CS2318

09/15/16

~~Usual~~ "Usual" flow

Fetch — Fetch Load inst
Inc pc

Execute

~~in stage~~ Sequential mode

Fetch: ~~1) IR ← M ← pc~~

1) IR ← M[pc]

2) pc ← pc + 4 (inc pc)

Execute — ADD subtract Load
store

2 in change of flow

Fetch Load IR IR ← M[PC]

PC ← PC + 4

Execute Modify PC

Condition un-conditional

Types of Branch Inst.

1) Classification conditional / unconditional

2) Relative (Relative to PC in MIPS) or Absolute

3) Range

3

BEQ, BNE
BLT BLE ...
] conditional Relativ ± 2^{15} words

B — Unconditional Relative

J (JAL) — Uncondition not relative Restricted. In 0 segment of 2^{28} words

JR $\$R_s$ — $PC \leftarrow \$R_s$ Uncond. Absolut. Range 2^{32} bytes

Assume I want to jump to unconditionaly

0X ABCD CAFE

4

BEQ 1 inst.

BLT 2 Real Inst

J B 1 Inst

JR R_s We have to place the Target in R_s

For now Assume that R_s contains an address

5)

```
...
a = factorial(n);
...
```

... ~~compiling it~~

```
factorial ( ) {
 return ( )
}
```

Transmit argument to function
(In MIPS $A0 to $A3
In "Every" CPU can use
the stack)

Provide return value
$V0 $V1 or stack

⋮
a = factorial (args)   change of flow
⋮

factorial (pars) {

return (value)
}

go back to the calling function
To the next instruction

in MIPS

1000 JAL factorial
1004

15000 factorial: start executing factorial
return

7

- JAL ~~function~~ Target

~~$RA ⇐ @(TARGET)~~

$RA ⇐ PC (Return Address)

PC ⇐ @(TARGET)

@(TARGET) absolute address within a segment

- Ret JR $RA return

⇒ Do not use $RA as general purpose

Only one level of nesting

To support "more" levels use a stack

8 Multiply / Divide

multiply $|n| \times |n| \rightarrow 2n$

add $|n| + |n| \rightarrow (n+1)$

R5 * R6 64

Hi

[Hi ; Lo] ⇐ R5 * R6

[Hi ; Lo] ← R5 / R6

Hi ← R5 % R6

Lo ⇐ R5 / R6

mfhi $RD $RD ← hi
mflo $RD $RD ⇐ Lo

mthi $R5 hiR5 ← hi
mtlo $R5 loR5 ← Lo

mult $Rs, $Rt

[Hi ; Lo] ⟵ $Rs * $Rt

mul $Rd, $Rs, $Rt

: mult $Rs, $Rt

mflo $Rd