

### System Timing with FFs

Clock Period =  $T_{c2q} + T_{cl} + T_{su} + T_{skew}$

$T_{c2q}$  : Clock to Q prop delay  
 $T_{cl}$  : max delay of combinational logic  
 $T_{su}$  : setup time of clock  
 $T_{skew}$  : max clock skew

Clock skew is the difference in clock edge arrival times at different FFs in the system. If Clock arrives  $T_{skew}$  early, then  $T_{skew}$  adds to  $T_{su}$ . If clock arrives  $T_{skew}$  late, then  $T_{skew}$  adds to  $T_{c2q}$ . Either way, additional delay is  $T_{skew}$ .

BR 9/04 5

### System Timing with Latches

At first glance, timing looks like (assume logic of FF example was broken evenly between two stages so total logic delay is same as FF example)

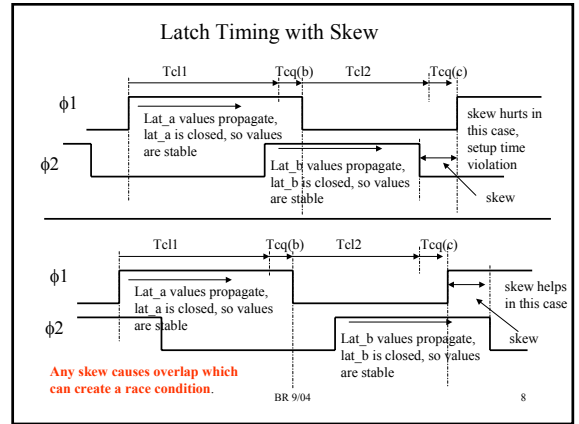
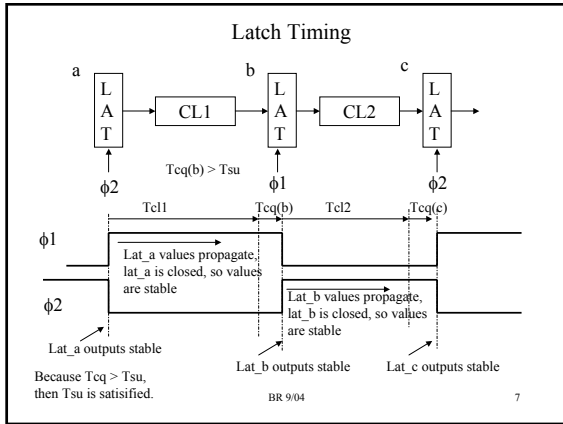
Clock Period =  $C2Q$  (first latch) +  $T_{logic}$  +  $T_{su}$  +  $C2Q$  (middle latch) +  $T_{su}$  +  $T_{skew}$

Is  $T_{su}$  really in this equation?

$T_{logic} = T_{CL1} + T_{CL2}$ , where  $T_{CL1} = T_{CL2}$

Also, typically  $C2Q > T_{su}$

BR 9/04 6

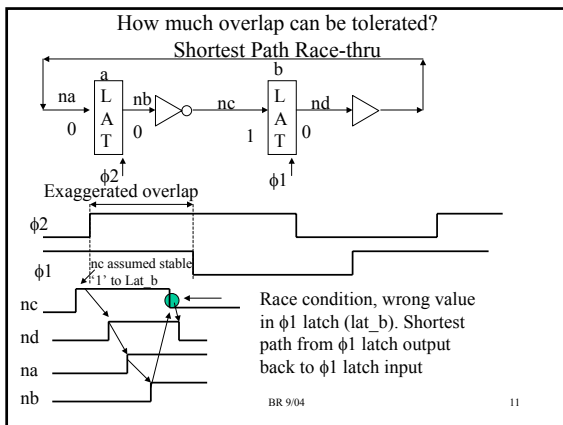
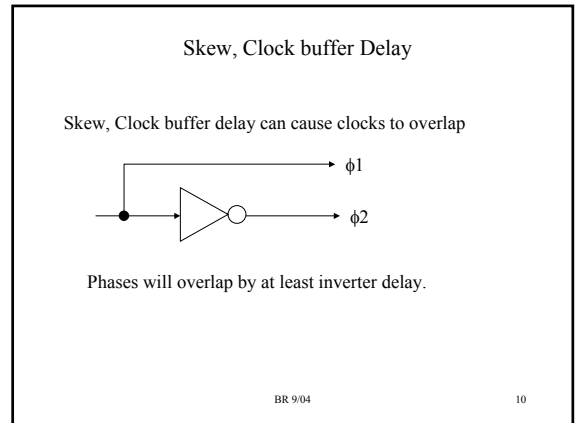


### System Timing with Latches (actual)

Actual latch equation is:  
 $\text{Clock Period} = 2 * C2Q + T_{logic} + T_{skew}$  ← assume harmful skew

$T_{logic} = T_{CL1} + T_{CL2}$ , where  $T_{CL1} = T_{CL2}$   
 $T_{cq} > T_{su}$

BR 9/04 9



### How much overlap can be tolerated?

Need to satisfy:

$$2 * T_{c2q} + T_{cl1} (\text{min path}) + T_{cl2} (\text{min path}) + T_{su} > T_{\text{overlap}}$$

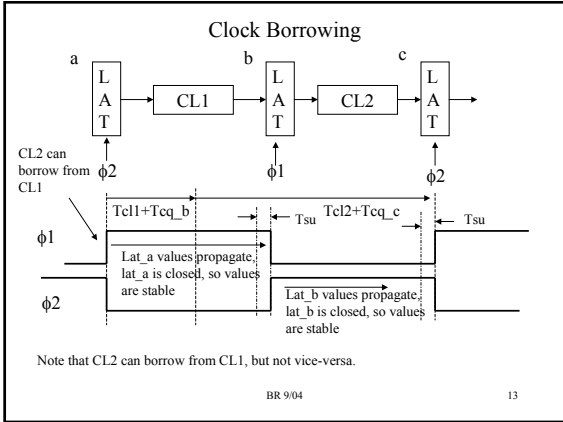
Let  $T_{cl} (\text{min path}) = T_{cl1} (\text{min path}) + T_{cl2} (\text{min path})$ .

If all of overlap due to  $T_{skew}$ , then:

$$2 * T_{c2q} + T_{cl} (\text{min path}) + T_{su} > T_{skew}$$

$$T_{cl} (\text{min path}) > T_{skew} - 2 * T_{c2q} - T_{su}$$

BR 9/04 12



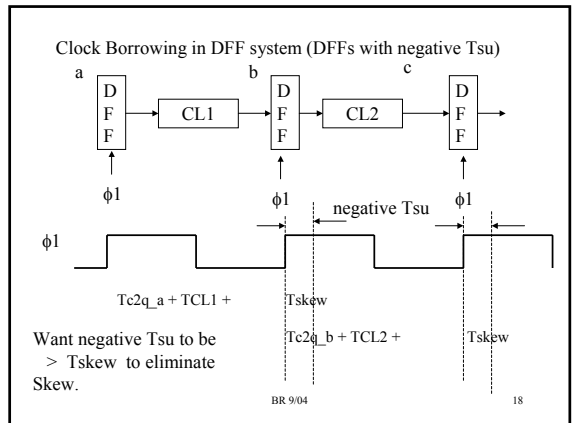
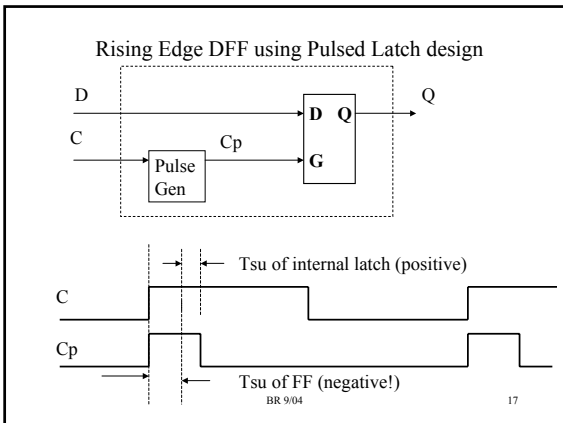
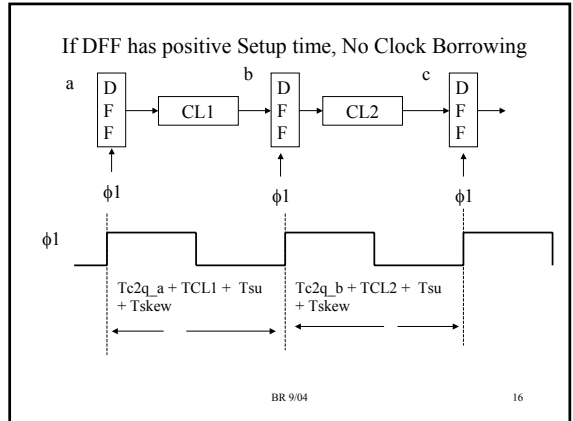
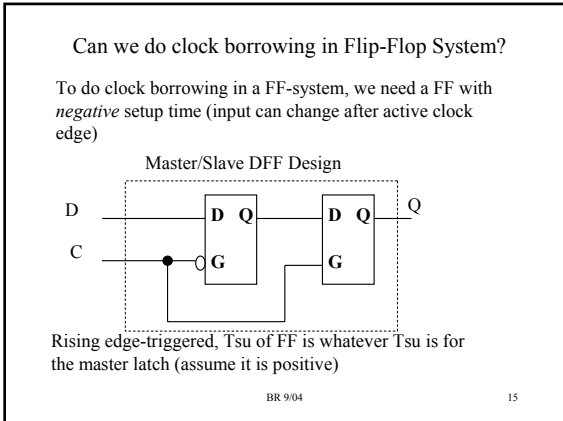
### Clock Borrowing can be useful

Not always possible to perfectly balance logic delays in each stage.

Latches allow us to do clock borrowing.

Helps alleviate the pipeline balancing problem.

BR 9/04 14



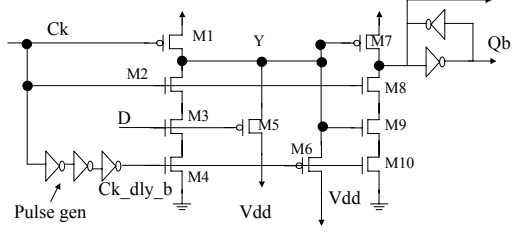
### Timing Summary

- Latch-based System
  - Allows clock borrowing
  - System timing is:
    - Clock period =  $T_{cl}(\text{phase1}) + T_{cl}(\text{phase2}) + 2 * T_{c2q} + T_{skew}$
  - If clock phase overlap, then:
    - $T_{cl}(\text{min path of } T_{cl1}, T_{cl2}) > T_{skew} - 2 * T_{cq} - T_{su}$
- DFF-based System
  - DFFs with positive Setup:
    - Clock Period =  $T_{c2q} + T_{cl}(\text{max}) + T_{su} + T_{skew}$
  - DFFs with negative Setup:
    - Clock period =  $T_{cq} + T_{cl}(\text{max})$
    - if  $T_{su}(\text{negative}) > T_{skew}$

BR 9/04

19

### Pulsed Latch Design (AMD-K6)

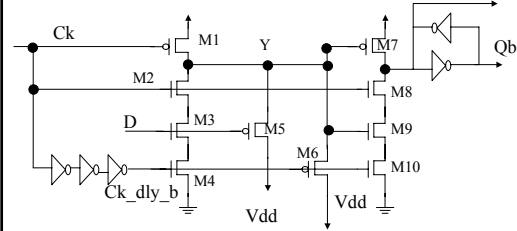


$Ck = 0 \rightarrow M1 \text{ on, } Y = V_{dd} \rightarrow M7 \text{ off} \rightarrow M8 \text{ off} \rightarrow Q = Q_{old}$   
 $Ck\_dly\_b = 1 \rightarrow M4 \text{ on} \rightarrow M10 \text{ on, } M6 \text{ off}$

BR 9/04

20

### Pulsed Latch Design (AMD-K6)

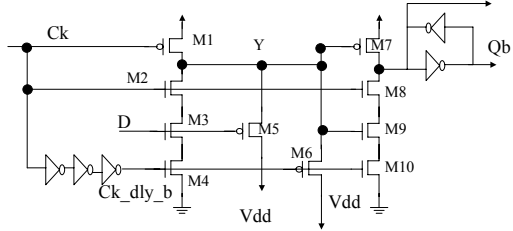


$Ck \ 0 \rightarrow 1 \rightarrow M1 \text{ off} \rightarrow D \text{ affects } Y \rightarrow Q \text{ new} = \text{Not}(Y)$   
 $Ck\_dly\_b = 1 \rightarrow M2 \text{ on, } M4 \text{ on, } M6 \text{ off} \rightarrow M10 \text{ on}$

BR 9/04

21

### Pulsed Latch Design (AMD-K6)

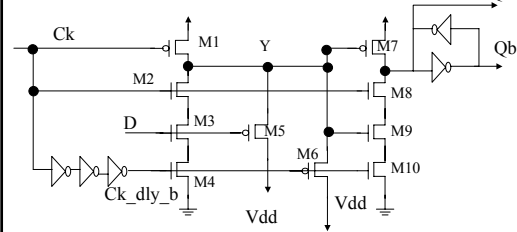


$Ck \ 1 \rightarrow M1 \text{ off} \rightarrow M2 \text{ on} \rightarrow Y = V_{dd} \rightarrow Q = Q_{old}$   
 $Ck\_dly\_b = 0 \rightarrow M4 \text{ off, } M6 \text{ on, } M10 \text{ off}$

BR 9/04

22

### Pulsed Latch Design (AMD-K6)

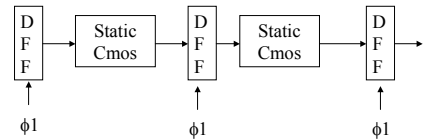


$Ck \ 1 \rightarrow 0 \rightarrow M1 \text{ on} \rightarrow Y = V_{dd} \rightarrow Q = Q_{old}$   
 $Ck\_dly\_b = 0 \rightarrow M2 \text{ off, } M8 \text{ off, } M4 \text{ off, } M6 \text{ on, } M10 \text{ off}$

BR 9/04

23

### Pipelined DFF System (pulsed Latches) with Static CMOS



Clock period =  $T_{cq} + T_{cl}(\text{max})$

if  $T_{su}(\text{negative}) > T_{skew}$

Desireable Feature: want DFFs with low latency (small  $T_{cq}$ )

BR 9/04

24

### Pipelined Latch System with Static CMOS

Clock Period =  $2 * T_{c2q} + T_{cl}(\text{max path})$

If clock phase overlap, then:

$T_{cl}(\text{min path}) > T_{skew} - T_{cq} - T_{su}$  (to avoid race thru)

BR 9/04 25

### Level-Sensitive Latch Design

	Half-cycle 1	Half-cycle 2
Traditional		
TSPC latch		

TSPC = True Single Phase Clocking (clock does not have to be inverted), use on Alpha 21064 (first Alpha, one LARGE clock driver). TSPC latches slower than traditional latches, easy to invert clock locally. BR 9/04 26

### Flip-Flop Design

- Can combine TSPC Latches in several ways to make a Flip-Flop

Node X is a dynamic node. If clock = 1 and D=0, then node is floating! Lowers noise immunity, susceptible to charge sharing, charge coupling. Note that this FF has a long latency.

(a) Positive edge-triggered D flip-flop BR 9/04 27

Charge coupling problem on this node. Node is truly dynamic, will be a case where it is not driven.

(b) Negative edge-triggered D flip-flop BR 9/04 28

### What is Charge Coupling?

Charge coupling is where a signal transition on one node causes a voltage change on another node via a coupling capacitor.

Charge coupling between wires due to sidewall capacitance or layer-to-layer capacitance

Charge coupling between gate and source/drain due to  $C_{gs}$ ,  $C_{gd}$ .

BR 9/04 29

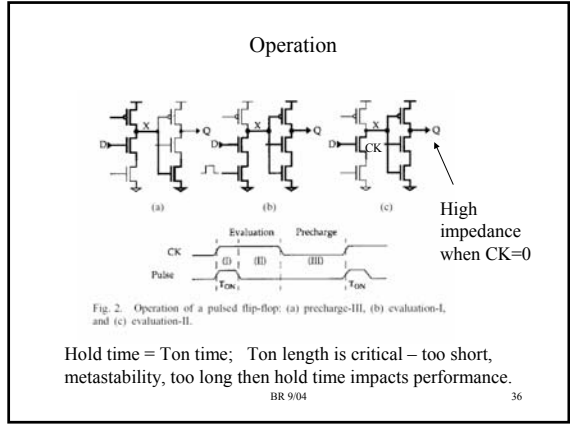
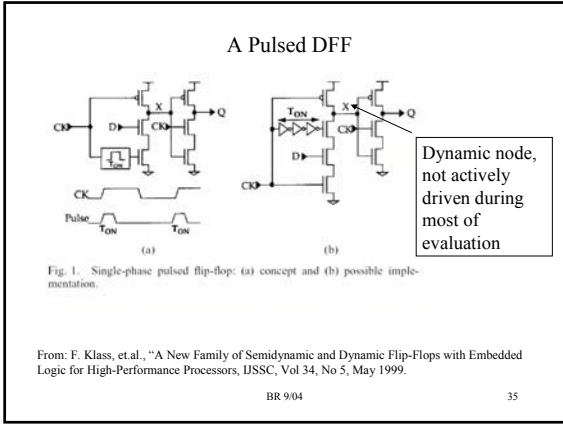
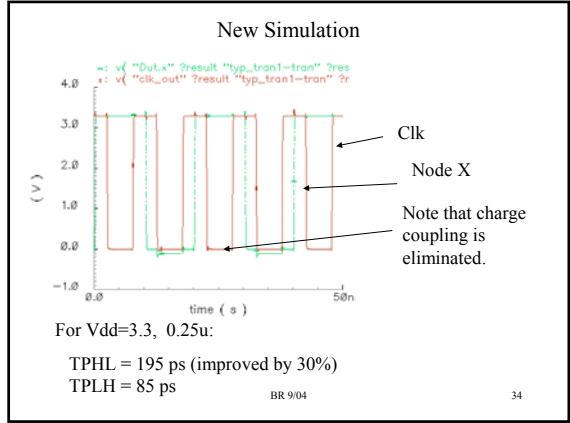
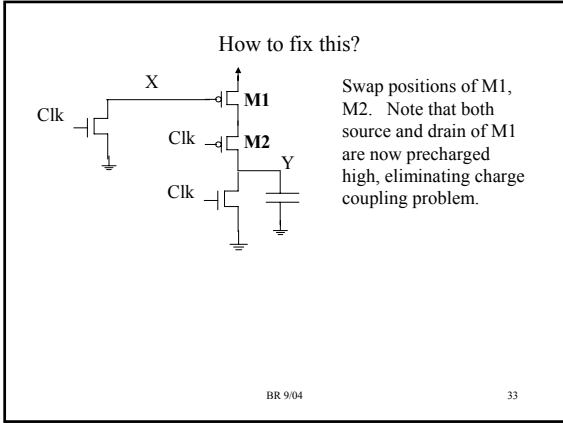
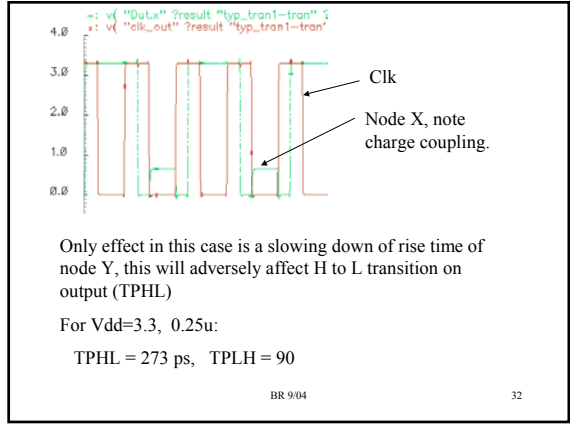
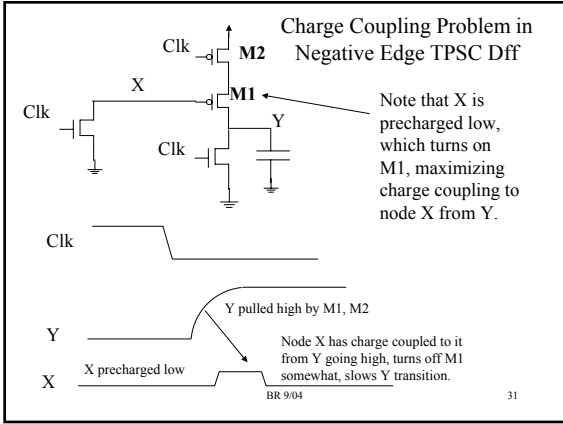
### Gate and Source/Drain Coupling

When channel is off,  $C_{gs} = C_{gd} = 0$

When channel is saturated (on),  $C_{gs}$ ,  $C_{gd}$  is maximized.

Gate and Source/Drain coupling is a problem when the coupled node is not being actively driven (which is the case for a dynamic node).

BR 9/04 30



### Improved Design

Keeper gets rid of dynamic node.

Keeper ensures output always actively driven.

Suitable for interfaces to static logic. Note that output is stable through precharge.

Fig. 3. Semidynamic edge-triggered flip-flop.

BR 9/04 37

### Why NAND Gate?

Pull-down path conditional with value of D.  
If D=1, then X=0, making S=1

Fig. 3. Semidynamic edge-triggered ff

In Precharge, CK=0, CKD=0, X=1, S=H, N1 = ON.

At Eval start, CK=1 (N3 on), CK=0, X=1. If D=1, then N2 on, so X pulled low. X going low forces S to stay high, resulting in more robust pull-down of node X. This is called "conditional shutoff" – N1 is not shutoff if D=1.

BR 9/04 38

### Spice Simulation Results

With conditional shutoff. Delay from CK to CKD decreased from 100ps to 10 ps in 10 ps steps (decreasing TON sampling window)

Original circuit without conditional shutoff. Note failure of pull-down of node X.

Conditional shutoff allows smaller TON times.

BR 9/04 39

### Embedded Logic

Fig. 5. SDF with embedded  $(A + B)(C + D) + (E + F)(G + H)$  function.

Embedded logic reduces number of logic stages between registers, but adds latency to DFF. Tradeoff is generally in favor of embedding logic in the DFF.

BR 9/04 40

### Dynamic DFF

Fig. 6. Single-rail dynamic flip-flop with embedded  $(A + B + C + D)$  function.

Used for interfacing to dynamic logic – output forced to a '0' during precharge.

BR 9/04 41

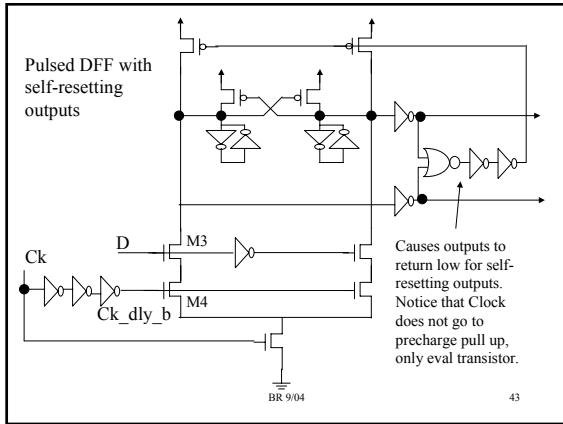
### Dynamic Dual Rail DFF

Weak keeper that keeps outputs at logic '1'. Do not need full keeper since outputs precharged to a '0'.

Fig. 7. Dual-rail dynamic flip-flop.

Useful when true and complement outputs are required for interfacing to dynamic logic because dynamic logic cannot implement inverting logic, so dual rail inputs are required.

BR 9/04 42



### Self-Resetting CMOS (SRCMOS)

- SRCMOS differs from Domino logic in that the reset signal is generated locally
  - Less clock load
- Outputs are pulsed 0→1 → 0, inputs assumed pulsed as well
  - Because inputs are '0' at start of evaluation, then no need for evaluation transistor!
  - No evaluation transistor means one less transistor in pulldown path means a faster circuit
- Many schemes for generating reset
  - Delayed version of the clock
  - Generate reset based on dual rail outputs
  - Generate reset for an entire block of gates at one time
- Will look at this in more detail for decode circuits for RAMs.

BR 9/04

44