

## Pipelining Part 2

Note Title

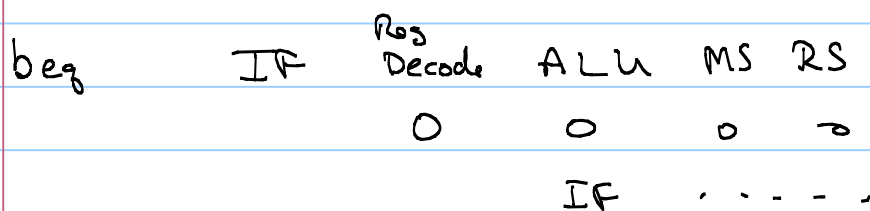
10/17/2008

### Control Hazards

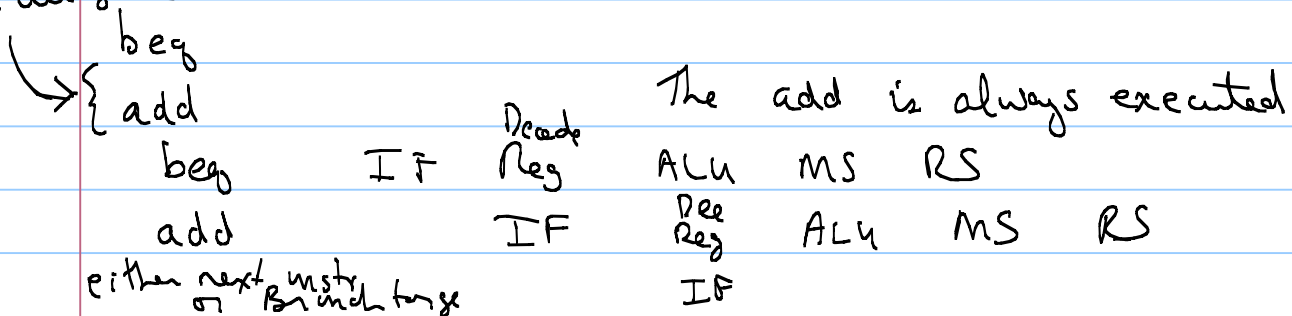


## Approaches to Control Hazards

- ① Put special branch comparator in the Decode Stage.



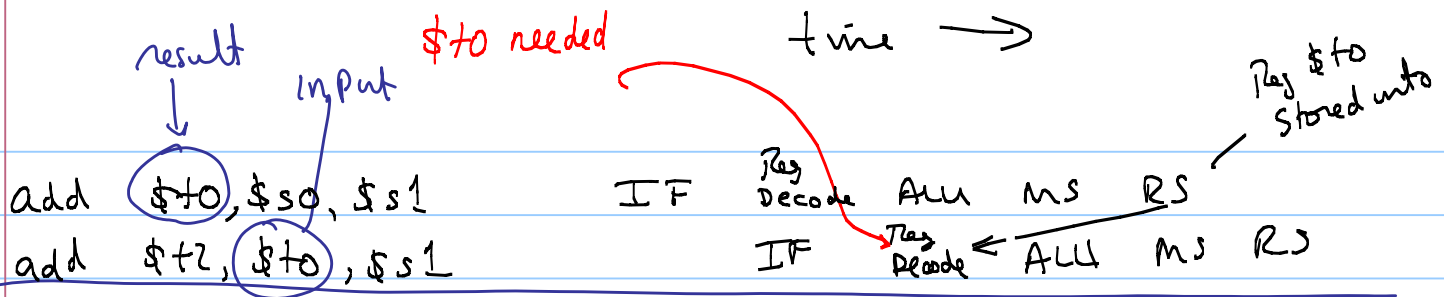
- ② Arrange to always execute the instructions after branch  
a delay slot



## Branch prediction 6.6

beg	IF	Reg Dec	ALU	MS	RS	
add		IF	0	0	0	if wrong
		IF	Reg Dec	ALU	...	if right

Branch instructions w/ prediction — Static prediction  
Dynamic branch prediction.

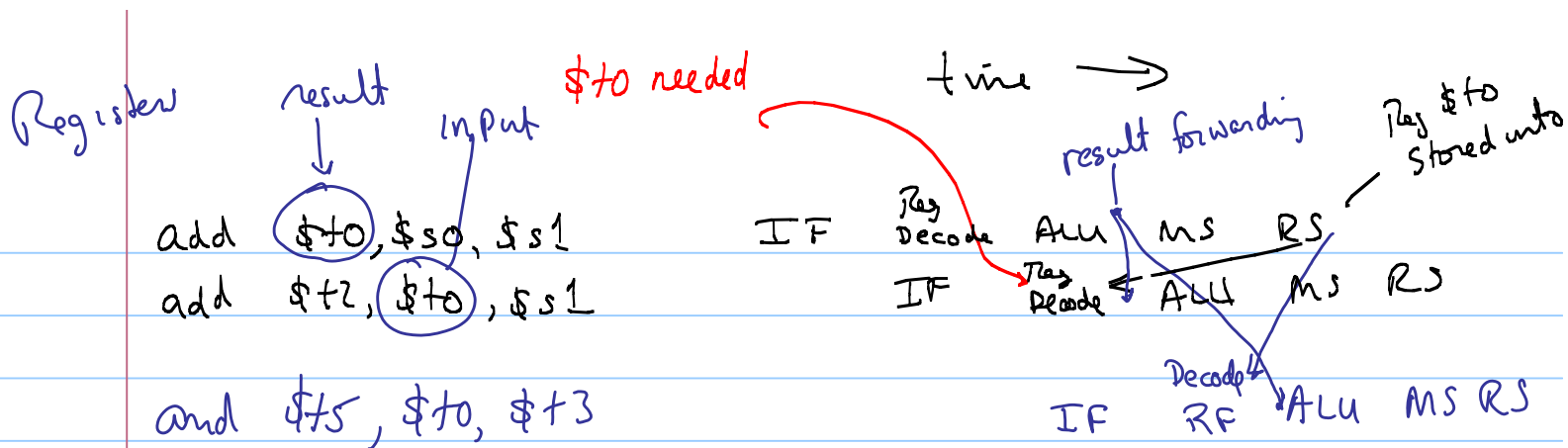


## Review

① Delay slot after branch — buy time

delay slot → beg — — —  
add \$t0, ... —

② Get the result sooner — add a comparator to the inst. decode



What about Loads

lw  $\$t0, 0(\$t1)$

add  $\$t2, \$t0, \$t1$

IF RegF Decode ALU MF RS

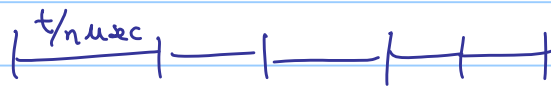
IF RF/Dec O ALU MS RS

- ① introduce a bubble — "stalling the pipeline" }
- ② define a delay slot for loads.
- ③ don't fix it in HW / Require programmer or compiler to insert NOP if necessary.

$t$   $\mu$ secs



$t/n \mu$ sec



- Single instr is no faster.
- speed up is actually  $t / \text{time of slowest stage.}$

$n$  times faster

# Memory Hierarchy

- Caching
- Virtual Memory

What kinds of memory are in a computer system?

Fast	Registers	Fractions of nano-seconds	
	L1 Cache		
	L2 Cache		
	(L3 Cache)		X100
	System Memory (R.A.M)	10's of nanoseconds	10000-100000
	Solid State Disks / Flash Memory	~ milliseconds	
	Rotating Disks	10's of ms	10-100
slow	Tapes	Minutes / Hours	

Processor

Small, Fast

\$\$\$\$

Cache

System memory

Perf

Big, Slow

Disks

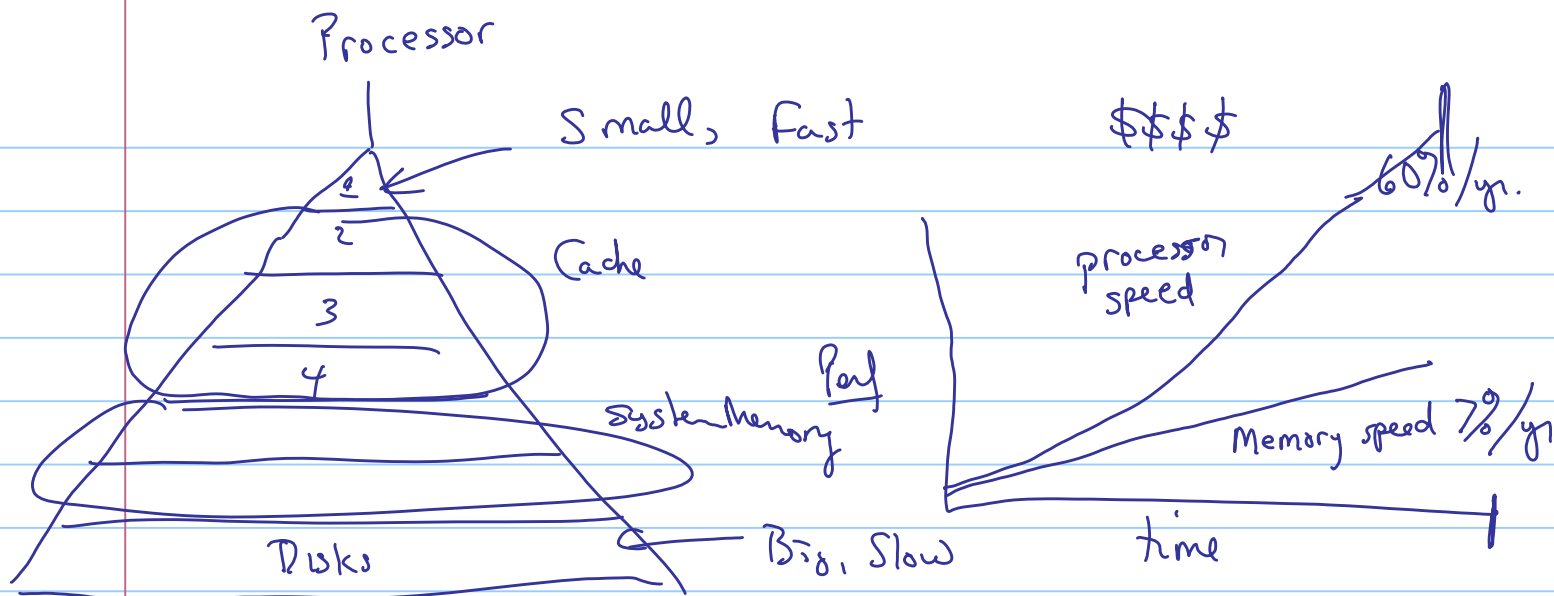
processor speed

Memory speed 7%/yr

time

2006

Processors hit a wall.





## General Approach to Caching -

- Cache is a copy of something stored lower in the hierarchy.

- it works because of spatial and temporal locality

Key idea behind caching.

