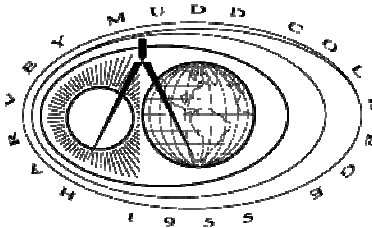


# Introduction to CMOS VLSI Design

## Lecture 10: Sequential Circuits

David Harris



Harvey Mudd College  
Spring 2004

# Outline

---

- Floorplanning
- Sequencing
- Sequencing Element Design
- Max and Min-Delay
- Clock Skew
- Time Borrowing
- Two-Phase Clocking

# Project Strategy

- ❑ Proposal
  - Specifies inputs, outputs, relation between them
- ❑ Floorplan
  - Begins with block diagram
  - Annotate dimensions and location of each block
  - Requires detailed paper design
- ❑ Schematic
  - Make paper design simulate correctly
- ❑ Layout
  - Physical design, DRC, NCC, ERC

# Floorplan

---

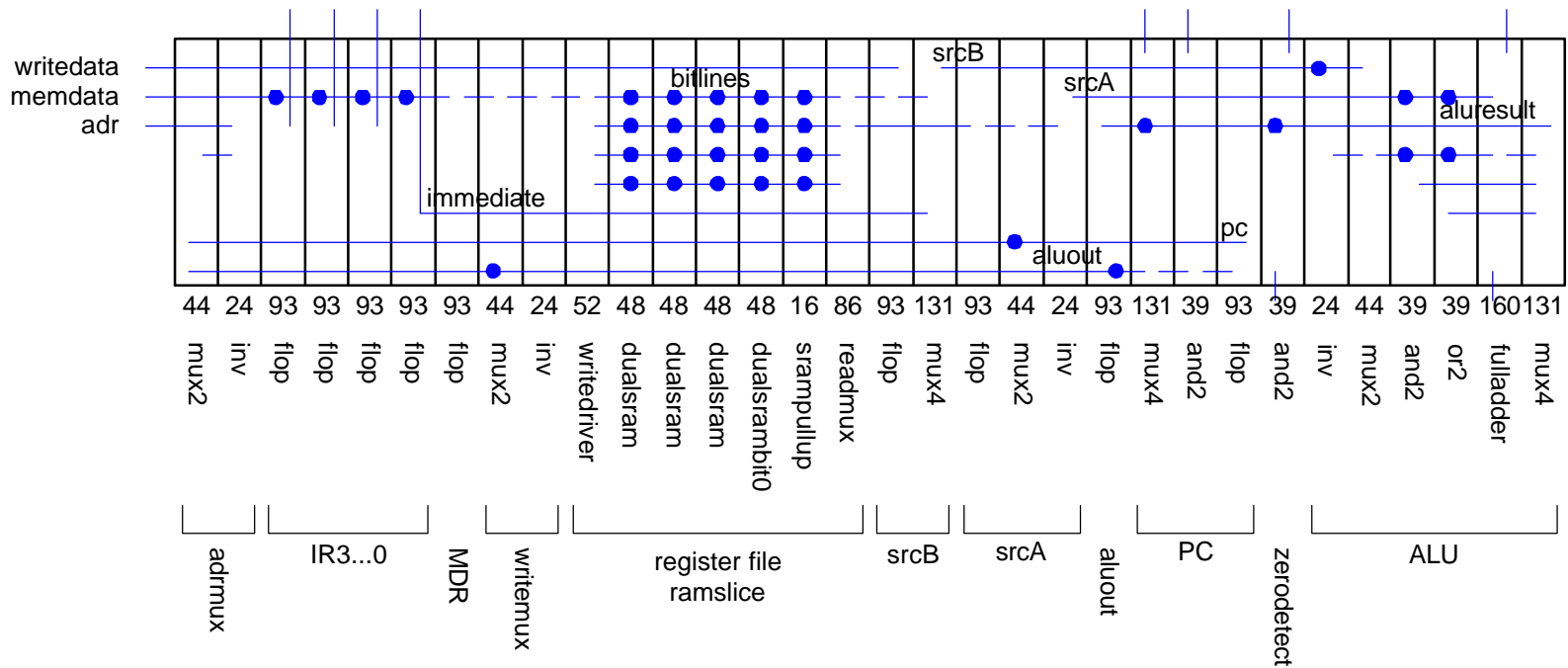
- ❑ How do you estimate block areas?
  - Begin with block diagram
  - Each block has
    - Inputs
    - Outputs
    - Function (draw schematic)
    - Type: array, datapath, random logic
- ❑ Estimation depends on type of logic



# Area Estimation

- ❑ Arrays:
  - Layout basic cell
  - Calculate core area from # of cells
  - Allow area for decoders, column circuitry
- ❑ Datapaths
  - Sketch slice plan
  - Count area of cells from cell library
  - Ensure wiring is possible
- ❑ Random logic
  - Compare complexity do a design you have done

# MIPS Slice Plan



# Typical Layout Densities

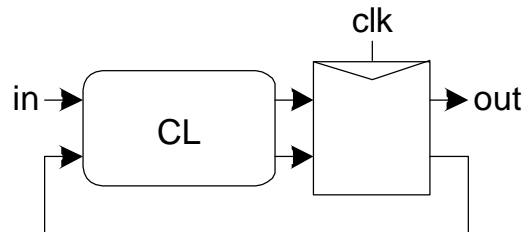
- ❑ Typical numbers of high-quality layout
- ❑ Derate by 2 for class projects to allow routing and some sloppy layout.
- ❑ Allocate space for big wiring channels

Element	Area
Random logic (2 metal layers)	1000-1500 $\lambda^2$ / transistor
Datapath	250 – 750 $\lambda^2$ / transistor Or 6 WL + 360 $\lambda^2$ / transistor
SRAM	1000 $\lambda^2$ / bit
DRAM	100 $\lambda^2$ / bit
ROM	100 $\lambda^2$ / bit

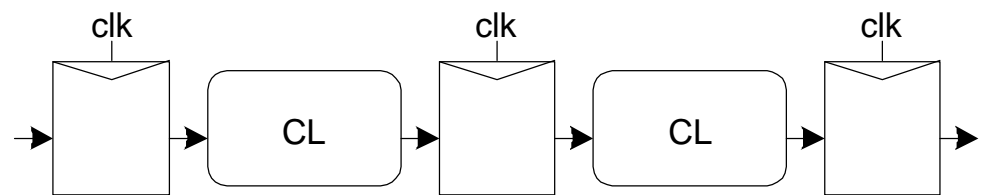


# Sequencing

- ❑ *Combinational logic*
  - output depends on current inputs
- ❑ *Sequential logic*
  - output depends on current and previous inputs
  - Requires separating previous, current, future
  - Called *state* or *tokens*
  - Ex: FSM, pipeline



Finite State Machine



Pipeline

# Sequencing Cont.

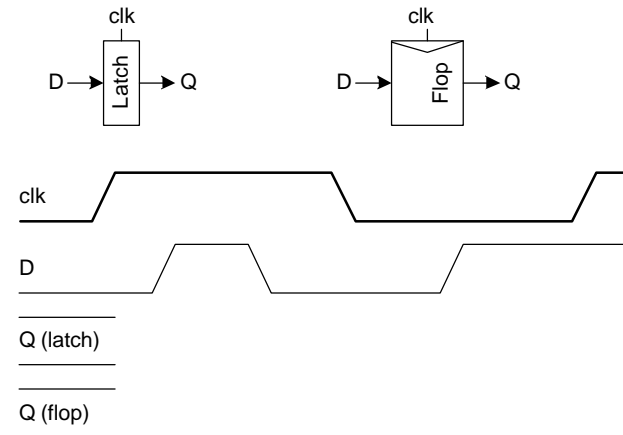
- ❑ If tokens moved through pipeline at constant speed, no sequencing elements would be necessary
- ❑ Ex: fiber-optic cable
  - Light pulses (tokens) are sent down cable
  - Next pulse sent before first reaches end of cable
  - No need for hardware to separate pulses
  - But *dispersion* sets min time between pulses
- ❑ This is called *wave pipelining* in circuits
- ❑ In most circuits, dispersion is high
  - Delay fast tokens so they don't catch slow ones.

# Sequencing Overhead

- ❑ Use flip-flops to delay fast tokens so they move through exactly one stage each cycle.
- ❑ Inevitably adds some delay to the slow tokens
- ❑ Makes circuit slower than just the logic delay
  - Called sequencing overhead
- ❑ Some people call this clocking overhead
  - But it applies to asynchronous circuits too
  - Inevitable side effect of maintaining sequence

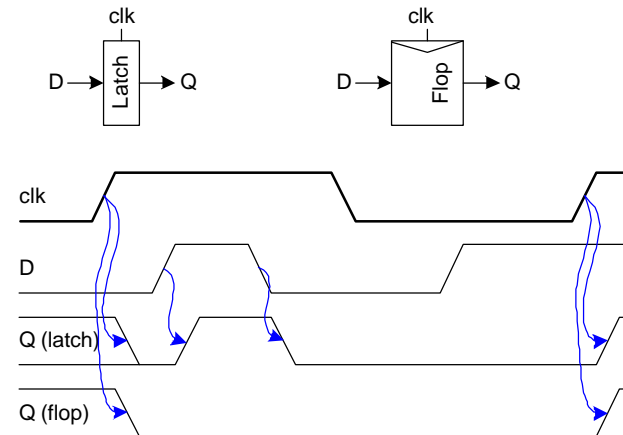
# Sequencing Elements

- ❑ **Latch:** Level sensitive
  - a.k.a. transparent latch, D latch
- ❑ **Flip-flop:** edge triggered
  - A.k.a. master-slave flip-flop, D flip-flop, D register
- ❑ **Timing Diagrams**
  - Transparent
  - Opaque
  - Edge-trigger



# Sequencing Elements

- ❑ **Latch:** Level sensitive
  - a.k.a. transparent latch, D latch
- ❑ **Flip-flop:** edge triggered
  - A.k.a. master-slave flip-flop, D flip-flop, D register
- ❑ **Timing Diagrams**
  - Transparent
  - Opaque
  - Edge-trigger



# Latch Design

- ❑ Pass Transistor Latch

- ❑ Pros

  - +

  - +

- ❑ Cons

  - 

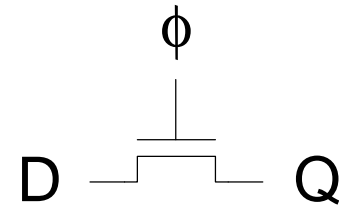
  - 

  - 

  - 

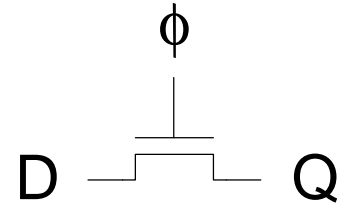
  - 

  - 



# Latch Design

- ❑ Pass Transistor Latch
- ❑ Pros
  - + Tiny
  - + Low clock load
- ❑ Cons
  - $V_t$  drop
  - nonrestoring
  - backdriving
  - output noise sensitivity
  - dynamic
  - diffusion input



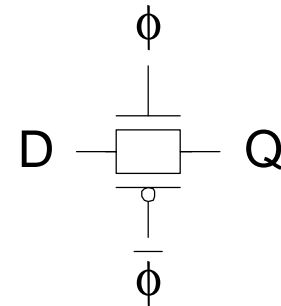
Used in 1970's

# Latch Design

□ Transmission gate

+

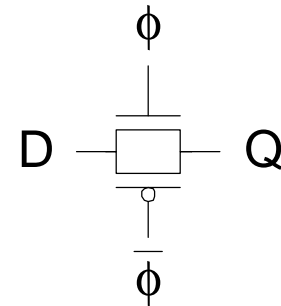
-





# Latch Design

- ❑ Transmission gate
  - + No  $V_t$  drop
  - Requires inverted clock



# Latch Design

## □ Inverting buffer

+

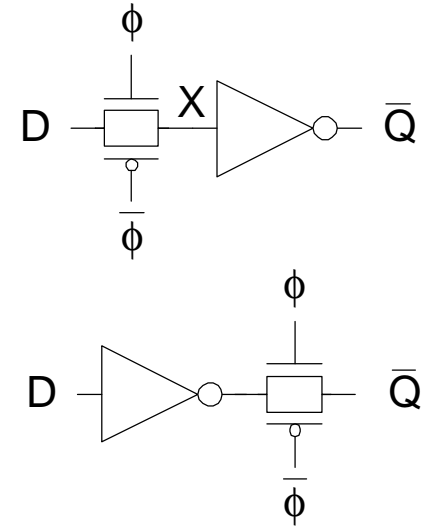
+

+ Fixes either

•

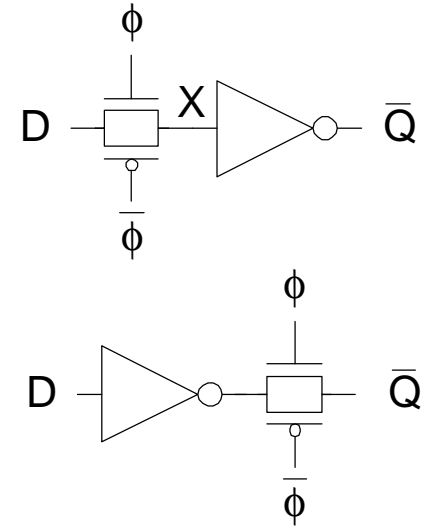
•

-



# Latch Design

- ❑ Inverting buffer
  - + Restoring
  - + No backdriving
  - + Fixes either
    - Output noise sensitivity
    - Or diffusion input
  - Inverted output

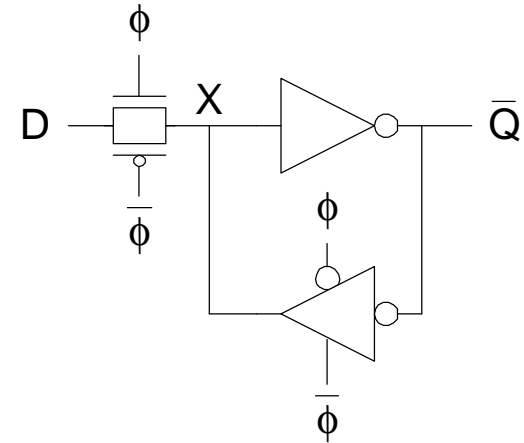


# Latch Design

❑ Tristate feedback

+

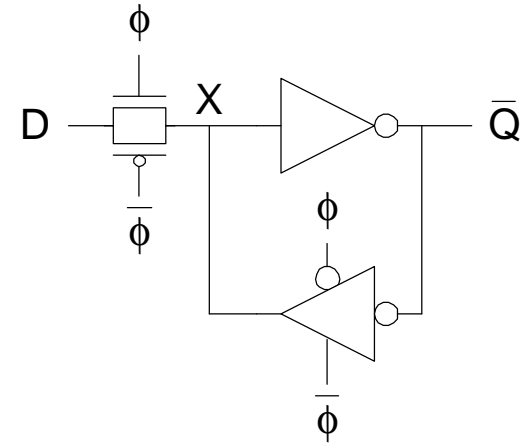
-



# Latch Design

- ❑ Tristate feedback
  - + Static
  - Backdriving risk

- ❑ Static latches are now essential

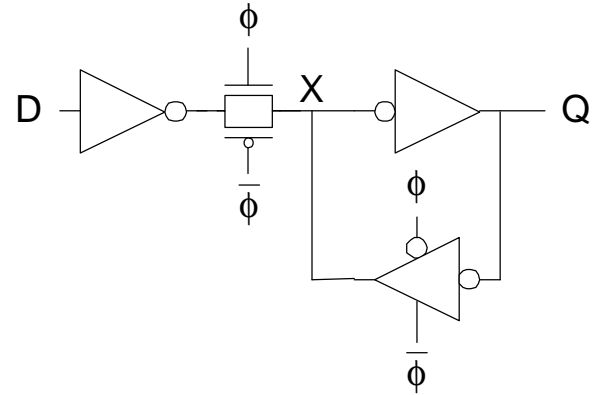


# Latch Design

□ Buffered input

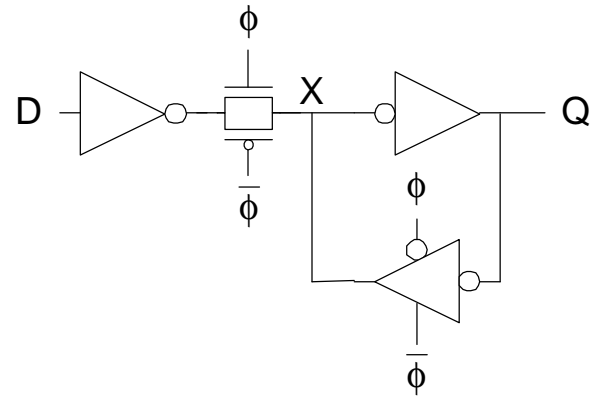
+

+



# Latch Design

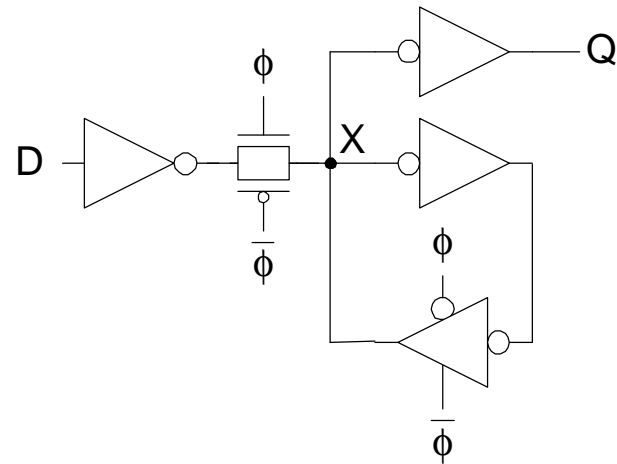
- Buffered input
  - + Fixes diffusion input
  - + Noninverting



# Latch Design

□ Buffered output

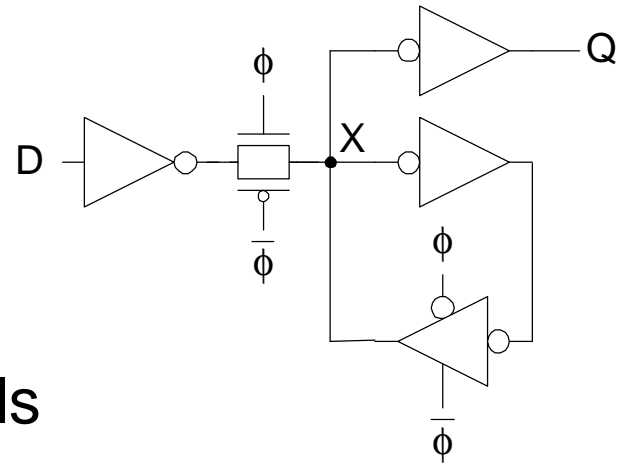
+





# Latch Design

- ❑ Buffered output  
+ No backdriving



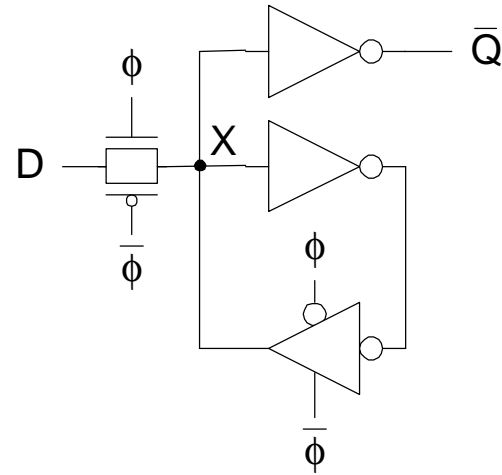
- ❑ Widely used in standard cells
  - + Very robust (most important)
  - Rather large
  - Rather slow (1.5 – 2 FO4 delays)
  - High clock loading

# Latch Design

## □ Datapath latch

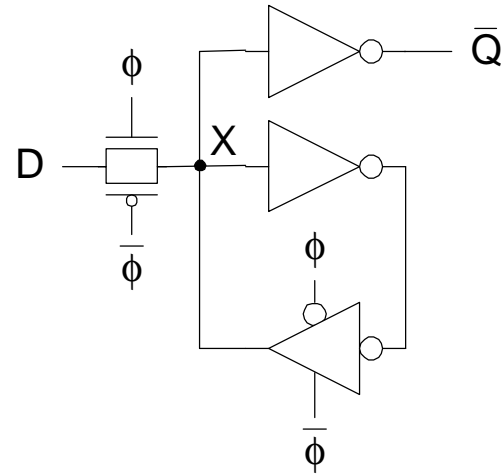
+

-



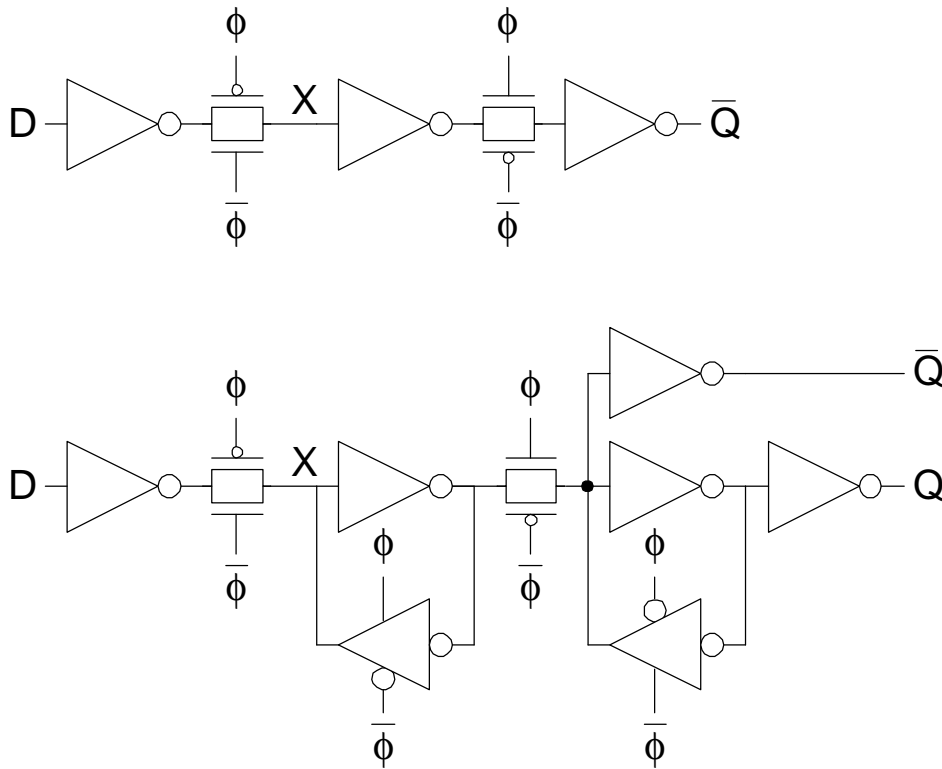
# Latch Design

- Datapath latch
  - + Smaller, faster
  - unbuffered input



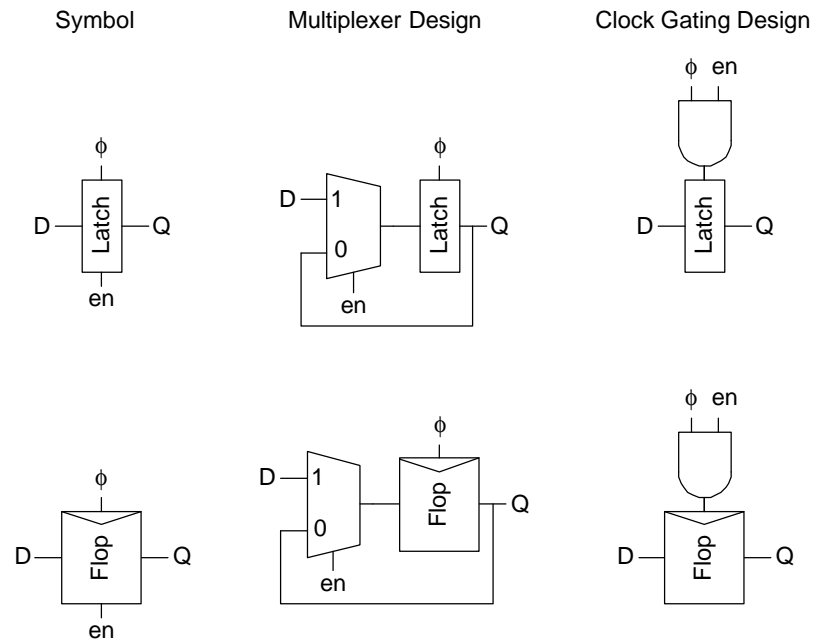
# Flip-Flop Design

- ❑ Flip-flop is built as pair of back-to-back latches



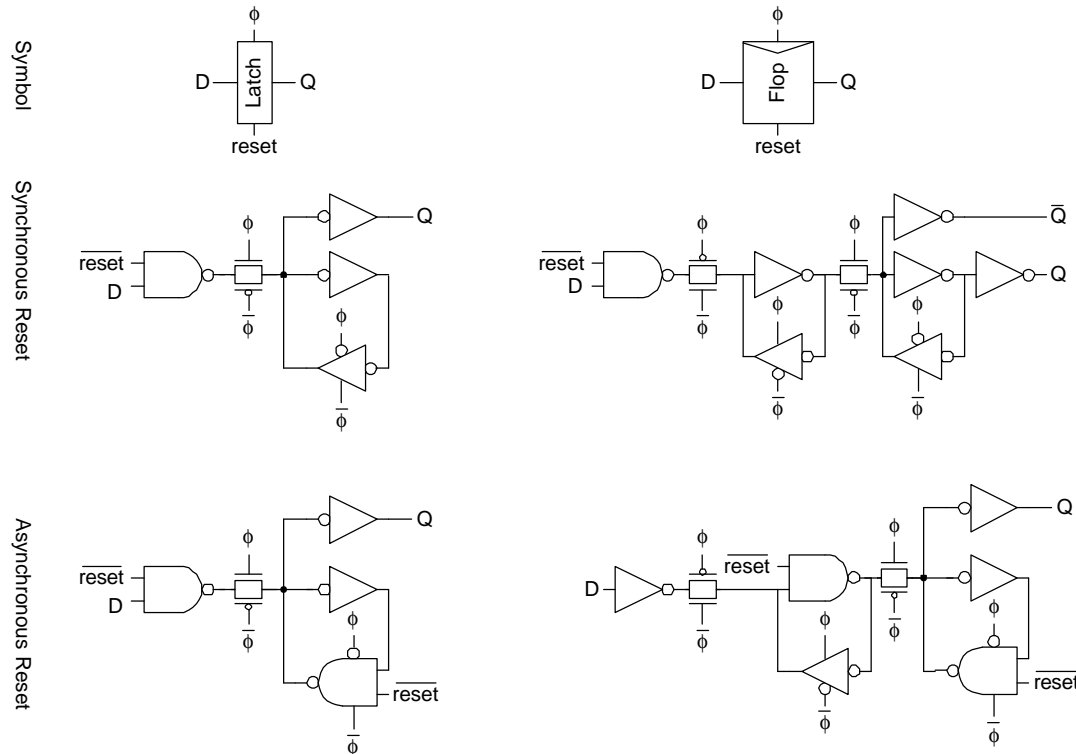
# Enable

- ❑ Enable: ignore clock when  $en = 0$ 
  - Mux: increase latch D-Q delay
  - Clock Gating: increase en setup time, skew



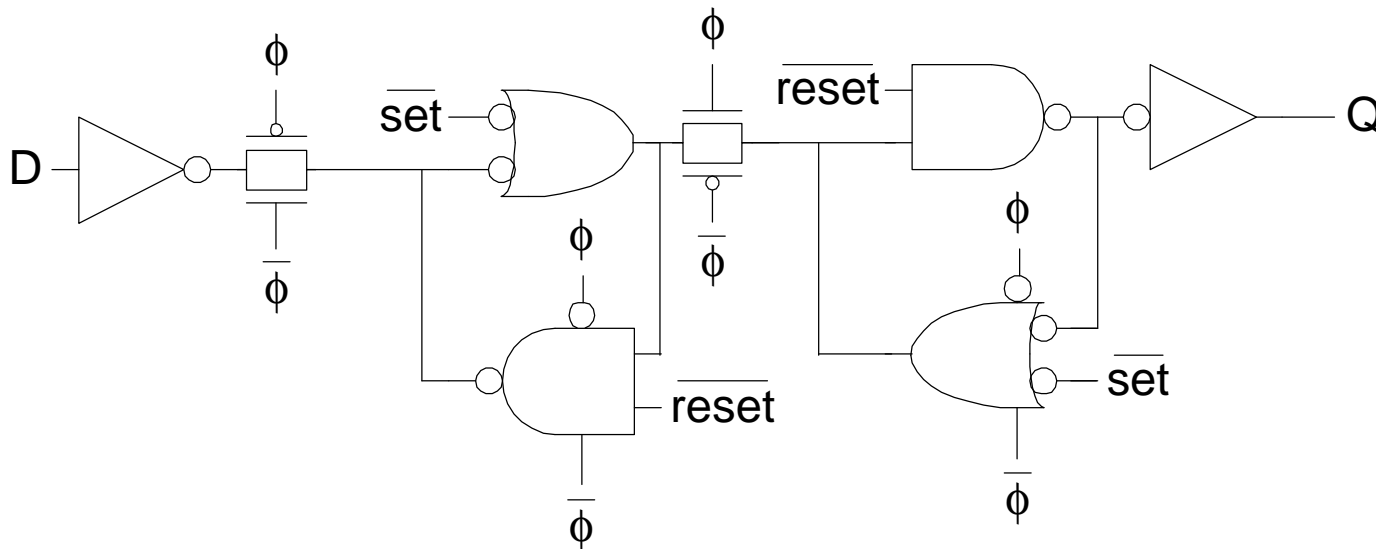
# Reset

- ❑ Force output low when reset asserted
- ❑ Synchronous vs. asynchronous



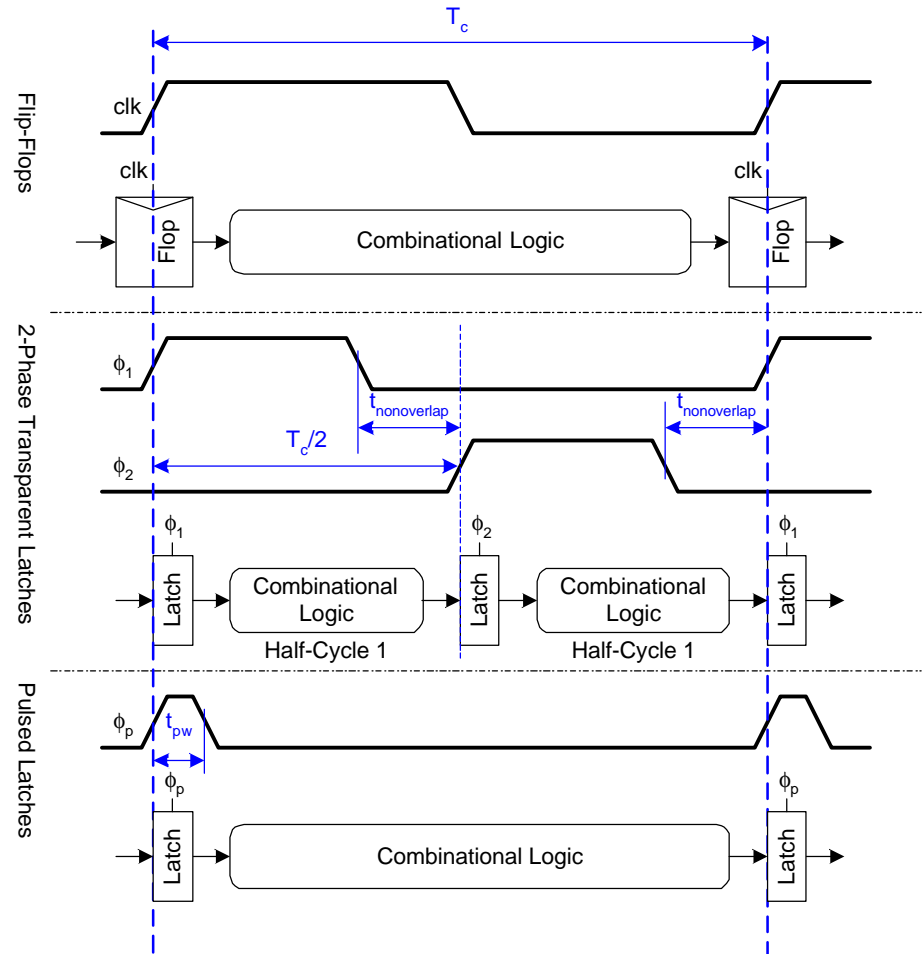
# Set / Reset

- ❑ Set forces output high when enabled
- ❑ Flip-flop with asynchronous set and reset



# Sequencing Methods

- ❑ Flip-flops
- ❑ 2-Phase Latches
- ❑ Pulsed Latches

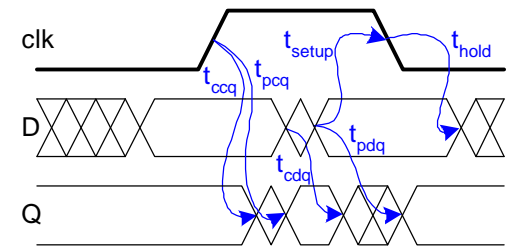
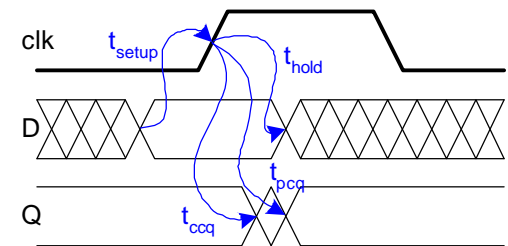
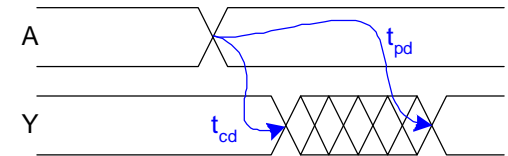
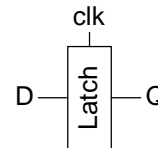
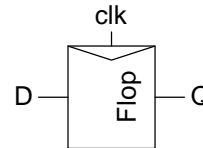
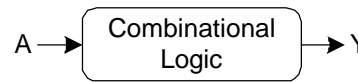




# Timing Diagrams

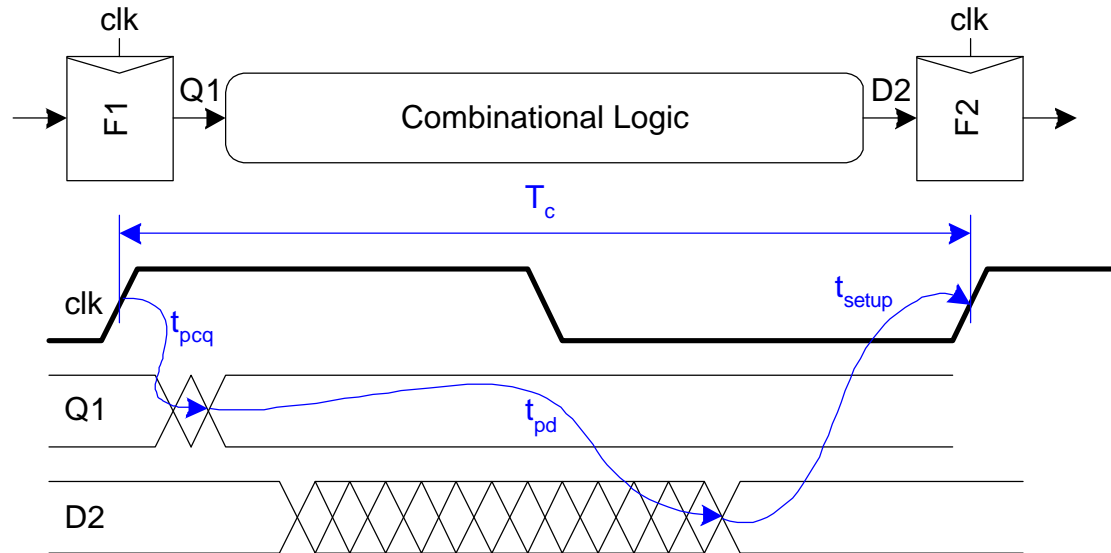
## Contamination and Propagation Delays

$t_{pd}$	Logic Prop. Delay
$t_{cd}$	Logic Cont. Delay
$t_{pcq}$	Latch/Flop Clk-Q Prop Delay
$t_{ccq}$	Latch/Flop Clk-Q Cont. Delay
$t_{pdq}$	Latch D-Q Prop Delay
$t_{pcq}$	Latch D-Q Cont. Delay
$t_{setup}$	Latch/Flop Setup Time
$t_{hold}$	Latch/Flop Hold Time



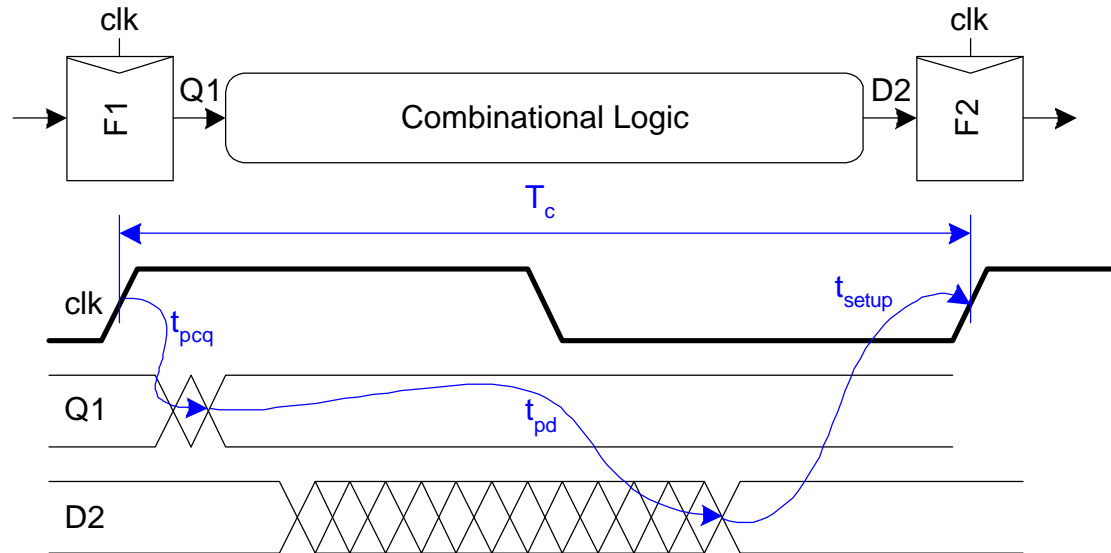
# Max-Delay: Flip-Flops

$$t_{pd} \leq T_c - \underbrace{\quad}_{\text{sequencing overhead}}$$



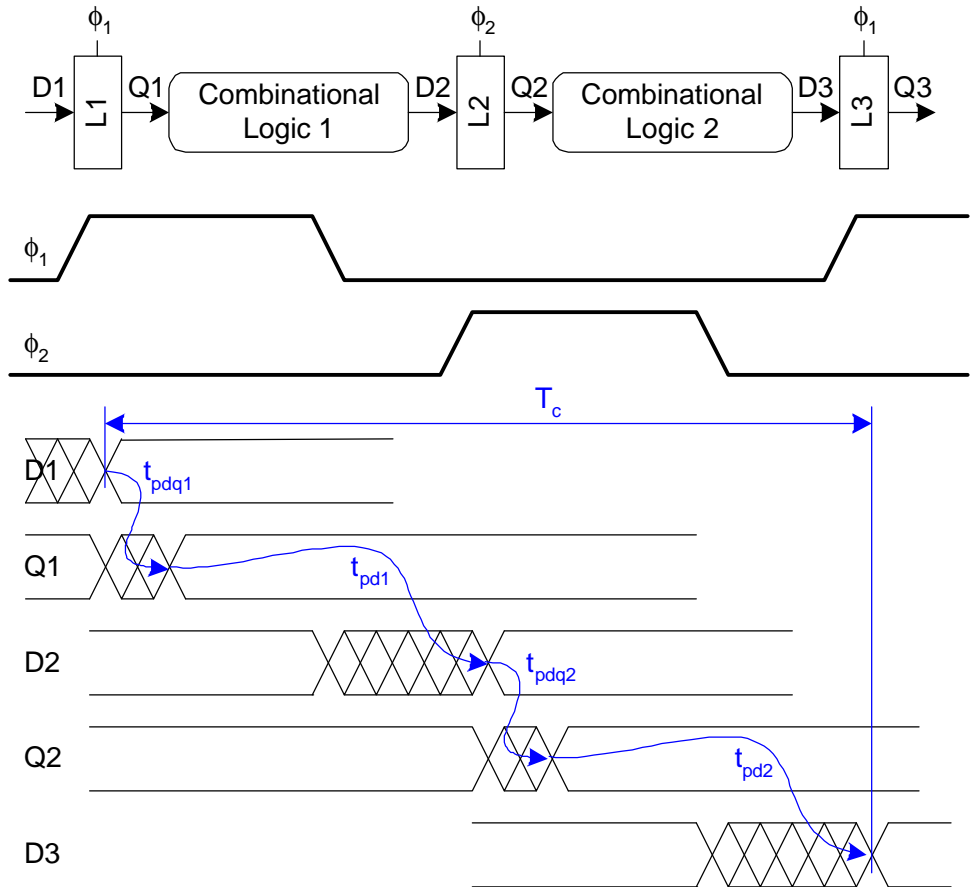
# Max-Delay: Flip-Flops

$$t_{pd} \leq T_c - \underbrace{(t_{\text{setup}} + t_{pcq})}_{\text{sequencing overhead}}$$



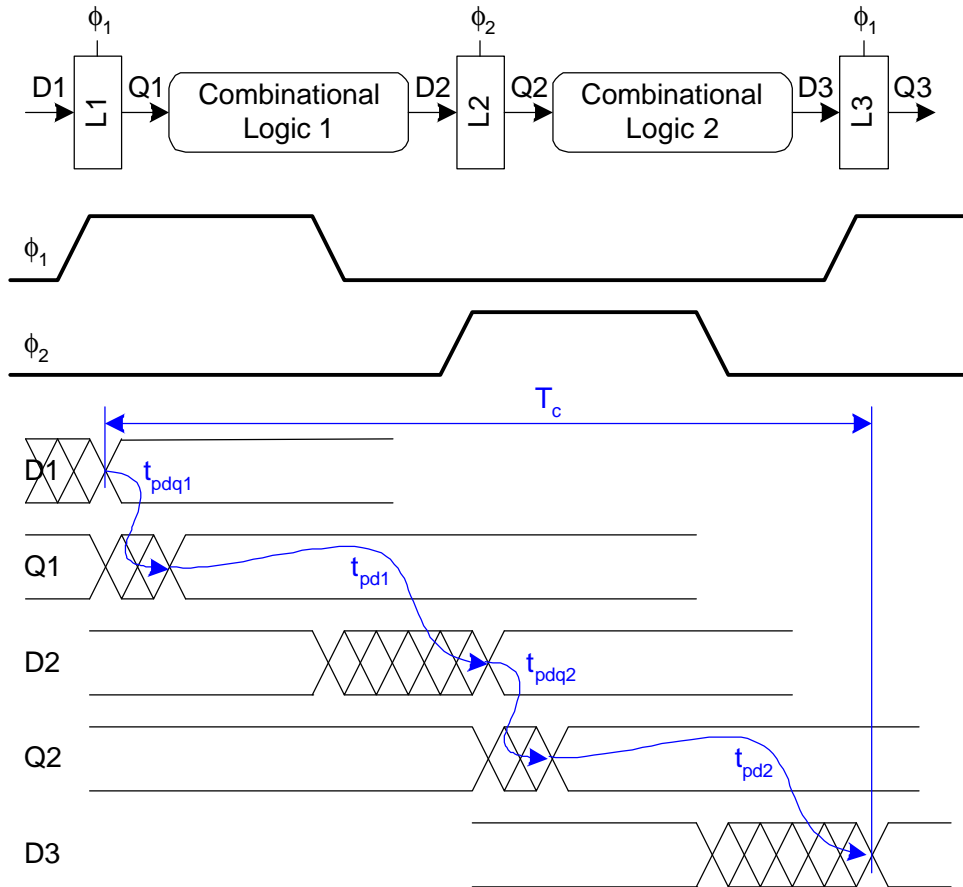
# Max Delay: 2-Phase Latches

$$t_{pd} = t_{pd1} + t_{pd2} \leq T_c - \underbrace{\hspace{2cm}}_{\text{sequencing overhead}}$$



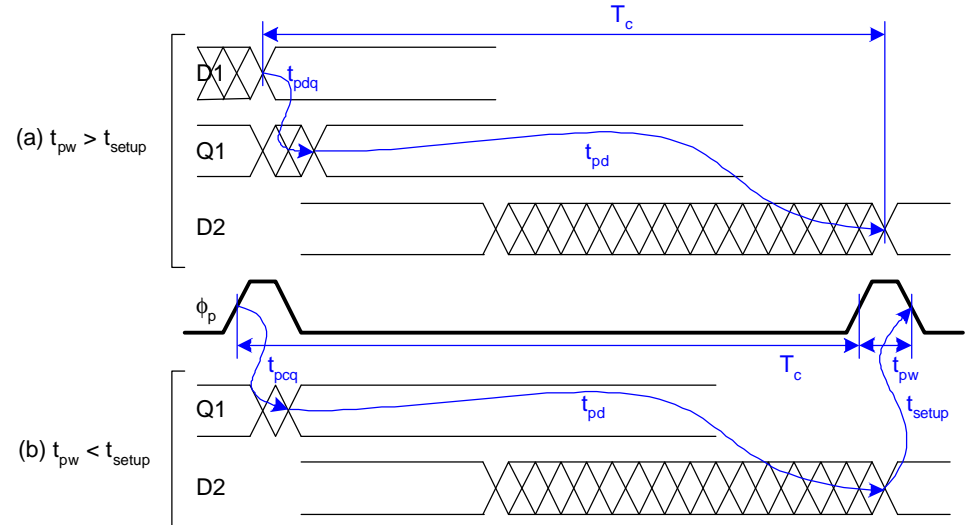
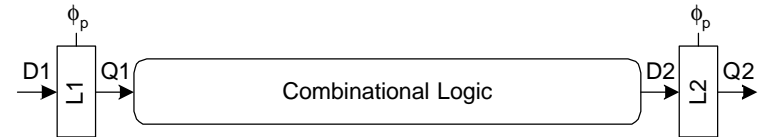
# Max Delay: 2-Phase Latches

$$t_{pd} = t_{pd1} + t_{pd2} \leq T_c - \underbrace{(2t_{pdq})}_{\text{sequencing overhead}}$$



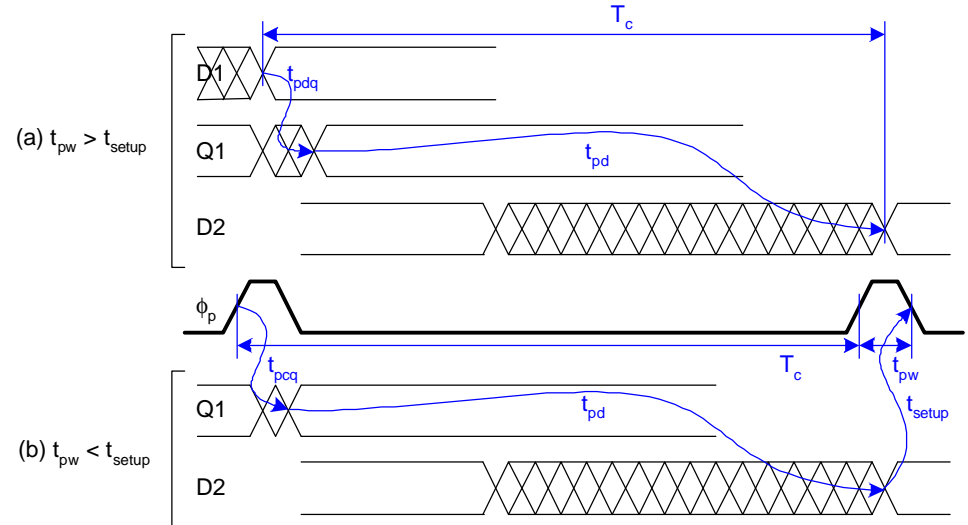
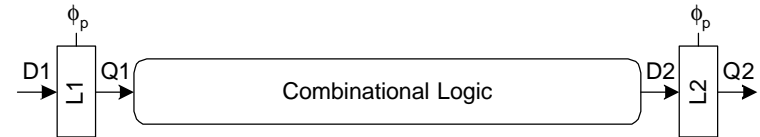
# Max Delay: Pulsed Latches

$$t_{pd} \leq T_c - \underbrace{\max(\quad)}_{\text{sequencing overhead}}$$



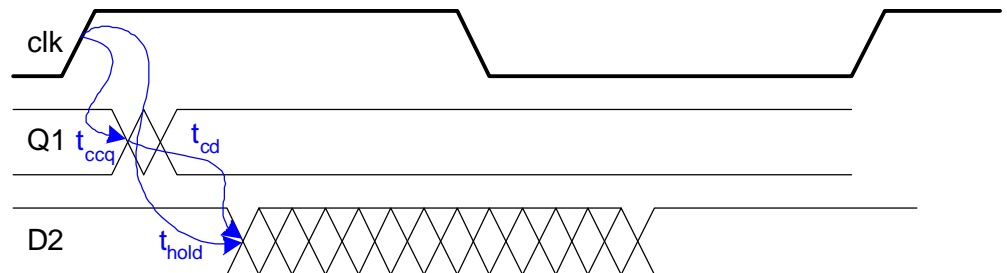
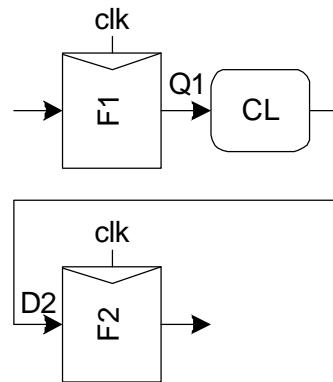
# Max Delay: Pulsed Latches

$$t_{pd} \leq T_c - \underbrace{\max(t_{pdq}, t_{pcq} + t_{setup} - t_{pw})}_{\text{sequencing overhead}}$$



# Min-Delay: Flip-Flops

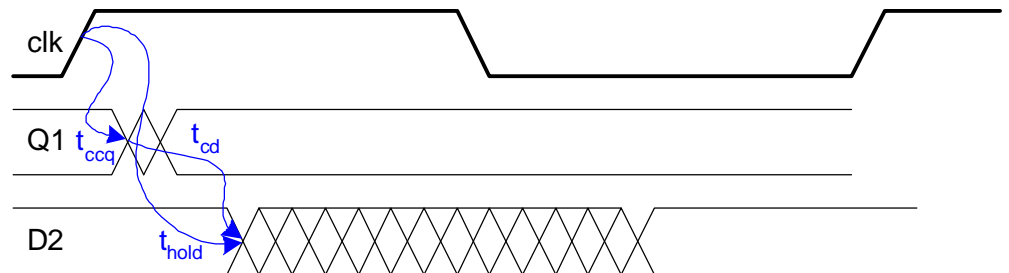
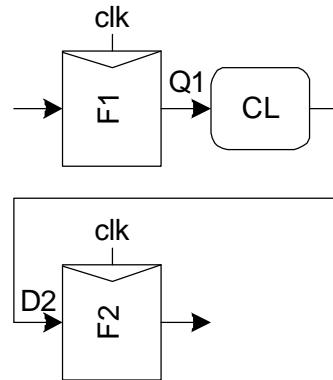
$$t_{cd} \geq$$





# Min-Delay: Flip-Flops

$$t_{cd} \geq t_{\text{hold}} - t_{ccq}$$



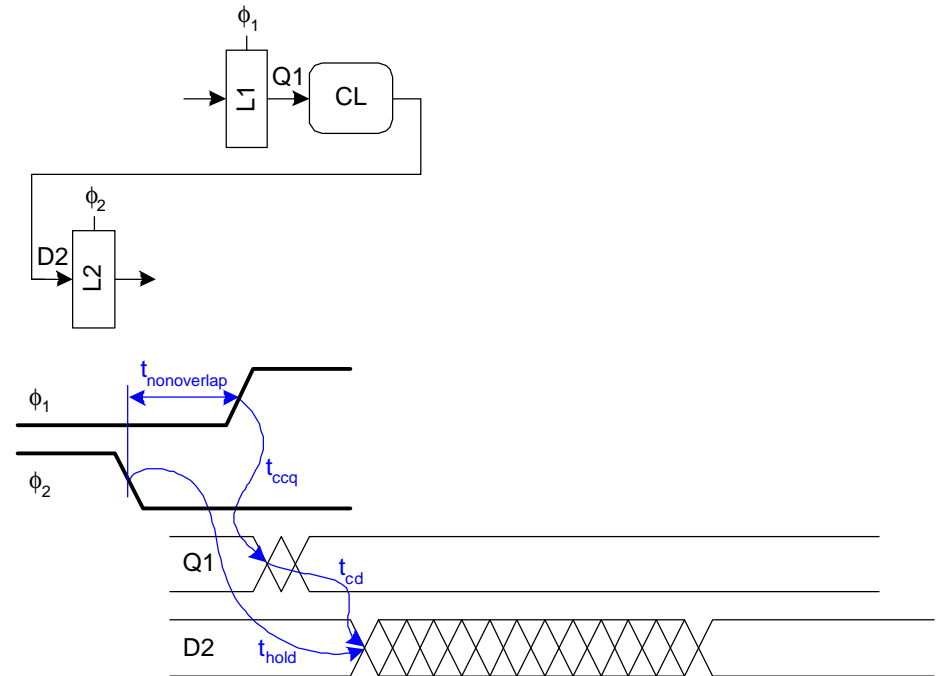
# Min-Delay: 2-Phase Latches

$$t_{cd1}, t_{cd2} \geq$$

Hold time reduced by nonoverlap

Paradox: hold applies twice each cycle, vs. only once for flops.

But a flop is made of two latches!



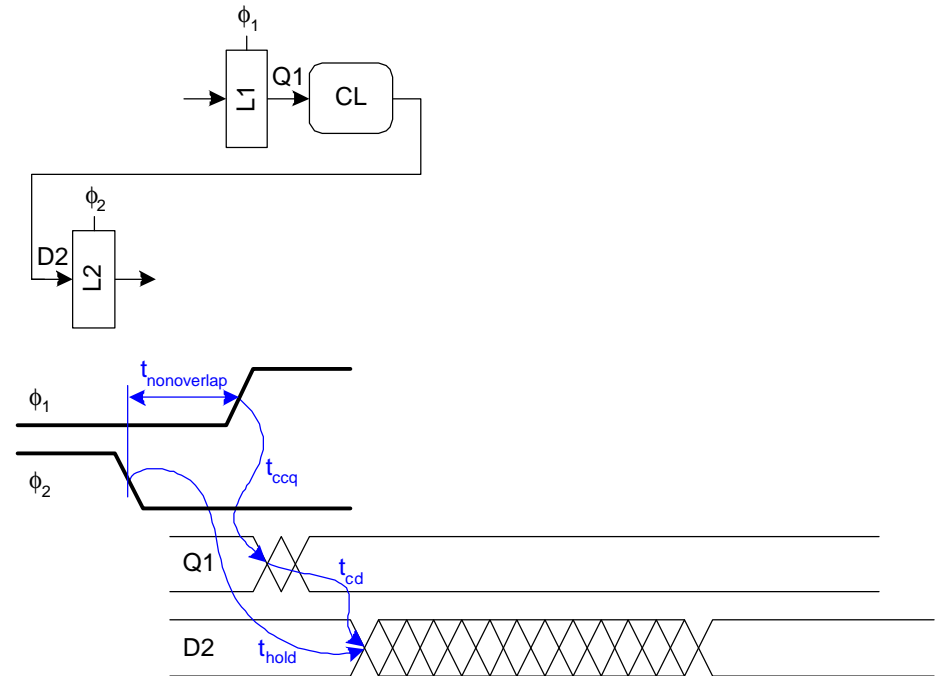
# Min-Delay: 2-Phase Latches

$$t_{cd1}, t_{cd2} \geq t_{\text{hold}} - t_{ccq} - t_{\text{nonoverlap}}$$

Hold time reduced by nonoverlap

Paradox: hold applies twice each cycle, vs. only once for flops.

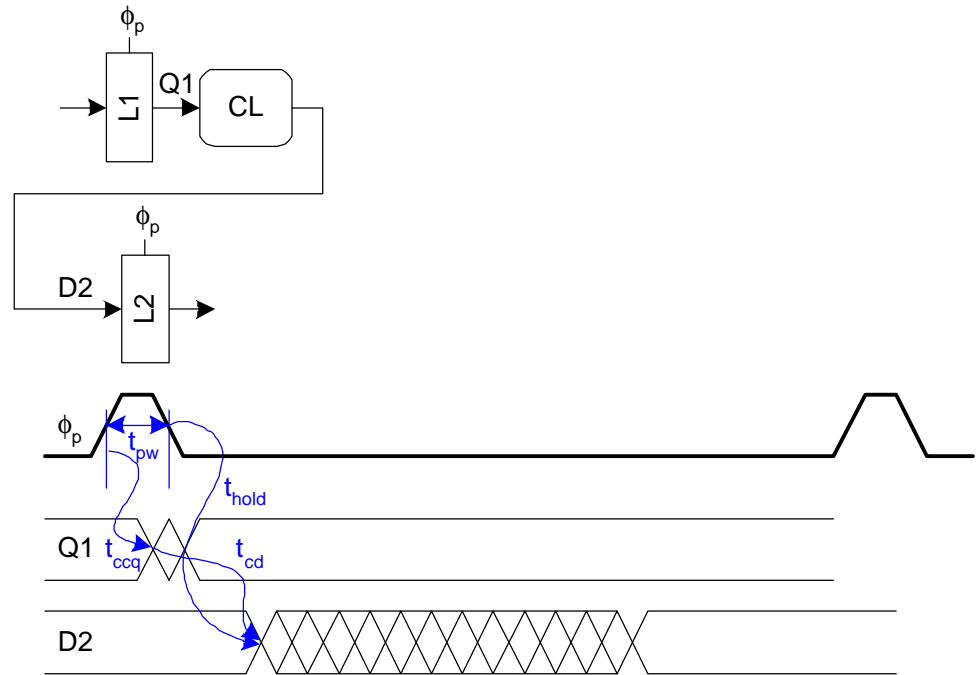
But a flop is made of two latches!



# Min-Delay: Pulsed Latches

$$t_{cd} \geq$$

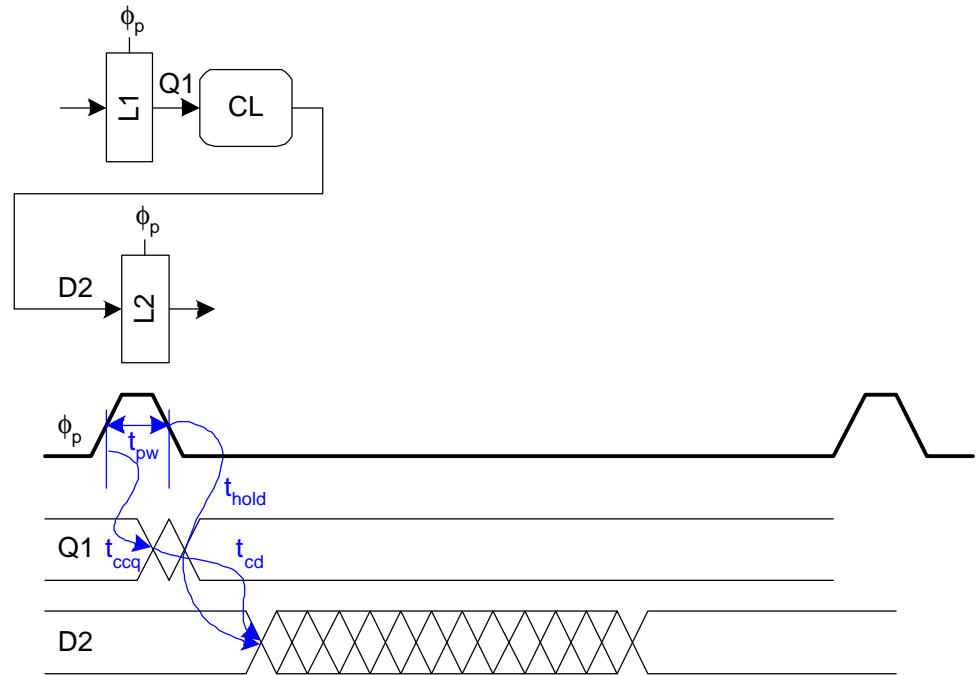
Hold time increased  
by pulse width



# Min-Delay: Pulsed Latches

$$t_{cd} \geq t_{hold} - t_{ccq} + t_{pw}$$

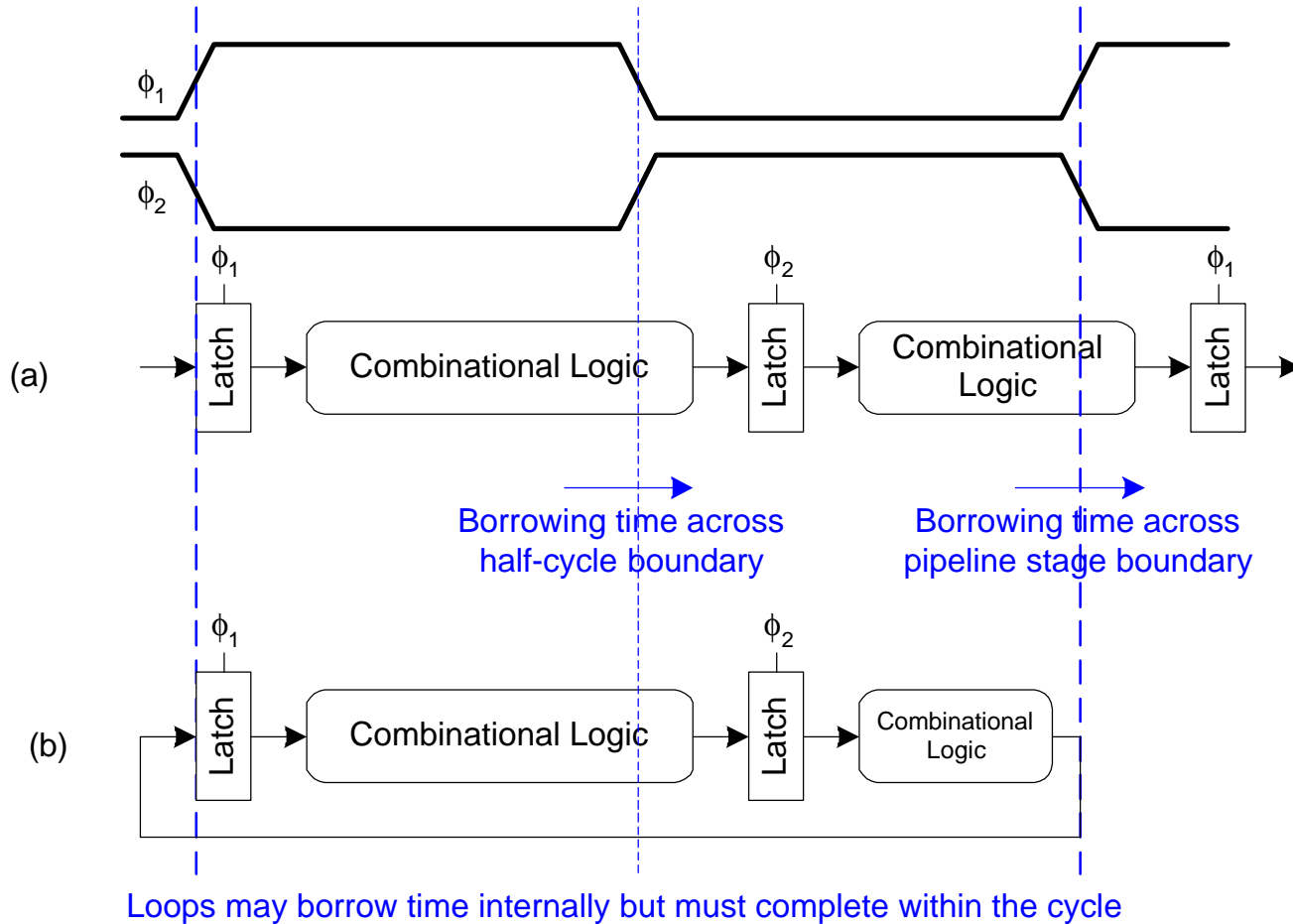
Hold time increased  
by pulse width



# Time Borrowing

- ❑ In a flop-based system:
  - Data launches on one rising edge
  - Must setup before next rising edge
  - If it arrives late, system fails
  - If it arrives early, time is wasted
  - Flops have hard edges
- ❑ In a latch-based system
  - Data can pass through latch while transparent
  - Long cycle of logic can borrow time into next
  - As long as each loop completes in one cycle

# Time Borrowing Example



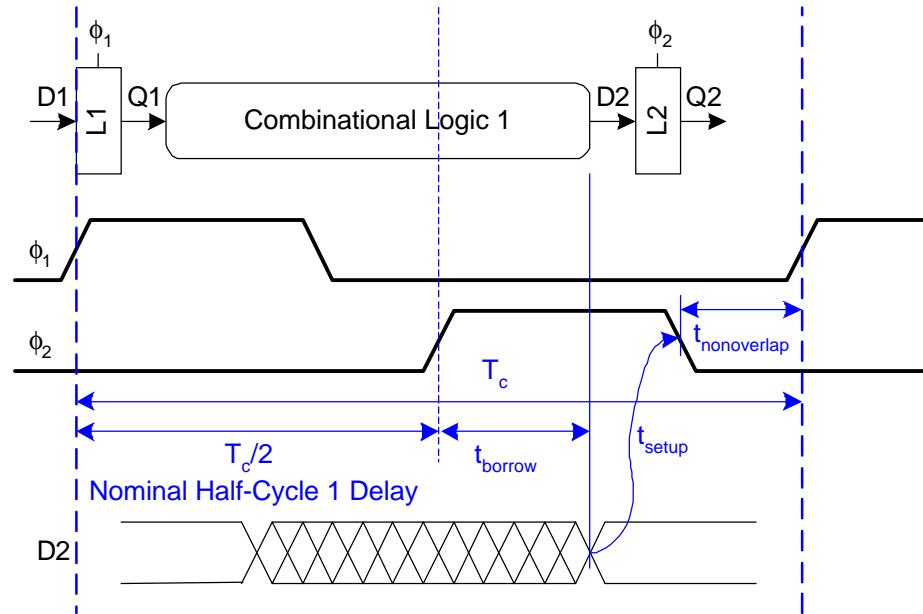
# How Much Borrowing?

## 2-Phase Latches

$$t_{\text{borrow}} \leq \frac{T_c}{2} - (t_{\text{setup}} + t_{\text{nonoverlap}})$$

## Pulsed Latches

$$t_{\text{borrow}} \leq t_{pw} - t_{\text{setup}}$$





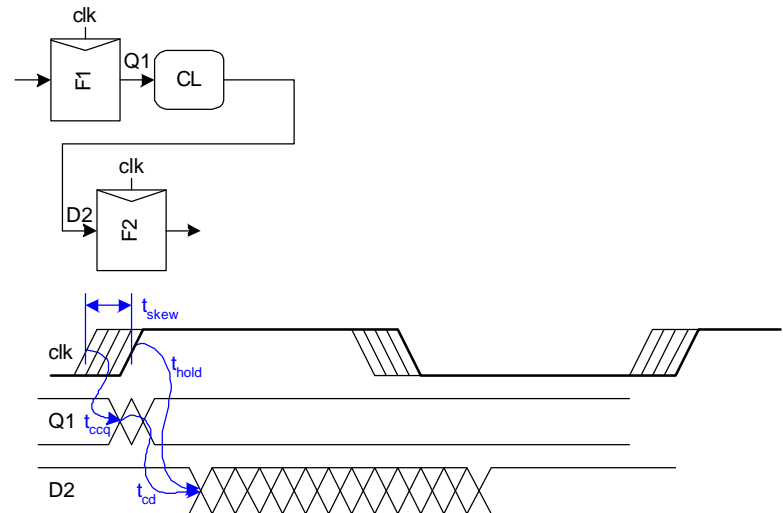
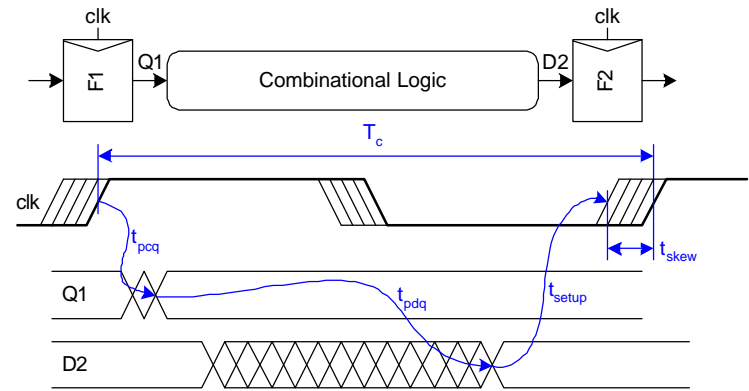
# Clock Skew

- ❑ We have assumed zero clock skew
- ❑ Clocks really have uncertainty in arrival time
  - Decreases maximum propagation delay
  - Increases minimum contamination delay
  - Decreases time borrowing

# Skew: Flip-Flops

$$t_{pd} \leq T_c - \underbrace{(t_{pcq} + t_{setup} + t_{skew})}_{\text{sequencing overhead}}$$

$$t_{cd} \geq t_{hold} - t_{ccq} + t_{skew}$$



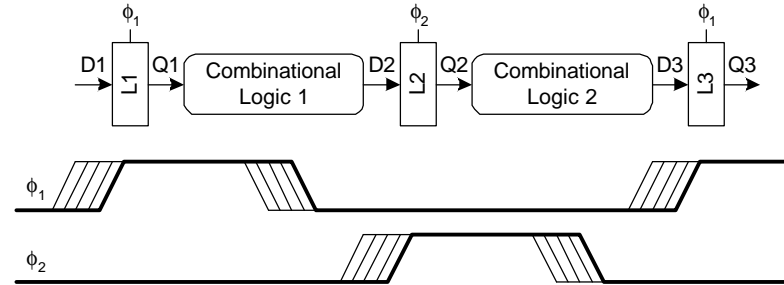
# Skew: Latches

## 2-Phase Latches

$$t_{pd} \leq T_c - \underbrace{(2t_{pdq})}_{\text{sequencing overhead}}$$

$$t_{cd1}, t_{cd2} \geq t_{\text{hold}} - t_{ccq} - t_{\text{nonoverlap}} + t_{\text{skew}}$$

$$t_{\text{borrow}} \leq \frac{T_c}{2} - (t_{\text{setup}} + t_{\text{nonoverlap}} + t_{\text{skew}})$$



## Pulsed Latches

$$t_{pd} \leq T_c - \underbrace{\max(t_{pdq}, t_{pcq} + t_{\text{setup}} - t_{pw} + t_{\text{skew}})}_{\text{sequencing overhead}}$$

$$t_{cd} \geq t_{\text{hold}} + t_{pw} - t_{ccq} + t_{\text{skew}}$$

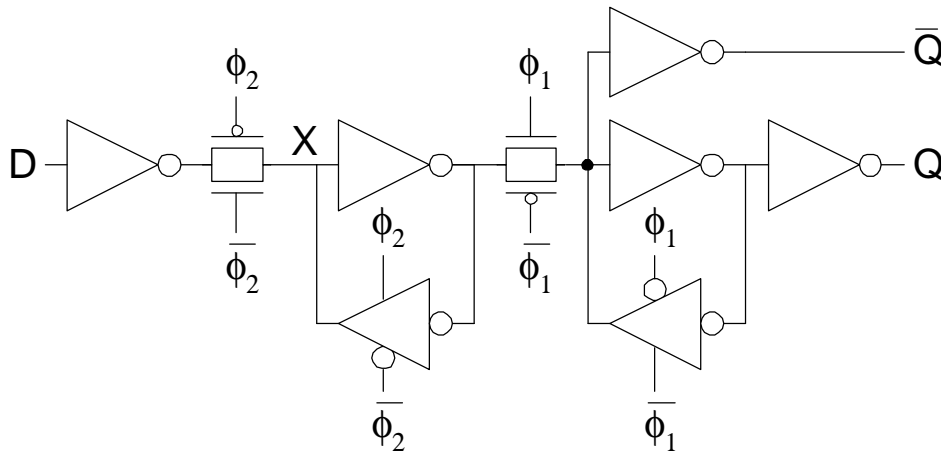
$$t_{\text{borrow}} \leq t_{pw} - (t_{\text{setup}} + t_{\text{skew}})$$

# Two-Phase Clocking

- ❑ If setup times are violated, reduce clock speed
- ❑ If hold times are violated, chip fails at any speed
- ❑ In this class, working chips are most important
  - No tools to analyze clock skew
- ❑ An easy way to guarantee hold times is to use 2-phase latches with big nonoverlap times
- ❑ Call these clocks  $\phi_1$ ,  $\phi_2$  (ph1, ph2)

# Safe Flip-Flop

- ❑ In class, use flip-flop with nonoverlapping clocks
  - Very slow – nonoverlap adds to setup time
  - But no hold times
- ❑ In industry, use a better timing analyzer
  - Add buffers to slow signals if hold time is at risk



# Summary

- ❑ Flip-Flops:
  - Very easy to use, supported by all tools
- ❑ 2-Phase Transparent Latches:
  - Lots of skew tolerance and time borrowing
- ❑ Pulsed Latches:
  - Fast, some skew tol & borrow, hold time risk

	Sequencing overhead ( $T_c - t_{pd}$ )	Minimum logic delay $t_{cd}$	Time borrowing $t_{borrow}$
Flip-Flops	$t_{pcq} + t_{setup} + t_{skew}$	$t_{hold} - t_{ccq} + t_{skew}$	0
Two-Phase Transparent Latches	$2t_{pdq}$	$t_{hold} - t_{ccq} - t_{nonoverlap} + t_{skew}$ in each half-cycle	$\frac{T_c}{2} - (t_{setup} + t_{nonoverlap} + t_{skew})$
Pulsed Latches	$\max(t_{pdq}, t_{pcq} + t_{setup} - t_{p\overline{w}} + t_{skew})$	$t_{hold} - t_{ccq} + t_{p\overline{w}} + t_{skew}$	$t_{p\overline{w}} - (t_{setup} + t_{skew})$