MIPS Format Exceptions

I-format Exceptions

- Still have 2 registers and a constant value immediately present in the instruction.
    - rs: operand or base address (5 bits)
    - rt: operand or data register (5 bits)
    - immediate: value or offset (16 bits)

Name	Format	Layout						Example
		6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	
		op	rs	rt	immediate			
beq	I	4	1	2	25 (offset)			beq $1, $2, 100
bne	I	5	1	2	25 (offset)			bne $1, $2, 100
lui	I	15	0	1	100			lui $1, 100
lw	I	35	2	1	100 (offset)			lw $1, 100($2)
sw	I	43	2	1	100 (offset)			sw $1, 100($2)

Example

```
lw $t0, 32($s3)        (registers 8 and 19)
```

op	rs	rt	immediate
35	19	8	32
100011	10011	01000	0000000000100000

Example

```
lui $t0, 1028        (register 8)
```

op	rs	rt	immediate

15	0	8	1028
001111	00000	01000	0000010000000100

Example: `beq`

```
beq $t0, $zero, ENDIF
```

The offset stored in a `beq` (or `bne`) instruction is the number of instructions from the PC (the instruction after the `beq` instruction) to the label (`ENDIF` in this example). Or, in terms of addresses, it is the difference between the address associated with the label and the PC, divided by four.

```
offset = (addrFromLabelTable - PC) / 4
```

In the example above, if the `beq` instruction is at address `1004`, and thus the PC is `1008`, and if `ENDIF` is at address `1028`, then the value stored in the machine instruction would be

```
offset = (1028 - 1008) / 4 = 5
```

op	rs	rt	immediate
4	8	0	5
000100	01000	00000	0000000000000101

R-format Exceptions

- Still have opcode `0` (all of them!), 3 registers, a shift amount, and a funct code.

Name	Format	Layout						Example
		6 bits	5 bits	5 bits	5 bits	5 bits	6 bits	
		op	rs	rt	rd	shamt	funct	
sll	R	0	0	2	1	10	0	sll $1, $2, 10
srl	R	0	0	2	1	10	2	srl $1, $2, 10
jr	R	0	31	0	0	0	8	jr $31

NOTE: op is 0, so funct disambiguates

Example

```
srl $s0, $s1, 1        (registers 16, 17)
```

op	rs	rt	rd	shamt	funct
0	0	17	16	1	2
000000	00000	10001	10000	00001	000010

Example

```
jr $ra        (register 31)
```

op	rs	rt	rd	shamt	funct
0	31	0	0	0	8
000000	11111	00000	00000	00000	001000

Previous Slide

Alyce Brady, Kalamazoo College

Next Slide