

CISC Simulator Design Document

Rev 1.0

Group 2

Jonathan Pritchett

Ziwei Li

Haorong Xiao

Siyuan Zhang

Table of Contents

Table of Contents	1
Description	2
Data Representation	2
Registers	2
Memory	2
Instructions	3
Front Panel GUI	3
Figures	4
UI Mockup	4
Component Diagram	5
Class Diagram	6

Description

The CISC Simulator is a program that creates a simple simulator of a rudimentary computer. In its current iteration, this simulated computer consists of a set of registers, a simple memory, a small set of instructions, and a graphical user interface to serve as the computer front panel.

Data Representation

Data in the computer shall be stored in 16-bit words. To represent this, a class "Word" will be created which stores a 16-bit word as a 16 char array. This class will have accessor methods allowing for easy translation of this data into ints and strings and will allow for accessing of individual and subsets of bits.

Registers

In this first iteration of the CISC simulator, the following 11 16-bit registers are used.

R0...R3	4 General-purpose registers
PC	Program counter
IR	Instruction Register
MAR	Memory Address Register
MBR	Memory Buffer Register
X1...X3	3 Index Registers

Each register consists of a single Word object. All of the registers are collectively stored within a single class, "Registers", which may be instantiated once and shared among the components of the simulator.

Memory

The Memory system for this version consists of a single class that contains an array of 2048 "Word" objects. The memory is instantiated once and shared throughout the simulator.

Instructions

Five instructions are currently implemented within the simulator.

Instruction	OpCode	Description
LDR	01	Load register from memory
STR	02	Store register to memory
LDA	03	Load register with address
LDX	41	Load index register from memory
STX	42	Store index register to memory

Front Panel GUI

The front panel GUI provides the user with a way to view the contents of all of the registers and memory locations used by the simulator. It also allows the user to override any of these values manually and to run the simulator from its current state.

Figures

UI Mockup

Simple CPU Simulator

Registers

PC: 2752

IR: 1000100010110

R0: 45

R1: 782

R2: 7852

R3: 783

X1: 123

X2: 654

X3: 457

MAR: 3123

MBR: 7839

MFR: 4537

Memory

Address: 16

Value: 2146

☐ Override Locked

Override All Values

Load Test Program

☐ AutoRun

Step

Input Mode:

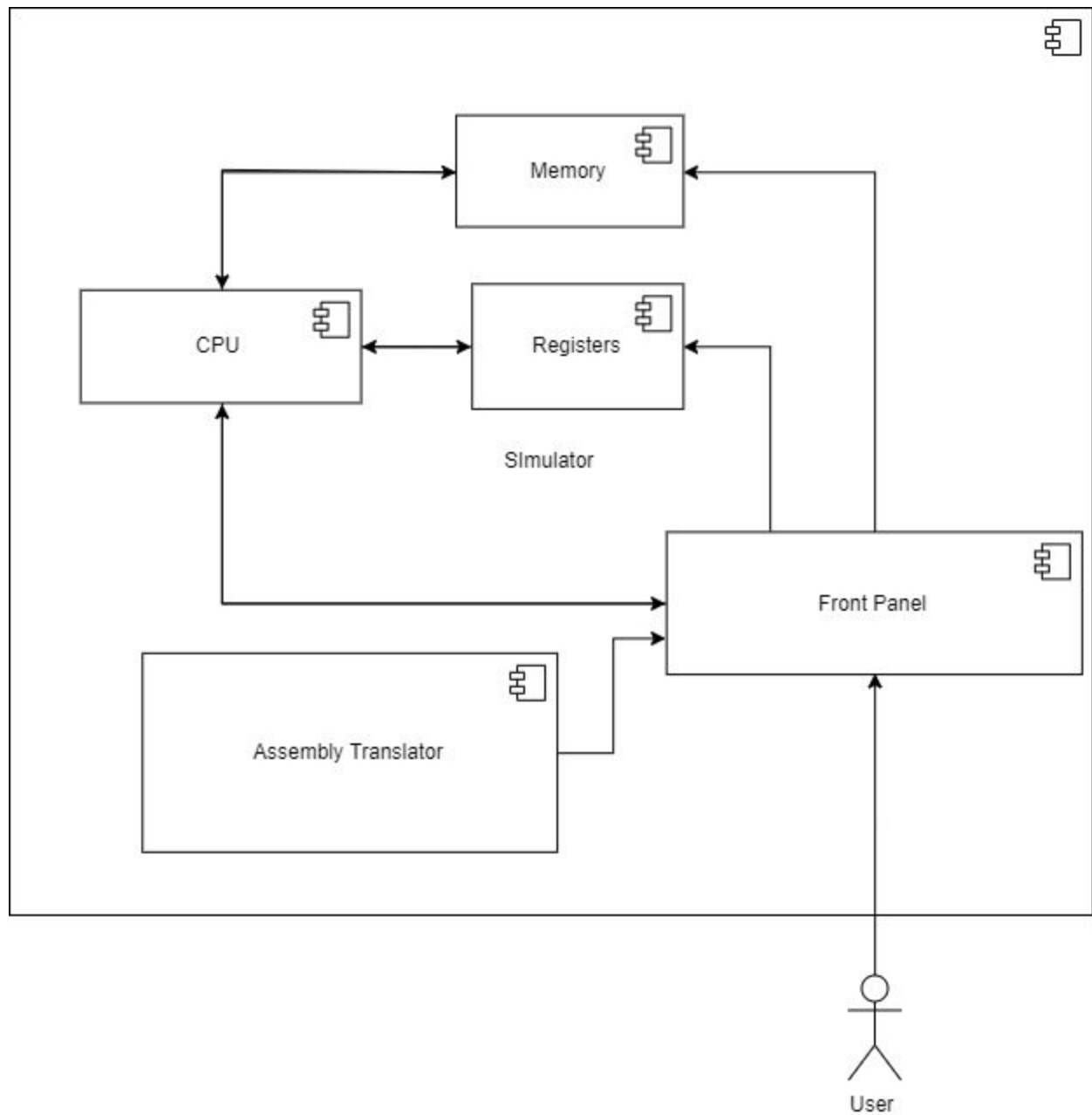
☐ Binary

☐ Octal

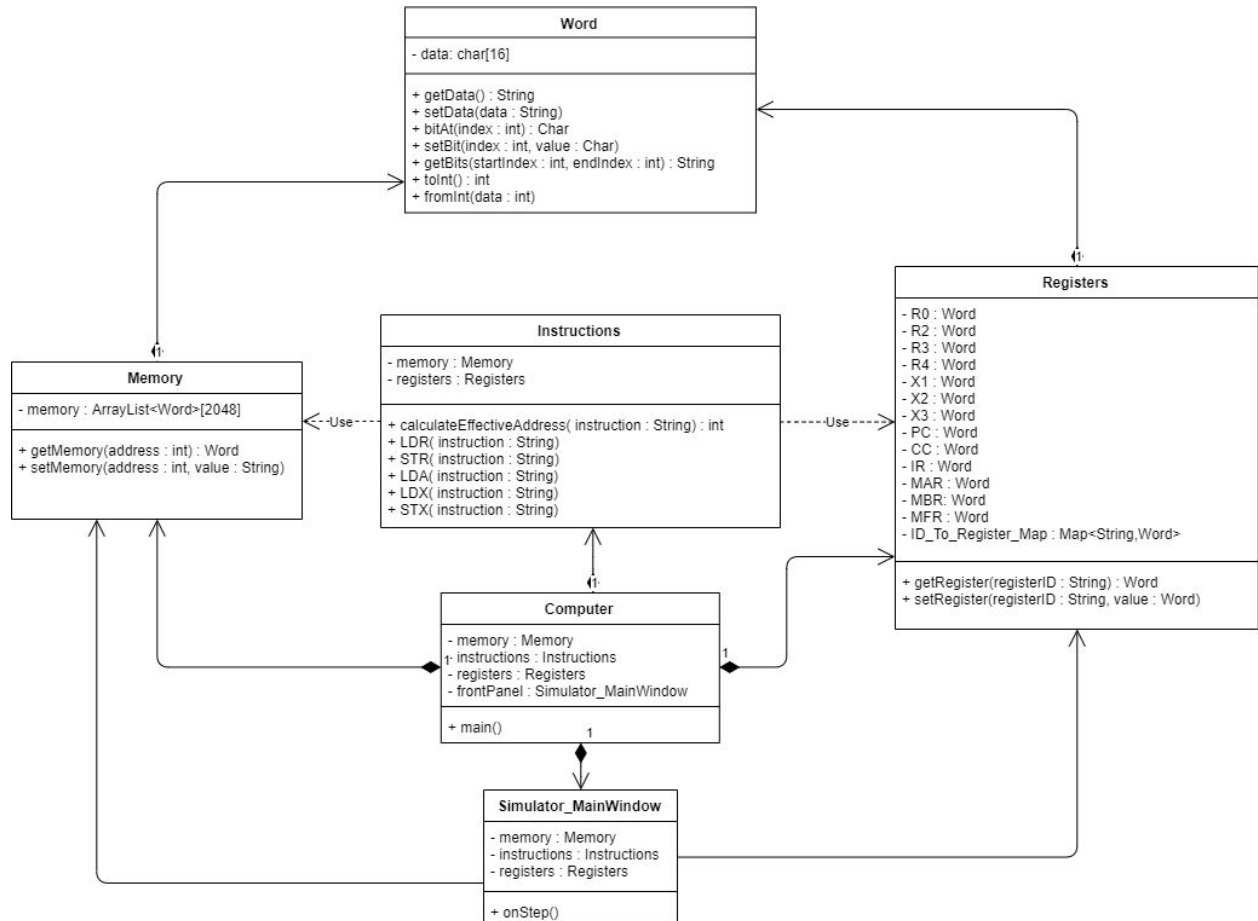
☒ Decimal

☐ Hex

Component Diagram



Class Diagram



Test Program

We initialize the test program by pressing the “load test program” button in the lower right corner.

We set up 4 memory addresses and 2 index registers:

- memory address 31 stores integer value 29
- memory address 30 stores integer value 31
- memory address 29 stores integer value 30
- index register X1 stores integer value 1 and index register X2 stores integer value 2.

Then, we have stored 5 instructions in memory:

- (1) Load register from memory: 000001 00 01 1 11110. (EA = memory[31] = 29)
This instruction loads effective address 29 by indexing and indirecting to register R0, which stores 30 representing by '11110'.

The screenshot shows a CISC Simulator window with a light green border. Inside, there are several input fields for registers and memory, along with control buttons. The registers are labeled R0 through R3, X1 through X3, PC, CC, IR, MAR, MBR, and MFR. The R0 register is highlighted with a red rectangle and contains the value 0000000000011110. The other registers contain various binary values. The PC register contains 1011. The CC register contains 0000000000000000. The IR register contains 0000010001111110. The MAR register contains 11101. The MBR register contains 0000000000011110. The MFR register contains 0000000000000000. Below the registers, there is a section labeled 'Memory' with two input fields: 'Address' and 'Value', both containing 0000000000000000. To the right of the memory fields, there are three buttons: 'Override All Values', 'Load test program', and 'Step'. There is also a checkbox labeled 'AutoRun?' which is currently unchecked. The 'Input Mode' dropdown menu is set to 'binary' and the 'Instruction' dropdown menu is set to 'Item 1'.

Register	Value
R0	0000000000011110
R1	0000000000000000
R2	0000000000000000
R3	0000000000000000
X1	0000000000000001
X2	0000000000000010
X3	0000000000000000
PC	1011
CC	0000000000000000
IR	0000010001111110
MAR	11101
MBR	0000000000011110
MFR	0000000000000000

Memory	
Address	0000000000000000
Value	0000000000000000

Buttons: Override All Values, Load test program, Step, AutoRun? (unchecked)

(2) Store register to memory: 000010 00 00 0 11100. (EA = 28)

This instruction stores the value of R0 to the effective address 28. Then, the content of memory address 28 become 30.

The screenshot shows a CISC Simulator window with a light green border. Inside, there are several input fields for registers and memory. The registers listed are R0, R1, R2, R3, X1, X2, X3, PC, CC, IR, MAR, MBR, and MFR. Each register has a corresponding binary value field. For example, R0 contains 0000000000011110. The Memory section is highlighted with a red rectangle and contains fields for Address (11100) and Value (0000000000011110). To the right of the registers, there are dropdown menus for 'Input Mode' (set to 'binary') and 'Instruction' (set to 'Item 1'). At the bottom right, there are buttons for 'Override All Values', 'Load test program', 'Lock Override' (with a checkbox), 'AutoRun?' (with a checkbox), and 'Step'.

Register	Value
R0	0000000000011110
R1	0000000000000000
R2	0000000000000000
R3	0000000000000000
X1	0000000000000001
X2	0000000000000010
X3	0000000000000000
PC	1011
CC	0000000000000000
IR	000010001111110
MAR	11101
MBR	0000000000011110
MFR	0000000000000000

Memory

Field	Value
Address	11100
Value	0000000000011110

Input Mode: binary
Instruction: Item 1

Buttons: Override All Values, Load test program, Lock Override, AutoRun?, Step

- (3) Load register with address : 000011 01 01 1 11101. (EA = memory[30] = 31)
This instruction loads the content in effective address to register R1 by indexing and indirecting. The effective address here is 31. Therefore, 31 is stored in R1.

The screenshot shows the CISC Simulator interface with the following components:

- Registers:** R0 (0000000000011110), R1 (11111), R2 (0000000000000000), R3 (0000000000000000), X1 (0000000000000001), X2 (0000000000000010), X3 (0000000000000000).
- PC (Program Counter):** 1101
- CC (Condition Codes):** 0000000000000000
- IR (Instruction Register):** 0000110101111101
- MAR (Memory Address Register):** 11100
- MBR (Memory Buffer Register):** 0000000000011110
- MFR (Memory Function Register):** 0000000000000000
- Memory:**
 - Address: 11100
 - Value: 0000000000011110
- Input Mode:** binary
- Instruction:** Item 1
- Buttons:** Override All Values, Load test program, Lock Override, AutoRun?, Step.

- (4) Load index register from memory: 101001 00 10 1 11100. (EA = memory[30] = 31)
This instruction loads the content in effective address to index register X2 by indexing and indirecting. Consequently, the content of Index Register 2 becomes 29.

The screenshot shows a CISC Simulator window with a green title bar. The interface is divided into several sections:

- Registers:** A list of registers on the left with their corresponding binary values in text boxes on the right:
 - R0: 0000000000011110
 - R1: 11111
 - R2: 0000000000000000
 - R3: 0000000000000000
 - X1: 0000000000000001
 - X2: 0000000000011101 (highlighted with a red rectangle)
 - X3: 0000000000000000
- Control and Status:** Below the registers are fields for PC (1110), CC (0000000000000000), IR (1010010010111100), MAR (11111), MBR (0000000000011101), and MFR (0000000000000000).
- Memory:** A section at the bottom left with 'Address' (11100) and 'Value' (0000000000011110) fields.
- Controls:** On the right side, there are buttons and checkboxes:
 - 'Input Mode' dropdown set to 'binary'.
 - 'Instruction' dropdown set to 'Item 1'.
 - 'Override All Values' button.
 - 'Load test program' button.
 - 'Lock Override' checkbox (unchecked).
 - 'AutoRun?' checkbox (unchecked).
 - 'Step' button.

- (5) Store index register to memory: 101010 00 10 0 00000. (EA = 29)
This instruction stores the value of index register 2 to memory[29].

CISC Simulator Design Document
Rev 1.0

The screenshot displays a CISC Simulator interface with a light green title bar and a grey main area. On the left, a list of registers and memory locations is shown, each with a corresponding binary value in a text box. The registers are R0, R1, R2, R3, X1, X2, X3, PC, CC, IR, MAR, MBR, and MFR. Below these is a 'Memory' section with 'Address' and 'Value' fields. The 'Address' field is highlighted with a red rectangle and contains the value '11101'. The 'Value' field contains '000000000011101'. On the right side, there are controls for 'Input Mode' (set to 'binary'), 'Instruction' (set to 'Item 1'), and buttons for 'Override All Values', 'Load test program', 'AutoRun?' (with a checkbox), and 'Step'.

Register	Value
R0	0000000000011110
R1	11111
R2	0000000000000000
R3	0000000000000000
X1	0000000000000001
X2	0000000000011101
X3	0000000000000000
PC	1111
CC	0000000000000000
IR	1010100010000000
MAR	11101
MBR	0000000000011110
MFR	0000000000000000

Memory

Address	Value
11101	000000000011101

Input Mode: binary
Instruction: Item 1

Buttons: Override All Values, Load test program, AutoRun? (checkbox), Step