the machine
Restart	Restart the machine

2.2 Classic Panel

The appearance and operational logic of the **Classic Panel** emulate the PDP-8 computer. Users will use switches to input and lights for indication.

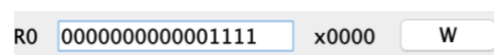
The **Classic Panel** has not been finished yet and will be released in the next version.

2 Operation

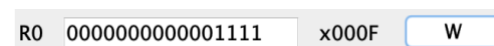
2.1 Writing Values to Registers

Following the steps below to write a value to a register.

Step 1: Input a value into the box



Step 2: Click the 'W' button at right to write the value to the register



Step 3: Done! The value will be written to the Register.

Error handling:

- Input too long: Remove the excess bits from the left
- Input too short: Add zeros from the left
- Input is not binary: Pop a Error window.

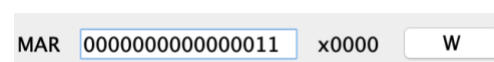


2.2 Writing Values to Memory

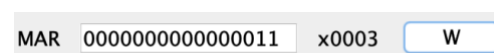
Two methods are acceptable to write a value to the Memory.

2.2.1 Using Memory Address Register and Memory Buffer Register

Step 1: Input a value into the MAR box



Step 2: Click the 'W' button of MAR



Step 3: Input a value into the MBR box

MBR x0000

Step 4: Click the 'W' button of MBR

MBR x0000

Step 5: Done! The value of MAR will be written to the Memory, and the MAR will automatically change to the next address.

MAR x0004
 MBR x0000

Memory			
Address	Value	Hex Value	Assemble Code
x0000	0000010010000100	x0484	LDR 0,2,4
x0001	0000000001100100	x0064	null
x0002	0000000000000000	x0000	null
x0003	0000100010000100	x0884	STR 0,2,4


2.2.2 Modifying the Memory Area

Step 1: Double click the memory row that needs to modify

Memory			
Address	Value	Hex Value	Assemble Code
x0000	0000010010000100	x0484	LDR 0,2,4
x0001	0000000001100100	x0064	null
x0002	0000000000000000	x0000	null
x0003	0000100010000100	x0884	STR 0,2,4
x0004	000000000100010	x0022	null
x0005	0000000000000000	x0000	null
x0006	0000000001100100	x0064	null

Step 2: An input window as the following will pop up. Input the value that needs to write to the memory

Input

 Input binary value

Step 3: Click the 'OK' button, and then the value will be written to the Memory.

Memory			
Address	Value	Hex Value	Assemble Code
x0000	0000010010000100	x0484	LDR 0,2,4
x0001	0000000001100100	x0064	null
x0002	0000000000000000	x0000	null
x0003	0000100010000100	x0884	STR 0,2,4
x0004	000000000100010	x0022	null
x0005	0000110101100011	x0D63	LDA 1,1,3,1
x0006	0000000001100100	x0064	null

2.3 Executing Instructions

Instruction can be executed step-by-step or automatically.

2.3.1 Executing Instructions Step-by-Step

Step 1: Store an instruction to the Memory

Memory			
x001A	0000000000000000	x0000	null
x001B	0000000001100100	x0064	null
x001C	0000000000000000	x0000	null
x001D	0000000000000000	x0000	null
x001E	1010010001010100	xA454	LDX 1,20
x001F	1010010010010110	xA496	LDX 2,22

Step 2: Write the address of the instruction to the Program Counter (PC)

PC	000000011110	x01E	W
----	--------------	------	---

Step 3: Click the 'Single Run' button, and then the instruction will be executed.

- The Program Counter will automatically point to the next address of Memory.
- The Instruction Register will store the last executed instruction.

Memory Register		Basic Indicators	
MAR	0000000000010100 x0014 W	PC	000000011111 x01F W
MBR	0000000001010000 x0050 W	IR	1010010001010100 xA454 W
		CC	0000 0 W
		MFR	0000 0 W
Memory			
x001B	0000000001100100	x0064	null
x001C	0000000000000000	x0000	null
x001D	0000000000000000	x0000	null
x001E	1010010001010100	xA454	LDX 1,20
x001F	1010010010010110	xA496	LDX 2,22
x0020	1010010011111000	xA4F8	LDX 3,24,1

2.3.1 Executing Instructions Automatically

Step 1: Store instructions to the Memory

Memory			
x001E	1010010001010100	xA454	LDX 1,20
x001F	1010010010010110	xA496	LDX 2,22
x0020	1010010011111000	xA4F8	LDX 3,24,1
x0021	0000011100001011	x0708	LDR 3,0,11
x0022	0000010000101011	x042B	LDR 0,0,11,1
x0023	0000010111000011	x05C3	LDR 1,3,3
x0024	0000011011100011	x06E3	LDR 2,3,3,1
x0025	0000101000000001	x0A01	STR 2,0,1
x0026	1010100011010000	xA8D0	STX 3,16
x0027	0000110100000100	x0D04	LDA 1,0,4
x0028	0000000000000000	x0000	null

Step 2: Write the address of the **starting** instruction to the Program Counter (PC)

PC	000000011110	x01E	W
----	--------------	------	---

Step 3: Click the 'Auto Run' button, and then the instructions will be executed automatically.

- The Program Counter will automatically point to the next address of Memory after an instruction being executed.
- All the indicators will be continuously updated while the program is running.

The screenshot shows a simulator interface with the following sections:

- Registers:** R0 (000000000110101, x0035, W), R1 (000000000000100, x0004, W), R2 (0000000001100100, x0064, W), R3 (0000000001001101, x004D, W).
- Index Register:** IX1 (0000000001010000, x0050, W), IX2 (0000000001011111, x005F, W), IX3 (0000000001100100, x0064, W).
- Memory Register:** MAR (0000000000000100, x0004, W), MBR (000000000100010, x0022, W).
- Basic Indicators:** PC (000000101000, x028, W), IR (0000110100000100, x0D04, W), CC (0000, 0, W), MFR (0000, 0, W).
- Memory:** A list of memory addresses and their contents. Address x0024 contains 0000011011100011. Address x06E3 contains LDR 2,3,1.
- Controller:** Includes buttons for IPL, Program 1, Program 2, Floating Test, Vector Test, Auto Run, Single Run, Pause, Stop, Restart, Reload, and Save. The Instruction field shows 0000000000000000.

Step 4: Click the 'Pause' button or the 'Stop' button to stop the program.

3 Instructions Reference

3.1 Load/Store Instructions

The basic instructions to load/store values from/to Registers or Memory. The binary instruction code format is as follows:

Opcode	R	IX	I	Address
0	5	6 7	8 9	1 1
			0 1	5

- Opcode:** 6 bits Specifies the instruction
- R:** 2 bits Specifies the General-Purpose Register
- IX:** 2 bits Specifies the Index Register
- I:** 1 bit Specifies Indirect Addressing
If I =1, indirect addressing; otherwise, no indirect addressing.
- Address:** 5 bits Specifies the location

3.1.1 LDR Instruction

- Instruction: LDR r, x, address[, I]
- Octal-Opcode: 01
- Binary-Opcode: 000001
- Function: Loads Register from Memory

3.1.2 STR Instruction

- Instruction: STR r, x, address[, I]

Octal-Opcode: 02
 Binary-Opcode: 000010
 Function: Stores Register to Memory

3.1.3 LDA Instruction

Instruction: LDA r, x, address[, I]
 Octal-Opcode: 03
 Binary-Opcode: 000011
 Function: Loads Register with Address

3.1.4 LDX Instruction

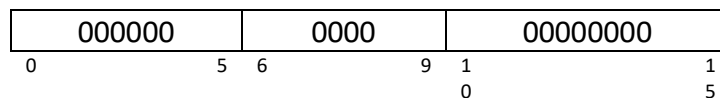
Instruction: LDX x, address[, I]
 Octal-Opcode: 41
 Binary-Opcode: 100001
 Function: Loads Index Register from Memory

3.1.5 STX Instruction

Instruction: STX x, address[, I]
 Octal-Opcode: 42
 Binary-Opcode: 100010
 Function: Stores Index Register to Memory

3.2 Other Instructions

3.2.1 Halt Instruction



Instruction: Halt
 Octal-Opcode: 00
 Binary-Opcode: 000000
 Function: Stops the machine