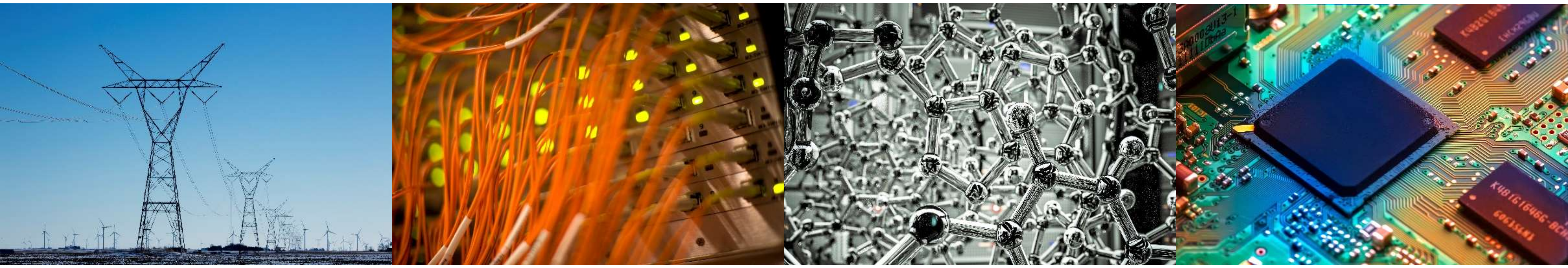
ECE120 Midterm 2 HKN Review Session



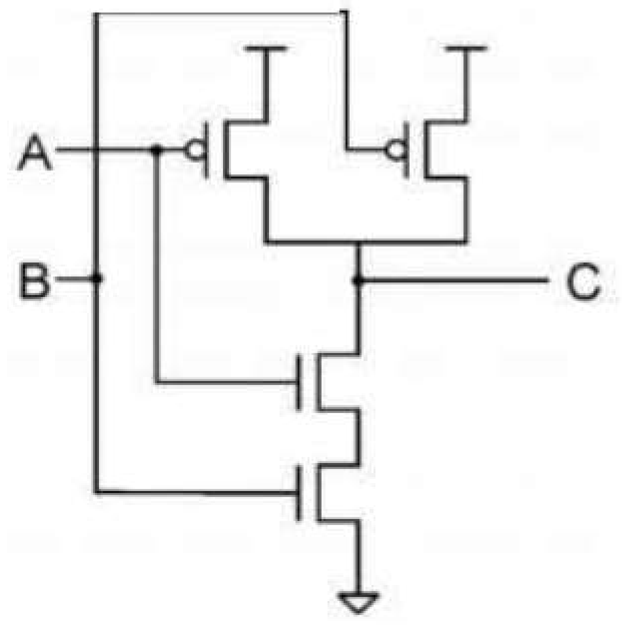
ECE ILLINOIS

ILLINOIS
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

CMOS

CMOS means Complementary Metal-Oxide-Semiconductor Transistor.

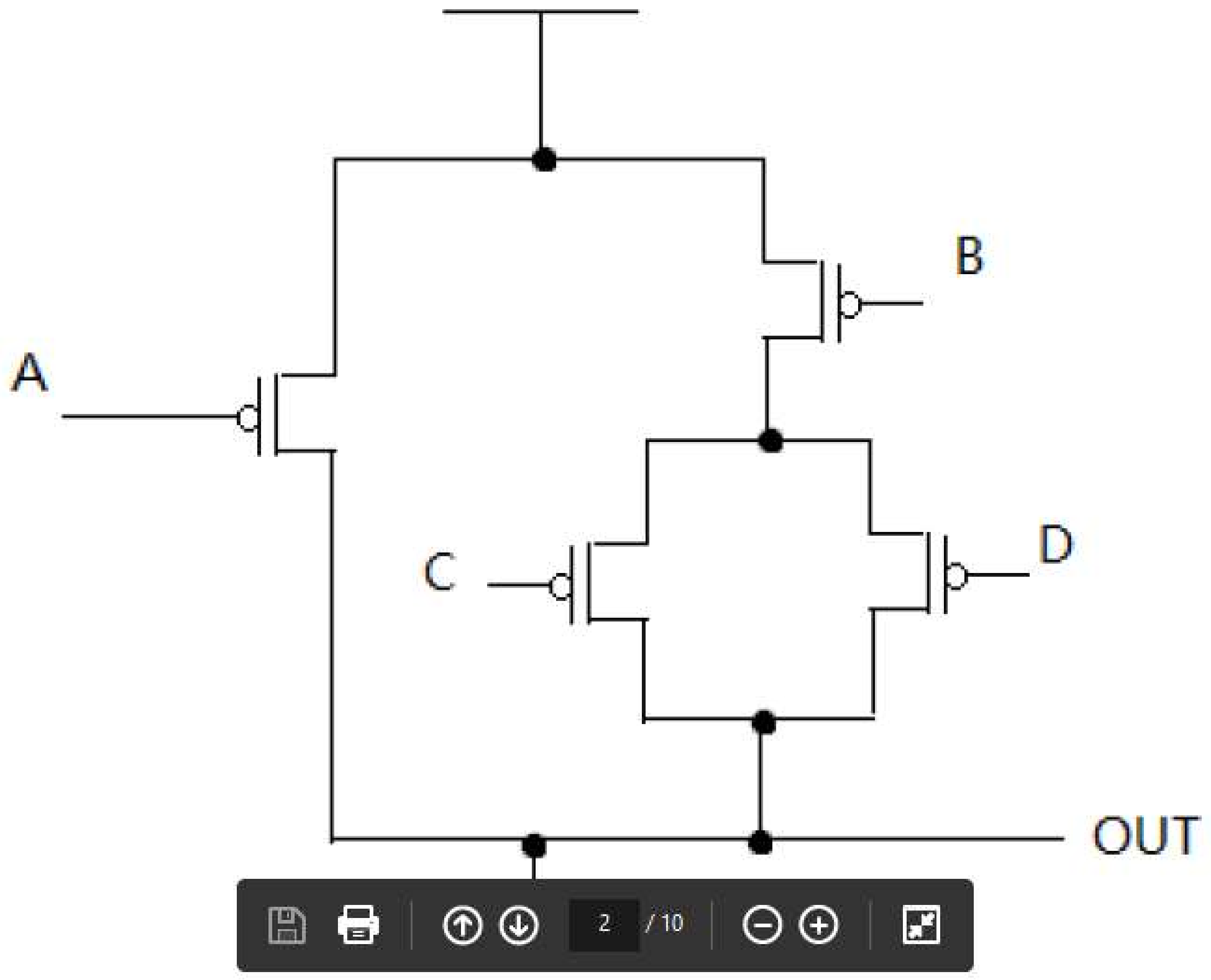
CMOS circuit contains PMOS transistor and NMOS transistor.



NMOS & PMOS

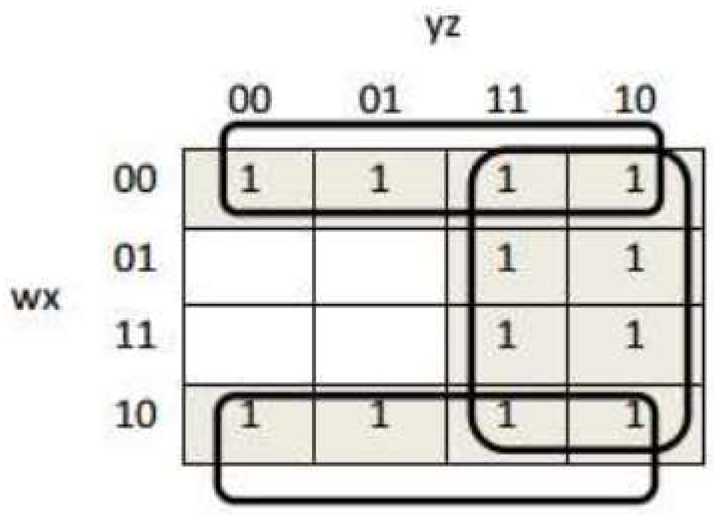
- In a NMOS, carriers are electrons
- When a high voltage is applied to the gate, NMOS will conduct
- In a PMOS, carriers are holes.
- When a high voltage is applied to the gate, PMOS will not conduct
- Parallel and series duality.

Practice



Karnaugh maps

- Gray code order: to put adjacent cells together
- Minimal expression can be derived by grouping neighboring cells into powers of 2



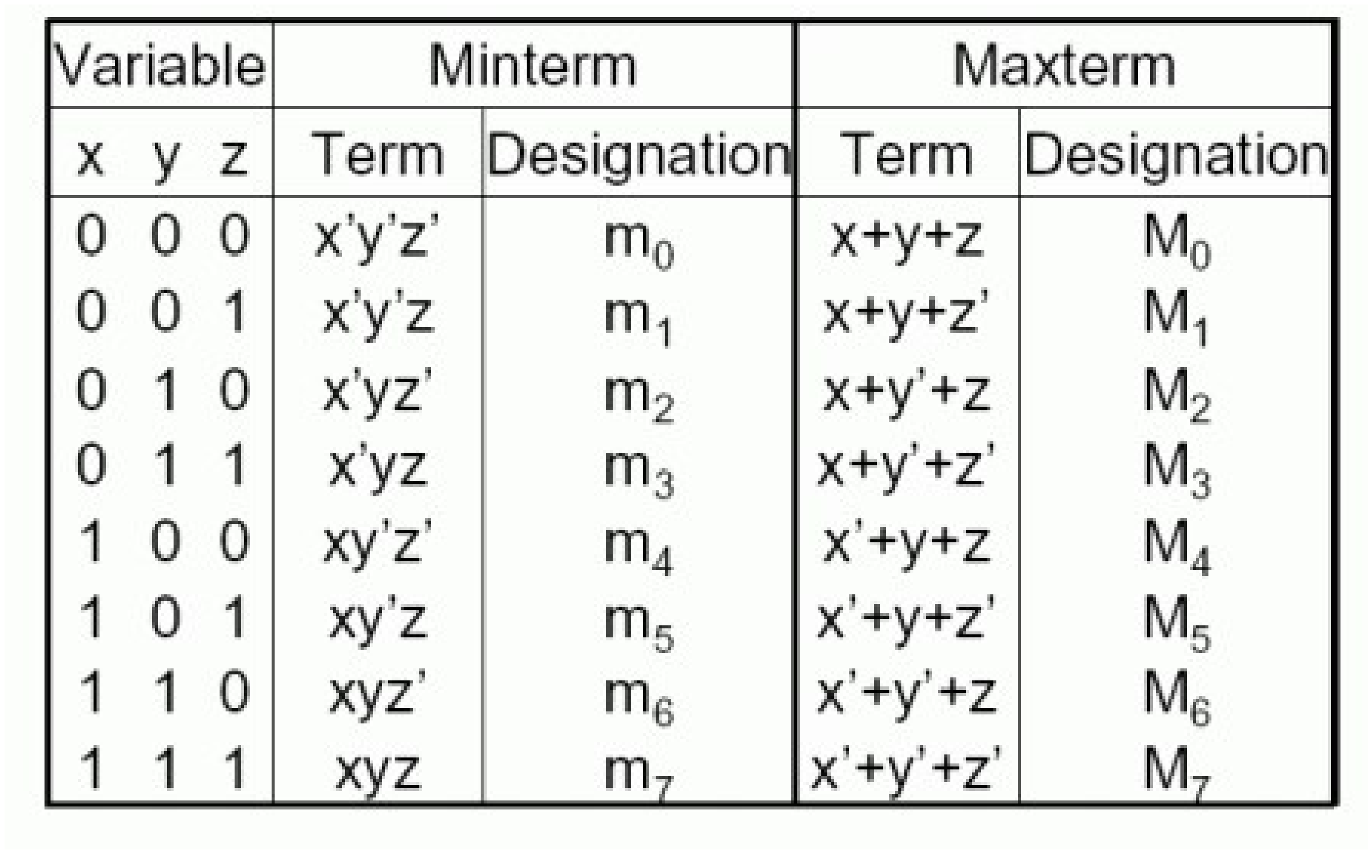
$x'+y$

- Implicant—a rectangular cover of 1's or X's
- Prime implicant—an implicant (containing at least one minterm) that is not wholly covered by a single other implicant
- Essential prime implicant—a prime implicant containing a minterm that is not covered by another prime implicant
- |Non-essential PI| + |Essential PI| = |PI|

- Cover the remaining 1’s using as few prime implicants as possible
- In other words, find minimum number of rectangles to cover all 1’s in K-map, each rectangle as large as possible

Minterms(1) and Maxterms(0)

Capitalize M for maxterm and lowercase m for minterm (have this table on your sheet)



Variable			Minterm		Maxterm	
x	y	z	Term	Designation	Term	Designation
0	0	0	x'y'z'	m_0	x+y+z	M_0
0	0	1	x'y'z	m_1	x+y+z'	M_1
0	1	0	x'yz'	m_2	x+y'+z	M_2
0	1	1	x'yz	m_3	x+y'+z'	M_3
1	0	0	xy'z'	m_4	x'+y+z	M_4
1	0	1	xy'z	m_5	x'+y+z'	M_5
1	1	0	xyz'	m_6	x'+y'+z	M_6
1	1	1	xyz	m_7	x'+y'+z'	M_7

Boolean Algebra Properties

- Suppose a and b are literals. Then, a·b as well as ab reads “a AND b”, a+b reads“a OR b”, a’ reads “NOT a” (same as)
- Writing truthtables
- POS: (a+b) • (x’+y)
  - Circle all rows where f(x, y, …) = 0

  SOP: ab+x’y+abxy
  - Circle all rows where f(x, y, …) = 1

Proof: exhaustion or find a counter example.

- POS to SOP:

  $(A+B)(C+D+E) = AC+AD+AE+BC+BD+BE$
- SOP to POS:
- use Boolean algebra distributivity property: $A+BC=(A+B)(A+C)$
- Example:

  $wx'+y+vz$

  $= (w+y)(x'+y)+vz$

  $= [(w+y)(x'+y)+v][(w+y)(x'+y)+z]$

  $= (w+y+v)(x'+y+v)(w+y+z)(x'+y+z)$

Canonical forms

- Boolean function expressed as a sum of minterms is termed the *canonical sum of products* form of the function
- Example:

$$f = x'y'z' + x'y'z + x'yz' + x'yz + xyz'$$

$$= m0 + m1 + m2 + m3 + m6$$

$$= \Sigma(m0,m1,m2,m3,m6)$$

- Boolean function expressed as a product of maxterms is termed the *canonical product of sum* form of the function
- Example:

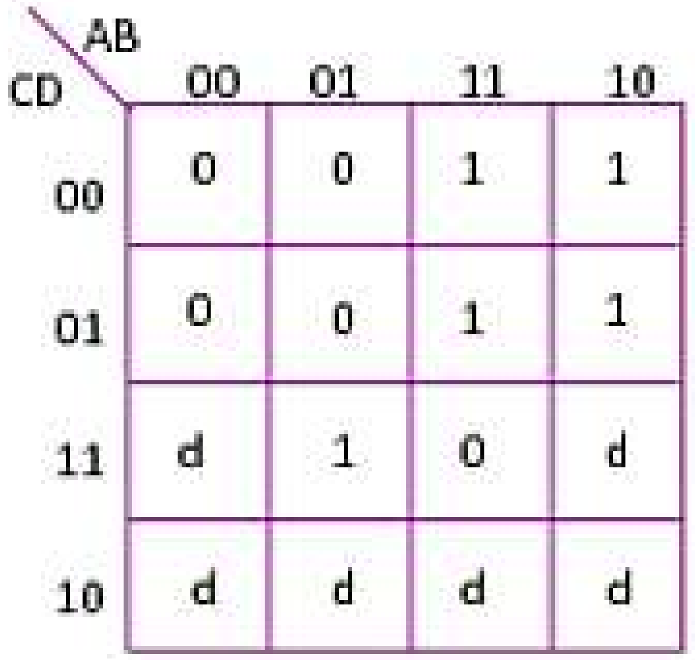
  $f = (x' + y + z)(x' + y + z')(x' + y' + z')$

  $= M4\ M5\ M7$

  $= \Pi(M4, M5, M7)$
- JUST a binary to decimal
- Max = 0, Min = 1

Don’t Cares

- Implicants can include Don’t cares but cannot be made up of only don’t cares.
- Use don’t cares to make your SOP or POS simpler. Include them only when it reduces the SOP/POS expression.



CD \ AB	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	d	1	0	d
10	d	d	d	d

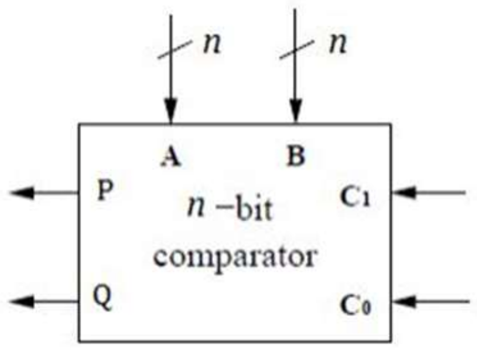
SOP/POS to CMOS

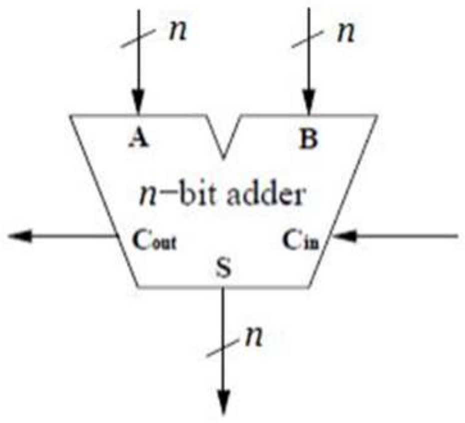
- SOP => AND/OR
- AND/OR => NAND
- Example: F = A'B + BC + D

- POS => OR/AND
- OR/AND => NOR
- Example: F = (A + B + C')(A + C)(B')

Bit-Slice Design

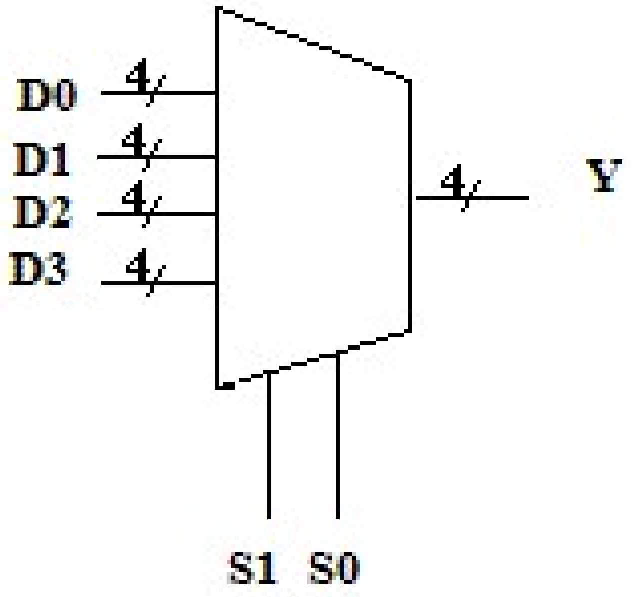
- Comparator has two outputs
  - Represent A=B, A>B, A<B (4th comb. unused)
- Adder produces sum bit and carry bit

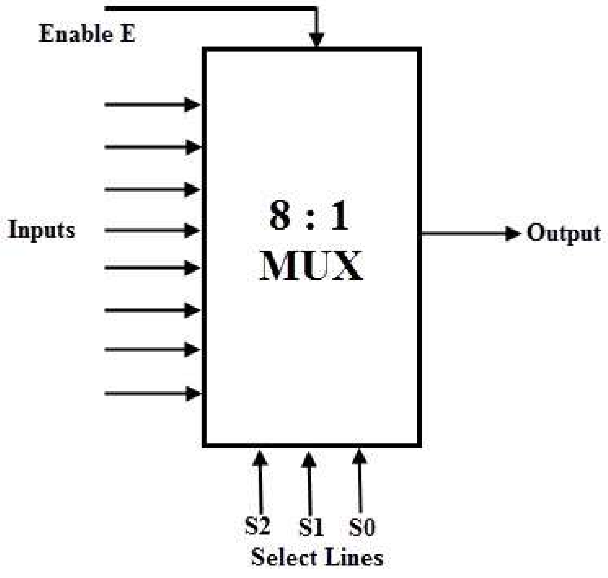


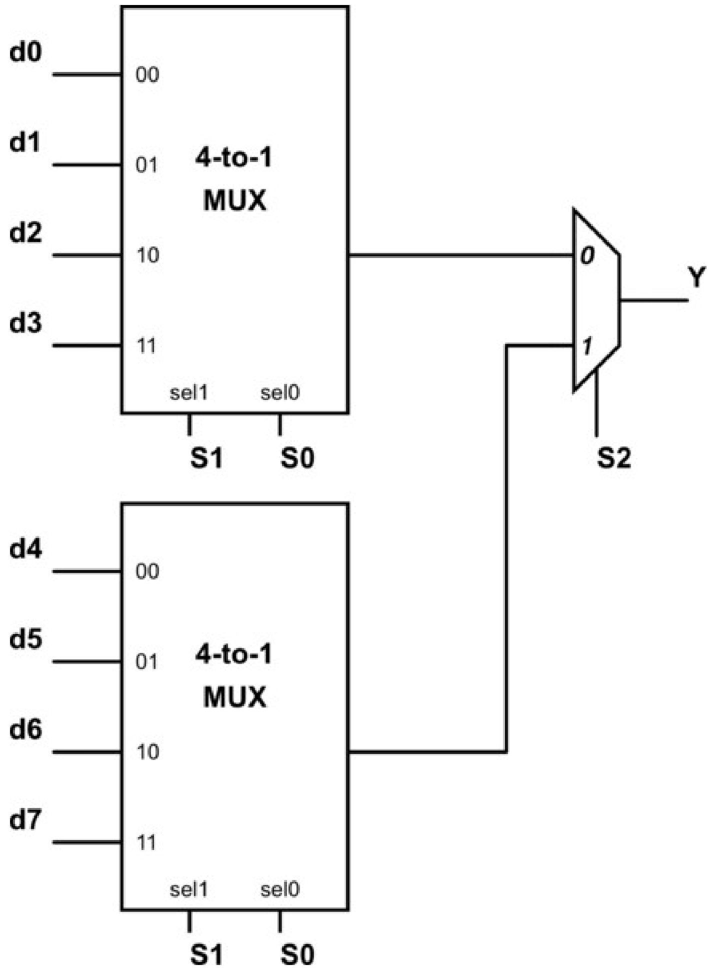


MUX

2^ns inputs for n select bits

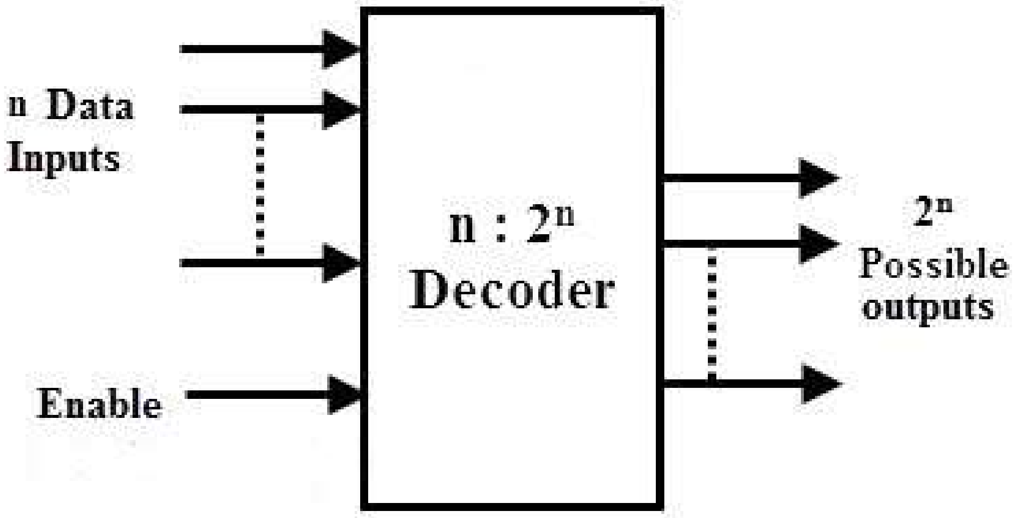






Decoders

- Decoder is a minterm generator. Any n-variable function can be implemented using a 2^n decoder and a single OR gate.
- N inputs 2^n outputs.

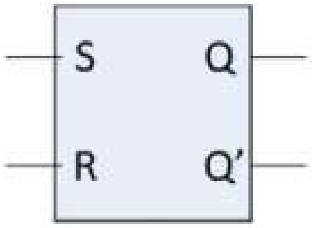


Combinational and Sequential Logic

- Combinational circuit: Output is a function of its input ONLY
- Sequential circuit: Depends on the current and past inputs.

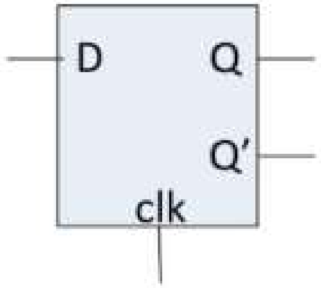
Latches

- SR Latch



S R	Q^+
0 0	Q (hold)
0 1	0 (reset)
1 0	1 (set)
1 1	forbidden

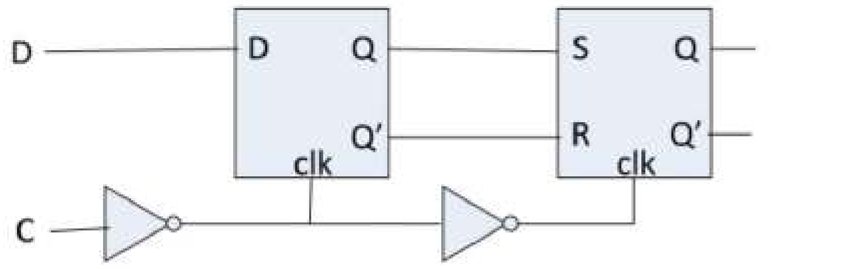
- D Latch



C	D	R	S	Q^+	
1	0	1	0	0	reset
1	1	0	1	1	set
0	0	0	0	Q	no change
0	1	0	0	Q	no change

Flip-flops

- D flip-flop

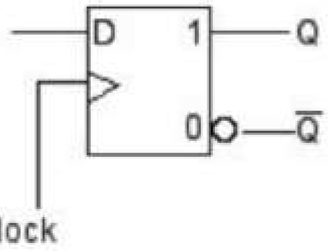


- Master D latch

Slave SR or D latch

D flip-flop

D	Q+
0	0
1	1



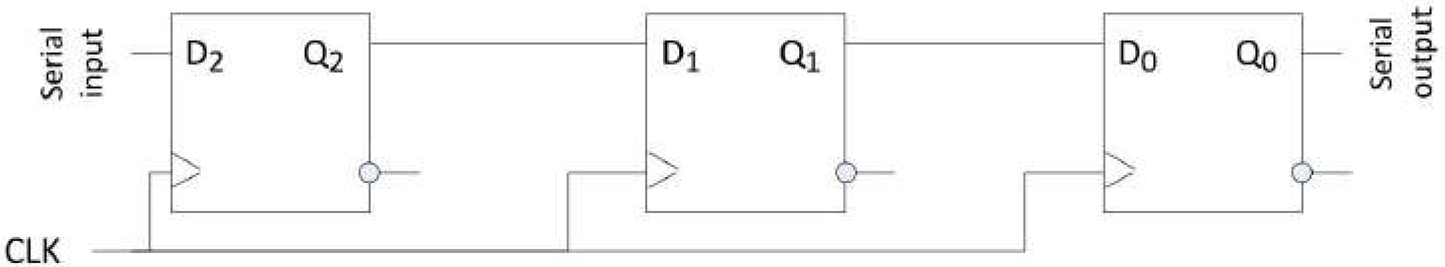
Changes happens only on the edges of clock (positive edge or negative edge)

- Other kinds of flip-flops: T, SR, JK….

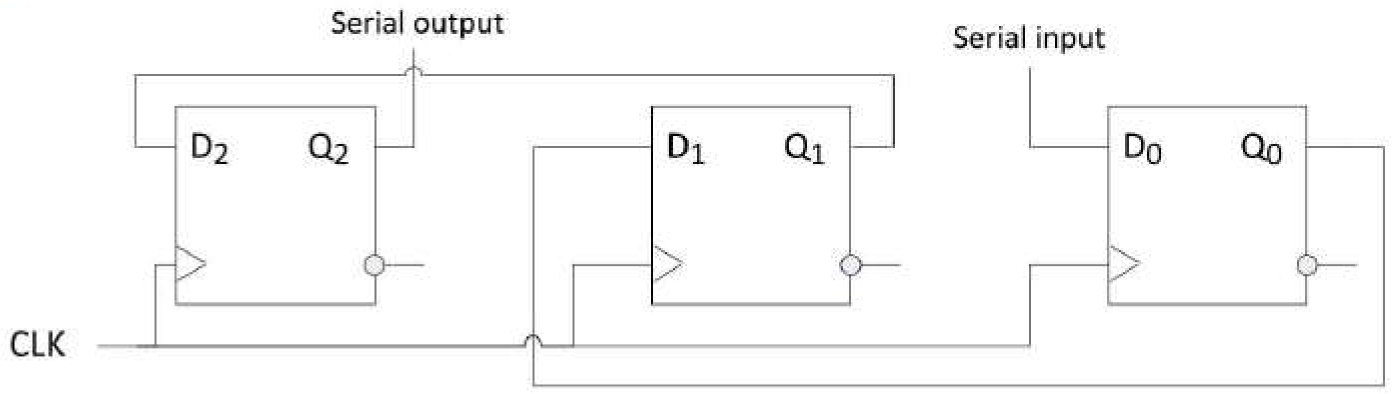
Registers

It seems like a group of flip-flops

- Logical Shift Right:

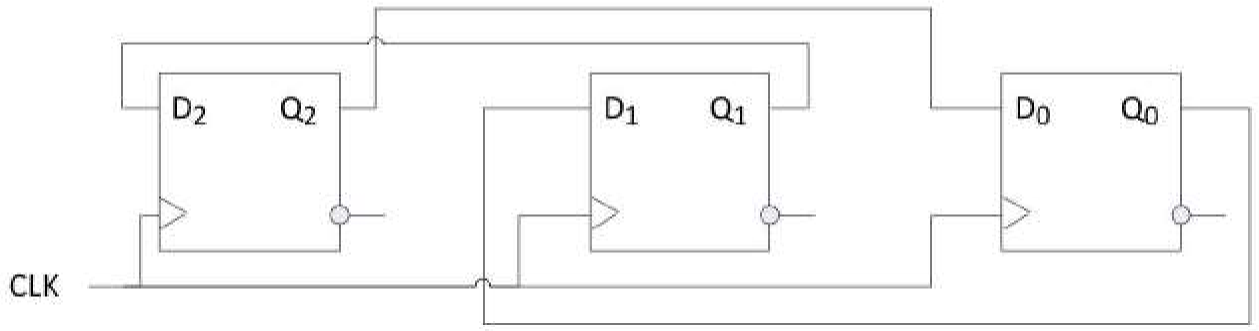


- Logical Shift Left:

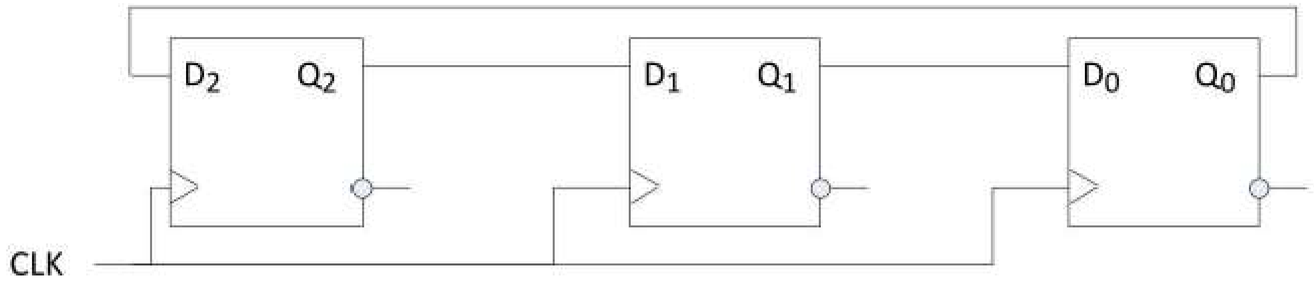


Registers

- **Circular Shift Left:**

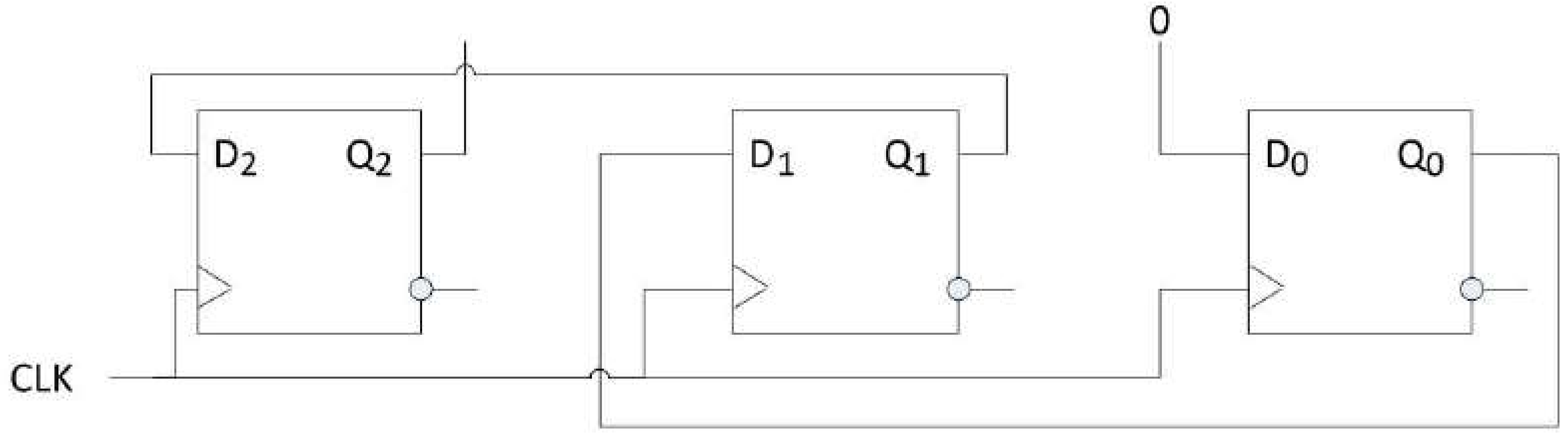


- **Circular Shift Right:**

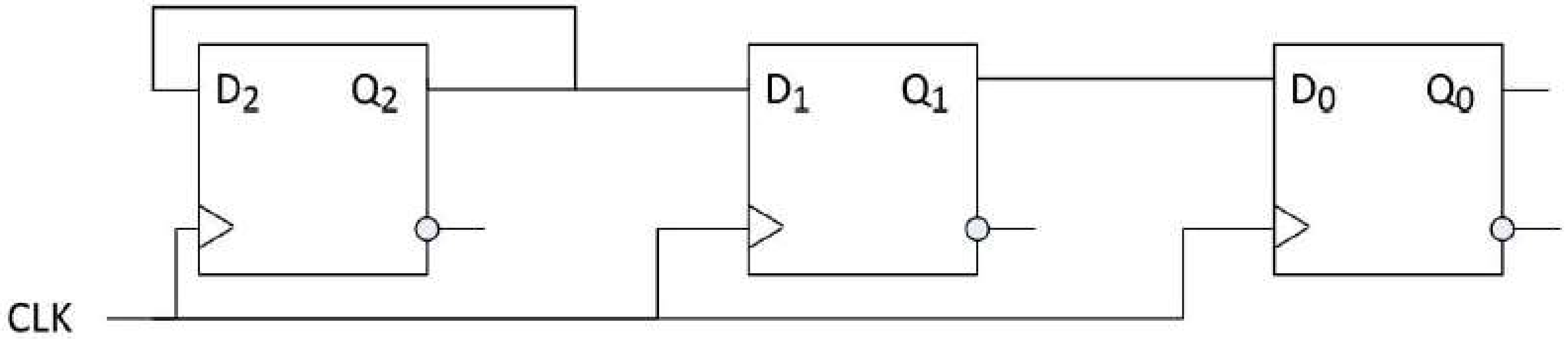


Registers

- Arithmetic Shift Left: Must shift in 0



- Arithmetic Shift Right: Can't change sign bit


D2
Q2
D1
Q1
D0
Q0
CLK