

Sincy Technical Manual

For Sincy Single Cycle MIPS Simulator

The screenshot displays the Sincy Single Cycle MIPS Simulator interface, which is divided into three main windows:

- Sincy Control Panel:** Contains buttons for "Start Execution", "Resume", "Pause", and "Step". It also has "View Registers", "View Data Memory", and "View Instruction Memory" buttons. A "Speed (In Seconds):" field is set to ".75". At the bottom, a "Finished" button is visible.
- Instruction Memory:** A table showing the instruction memory contents. The columns are "Address", "Instruction", and "Binary Value". The address range is from 0 to 35, and all instructions are currently 0.
- Registers:** A table showing the register file. The columns are "Register(b)", "Register", "Data(b)", and "Data(d)". The registers are numbered 0 to 31, and all data values are currently 0.

The "Enter Instructions:" window is also visible, showing a list of MIPS instructions entered for execution:

```
addi $s1, $s1 200
addi $s2, $s1 300
add $s3, $s1 $s2
addi $s4, $s3 126700
sub $s5, $s4 $s3
addi $s6, $zero 255
and $s7 $s2, $s6
or $t8 $s7 $s6,
or $t8 $s4, $t8
mul $s1, $s1, $t8
addi $s0 $zero 64
mul $t9 $s1 $s0
addi $t2 $zero 2
srl $t7 $t9 2
sll $t6 $t7 2
```

Patrick Martinez

Contents

Warnings	2
Introduction	3
Instruction Guide	4
Getting Started	4
Downloading Sincy	4
Opening Program and Windows	4
Using Sincy	5
Running an Arithmetic Program	5
Running a Loop.....	6
Changing Speed	8
Using Step-By-Step Execution	8
Loading .asm Files from Command Line	8
Conclusion	10



Warnings

- Sincy Simulator is provided “as is” and “with all faults.”
- The developer makes no guarantees of any kind concerning the safety, suitability, typographical errors or other harmful components of this software.
- There are no guarantees that this program will work on your system.
- You are solely responsible for the protection and backup of your data.
- The developer will not be liable for any damages one may suffer in connection when using, modifying or distributing this software.
- Due to limited resources and the nature of this program (the end user being able to type in commands) we cannot possibly test every single possible combination of commands. As such, untested, buggy and undocumented situations may be encountered. All bug reports may be sent to patrickmartinez@txstate.edu.
- We do not recommend using this program as a text editor to write code. There is no save function. We advise to write code in a separate text editor and to copy and paste it into the simulator.
- The only commands currently supported are J, BEQ, ADD, ADDI, SUB, SW, LW, SLL, SRL, MUL AND and OR. Any other assembly commands will cause the program to crash.

Introduction

Thank you for considering the download and use of the Sincy Single Cycle MIPS Simulator! We worked very hard to bring you this program and hope that it supplements your understanding of the MIPS process lifecycle. This program is intended for students who are taking or are interested in taking a computer architecture course. In order to use the software, one must be familiar with basic assembly language syntax and its use. The only thing you need to begin to use this software is a computer capable of running a simple Java Applet. The program is fairly basic and about 1.5MB and should run on virtually any machine made in the last 30 years. Although it is software, we have taken care to avoid making promises and accepting liability for things we cannot be held responsible for. Please be sure to review the brief warning section just before this page. It is assumed that the user has basic computer and command line use skills.

In this guide, a user new to Sincy will learn where Sincy can be downloaded, how to run the program, how to use the simulator to view the registers and memory interact with basic instructions, how to load a program from the command line and how to contribute to this open-source software.

Instruction Guide

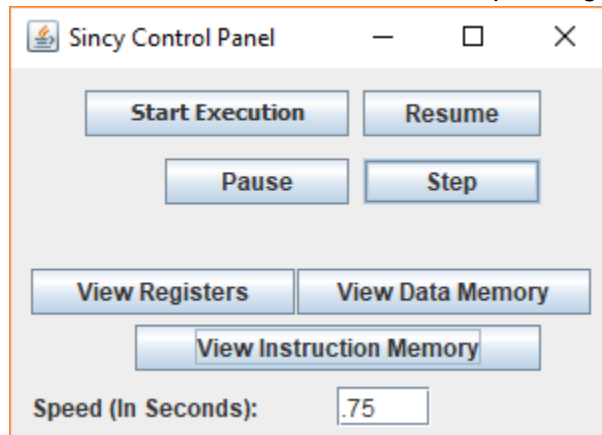
Getting Started

Downloading Sincy

1. Open an internet browser of your choice.
2. Enter <http://patrickmartinez.us/sincy.jar> in your search bar.
3. Save the file.

Opening Program and Windows

1. Run the file you have just downloaded. (You should see the following launch screen)
Note: You may need to move or resize the windows depending on your system settings.



2. Click "View Registers"
3. Click "View Data Memory"
4. Click "View Instruction Memory"
5. Move and resize windows to preference.
6. Click the 'X' at the top right to close individual windows.

Note: Closing the "Sincy Control Panel" will exit the program.

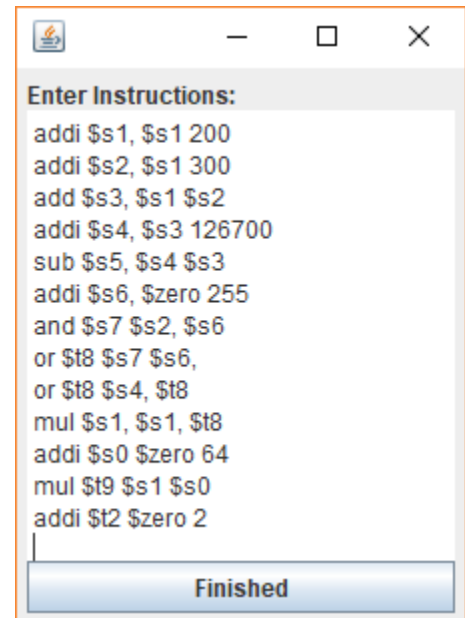
There is no way to re-open the "Instruction Window" when it has been closed.

Using Sincy

Running an Arithmetic Program

1. Enter the following into the Instruction Window labeled “Enter Instructions”.

```
addi $s1, $s1 200
addi $s2, $s1 300
add $s3, $s1 $s2
addi $s4, $s3 126700
sub $s5, $s4 $s3
addi $s6, $zero 255
and $s7 $s2, $s6
or $t8 $s7 $s6,
or $t8 $s4, $t8
mul $s1, $s1, $t8
addi $s0 $zero 64
mul $t9 $s1 $s0
addi $t2 $zero 2
```



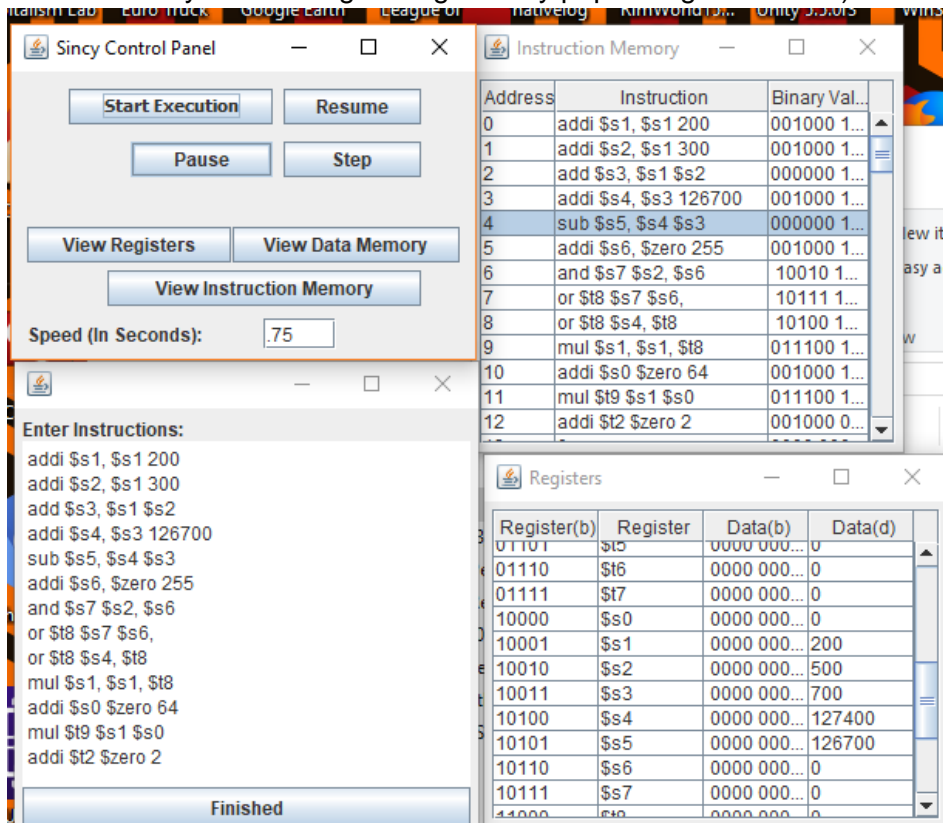
Note: Text has been typed above so it may be copy/pasted. If instructions window does not appear, restart the program. The window will auto open on start-up.

2. Click “Finished” on the Instruction Window.
3. Click “View Instruction Memory” to open the Instruction Memory Window. (You should see the instruction we typed in Instruction Memory)

Note: If the instructions don't appear like below, repeat step 2.

Instruction Memory		
Address	Instruction	Binary Value
0	addi \$s1, \$s1 200	001000 10001 10001 0000000011001000
1	addi \$s2, \$s1 300	001000 10010 10001 0000000100101100
2	add \$s3, \$s1 \$s2	000000 10001 10010 10011 00000 100000
3	addi \$s4, \$s3 126700	001000 10100 10011 11110111011101100
4	sub \$s5, \$s4 \$s3	000000 10100 10011 10101 00000 100010
5	addi \$s6, \$zero 255	001000 10110 00000 0000000011111111
6	and \$s7 \$s2, \$s6	10010 10110 10111 100100
7	or \$t8 \$s7 \$s6,	10111 10110 11000 100101
8	or \$t8 \$s4, \$t8	10100 11000 11000 100101
9	mul \$s1, \$s1, \$t8	011100 10001 11000 10001 000010
10	addi \$s0 \$zero 64	001000 10000 00000 0000000001000000
11	mul \$t9 \$s1 \$s0	011100 10001 10000 11001 000010
12	addi \$t2 \$zero 2	001000 01010 00000 0000000000000010
13	0	0000 0000 0000 0000 0000 0000 0000 0000
14	0	0000 0000 0000 0000 0000 0000 0000 0000
15	0	0000 0000 0000 0000 0000 0000 0000 0000

- Click "View Registers".
- Click "Start Execution".
Note: Speed is set to .75 by default. Be sure speed is set to this value before following the next step.
- Move or resize the windows so both Instruction Memory and Registers are visible.
- Click "Start Execution". (You should now see a highlighted bar traversing through the instruction memory and the registers gradually populating like below.)



Running a Loop

- Enter the following into the Instruction Window labeled "Enter Instructions".

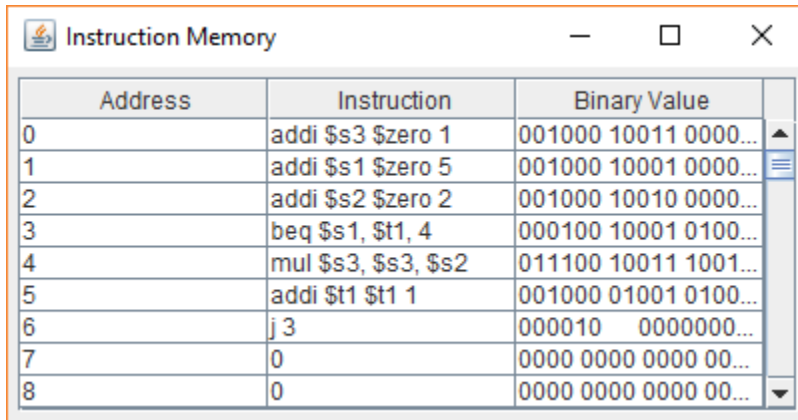
```
addi $s3 $zero 1
addi $s1 $zero 5
addi $s2 $zero 2
beq $s1, $t1, 4
mul $s3, $s3, $s2
addi $t1 $t1 1
j 3
```

Note: Text has been typed above so it may be copy/pasted. If instructions window does not appear, restart the program. The window will auto open on start-up.

- Click "Finished" on Instruction Window.

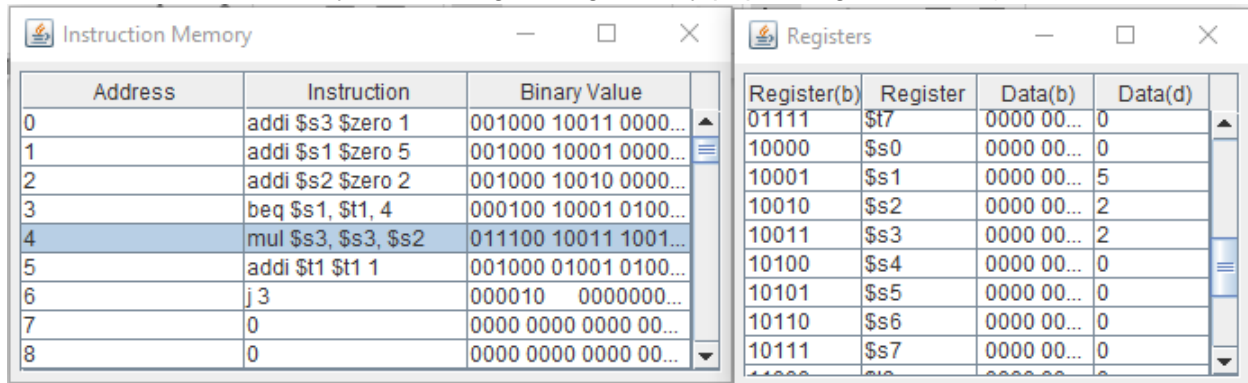
- Click “View Instruction Memory” to open the Instruction Memory Window. (You should see the instruction we typed in Instruction Memory)

Note: If the instructions don't appear like below, repeat step 2.



Address	Instruction	Binary Value
0	addi \$s3 \$zero 1	001000 10011 0000...
1	addi \$s1 \$zero 5	001000 10001 0000...
2	addi \$s2 \$zero 2	001000 10010 0000...
3	beq \$s1, \$t1, 4	000100 10001 0100...
4	mul \$s3, \$s3, \$s2	011100 10011 1001...
5	addi \$t1 \$t1 1	001000 01001 0100...
6	j 3	000010 0000000...
7	0	0000 0000 0000 00...
8	0	0000 0000 0000 00...

- Click “View Registers”.
- Click “Start Execution”.
 - Note: Speed is set to .75 by default. Be sure it is set to this value before following the next step.*
- Move or resize the windows so both Instruction Memory and Registers are visible.
- Click “Start Execution”. (You should now see a highlighted bar looping through the instruction memory and the registers gradually populating like below.)



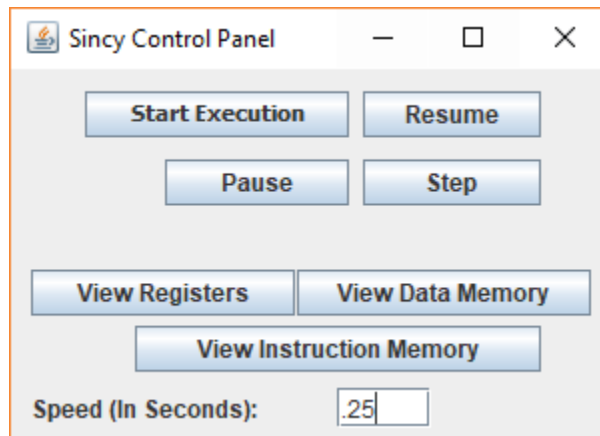
Address	Instruction	Binary Value
0	addi \$s3 \$zero 1	001000 10011 0000...
1	addi \$s1 \$zero 5	001000 10001 0000...
2	addi \$s2 \$zero 2	001000 10010 0000...
3	beq \$s1, \$t1, 4	000100 10001 0100...
4	mul \$s3, \$s3, \$s2	011100 10011 1001...
5	addi \$t1 \$t1 1	001000 01001 0100...
6	j 3	000010 0000000...
7	0	0000 0000 0000 00...
8	0	0000 0000 0000 00...

Register(b)	Register	Data(b)	Data(d)
01111	\$t7	0000 00...	0
10000	\$s0	0000 00...	0
10001	\$s1	0000 00...	5
10010	\$s2	0000 00...	2
10011	\$s3	0000 00...	2
10100	\$s4	0000 00...	0
10101	\$s5	0000 00...	0
10110	\$s6	0000 00...	0
10111	\$s7	0000 00...	0

Note: Notice that in the loop the highlighted portion jumps back up much like a real loop should do.

Changing Speed

1. Follow steps 1 - 4 on “Running a Loop” section on page 7.
2. Set the speed to .25 like below.



3. Click “Start Execution”. (The highlighted line showing the currently running instruction should be travelling significantly faster now)

Using Step-By-Step Execution

1. Follow steps 1-4 on “Running a Loop” section on page 7.
2. Click “Step”. (The current command line should appear at line 0.)
3. Click “Step” again. (The line will jump to line 1.)
4. Continue until you have analyzed all the steps you would like to see or the program finishes executing.

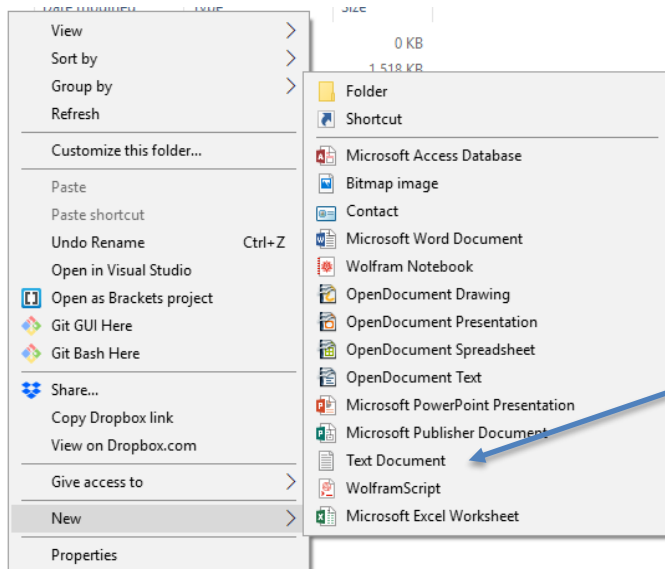
Loading .asm Files from Command Line

Sincy also supports a command line argument using filenames. To use this feature, a file must be saved as .asm.

Saving a File as .asm

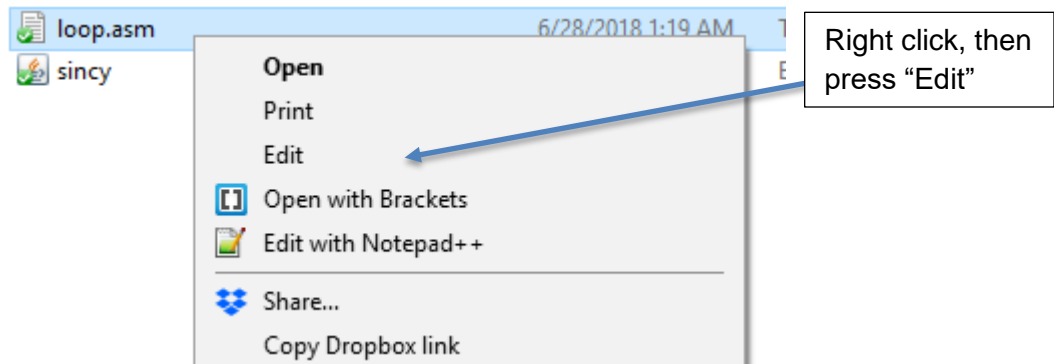
1. Right-click on the window where Sincy is located and follow the menu options as follows.

New > Text Document



Click “Text Document”
under the “New” option.

2. Right-click and click "Edit"



3. Paste some code into the file and save it. The code examples on pages 6 or 5 will work.

Loading Sincy from a .asm File

1. Open a command prompt and navigate to the folder containing Sincy and the .asm file.
Note: It is assumed that the person using this manual is familiar with their operating systems command line or terminal program. The steps associated with command line or terminal use are outside of the scope of this document.

2. Type `java -jar sincy.jar [name of file]`.

Note: Replace [name of file] with your filename. See example below

A screenshot of a Windows command prompt window. The title bar reads 'MINGW64:/c:/Users/martip23/Dropbox/Tech Writing...'. The prompt shows the user 'martip23@LivingRoom-PC' in a 'MINGW64' environment at the path '~/Dropbox/Tech Writing/proj4'. The command '\$ java -jar sincy.jar loop.asm' has been entered at the prompt.

3. Type enter. (You should see Sincy open, except now there is no enter instruction window and the instruction memory is preloaded.)

Note: The command line version of Sincy provides more feedback on the process and cycle information.

A screenshot of a Windows command prompt window showing the output of the command 'java -jar sincy.jar loop.asm'. The title bar is the same as the previous screenshot. The output text is as follows:
Waiting...
martip23@LivingRoom-PC MINGW64 ~/Dropbox/Tech Writing/proj4
\$ java -jar sincy.jar loop.asm
Starting sincySimulator (prototype)...
Pre-Read...
Running sincySimulator...
Waiting...
Stepping through Execution
FETCH
DECODE: addi \$s3 \$zero 1
EXECUTE
MEMORY
WRITEBACK
Waiting...

Conclusion

Thank you for using this software. It should help you along your journey to study for a career in Computer Science. It should also help you understand and pass an important test in your Computer Architecture class. This is an open-source project and thus is open to public contribution from other developers. The codebase has been intentionally designed to be easy to use and is object-oriented to encourage collaboration between various users. If you have any questions, concerns, bug reports or suggestions for this software please email me at patrickmartinez@txstate.edu.