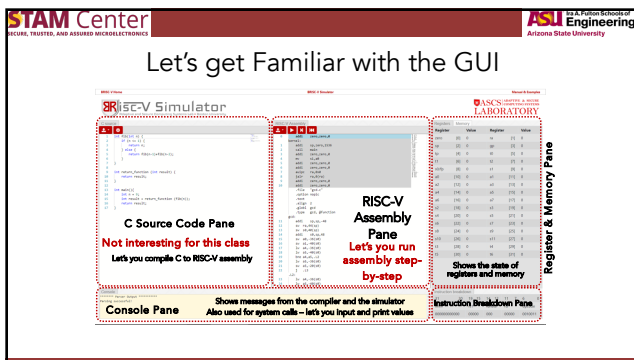




1



2



3

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

Don't have valid RISC-V assembly code to start with?

SRISCV Simulator

Load your code here!

Examples available here!

Click the load button to open a drop down menu for loading examples

Documentation and example code here!

ASCS: ARIZONA STATE UNIVERSITY LABORATORY

Register	Value	Register	Value
x0	0	x1	1
x2	2	x3	3
x4	4	x5	5
x6	6	x7	7
x8	8	x9	9
x10	10	x11	11
x12	12	x13	13
x14	14	x15	15
x16	16	x17	17
x18	18	x19	19
x20	20	x21	21
x22	22	x23	23
x24	24	x25	25
x26	26	x27	27
x28	28	x29	29
x30	30	x31	31

4

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

Don't have valid RISC-V assembly code to start with?

SRISCV Simulator

Load your code here!

Examples available here!

Load the GCD example

Documentation and example code here!

ASCS: ARIZONA STATE UNIVERSITY LABORATORY

Register	Value	Register	Value
x0	0	x1	1
x2	2	x3	3
x4	4	x5	5
x6	6	x7	7
x8	8	x9	9
x10	10	x11	11
x12	12	x13	13
x14	14	x15	15
x16	16	x17	17
x18	18	x19	19
x20	20	x21	21
x22	22	x23	23
x24	24	x25	25
x26	26	x27	27
x28	28	x29	29
x30	30	x31	31

5

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

Kernel and User Instructions

SRISCV Simulator

Our code got loaded!

The console says that it passed the file without problems if there were problems, they would pop up here

Documentation and example code here!

ASCS: ARIZONA STATE UNIVERSITY LABORATORY

Register	Value	Register	Value
x0	0	x1	1
x2	2	x3	3
x4	4	x5	5
x6	6	x7	7
x8	8	x9	9
x10	10	x11	11
x12	12	x13	13
x14	14	x15	15
x16	16	x17	17
x18	18	x19	19
x20	20	x21	21
x22	22	x23	23
x24	24	x25	25
x26	26	x27	27
x28	28	x29	29
x30	30	x31	31

6

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

Kernel and User Instructions

SRISecV Simulator

Gray instructions are kernel instructions

They setup some registers like the stack pointer, and jump to label "main"

Register

Value

Register

Value

7

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

Kernel and User Instructions

SRISecV Simulator

White instructions are user instructions

All the assembly you write will be here

Register

Value

Register

Value

8

STAM Center

SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering

Arizona State University

Simulator Controls

SRISecV Simulator

Here are some simulator controls:

Load code

Run code

Reset Simulator

Step through code

Register

Value

Register

Value

9

3

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Stepping Through a Program

SRISecV Simulator

Let's click the step button twice

The blue line shows which instruction will be executed next

Register

Register	Value	Register	Value
zero	0	ra	0
1	0	sp	0
2	0	gp	0
3	0	tp	0
4	0	t0	0
5	0	t1	0
6	0	t2	0
7	0	t3	0
8	0	t4	0
9	0	t5	0
10	0	t6	0
11	0	t7	0
12	0	t8	0
13	0	t9	0
14	0	s0	0
15	0	s1	0
16	0	s2	0
17	0	s3	0
18	0	s4	0
19	0	s5	0
20	0	s6	0
21	0	s7	0
22	0	s8	0
23	0	s9	0
24	0	ra	0

10

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

Stepping Through a Program

SRISecV Simulator

Click the step button again

The blue line moved two lines down

These instructions did not do anything, but the next one will

The instruction breakdown pane shows how the instruction is stored in memory

The top row represents the bit ranges, the middle row shows range names, the bottom row shows binary values for the specific instruction

Register

Register	Value	Register	Value
zero	0	ra	0
1	0	sp	0
2	0	gp	0
3	0	tp	0
4	0	t0	0
5	0	t1	0
6	0	t2	0
7	0	t3	0
8	0	t4	0
9	0	t5	0
10	0	t6	0
11	0	t7	0
12	0	t8	0
13	0	t9	0
14	0	s0	0
15	0	s1	0
16	0	s2	0
17	0	s3	0
18	0	s4	0
19	0	s5	0
20	0	s6	0
21	0	s7	0
22	0	s8	0
23	0	s9	0
24	0	ra	0

11

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Engineering
Arizona State University

addi Changed the Register File

SRISecV Simulator

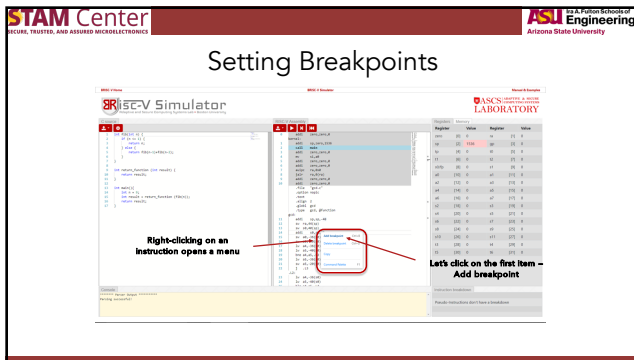
addi sp, zero, 1536 got executed. It summed 0 and 1536, and put it in register sp

Register sp is highlighted! sp stands for stack pointer

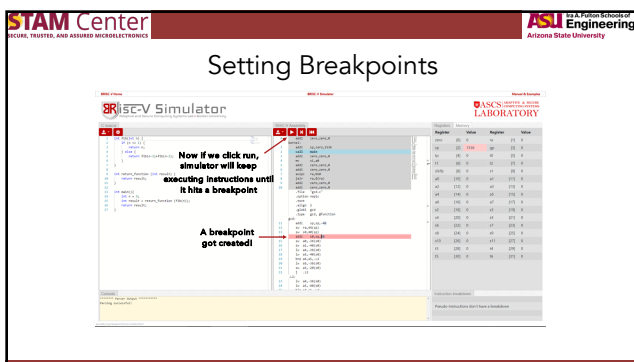
Register

Register	Value	Register	Value
zero	0	ra	0
1	0	sp	1536
2	0	gp	0
3	0	tp	0
4	0	t0	0
5	0	t1	0
6	0	t2	0
7	0	t3	0
8	0	t4	0
9	0	t5	0
10	0	t6	0
11	0	t7	0
12	0	t8	0
13	0	t9	0
14	0	s0	0
15	0	s1	0
16	0	s2	0
17	0	s3	0
18	0	s4	0
19	0	s5	0
20	0	s6	0
21	0	s7	0
22	0	s8	0
23	0	s9	0
24	0	ra	0

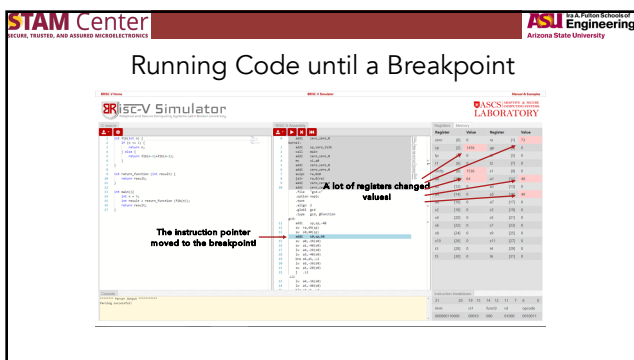
12



13



14



15

System Calls

- We also provide some simple system calls
- System calls are used for functionalities provided by the operating system
 - Think file systems, IO, etc.
- In RISC-V, system calls look something like:
 - Put the type of system call you want in register `t0`
 - More about that on the next slide
 - Put any arguments you may have in `a0` and `a1`
 - Call instruction `ECALL`
 - If the system call has return values, they will be in `a0`
- To really get familiar with syscalls, try running the example syscall file in the simulator

19

Supported Syscalls

Syscall	Syscall ID (put this in t0)	Description
Print integer	1	Print integer value in <code>a0</code> to console
Print char	2	Print ascii value in <code>a0</code> to console
Print string	3	Print string with address in <code>a0</code> and length in <code>a1</code> to console
Read integer	4	Read integer from console into <code>a0</code>
Read char	5	Read character from console into <code>a0</code> as an ascii value
Read string	6	Read string of length given in <code>a1</code> from console and store it at address in <code>a0</code>
SBRK	7	Dynamically allocate the amount of bytes specified in <code>a0</code> . The pointer to the beginning of the newly allocated memory will be stored in <code>a0</code> . The value in <code>a0</code> can be negative, if you want to deallocate some memory!

20

21
