Lecture 4: Instructions

Wednesday, January 17, 2018 5:27 PM

Outline

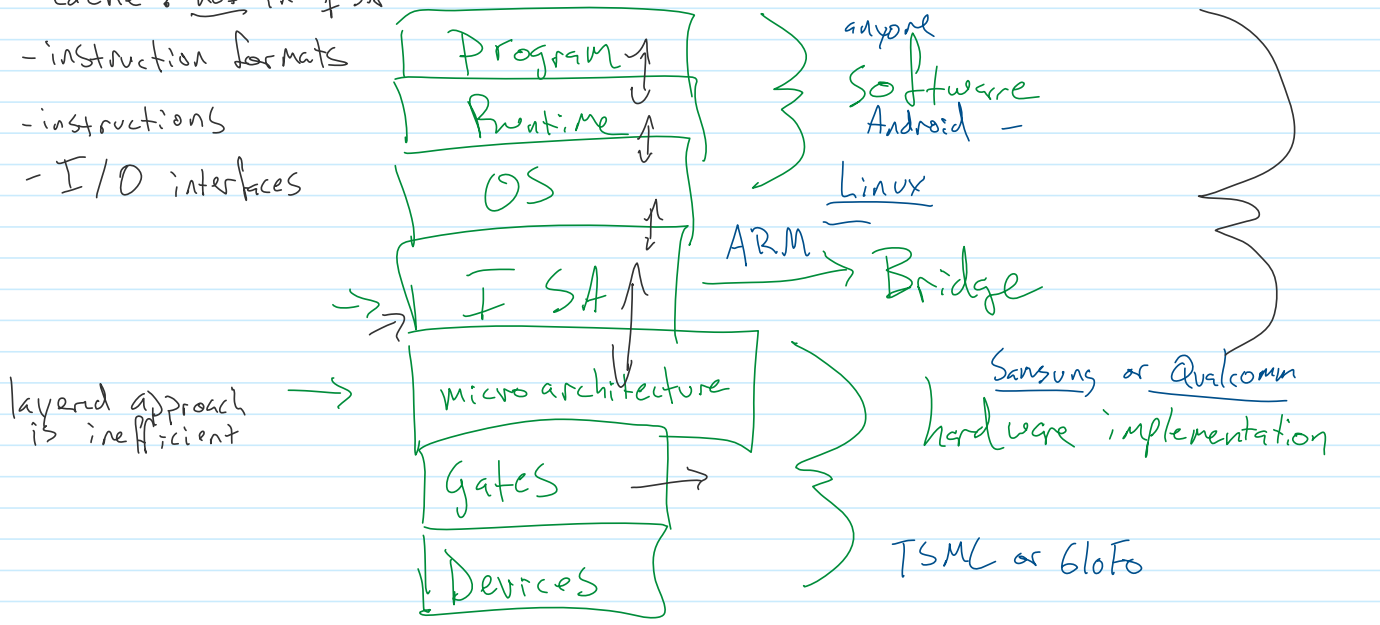
- ISA "definition"
- MIPS details
- From higher-level languages to machine code
- Other system architecture details

Quiz tomorrow
End 1/2 of discussion
Chapter 1 technology + performance
Chapter 2 at high level ↳ expect to look like book question

What is an ISA?

Instruction Set Architecture
- Registers (e.g, number)
- Cache? not in ISA
- instruction formats
- instructions
- I/O interfaces

Contract between programmer and the hardware



layered approach is inefficient →

Key features of ISAs:

# register	address range/ pointer size	Memory consistency model
inst format	Word size	- allowed interleavings of memory operations
size of instructions	Virtual memory	
I/O interfaces - interrupts	Security	

Question: How many registers does this ISA have?

32 registers
↳ bit fields are 5 bits

↳ bit fields are 8 bits

MIPS

→
→ 0x1000: jal → 0x 100000
→ 0x1004: add $2, $3, $4
→ 0x1008: sub $5, $6, $7

0x10000: add $8, $8, 7
→ jr $31
→
j 0x1000

	PC	$2	$3	$4	$5	$6	$7	$8	$31
		7	11	15	42	-1	8	2	
0:	0x1000								0x1008
1:	0x1004	26							
→ 2:	0x10000							9	
3:	0x10004								
4:	0x1008					-9			
5:									

---

li $3, 12000
lw $4, 8($3)

Result? load data at memory location 12008 into register 4

3 types of operands
- immediate
- registers
- memory (index)

dest.
source + imm. offset

Conditional branches

```
0  li  $2, 7
4  li  $3, 12
8  beq $3, $2, 8
12 ~~
16 ~~
20 ~~
24 ~~
```

→ taken? / not taken?

From higher-level languages to machine code

C code —compiler→ assembly —assembler→ object code —linker→ object code → binary

if(_)

add
lw
beq Label

0x12 0x3 0x2

0x1000
???

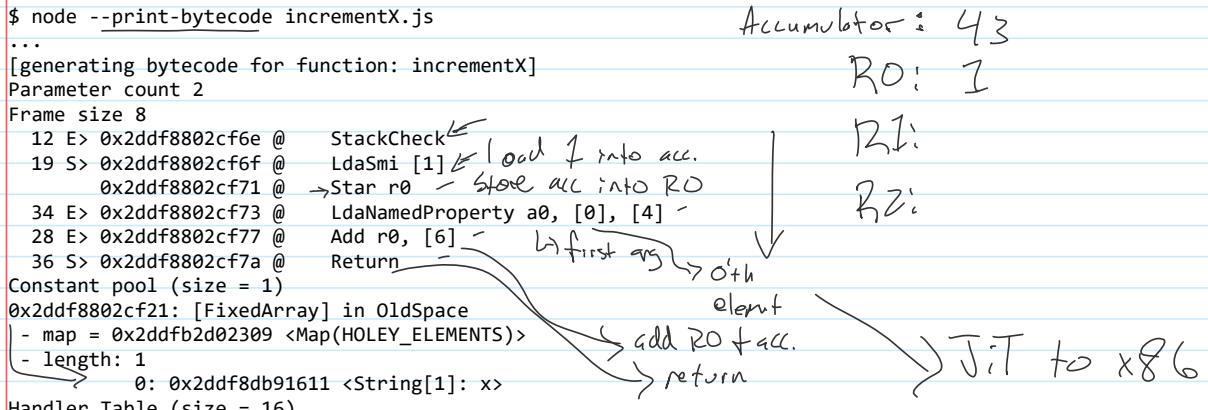
javascript is very common

Javascript?

```
function incrementX(obj) {
  return 1 + obj.x;
}
incrementX({x: 42});
```

```
function incrementX(obj) {
  return 1 + obj.x;
}
incrementX({x: 42});
```

```
$ node --print-bytecode incrementX.js
...
[generating bytecode for function: incrementX]
Parameter count 2
Frame size 8
  12 E> 0x2ddf8802cf6e @    StackCheck
  19 S> 0x2ddf8802cf6f @    LdaSmi [1]
        0x2ddf8802cf71 @    Star r0
  34 E> 0x2ddf8802cf73 @    LdaNamedProperty a0, [0], [4]
  28 E> 0x2ddf8802cf77 @    Add r0, [6]
  36 S> 0x2ddf8802cf7a @    Return
Constant pool (size = 1)
0x2ddf8802cf21: [FixedArray] in OldSpace
 - map = 0x2ddfb2d02309 <Map(HOLEY_ELEMENTS)>
 - length: 1
           0: 0x2ddf8db91611 <String[1]: x>
Handler Table (size = 16)
```



Accumulator: 43
R0: 1
R1:
R2:

Interpreting one "instruction" at a time

Just in time (JIT) compiling