CPSC 213 Lab 6

Static Control Flow

Slides available at randyzhu.com/cpsc213

Randy Zhu

Course Updates

- Hope everyone had a good reading break!
- Assignment 6 due this Friday
- Quiz 3 next week, Mar 2-6
- Course starts getting more interesting (read: harder) IMO

Branching Offset

Q1

	`7dss` $(ss < 0)$	$r[d] \leftarrow r[d] >> (v = -ss)$	`shr $v, rd`
branch	`8-pp`	$pc \leftarrow (a = pc + p \times 2)$	`br a`
branch if equal	`9rpp`	if $r[r] == 0 :\ pc \leftarrow (a = pc + p \times 2)$	`beq rr, a`
branch if greater	`Arpp`	if $r[r] > 0 :\ pc \leftarrow (a = pc + p \times 2)$	`bgt rr, a`
jump	`B---aaaaaaaa`	$\text{pc} \leftarrow a$	`j a`
get program counter	`6Fpd`	$r[d] \leftarrow \text{pc} + (o = 2 \times p)$	`gpc $o, rd`

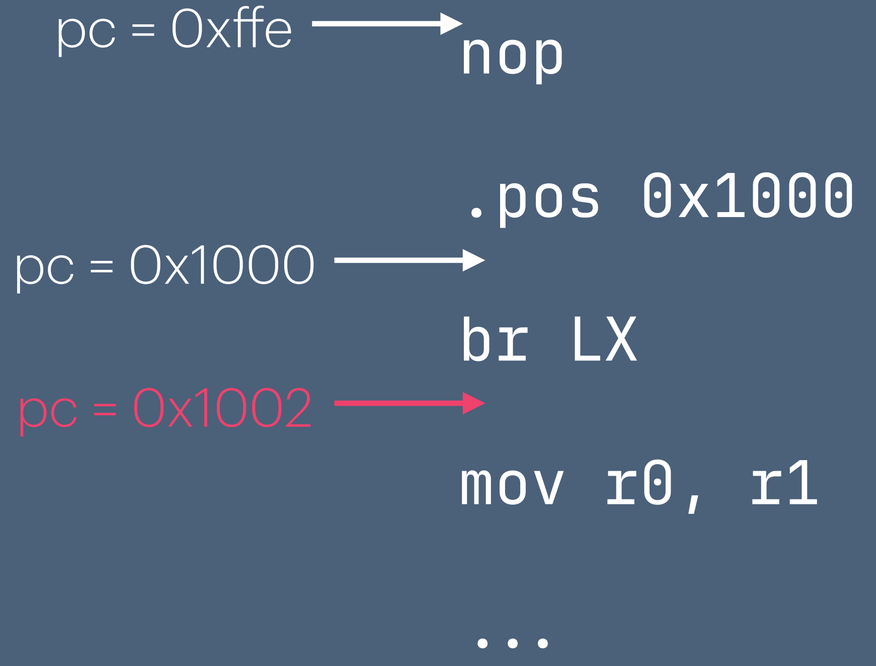
Branching Offset

Q1

- What's the state of PC after `br LX`?
- Recall:
  - Fetch -> Update PC w/ instr size
  - Execute -> Run instruction semantics
- Since pc = 0x1002 after fetch:

$$pc \leftarrow (a = pc + p \times 2)$$

this pc = 0x1002



Branching Offset

Q1

- How far can I jump from 0x1002?
- br instruction encoded as 8-pp
- pp is signed = 8 bits -> max value = 2^7 - 1 = 127
- p is multiplied by 2 in semantics = 0xFE
- 0x1002 + 0x00FE = 0x1100 -> **L5**

$$pc \leftarrow (a = pc + p \times 2)$$

Helpful: templates for if/for

1d pg 3, pg 7

Implementing *for* loops

```
for (i=0; i<10; i++)
  s += a[i];
```

- General Form (**C** and **Java**):

```
for (<init>; <continue_condition>; <step>) <statement_block>
```

  - each of init, continue_condition, and step is optional or can be a compound expression

```
for (;;)
for (int i=0, j=10; i<=j; i++, j--)
```

  - pseudocode template using goto statements

```
          <init>
loop:     goto end_loop if not <continue_condition>
          <statement_block>
          <step>
          goto loop
end_loop:
```


Implementing Conditionals (if-then-else)

```
if (a > b) max = a;
else max = b;
```

- General Form (**C** and **Java**):

```
if (<condition>) <then_statements> else <else_statements>
```

  - pseudocode template using goto statements

```
          c’ = not <condition>
          goto then if (c’ == 0)
else:     <else_statements>
          goto end_if
then:     <then_statements>
end_if:
```

Or sometimes:

```
c’ = <condition>
goto then if c’ > 0
```