EE141-Spring 2012 Digital Integrated Circuits

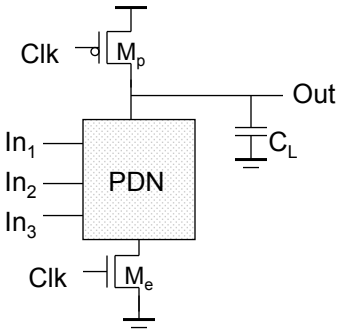
Lecture 19
Layout

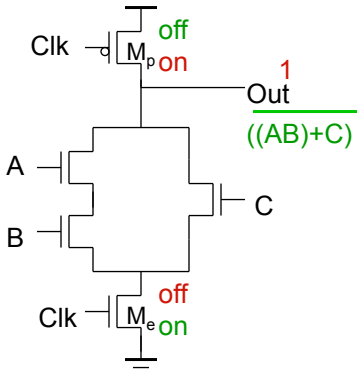
Administrativia

- No regrades on MT2 after next We!
- Phase 2 now in full motion

Dynamic Logic

Dynamic Gate





Two phase operation

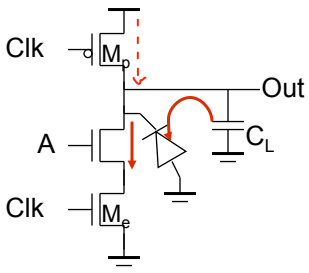
Precharge (Clk = 0)

Evaluate (Clk = 1)

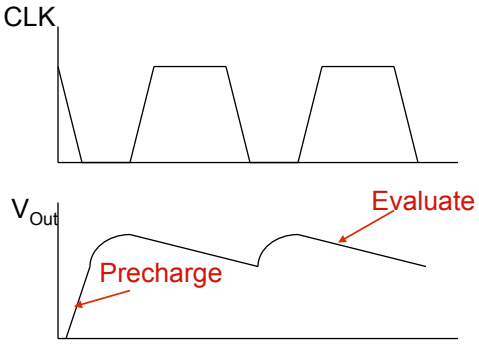
Challenges of Dynamic Gates

- **Noise sensitivity and small noise margins**
  - Leakage
  - Charge sharing
  - Clock feedthrough

Issues in Dynamic Design 1: Charge Leakage



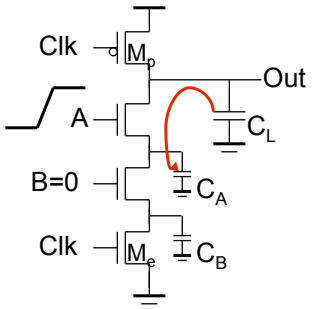
Leakage sources



Dominant component is subthreshold current

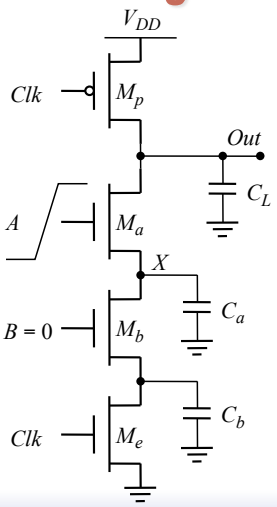
Issues in Dynamic Design 2: Charge Sharing

- Charge initially stored on C_L
  - C_A previously discharged



- When A rises, this charge is redistributed (shared) between C_L and C_A
- Makes Out drop below V_{DD}

Charge Sharing



- Two cases:
  - M_a stays on – complete charge share
  - M_a turns off – incomplete charge share

- Complete charge share:
  - $Q_{Ca} = V_{Out}C_a$
    $\Delta Q_{CL} = -V_{Out}C_a$

$V_{DD}C_L = V_{out}(C_L + C_a)$

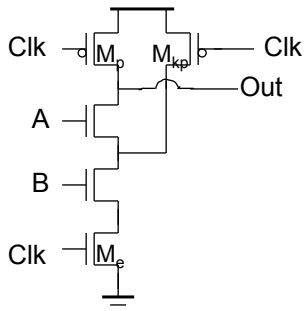
→ $\boxed{\Delta V_{Out} = -V_{DD}C_a/(C_a+C_L)}$

- Incomplete charge share:
  - $Q_{Ca} = (V_{DD}-V_{TN}^*)C_a$
    $\Delta Q_{CL} = -(V_{DD}-V_{TN}^*)C_a$

→ $\boxed{\Delta V_{Out} = -(V_{DD}-V_{TN}^*)C_a/C_L}$

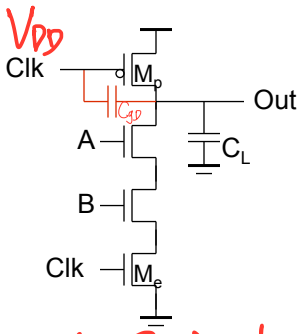
$V_{DD}C_L = V_{out}C_L + (V_{DD}-V_{TH})C_a$

Solution to Charge Sharing



- Keeper helps a lot
  - Can still get failures if Out drops below inverter V_{sw}
- Another option: precharge internal nodes
  - Increases power and area

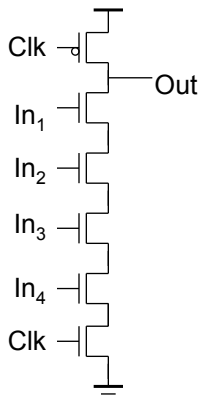
Issues in Dynamic Design 3: Clock Feedthrough

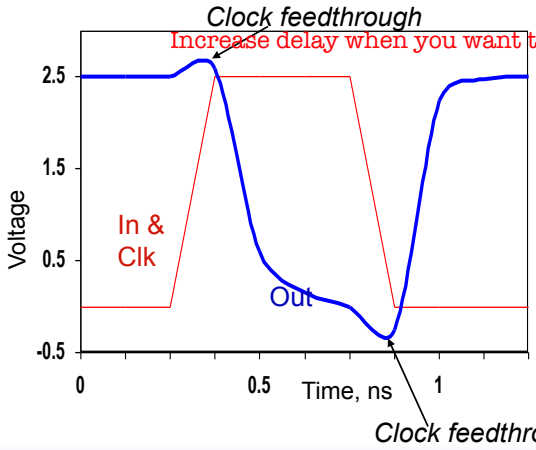


Coupling between Out and Clk input of the precharge device due to the gate to drain capacitance. So voltage of Out can rise above V_{DD}. The fast rising (and falling edges) of the clock couple to Out.

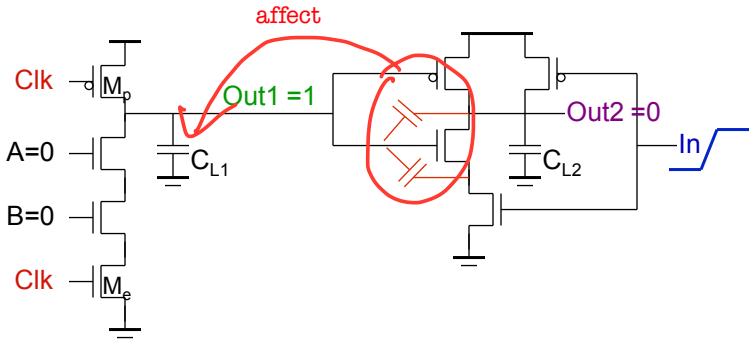
$$V_{DD}C_L + V_{DD}C_{gp} = V_{out}C_L + (V_{out} - V_{DD})C_{gp}$$

Clock Feedthrough





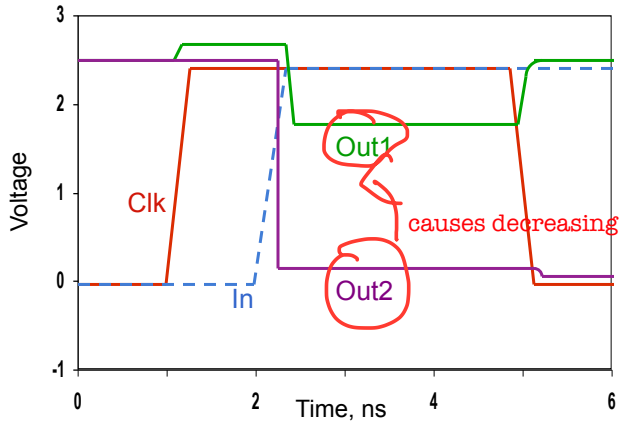
Issues in Dynamic Design 4: Backgate Coupling



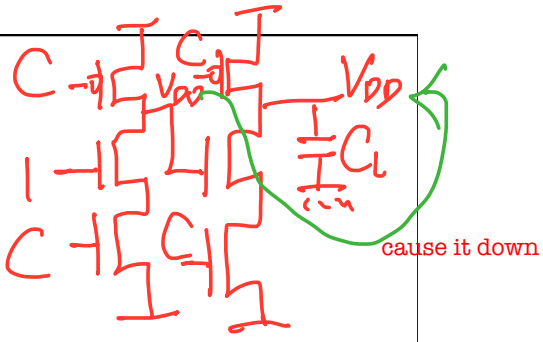
Dynamic NAND Static NAND

In digital circuit, when the circuit changes from one state to another state, the wire will product a big peak current, which will form a transient noise voltage and affect normal operation of the previous level.

Backgate Coupling Effect



Other Effects

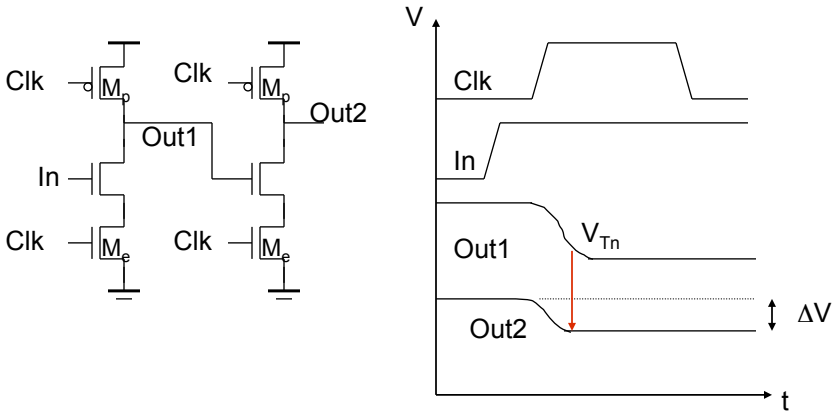


- Capacitive coupling
- Substrate coupling
- Minority charge injection
- Supply noise (ground bounce)

We can not couple two dynamic gates directly to each other because they all would start going simultaneously down when you turn on the clock.

Domino Logic

Cascading Dynamic Gates

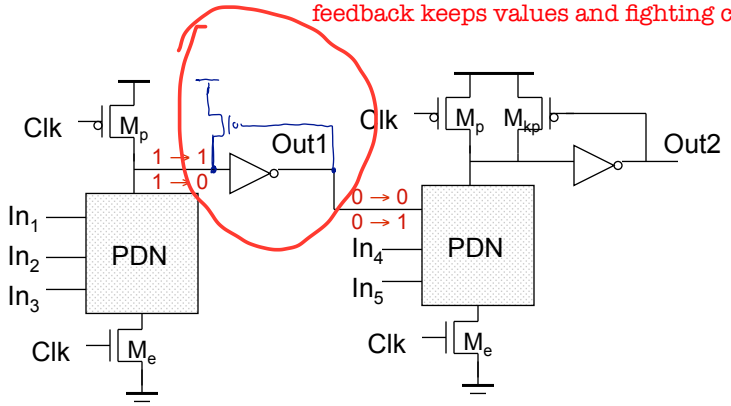


Rule: Only 0 → 1 transitions allowed at inputs!

The state the circuit can have only are 0 ore 0 to 1 at the input

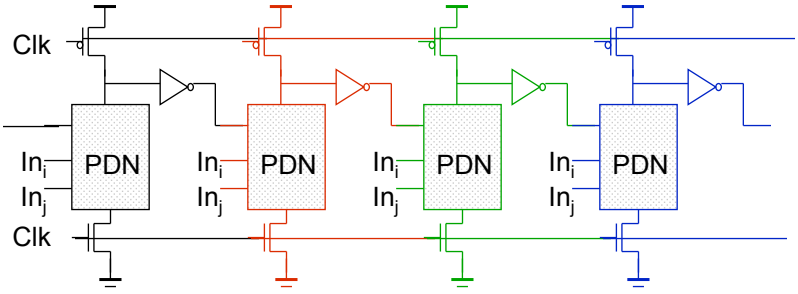
Domino Logic

feedback keeps values and fighting charge leakage.



If one gate down, other gates will down one after another.
So it calls Domino Logic.

Why Named Domino?



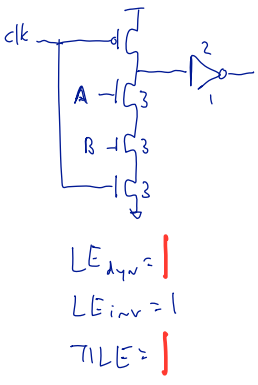
Like falling dominos!

Properties of Domino Logic

- Only non-inverting logic can be implemented
- Very high speed
  - static inverter can be skewed, only L-H transition critical
  - Input capacitance reduced – smaller logical effort

Because it has to have a inverter all the time.

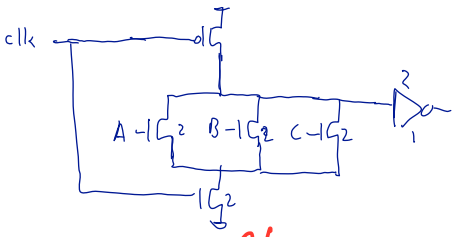
Domino Logic LE



$LE_{dyn} = 1$

$LE_{inv} = 1$

$\pi LE = 1$



$LE_{dyn} = 2/3$ $\quad LE_{static} = \frac{7}{3}$

$LE_{inv} = 1$

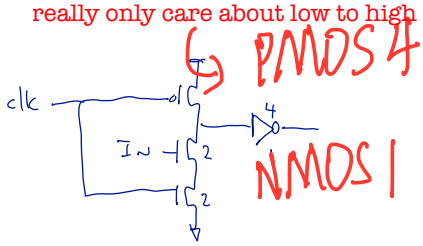
$\pi LE = 2/3$

LE is independent of the numbers of input.

PMOS wider, VM increases, inverter switches earlier, 0 to 1 faster, 1 to 0 lower

Domino Logic LE (skewed static gate)

really only care about low to high transition



6 PMOS 4

NMOS 1

reference inverter:

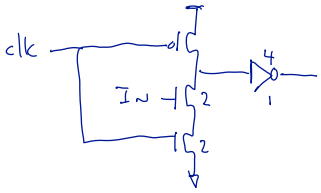
$LE_{dinv} = 2/3$

$LE_{inv} = 5/6$

$\Pi LE =$

Buffer "Average" LE

Domino buffer:



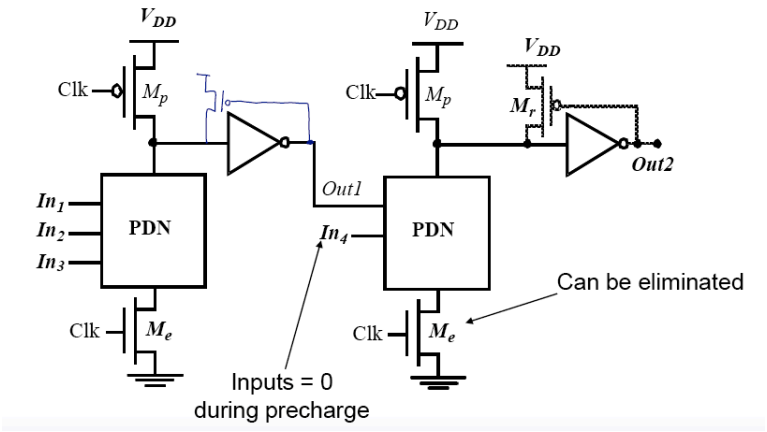
$LE_{dinv} = \frac{2}{3}$

$LE_{sinv} = \frac{5}{6}$

$\Pi LE = \frac{10}{18}$

"Average" $LE = \sqrt{10/18} \approx \frac{3}{4}$

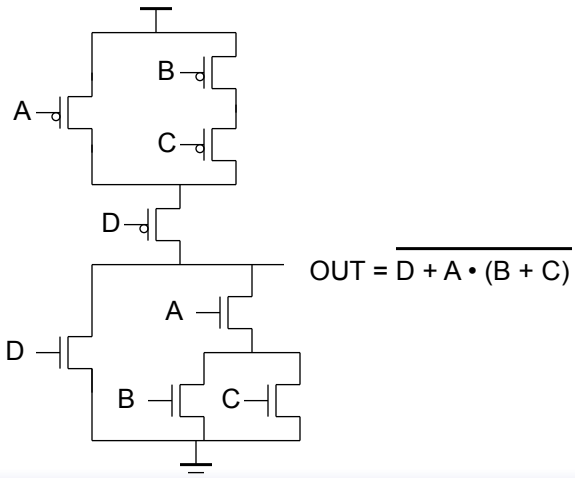
Designing with Domino Logic



Beside the first stage have to keep Me transistor, others can eliminate it.

CMOS Layout

Complex CMOS Gate

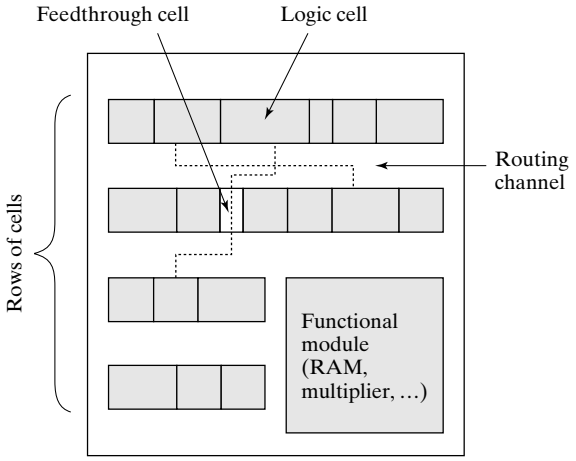


$OUT = \overline{D + A \cdot (B + C)}$

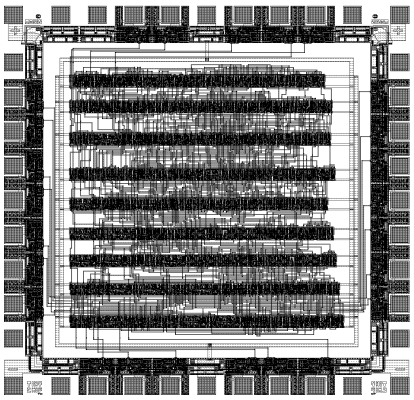
Cell Design

- Standard Cells
  - General purpose logic
  - Used to synthesize RTL/HDL
  - Same height, varying width sometimes, same width, varying height
- Datapath Cells
  - For regular, structured designs (arithmetic)
  - Includes some wiring in the cell

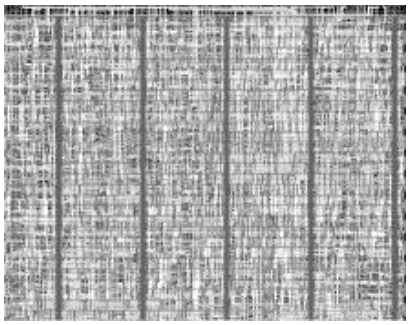
Standard Cell Methodology



Standard Cells – Then and Now

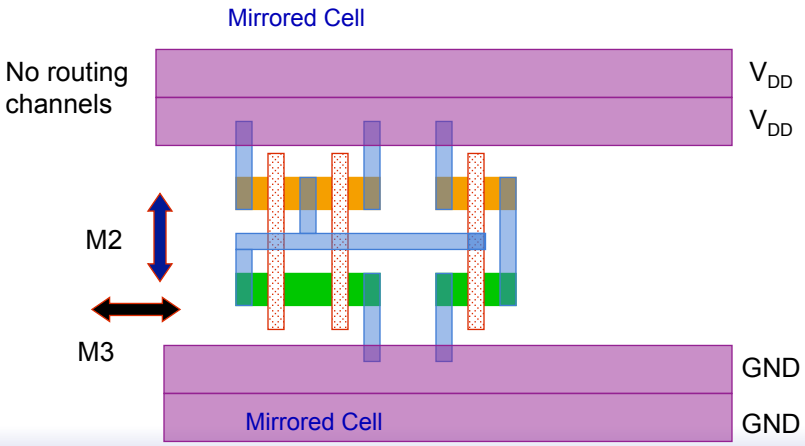


(a)

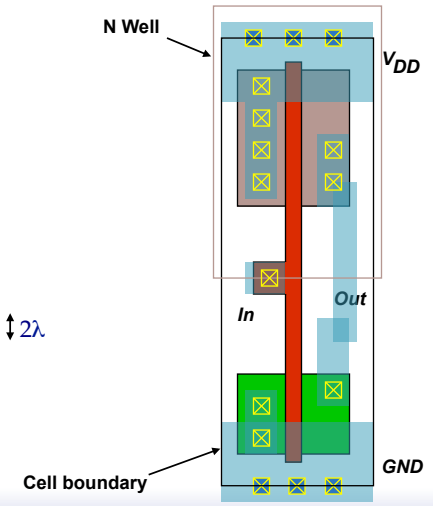


(b)

Standard Cell Layout Methodology



Standard Cells



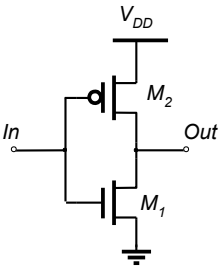
Cell height 12 metal tracks
Metal track is approx. 3λ + 3λ
Pitch =
repetitive distance between objects

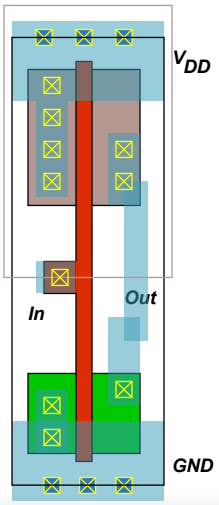
Cell height is "12 Mn pitch"

Rails ~10λ

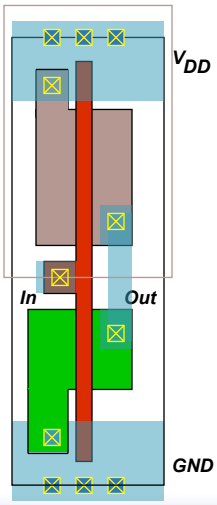
Standard Cells

With minimal diffusion routing

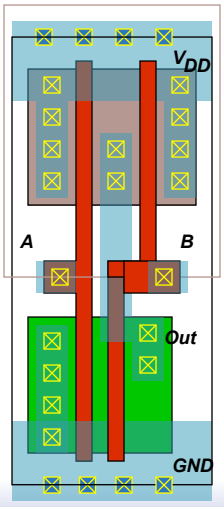




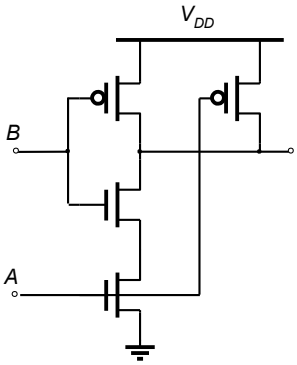
With silicided diffusion



Standard Cells



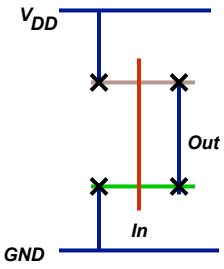
2-input NAND gate



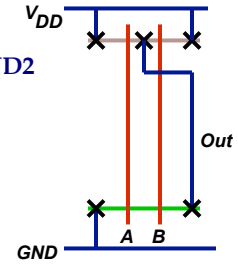
Stick Diagrams

Contains no dimensions
Represents relative positions of transistors

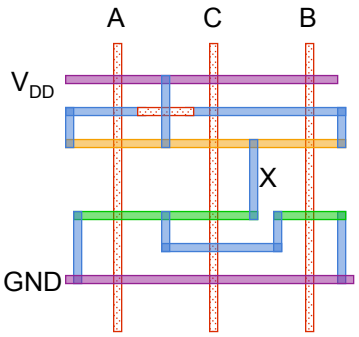
Inverter

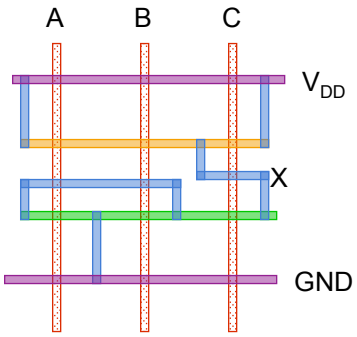


NAND2

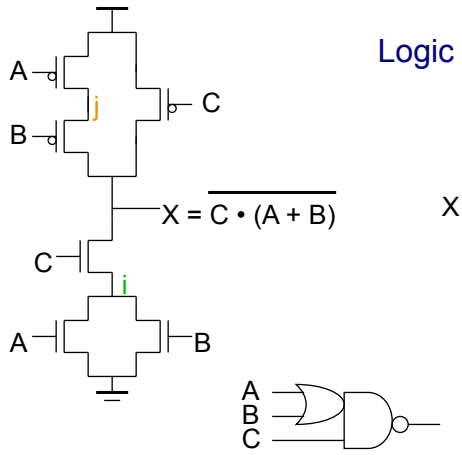


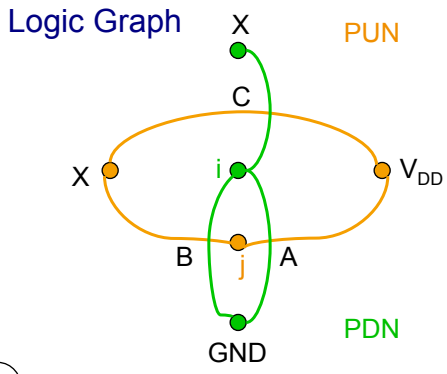
Two Versions of $\overline{C \cdot (A + B)}$





Logic Graphs





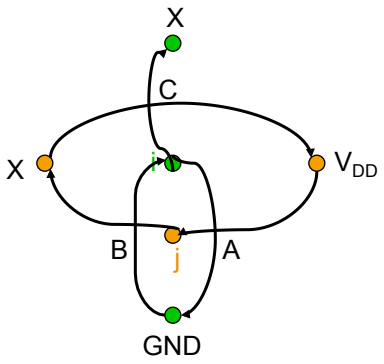
Consistent Euler Path

A B C

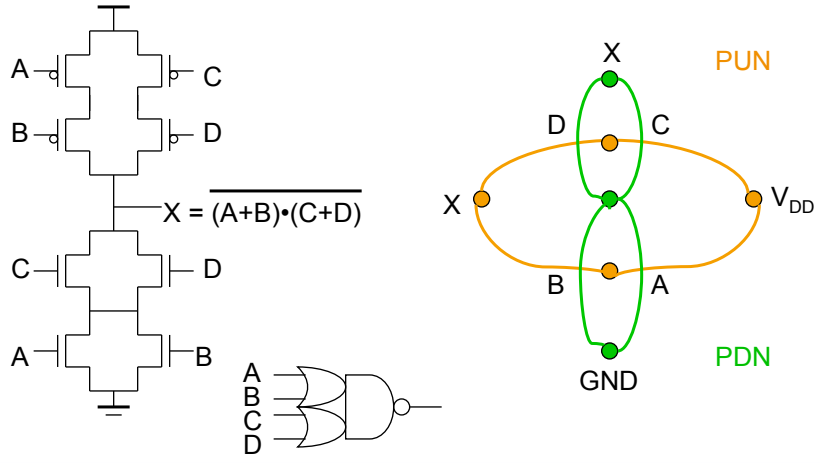
Has PDN and PUN

B C A

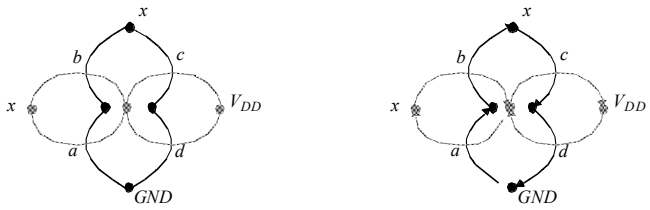
Has PUN, but no PDN



OAI22 Logic Graph

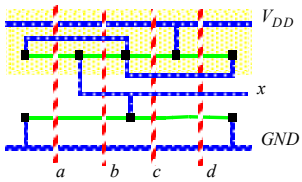


Example: x = ab+cd



(a) Logic graphs for $\overline{(ab+cd)}$

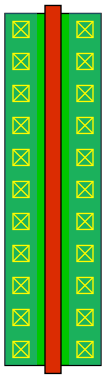
(b) Euler Paths {*a b c d*}



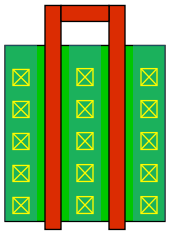
(c) stick diagram for ordering {a *b c d*}

Multi-Fingered Transistors

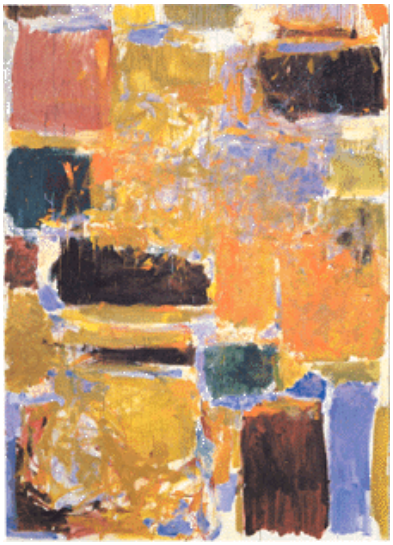
One finger



Two fingers (folded)

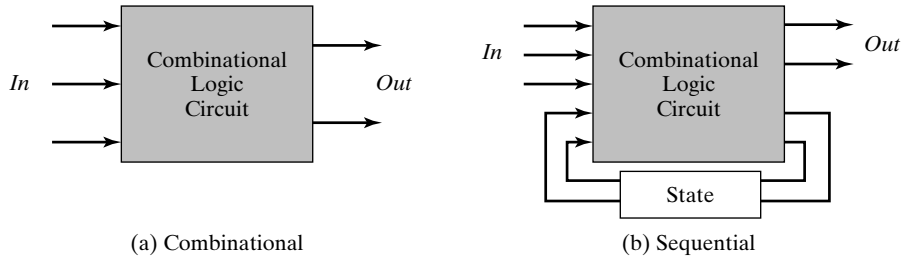


Less diffusion capacitance



Sequential Logic

Combinational vs. Sequential Logic



(a) Combinational

(b) Sequential

Output = *f*(*In*)

Output = *f*(*In, Previous In*)

Why Sequencing?

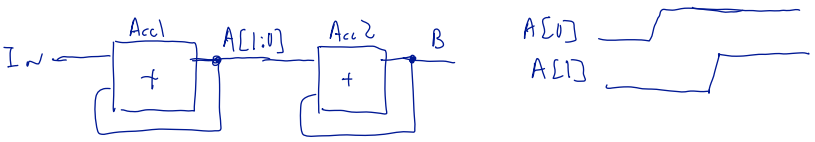
Two key (related) reasons that we need sequencing:

(1) Need to know when we are finished with a job

Why Sequential Logic?

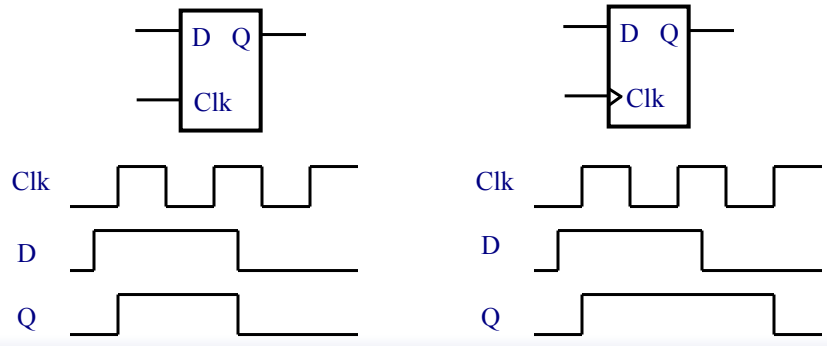
Two key (related) reasons that we need sequencing:

(2) Need to order events (aligning fast and slow)

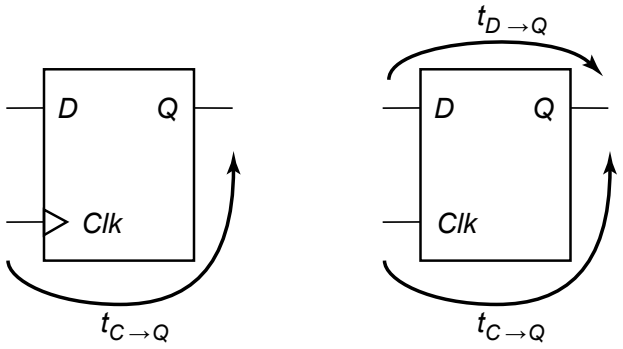


Latch versus Register (Flip-flop)

- Latch: level-sensitive
  clock is low - hold mode
  clock is high - transparent
- Register: edge-triggered
  stores data when
  clock rises

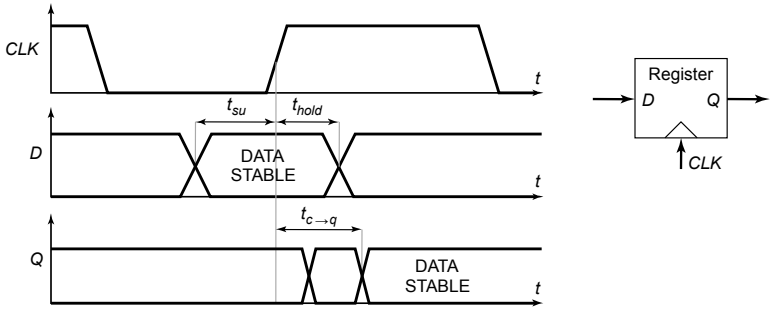


Characterizing Timing



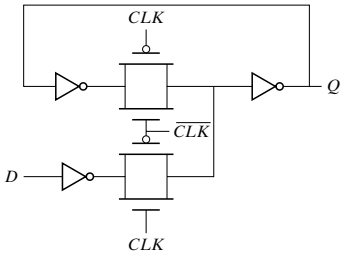
Register Latch

Timing Definitions – Set-up and Hold



Storage Mechanisms

Static



Dynamic