STAM Center — SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering — Arizona State University

BRISC-V Simulator

Adaptive and Secure Computing Systems Lab • Boston University

A Browser-based RISC-V Simulator

Adaptive and Secure Computing Systems (ASCS) Laboratory

1

STAM Center — SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering — Arizona State University

BRISC-V Simulator

- BRISC-V Simulator let's you:
  - Run RISC-V assembly code in the browser
  - Debug hand-written assembly
  - Run code until completion or until it hits a breakpoint
  - Step through execution, instruction-by-instruction
  - View the state of memory and registers at each step
  - View how each instruction is constructed – opcodes, registers, immediate values etc.

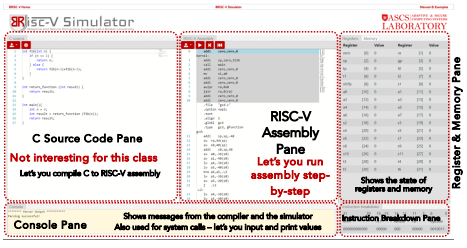
https://ascslab.org/research/briscv/simulator/simulator.html

2

STAM Center — SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering — Arizona State University

Let's get Familiar with the GUI

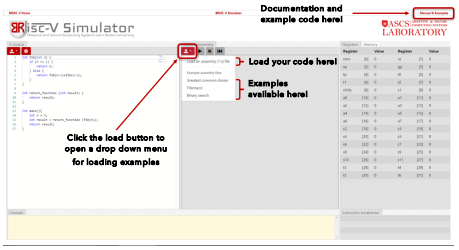


3

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

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

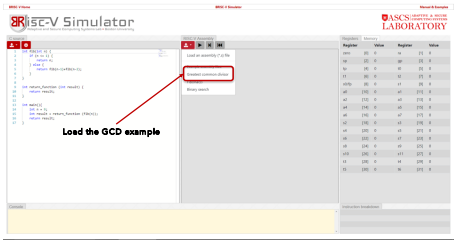


4

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

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

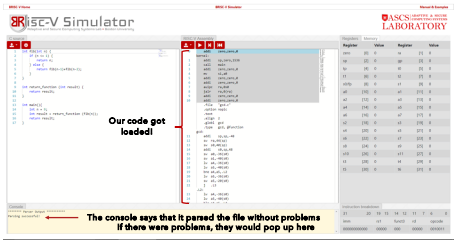


5

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Kernel and User Instructions

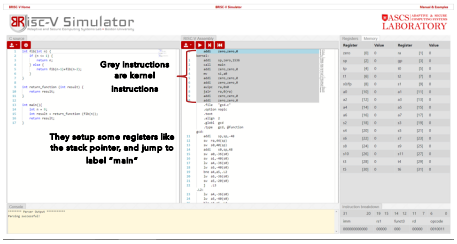


6

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Kernel and User Instructions

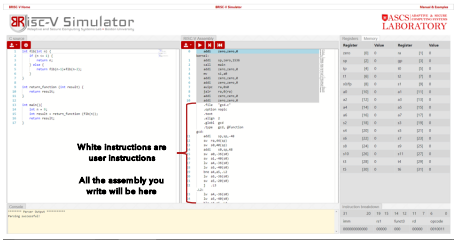


7

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Kernel and User Instructions

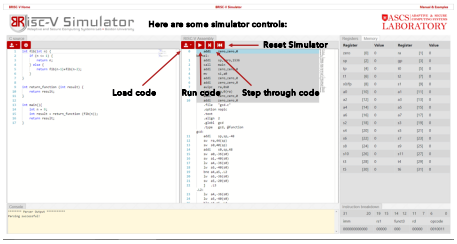


8

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Simulator Controls

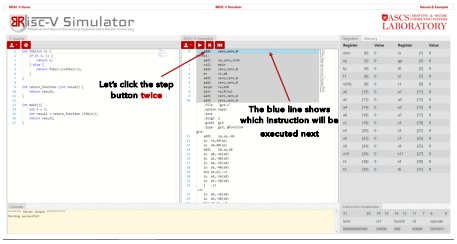


9

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Stepping Through a Program

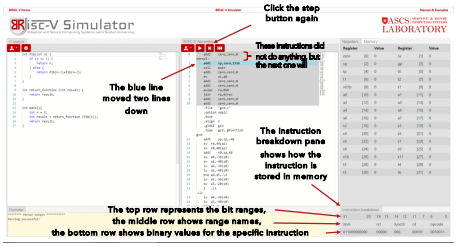


10

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Stepping Through a Program

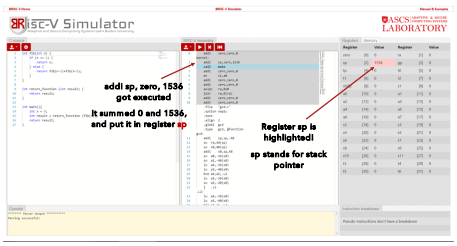


11

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

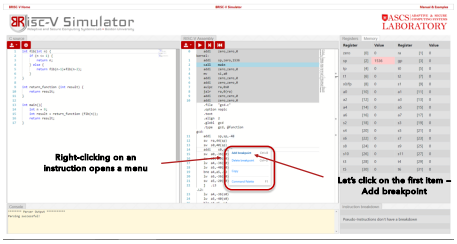
ASU Ira A. Fulton Schools of Engineering
Arizona State University

addi Changed the Register File



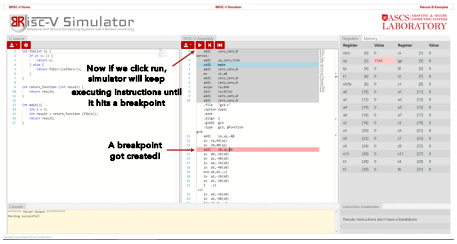
12

Setting Breakpoints



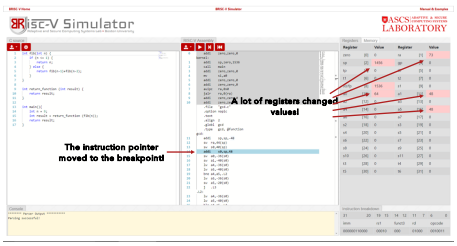
13

Setting Breakpoints



14

Running Code until a Breakpoint

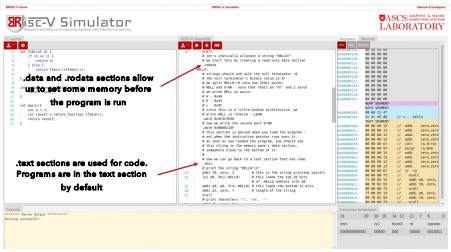


15

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Text and Data Sections

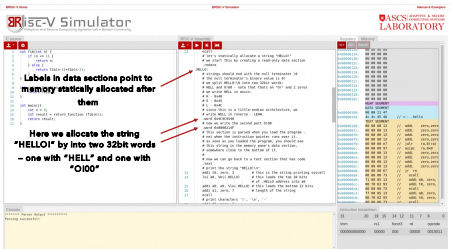


16

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Text and Data Sections

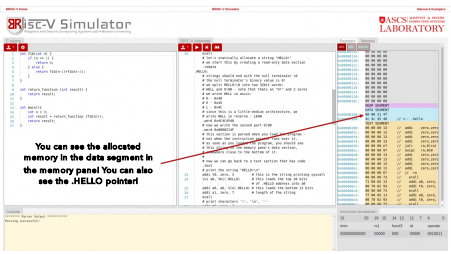


17

STAM Center
SECURE, TRUSTED, AND ASSURED MICROELECTRONICS

ASU Ira A. Fulton Schools of Engineering
Arizona State University

Text and Data Sections



18

System Calls

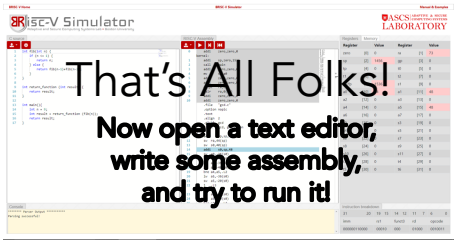
- We also provide some simple system calls
- System calls are used for functionalities provided by the operating system
  - Think file systems, IO, etc.
- In RISC-V, system calls look something like:
  - Put the type of system call you want in register t0
    - More about that on the next slide
  - Put 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 a0 to console
Print char	2	Print ascii value in a0 to console
Print string	3	Print string with address in a0 and length in a1 to console
Read integer	4	Read integer from console into a0
Read char	5	Read character from console into a0 as an ascii value
Read string	6	Read string of length given in a1 from console and store it at address in a0
SBRK	7	Dynamically allocate the amount of bytes specified in a0. The pointer to the beginning of the newly allocated memory will be stored in a0. The value in a0 can be negative, if you want to deallocate some memory!

20



That's All Folks!

Now open a text editor, write some assembly, and try to run it!

21