**Computer Simulator Simple Documentation**

# Introduction:

This document describes how to use our simulator in simple steps. The document is split into five sections:

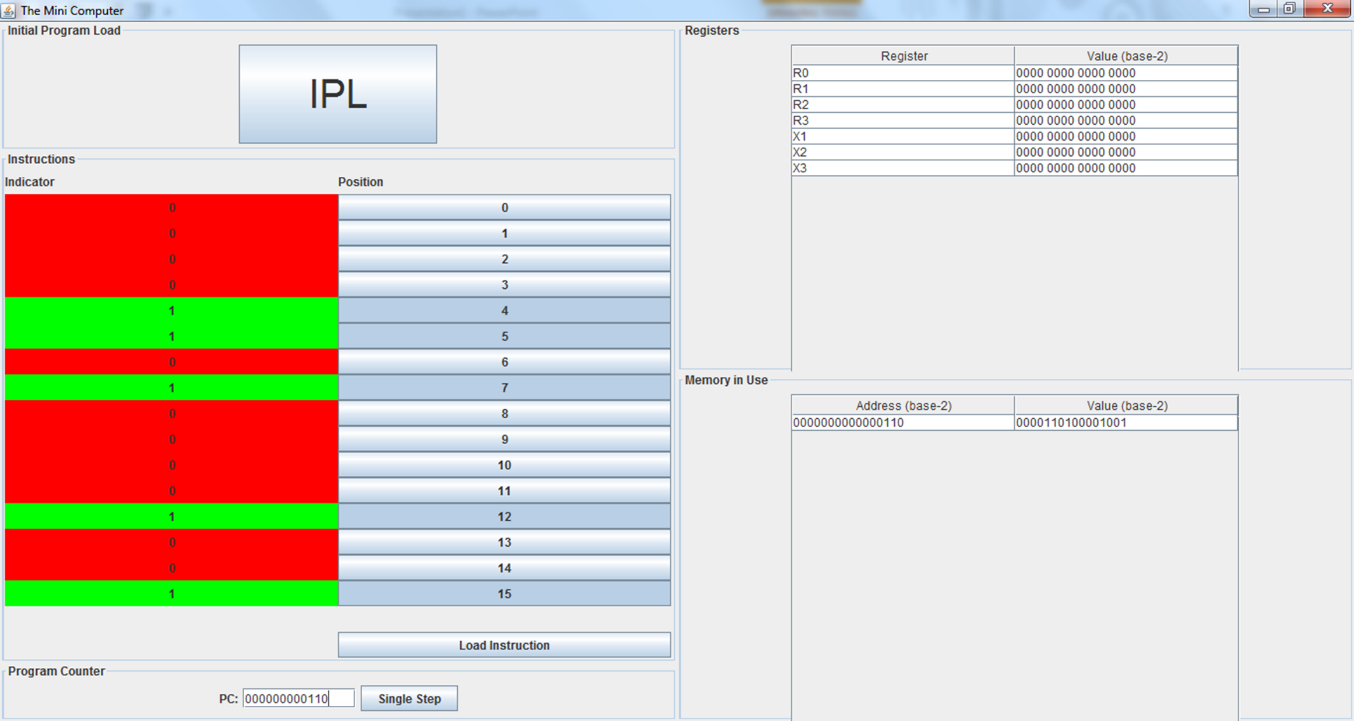
1. Introduction
2. How to launch the Simulator
3. What is the console layout
4. How to operate the simulator from the console.
5. User’s Bit-Instruction Manual

This document will evolve to include additional steps as the simulator project progresses through the four planned phases of agile development.

# How to launch the Simulator?

1. Please make sure JAVA 1.8 or later is installed.
2. Download the JAR file and boot program file
3. The boot program file and the JAR file need to be in the same directory
4. Double-click the JAR file to launch the console

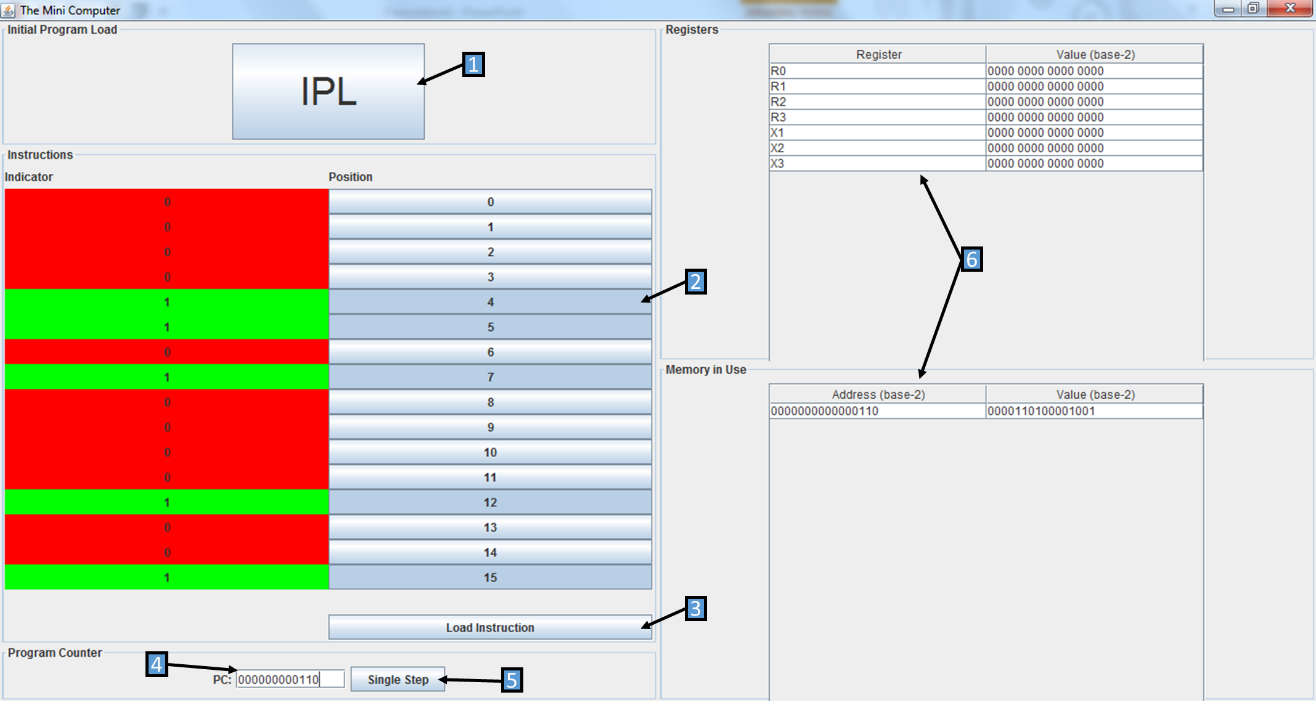
# What is the console layout?



Our mini-computer has a graphical user interface (GUI) console. It is split into five panels:

1. Initial Program Load (IPL) – Contains the initial booting sequence options.
2. Instructions – the console allows 16 bits of information to be loaded into the computer one instruction at a time from the console.
3. Program Counter – Allows the user to specify the program counter to single step
4. Registers – displays the register and the value contained within the register
5. Memory in Use – displays the address and value stored in memory in base-2 notation.

# How to operate the simulator from the console.



Step 1: From the console the minicomputer simulator can be launched by activating (clicking) the IPL switch. This IPL switch executes the boot program, which loads values into memory and registers (see label 6 for memory and register values).

Step 2: Instructions to the minicomputer can be entered into the instruction panel by activating bit-toggles (i.e. clicking buttons enumerated from 0-15). A green light indicates the corresponding bit position’s inclusion into the instruction. The user manual explains the available commands and basic bit-instruction usage.

Step 3: Activating the ‘Load Instruction’ will load the operator’s instruction into the minicomputer. Note: The default location for console instructions is “000000000110”

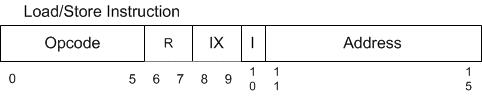
Step 4: From the Program Counter Panel the operator can elect to execute step-by-step instructions that have been loaded into the computer. The PC should contain the address of the instruction to be executed.

Step 5: Activate the Single Step button to execute the instruction.

Step 6: Observe the effect of your bit instruction in the minicomputer’s Registers and Memory.

# User’s Bit-Instruction Manual

Instructions for the minicomputer and the accompanying options for that instruction are broken down by the bit-position ranges in the figures and tables below. The Opcodes are the basic instruction options for the minicomputer. The general purpose register (R) will be used if an Opcode indicates a Load from register, or Store to register instruction. The index register (IX) will be used if an Opcode specifies a Load from index, or Store to index instruction. The indirect addressing indicator (I) tells the Opcode how to interpret the instruction’s address field. If I=0 then the Opcode will read the address value directly. However, if I=1 then the Opcode field will be a pointer to another address location.



The basic interpretation of the bit-wise instruction can be summarized in the table below.

|  |  |  |
| --- | --- | --- |
| **Field** | **#Bits** | **Description** |
| Opcode | 6 | Specifies one of 64 possible **instructions**;  Not all may be defined in this project |
| R | 2 | Specifies one of four general purpose **registers**; may be referred to by R0 – R3 |
| IX | 2 | Specifies one of three **index registers**; may be referred to by X1 – X3. O value indicates no indexing. |
| I | 1 | If I =1, specifies **indirect addressing indicator**; otherwise, no indirect addressing. |
| Address | 5 | Specifies one of 32 address **locations in memory** |

The currently available instruction OpCodes are as defined below.

|  |  |  |
| --- | --- | --- |
| **OpCode2** | **Instruction\*** | **Description** |
| 000001 | LDR r, x, address[,I] | Load Register From Memory |
| 000010 | STR r, x, address[,I] | Store Register To Memory |
| 000011 | LDA r, x, address[,I] | Load Register with Address |
| 100001 | LDX x, address[,I] | Load Index Register from Memory |
| 100010 | STX x, address[,I] | Store Index Register to Memory |

\*[,I] = the Boolean indicator for using indirect addressing