Warmup

Download Ripes from ripes.me/Ripes

Or use it in your browser at ripes.me

EE 201: RISC-V assembly

Steven Bell

26 March 2024

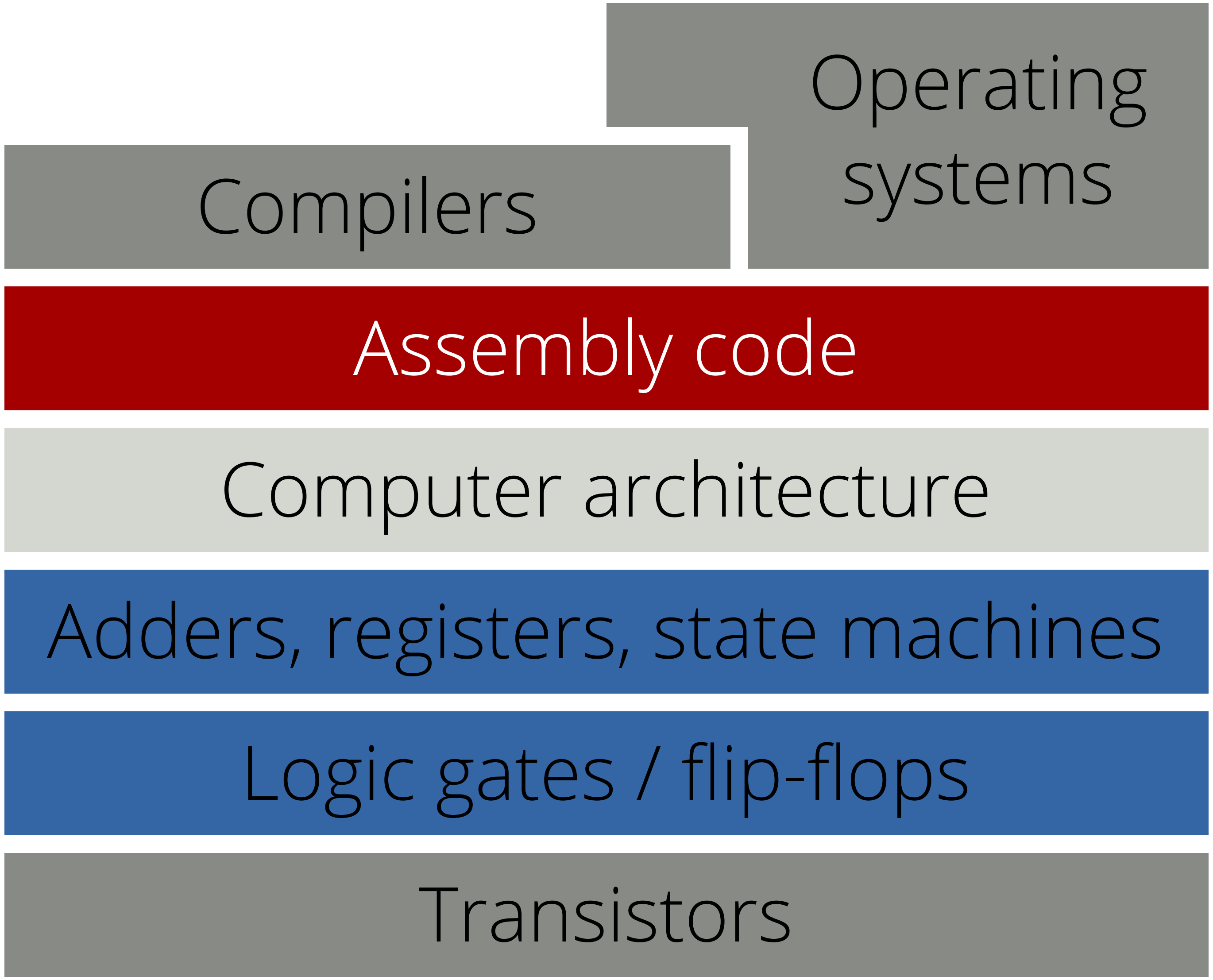
By the end of class today, you should be able to:

- Explain what an instruction set architecture (ISA) is
- Describe the registers available on a RISC-V processor

- Write RISC-V (RV32I) assembly code to do basic math
- Use branch instructions to implement if-else/loops/etc

We'll talk about memory and functions on Thursday!

The big picture



What's an architecture?

"The programmer's view of the computer."

The contract between hardware and software: the set of things the software can ask the hardware to do, and what happens as a result.

Software tools

ISA

Hardware design

Intel x86: a success/horror story

A modern processor can run code from ~40 years ago

A modern processor has to support code from ~40 years ago!

RISC-V

Pronounced "risk five"

A relatively new ISA developed at Berkeley

Follows reduced instruction-set computing (RISC) principles

- Support a small number of simple instructions

Defined as a base instruction set with optional extensions

- We'll be using RV32I, which is the 32-bit version with no extensions

Why RISC-V in this course?

It's a real ISA, growing in commercial adoption

- We can use a normal C compiler (gcc/clang) to write code for it

Simple enough to understand in one semester

- A summary of all of RV32I fits on one page

RISC-V general-purpose registers

RV32I defines 32 general purpose registers, `x0` through `x31`

`x0` is special: it is always 0

All other registers work the same, but by convention
some are used for specific things (e.g., function arguments)

Let's write some code!

Destination register

```
li x1,100
```

Mnemonic

Immediate operand

Load Immediate

ripes.me / Ripes

Pseudo-instructions??

addi x1, x0, 1

x1 = x0 + 1

zero!

OPERATION dst,src1,src2 (2 registers)		OPERATION dst,src1,immediate (reg + value)	
ADD	src1 + src2	ADDI	
SUB	src1 - src2	No SUBI!	use negative immediate
SLL	shift left	SLLI	
SRL	logical right	SRLI	
SRA	arithmetic	SRAI	
SLT	src1 < src2	SLTI	
SLTU	1111 < 0110 ?	SLTIU	
OR	logical OR	ORI	
AND	" AND	ANDI	
XOR	" XOR	XORI	

- 2 shift by 1 4: 0010 → 0100
- 2: 0010 → 0001 (1)
- −8 → −4: 1000 → 1100 "sign-extend"

Practice time!

Write an assembly program which calculates

$$100 + 100 + 5$$

Try some of the other operators!

How do I make choices?

Use a branch!

BEQ   branch if equal

Practice time

Write a program that computes the absolute value of the value stored in `x1`. (Use li to load various values to test it!)

Loops

A while loop has the form:

```
TOP
    check if condition is false, and branch to BOTTOM if so
    [ body of loop ]
    unconditionally branch back to TOP
BOTTOM
```

Practice time

Write a program that computes the sum of the natural numbers from 0 to 10.

```
if ( ~~~ ) {

}else {

}
```

```
B~~~ else
    stuff if true
    BEQ x0,x0, bottom
else:
    stuff if false
    ...
bottom:
    ...
```

while (~) {

}

top:
B~ bottom (check condition)

loop stuff ...

BEQ x0, x0, top

bottom