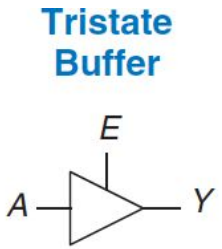
Digital Logic got BORING !!!

Un-Boring Digital Logic !!!

Tri-State Buffer

Tri-State Buffer

- A tri-state buffer enables gating of different signals onto a wire



E	A	Y
0	0	Z
0	1	Z
1	0	0
1	1	1

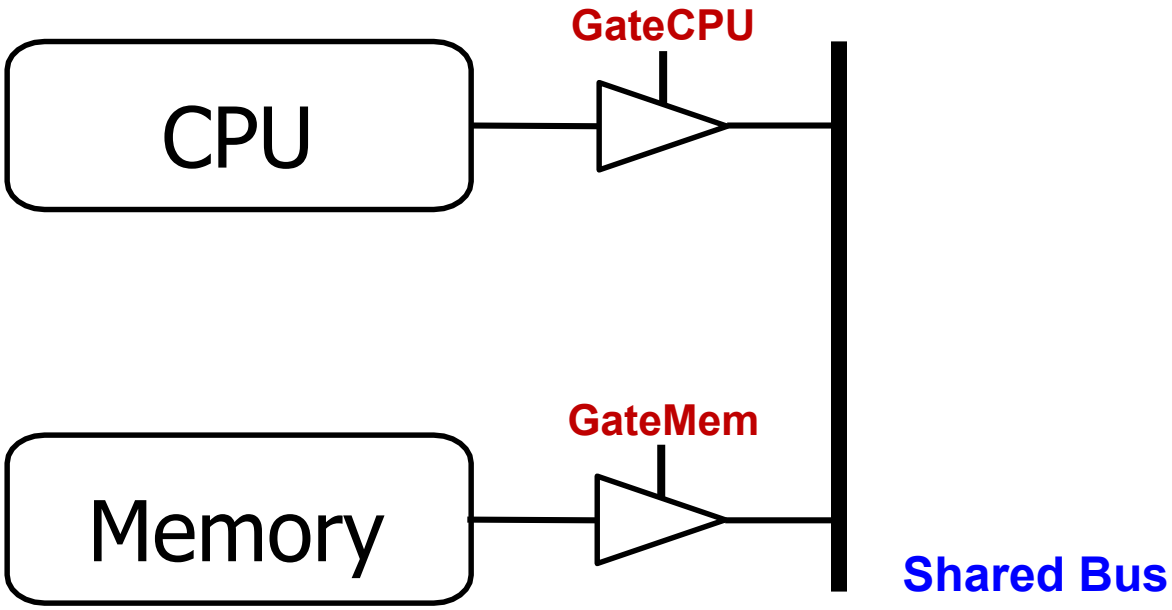
Figure 2.40 Tristate buffer

- Floating signal (Z): Signal that is not driven by any circuit
  - Open circuit, floating wire

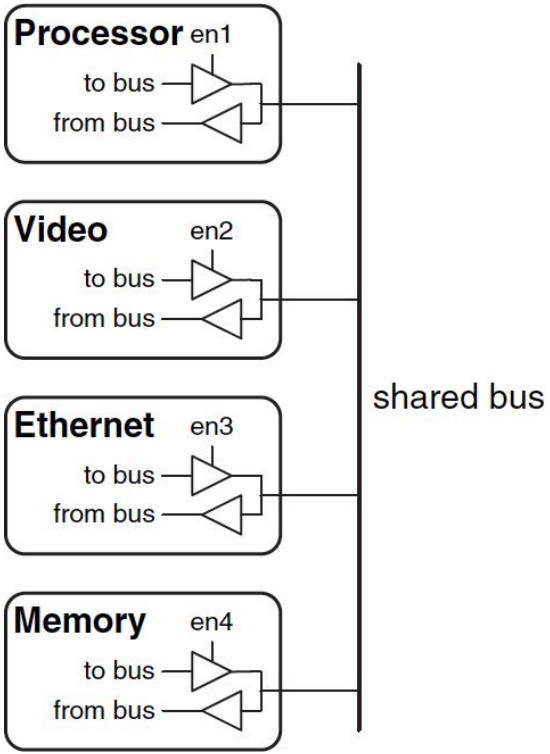
Example: Use of Tri-State Buffers

- Imagine a wire connecting the CPU and memory
  - At any time only the CPU or the memory can place a value on the wire, both not both
  - You can have two tri-state buffers: one driven by CPU, the other memory; and ensure at most one is enabled at any time

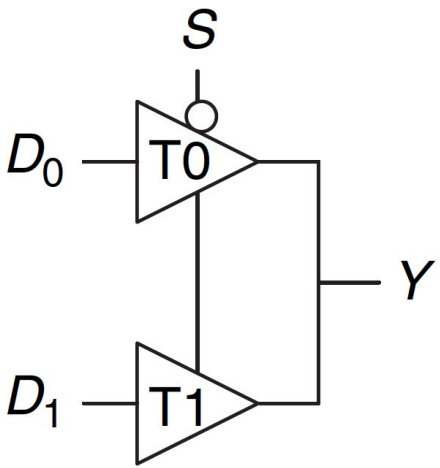
Example Design with Tri-State Buffers



Another Example

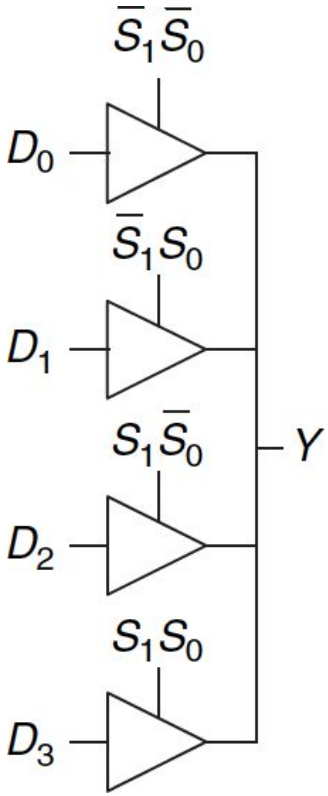


Multiplexer Using Tri-State Buffers

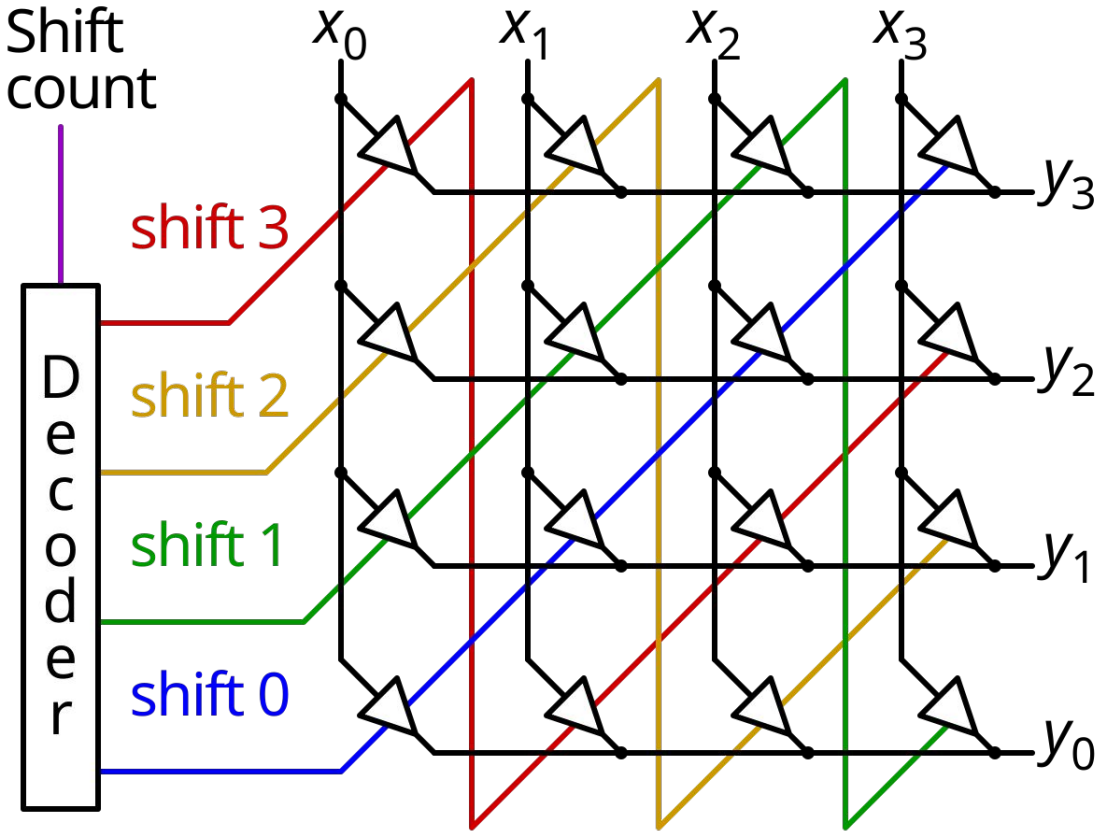


$$Y = D_0\overline{S} + D_1 S$$

Figure 2.56 **Multiplexer using tristate buffers**



Barrel Shifter Using Tri-State Buffers



4-valued logic

buf	
input	output
0	0
1	1
x	x
z	x

not	
input	output
0	1
1	0
x	x
z	x

4-valued logic

and	0	1	x	z
0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

or	0	1	x	z
0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

xor	0	1	x	z
0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

nand	0	1	x	z
0	1	1	1	1
1	1	0	x	x
x	1	x	x	x
z	1	x	x	x

nor	0	1	x	z
0	1	0	x	x
1	0	0	0	0
x	x	0	x	x
z	x	0	x	x

xnor	0	1	x	z
0	1	0	x	x
1	0	1	x	x
x	x	x	x	x
z	x	x	x	x

4-valued logic

bufif0		control input			
		0	1	x	z
data input	0	0	z	L	L
	1	1	z	H	H
	x	x	z	x	x
	z	x	z	x	x

bufif1		control input			
		0	1	x	z
data input	0	z	0	L	L
	1	z	1	H	H
	x	z	x	x	x
	z	z	x	x	x

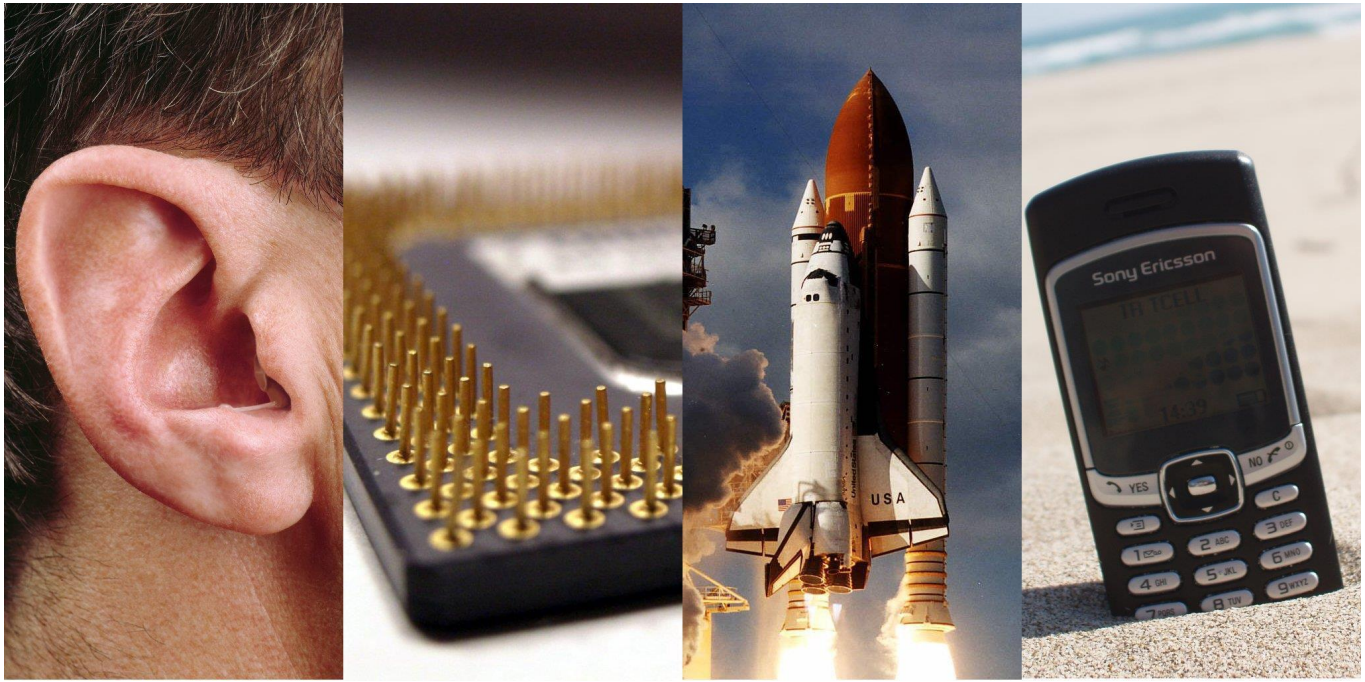
- The L and H symbols have a special meaning.
  - The **L symbol** means that the output has 0 or z value.
  - The **H symbol** means that the output has 1 or z value.
  - Any transition to H or L is treated as a transition to x.

Tradeoffs in Circuit Design

Circuit Design is a Tradeoff Between:

- Area
  - Circuit **area** is proportional to the **cost** of the device
- Speed / Throughput
  - We want **faster**, more **capable** circuits
- Power / Energy
  - Mobile devices need to work with a **limited** power supply
  - High performance devices **dissipate** more than 100W/cm^2
- Design Time
  - Designers are **expensive** in time and money
  - The **competition** will not wait for you

Requirements and Goals Depend On Application



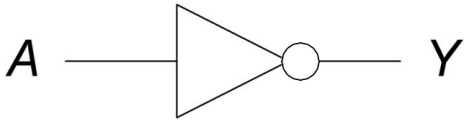
Circuit Timing

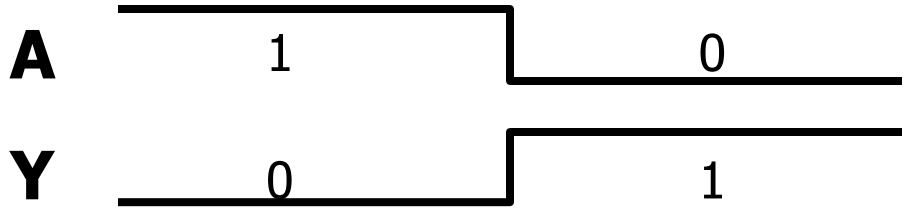
- Until now, we investigated **logical functionality**
- What about **timing**?
  - How **fast** is a circuit?
  - How can we make a circuit **faster**?
  - What happens if we run a circuit **too fast**?
- A design that is logically correct can still **fail** because of real-world **implementation issues**!

Combinational Circuit Timing

Digital Logic Abstraction

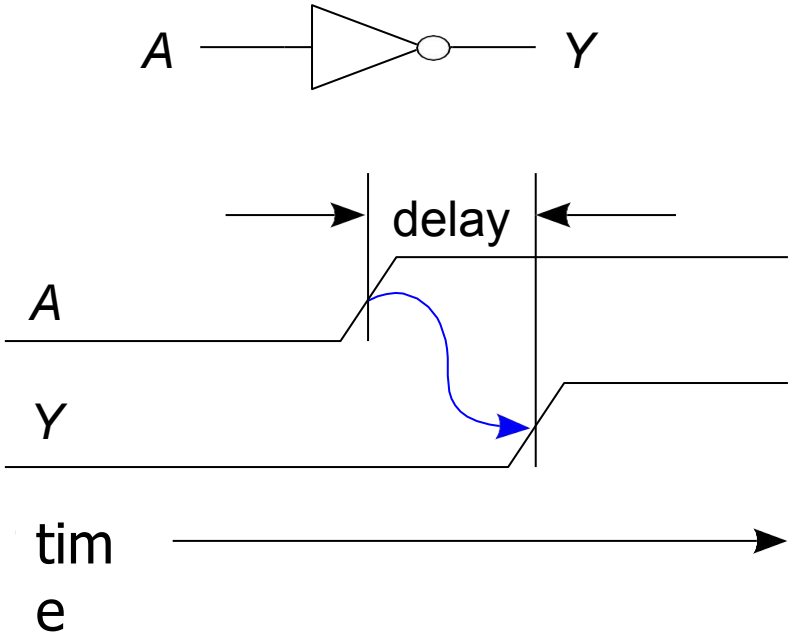
- **"Digital logic"** is a convenient **abstraction**
  - Output changes **immediately** with the input





Combinational Circuit Delay

- In reality, **outputs** are **delayed** from **inputs**
  - Transistors take a finite amount of time to switch



Real Inverter Delay Example

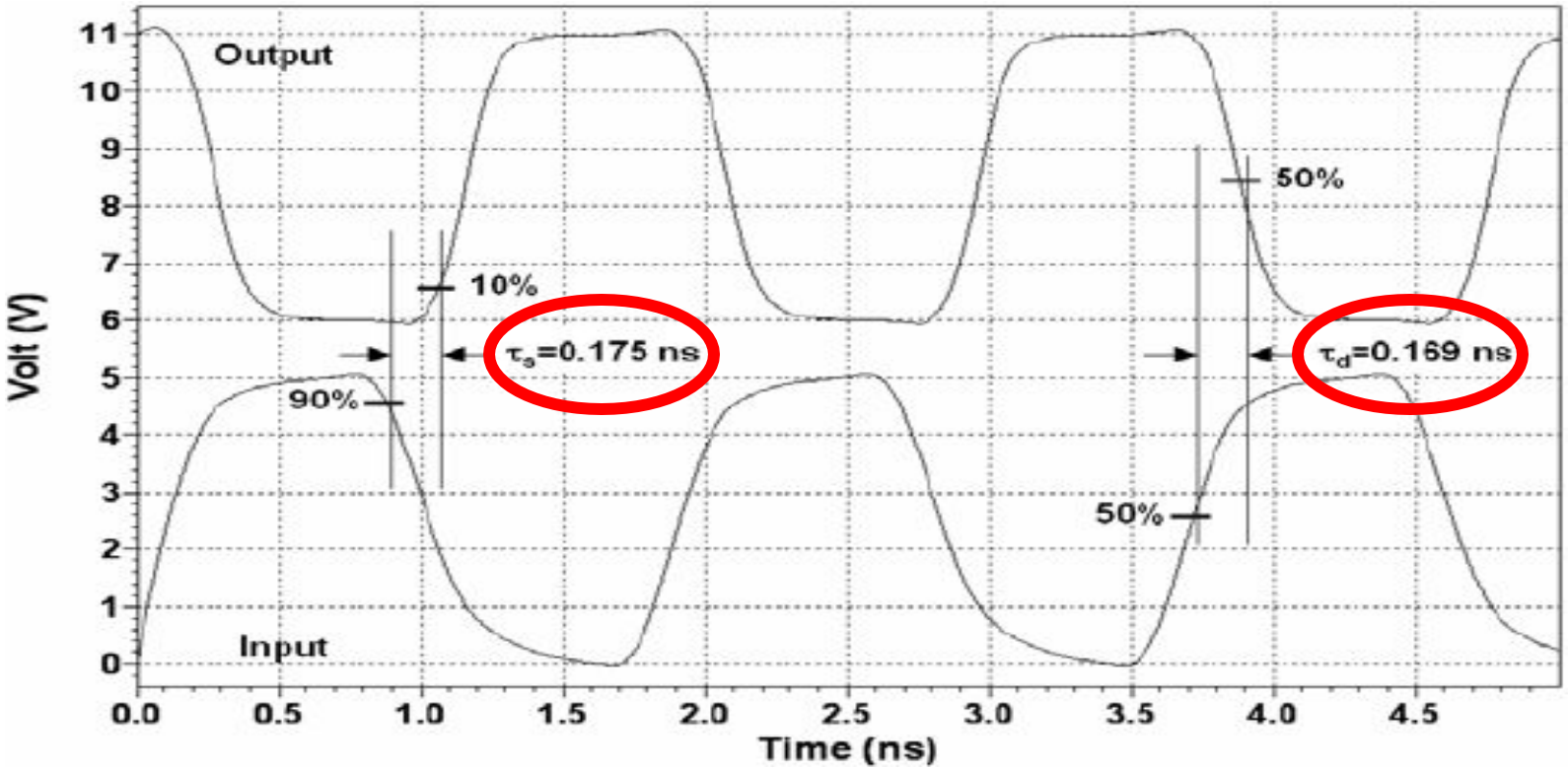


Image source: Sandoval-Ibarra, F., and E. S. Hernández-Bernal. "Ring CMOS NOT- based oscillators: Analysis and design." *Journal of applied research and technology,* 2008.

Circuit Delay and Its Variation

- Delay is fundamentally caused by
  - **Capacitance** and **resistance** in a circuit
  - Finite **speed of light** (not so fast on a nanosecond scale!)

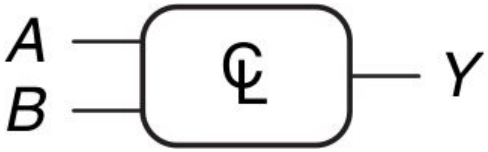
- **Anything** affecting these quantities can change delay:
  - **Rising** (i.e., 0 -> 1) vs. **falling** (i.e., 1 -> 0) inputs
  - Different **inputs** have different **delays**
  - Changes in **environment** (e.g., temperature)
  - **Aging** of the circuit

- We have a **range of possible delays** from input to output

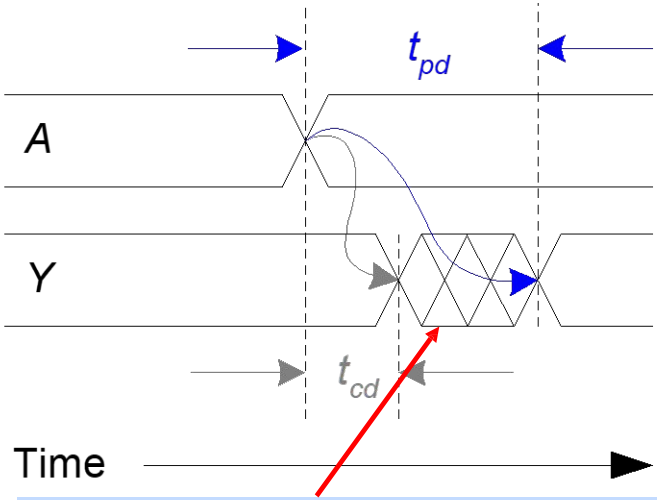
Delays from Input to Output

- **Contamination delay (t_{cd}):** delay until Y **starts changing**
- **Propagation delay (t_{pd}):** delay until Y **finishes changing**

Example Circuit



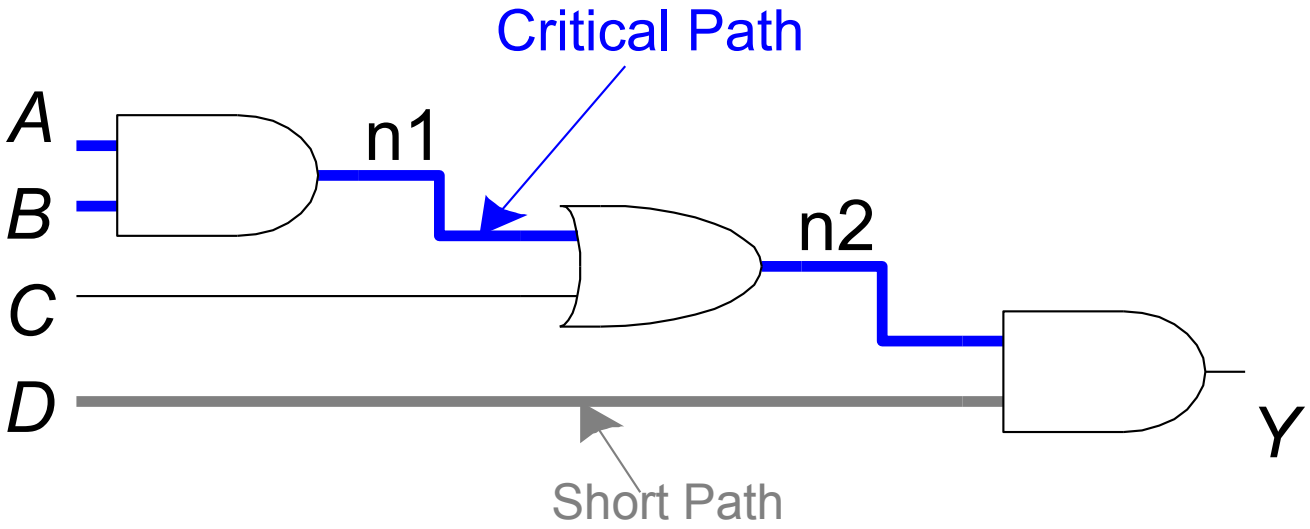
Effect of Changing Input 'A'



Cross-hatching means value is

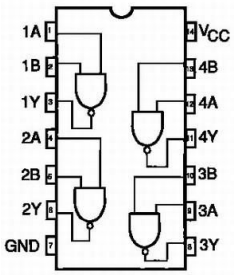
Calculating Long/Short Paths

- We care about **both** the **longest** and **shortest** paths in a circuit



- **Critical (Longest) Path:** $t_{pd} = 2t_{pd\_AND} + t_{pd\_OR}$
- **Shortest Path:** $t_{cd} = t_{cd\_AND}$

Example t_{pd} for a Real NAND-2 Gate



Symbol	Parameter	Conditions	25 °C			–40 °C to +125 °C		Unit
			Min	Typ	Max	Max (85 °C)	Max (125 °C)	
74HC00								
t_{pd}	propagation delay	nA, nB to nY; see Figure 6 [1]						
		V_{CC} = 2.0 V	-	25	-	115	135	ns
		V_{CC} = 4.5 V	-	9	-	23	27	ns
		V_{CC} = 5.0 V; C_L = 15 pF	-	7	-	-	-	ns
		V_{CC} = 6.0 V	-	7	-	20	23	ns

- Heavy **dependence** on **voltage** and **temperature**!

Source: Nexperia 2-input NAND (74HC00) Datasheet, Section 10

Example Worst-Case t_{pd}

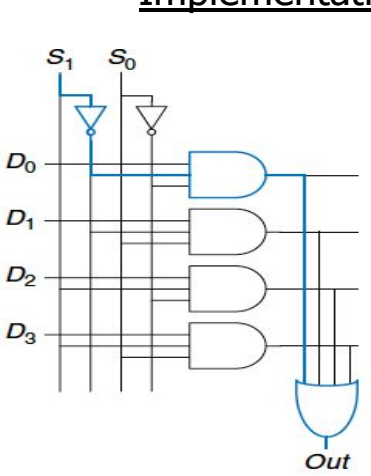
- Two different **implementations** of a **4:1 multiplexer**

Gate Delays

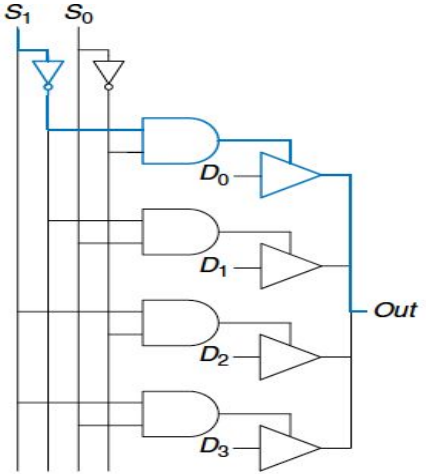
Gate	t_{pd} (ps)
NOT	30
2-input AND	60
3-input AND	80
4-input OR	90
tristate (A to Y)	50
tristate (enable to Y)	35

- **Different designs** lead to very **different delays**

Implementation 1



Implementation 2



(a)

$t_{pd\_sy} = t_{pd\_INV} + t_{pd\_AND3} + t_{pd\_OR4}$
$= 30\text{ ps} + 80\text{ ps} + 90\text{ ps}$
$= 200\text{ ps}$

$t_{pd\_dy} = t_{pd\_AND3} + t_{pd\_OR4}$
$= 170\text{ ps}$

(b)

$t_{pd\_sy} = t_{pd\_INV} + t_{pd\_AND2} + t_{pd\_TRI\_SY}$
$= 30\text{ ps} + 60\text{ ps} + 35\text{ ps}$
$= 125\text{ ps}$

$t_{pd\_dy} = t_{pd\_TRI\_AY}$
$= 50\text{ ps}$

Disclaimer: Calculating Long/Short Paths

- It's **not** always this easy to determine the long/short paths!
  - Not all **input transitions** affect the **output**
  - Can have **multiple different paths** from an input to output

- In reality, circuits are **not** all built equally
  - Different instances of the **same gate** have **different delays**
  - **Wires** have **nonzero delay** (increasing with length)
  - Temperature/voltage affect circuit speeds
    - Not all circuit elements are affected the same way
    - Can even **change the critical path**!

- Designers assume **"worst-case" conditions** and run many **statistical simulations** to balance yield/performance

Combinational Timing Summary

- Circuit outputs change some time **after** the inputs change
  - Caused by finite speed of light (not so fast on a ns scale!)
  - Delay is dependent on inputs, environmental state, etc.

- The range of possible delays is characterized by:
  - **Contamination delay (t_{cd}):** minimum possible delay
  - **Propagation delay (t_{pd}):** maximum possible delay

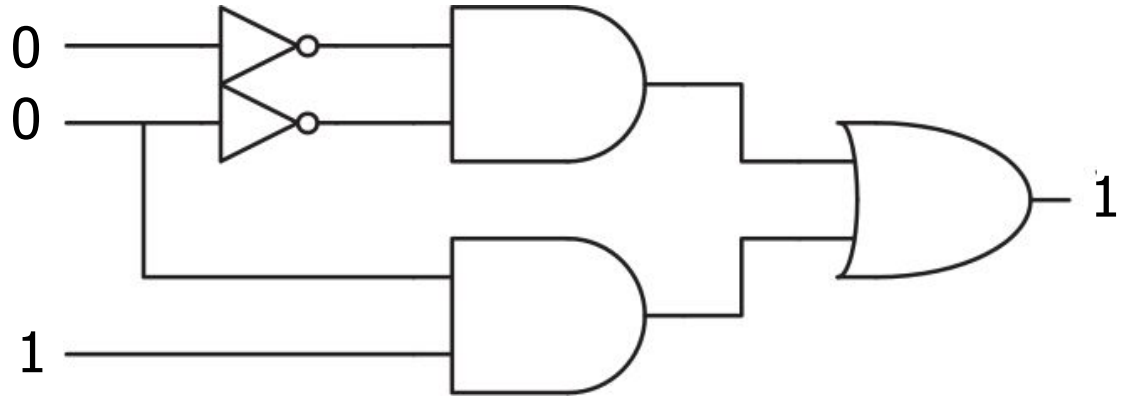
- Delays **change** with:
  - Circuit design (e.g., topology, materials)
  - Operating conditions

Output Glitches

Glitches

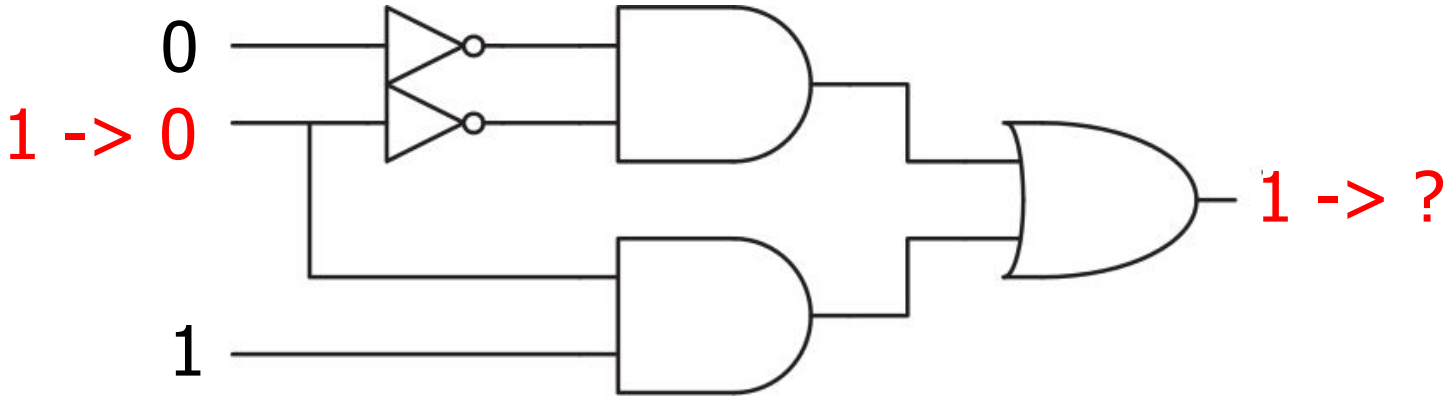
- **Glitch:** **one** input transition causes **multiple** output transitions

Circuit initial state



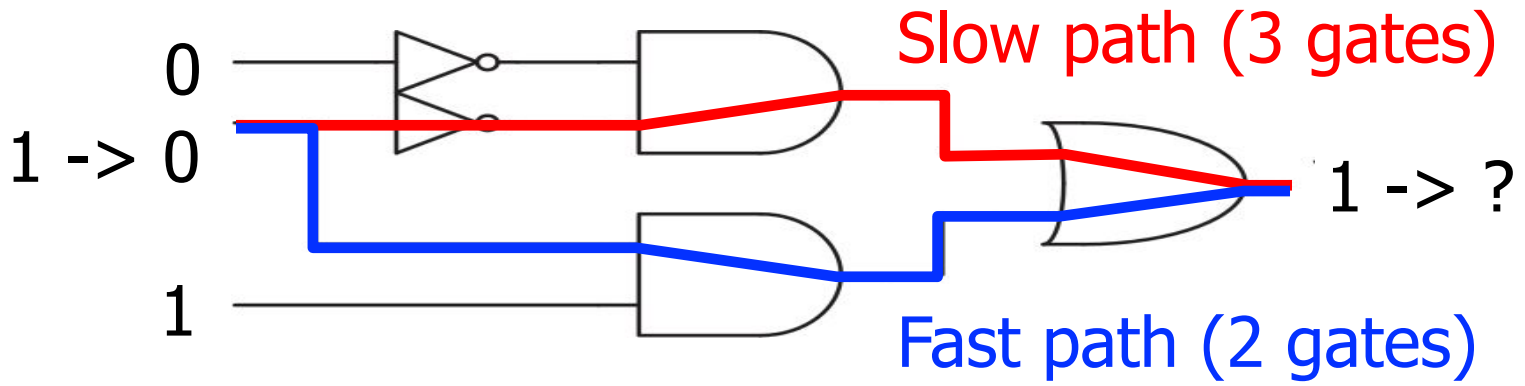
Glitches

- **Glitch:** **one** input transition causes **multiple** output transitions



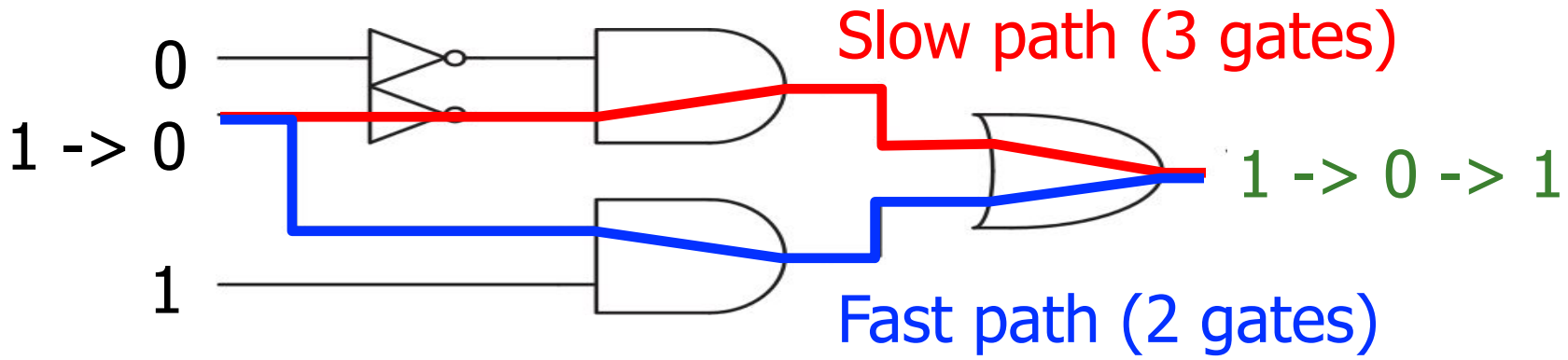
Glitches

- **Glitch: one** input transition causes **multiple** output transitions



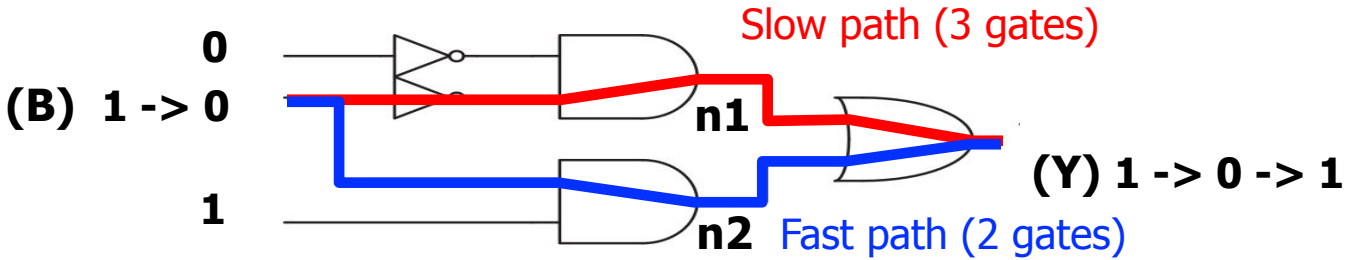
Glitches

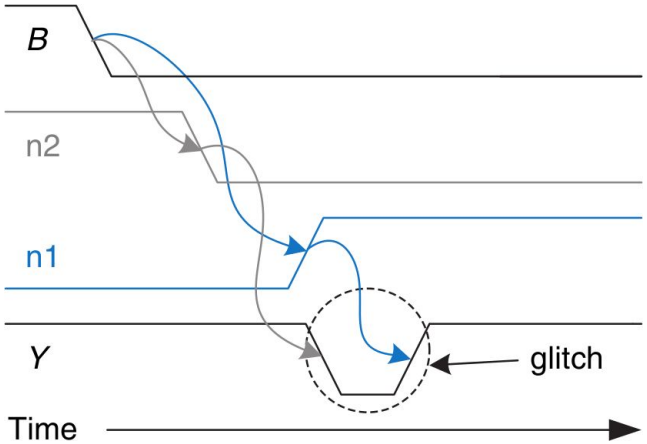
- **Glitch:** **one** input transition causes **multiple** output transitions



Glitches

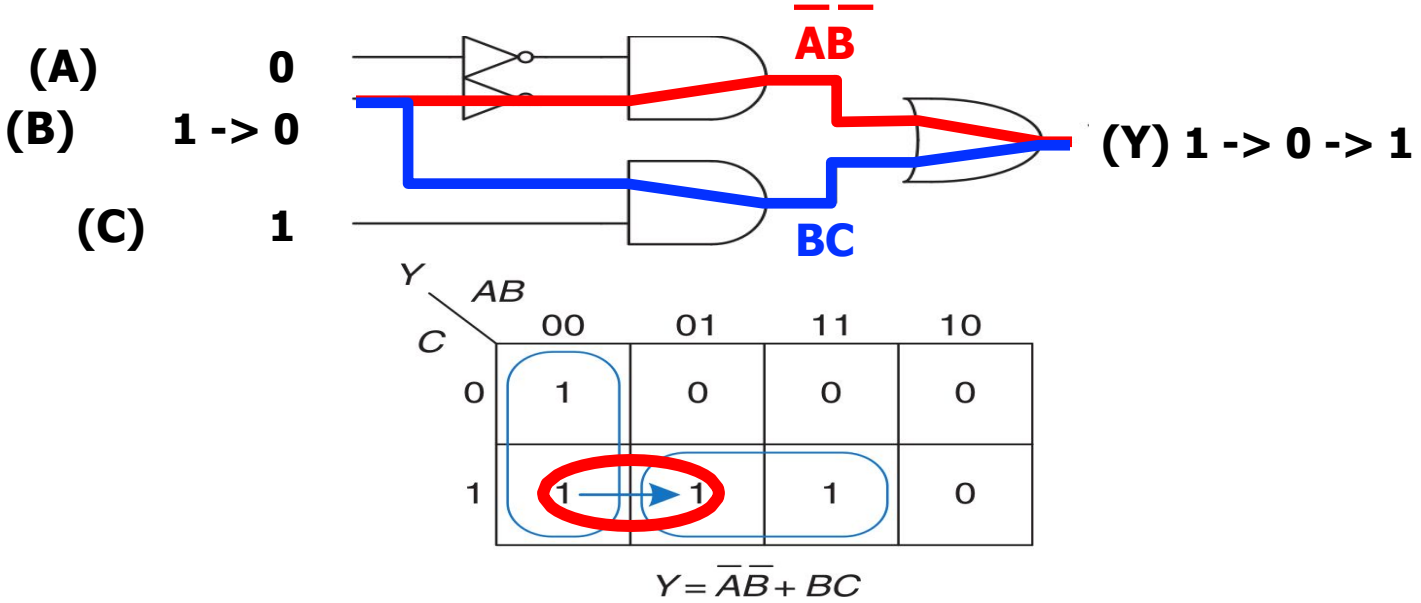
- **Glitch: one** input transition causes **multiple** output transitions





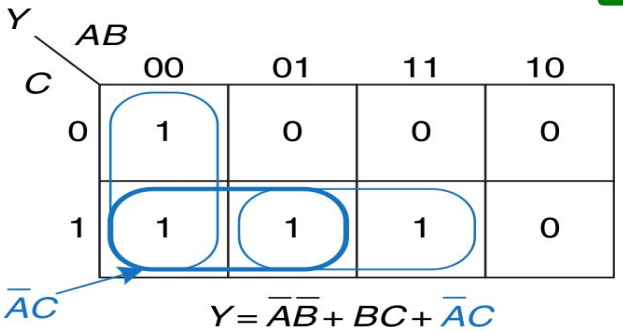
Avoiding Glitches Using K-Maps

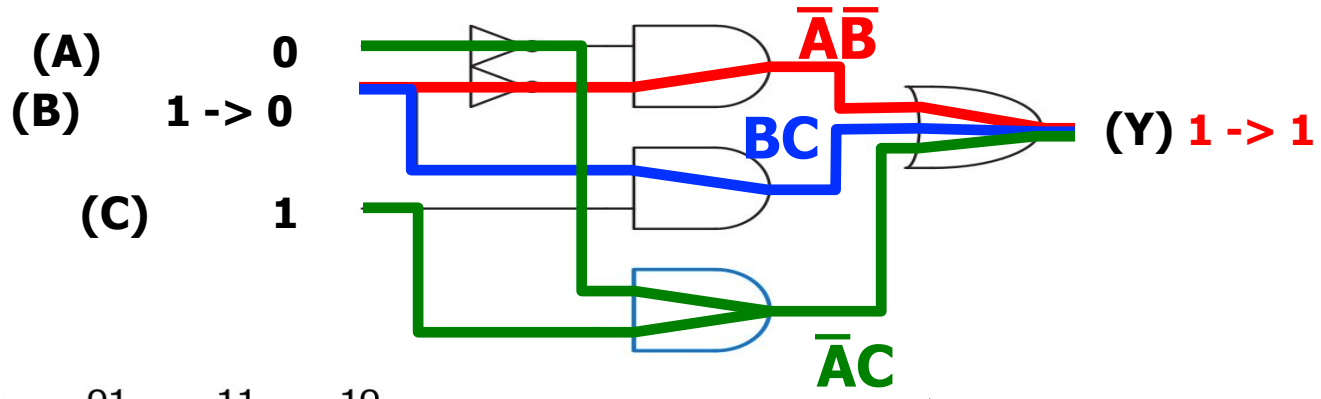
- Glitches are **visible** in **K-maps**
  - Recall: K-maps show the results of a change in a **single input**
  - A glitch occurs when **moving between prime implicants**



Avoiding Glitches Using K-Maps

- We can **fix** the issue by adding in the **consensus** term
  - Ensures **no transition** between different **prime implicants**





No dependence on B
=> No glitch!

Avoiding Glitches

- **Q:** Do we **always** care about glitches?
  - **Fixing** glitches is **undesirable**
    - More chip **area**
    - More **power consumption**
    - More **design effort**
  - The circuit is **eventually** guaranteed to **converge** to the **right value** regardless of glitchiness

- **A:** No, not always!
  - If we only care about the **long-term steady state output**, we can **safely ignore** glitches
  - Up to the **designer to decide** if glitches matter in their application