
 BROWN

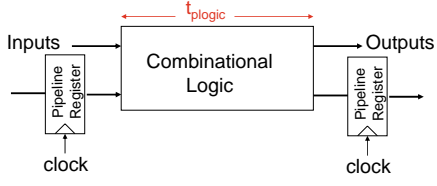
## Computer System Design Lecture 9: Pipeline Hazards

Prof. R. Iris Bahar  
EN164  
February 14, 2007

Reading: Appendix A, section A.2, A.3

 BROWN


## Designing the Pipeline Register



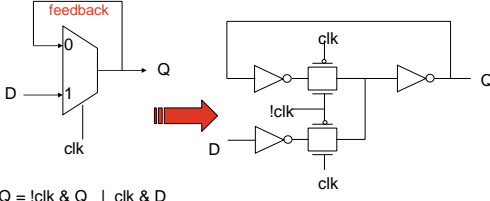
*When is input data latched by the pipeline register?*

*When are outputs of the pipeline register made visible to the next stage?*

EN164  
Lecture 8-2

 BROWN

## MUX Based Latches




$Q = !clk \& Q \mid clk \& D$

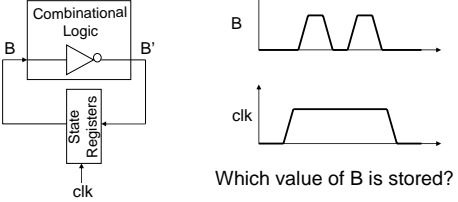
Data passes to output when clock is high

Latch is either in *transparent* or *hold* phase

EN164  
Lecture 8-3

 BROWN


## Latch Race Problem



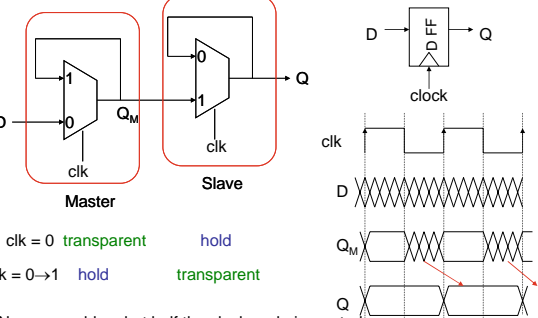
Which value of B is stored?

What we really want is to allow output of latch to change only on a clock edge (e.g., only when  $clk$  switching from 0  $\rightarrow$  1 )

EN164  
Lecture 8-4

 BROWN


## Master Slave Edge-Triggered Flip-flop



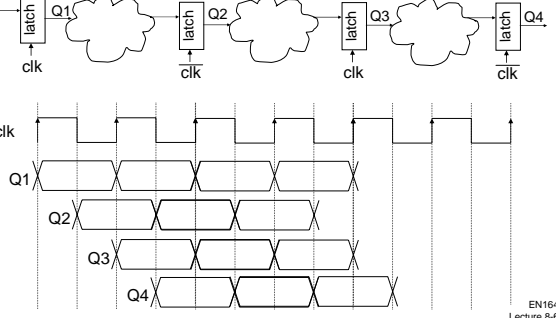
$clk = 0$  transparent      hold  
 $clk = 0 \rightarrow 1$  hold      transparent

No race problem but half the clock cycle is wasted

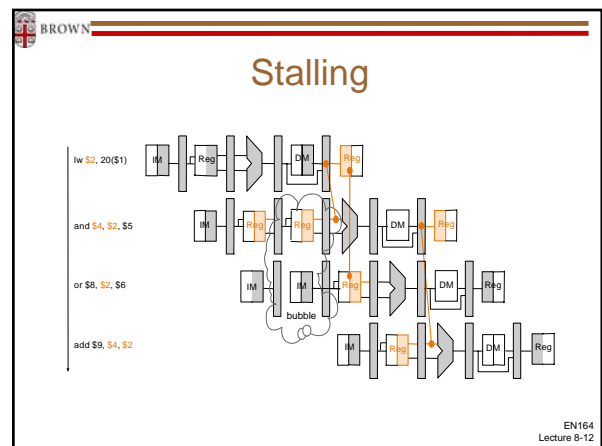
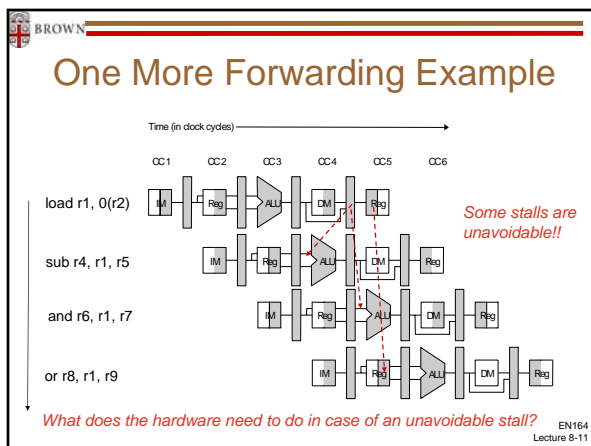
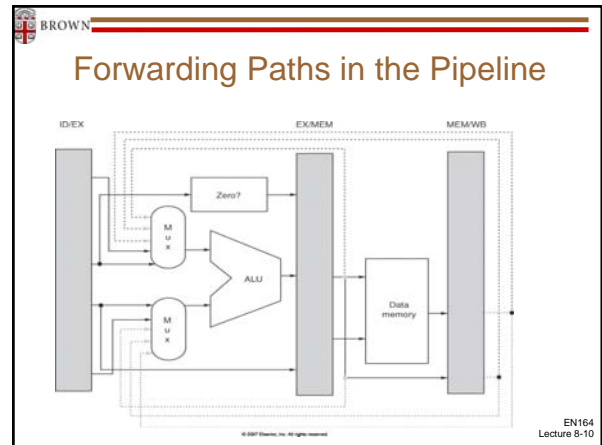
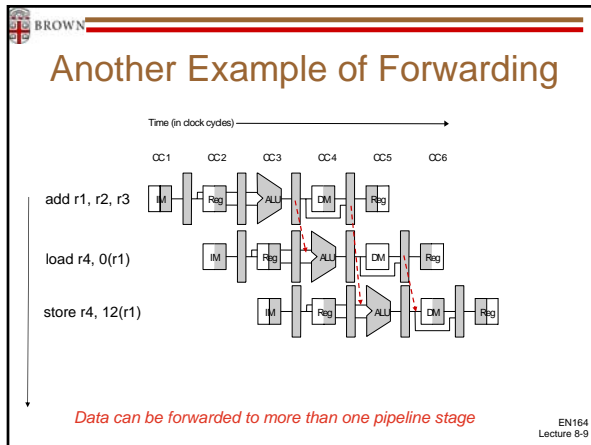
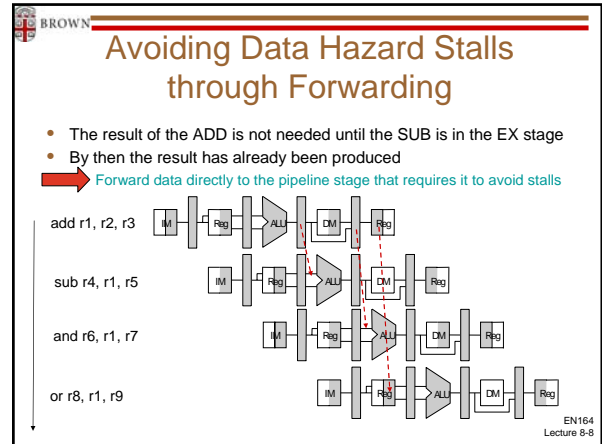
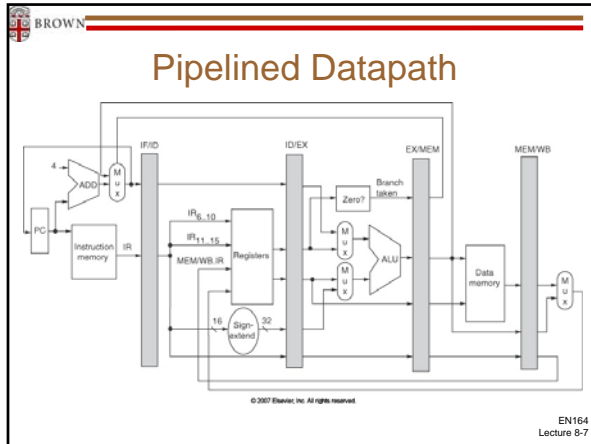
EN164  
Lecture 8-5


 BROWN

## Latches in a Pipeline



EN164  
Lecture 8-6




 BROWN

## Slowdown from Stalls

- Perfect pipelining with no hazards
  - ➔ an instruction completes every cycle (total cycles ~ num instructions)
  - ➔ speedup = increase in clock speed = num pipeline stages
- With hazards and stalls, some cycles (= stall time) go by during which no instruction completes, and then the stalled instruction completes
  - ➔ Total cycles = number of instructions + stall cycles
- Slowdown because of stalls =  $1 / (1 + \text{stall cycles per instruction})$

EN164  
Lecture 8-14

 BROWN

## Detecting Data Dependencies

Situation	Example code	Action
No dependence	LD R1, 45(R2) DADD R5, R6, R7 DSUB R8, R6, R7 OR R9, R6, R7	No hazards
Dependence requiring stall	LD R1, 45(R2) DADD R5, R1, R7 DSUB R8, R6, R7 OR R9, R6, R7	Detect use of R1 during ID of DADD and stall: Insert NOP in ID/EX pipeline register Recirculate contents of IF/ID register
Dependence overcome by forwarding	LD R1, 45(R2) DADD R5, R6, R7 DSUB R8, R1, R7 OR R9, R6, R7	Detect use of R1 during ID of DSUB and set mux control signal that accepts result from bypass path
Dependence with accesses in order	LD R1, 45(R2) DADD R5, R6, R7 DSUB R8, R6, R7 OR R9, R1, R7	No action required (dependencies are far enough apart)

EN164  
Lecture 8-15