Lecture 7 key concepts

- Minterms, maxterms
- Minterm numbers, maxterm numbers
- Canonical: sum of minterms, product of maxterms expressions
- Standard: sum of products, product of sums expressions
- Active high, active low signals

Which concepts are unclear to you after viewing L7?

A. Minterm, maxterm
B. SOP, POS
C. Active high, active low
D. None

XOR Boolean expressions

Som:

XOR(a,b)= m1+m2

= a'b + ab'

PoM:

XOR(a,b)= M0.M3

= (a + b)(a' + b')

a	b	a XOR b
0	0	0
0	1	1
1	0	1
1	1	0

$$(a + b)(a' + b') = aa' + a'b + ab' + bb'$$

$$= a'b + ab'$$

x: full adder carry output

x(a,b,c) =

m3 + m5 + m6 + m7

= ∑ m (3, 5, 6, 7)

= a'bc + ab'c + abc'

+ abc

An algebraic description of all the input combinations that make x=1

a	b	c	x
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

x: full adder carry output

x(a,b,c) =

M0 ● M1 ● M2 ● M4

= π M (0, 1, 2, 4)

= (a+b+c)(a+b+c')
(a+b'+c)(a'+b+c)

An algebraic description of all the input combinations that make x=0

a	b	c	x
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Given $F(a,b,c) = \sum m(1,2)$,
and $G(c,b,a) = \sum m(1,2)$.
Is F=G?

A. Yes
B. No

Given 5-bit unsigned input, output Z=1 iff decimal value of input is between 27 and 29 (inclusive). Z in Sum-of-minterm is:

A. $\sum m(27, 28, 29)$

B. $\sum m(20, 21, 22)$

C. $\sum m(30, 31, 32)$

Given 5-bit unsigned input, output F*=0 iff decimal value of input is between 27 and 29 (inclusive). F* in Product-of-maxterm is:

A. $\Pi M(27, 28, 29)$

B. $\Pi M(20, 21, 22)$

C. $\Pi M(30, 31, 32)$

Task: show algebraically that F*’ = Z

F* = (a’+b’+c+d’+e’)(a’+b’+c’+d)

Hint:

- Take (F*)’,
- apply DeMorgan’s theorems

Z = abc’de + abcd’

Canonical vs standard form

Canonical (examples):

F(a,b,c,d,e) = abcde + a’b’c’d’e’

G(a,b,c,d,e) = (a’+b+c’+d+e’) (a+b’+c+d’+e)

Standard (examples):

U(a,b,c,d,e) = abd + c’d’e’ **SOP**

V(a,b,c,d,e) = (b+e’) (a+c+d’) **POS**

Sum of product (minterm)

F = product_*i* + product_*k* + … + product_*m*

F is 1 if any of the product is 1

- Suitable for expressing active high output F (more in L9)
- AND-OR circuit structure
- Easily implemented with purely NAND

Product of Sum (maxterm)

G = (sum_*i*)(sum_*k*) … (sum_*m*)

G is 0 if any of the sum is 0

- Suitable for expressing active low output G (more in L9)
- OR-AND circuit structure
- Easily implemented with purely NOR

NAND/NOR implementation

Apply DeMorgan's theorems to re-arrange expression into required form

SOP with NAND implementation:

abc + cde = [(abc)' (cde)'] '

r + uv' + w'xz = [r' (uv')' (w'xz)'] '

POS with NOR implementation:

(a+b+c) (c+d+e) = [(a+b+c)' + (c+d+e)'] '

(r+v')(w'+x+z') = [(r+v')' + (w'+x+z')']'

Common examples of active-high/active-low signals

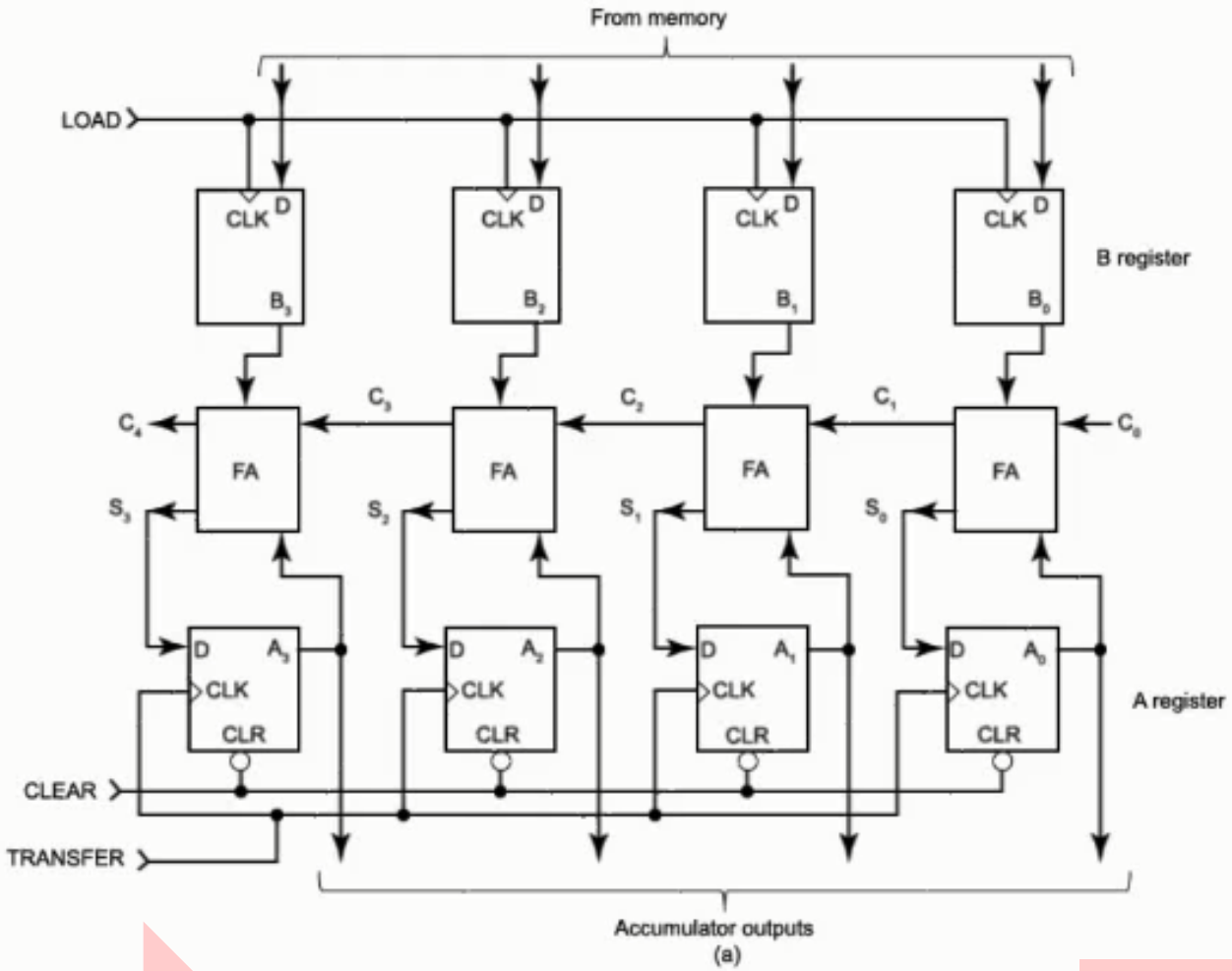
- Add*/Subtract
- Read/Write* or Read*/Write
- Pass/Fail* or Pass*/Fail
- On/Off* or On*/Off
- Odd/Even* or Odd*/Even
- Enable/Disable* or Dis/En*
- Run/Stop* or Stop/Run*
- Up/Down* or Up*/Down

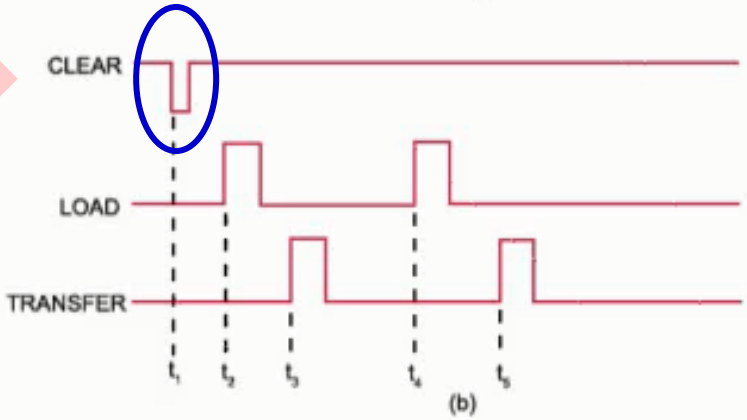
Notice that that each pair is named so as to clearly represent 2 opposite states

XOR gate: same or different?

a	b	a XOR b	SAME*	DIFFERENT
0	0	0	a=b	a=b
0	1	1	a≠b	a≠b
1	0	1	a≠b	a≠b
1	1	0	a=b	a=b

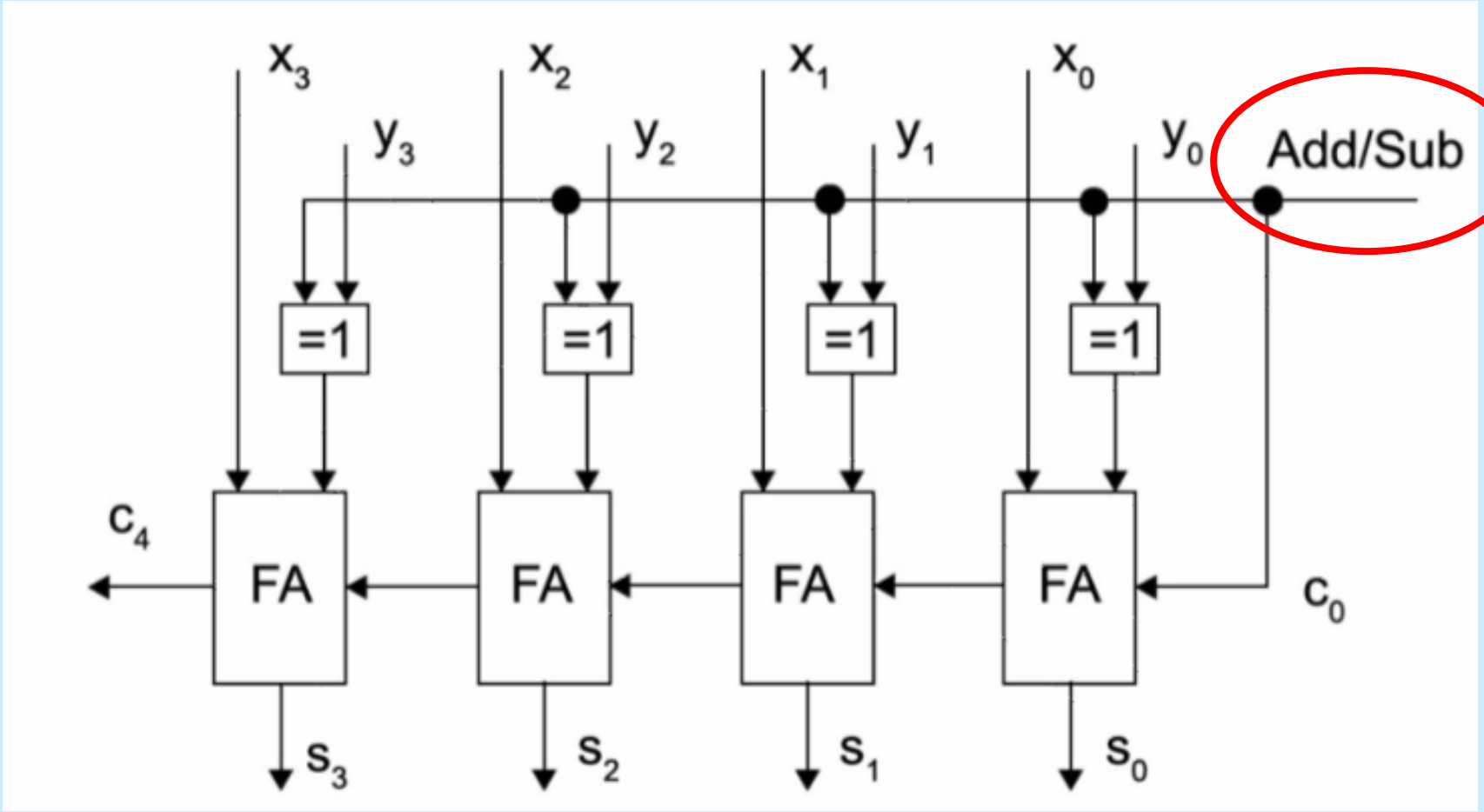
- "DIFFERENT" is active-high: 1 means "a is different from b"
- "SAME*" is active-low: 0 means "a is same as b"





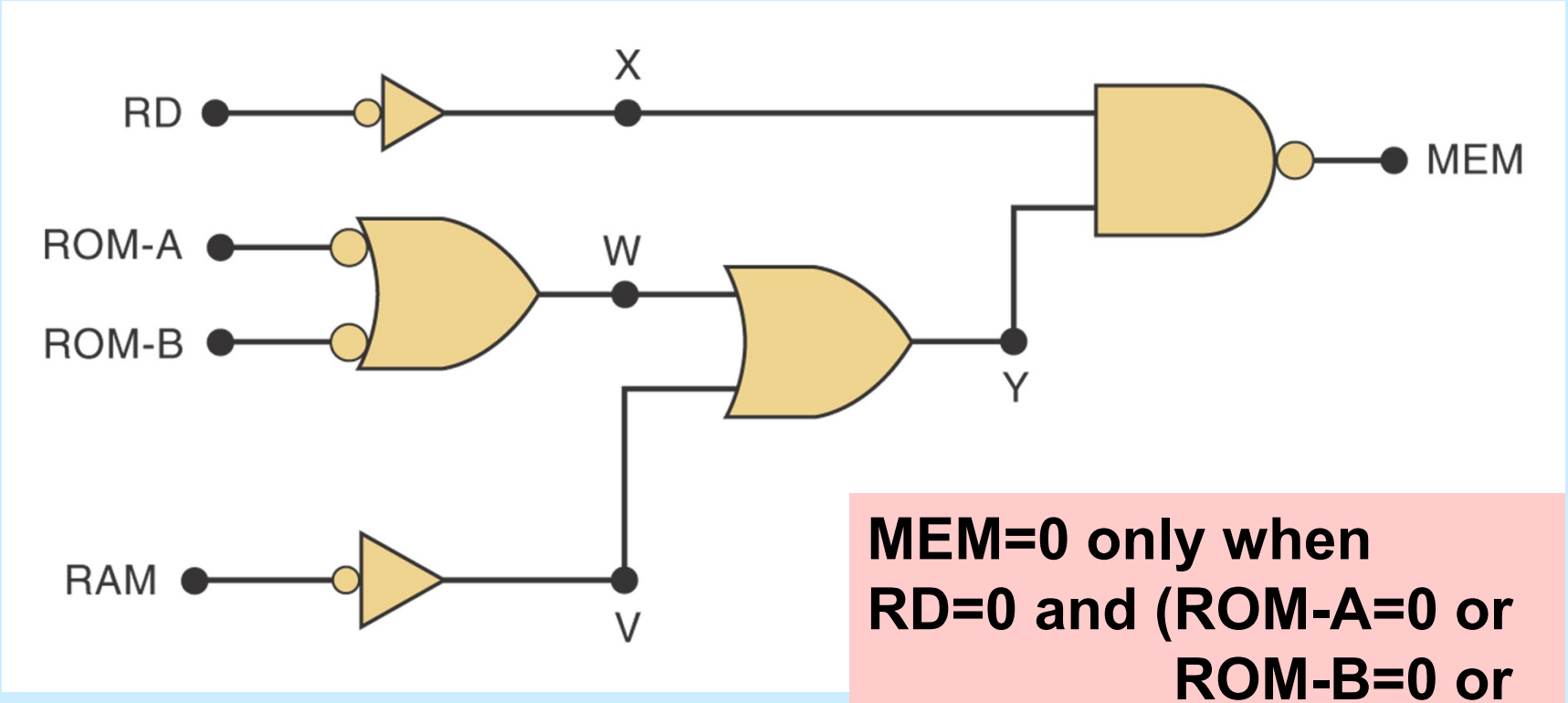
Active-low

Register A is **clear** to 0000 When **CLEAR=0**



Add is **active-low**: **Add=0** will cause the circuit to **add** x with y

Task: write POS for active-low MEM



MEM=0 only when
RD=0 and (ROM-A=0 or
ROM-B=0 or
RAM=0)

MEM = (RD + ROM-A)(RD + ROM-B)(RD + RAM)

Rename the active-low signals to make it clearer

Rename the inputs to RD*, ROM-A*, ROM-B*, RAM*

Rename the output to MEM*

Write POS:

MEM* = (RD*+ROM-A*) •
(RD* + ROM-B*) •
(RD* + RAM*)

End of L7 summary

Lecture 8 key concepts

- Karnaugh map method for logic expression simplication/minimisation
- How to construct kmap
- How to form loops: fewest, largest, loop more than once only if helpful
- How to write SOP or POS expressions
- Circuit design with “don’t care” inputs
- Combinational logic circuit design process
- Enable/Disable

Which concepts are unclear to you after viewing L8?

A. Constructing Kmap
B. From loop to expression
C. Don’t cares
D. Enable/disable
E. None

How many loops needed for minimum-cost SOP on this K-map?

X		C,D			
		00	01	11	10
A,B	00	0	0	0	0
	01	1	0	0	1
	11	0	0	1	1
	10	0	0	1	0

A. 1

B. 2

C. 3

D. 4

How many loops needed for minimum-cost POS on the same K-map?

X		C,D			
		00	01	11	10
A,B	00	0	0	0	0
	01	1	0	0	1
	11	0	0	1	1
	10	0	0	1	0

A. 1

B. 2

C. 3

D. 4

Recall Tutorial 2 Q3a

X = A'B'C'D' + A'B'CD' + A'BCD' + ABCD' + AB'CD'

X		C,D = 00	01	11	10
A,B	00	1	0	0	1
	01	0	0	0	1
	11	0	0	0	1
	10	0	0	0	1

X = A'B'(C'+C)D' + CD' (A'B' + A'B+ AB + AB')

= A'B'D' + CD'

Enable/Disable Circuits (summary)

Gate	Enable input	Enabled output	Disabled output
AND	Active Hi	Non-inverted	0
NAND	Active Hi	Inverted	1
OR	Active Lo	Non-inverted	1
NOR	Active Lo	Inverted	0

Enable/Disable Circuits (summary)

Gate	Enable input	Enabled output	Disabled output
AND	Active Hi	Non-inverted	0
NAND	Active Hi	Inverted	1
OR	Active Lo	Non-inverted	1
NOR	Active Lo	Inverted	0

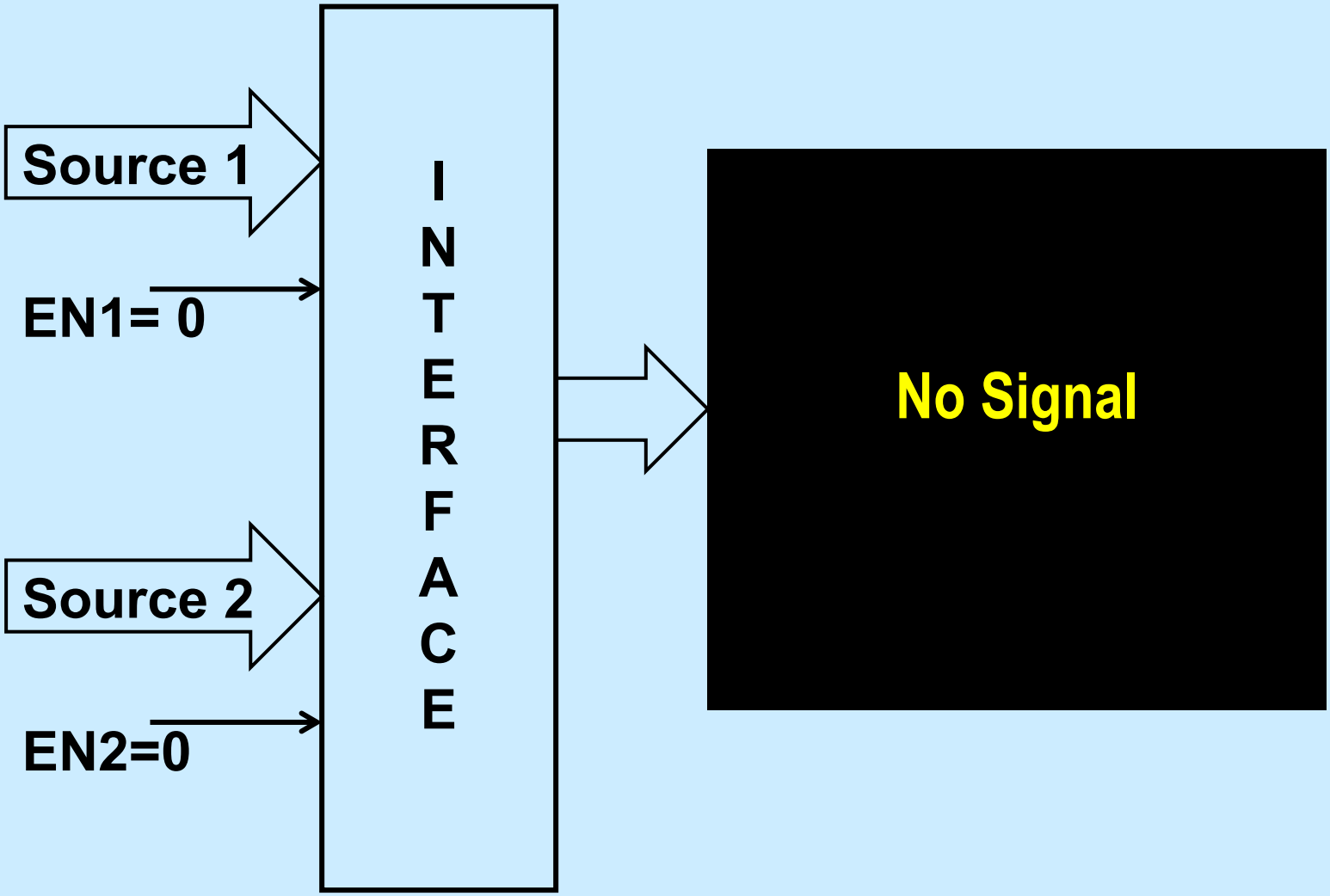
Enable/Disable Circuits (summary)

Gate	Enable input	Enabled output	Disabled output
AND	Active Hi	Non-inverted	0
NAND	Active Hi	Inverted	1
OR	Active Lo	Non-inverted	1
NOR	Active Lo	Inverted	0

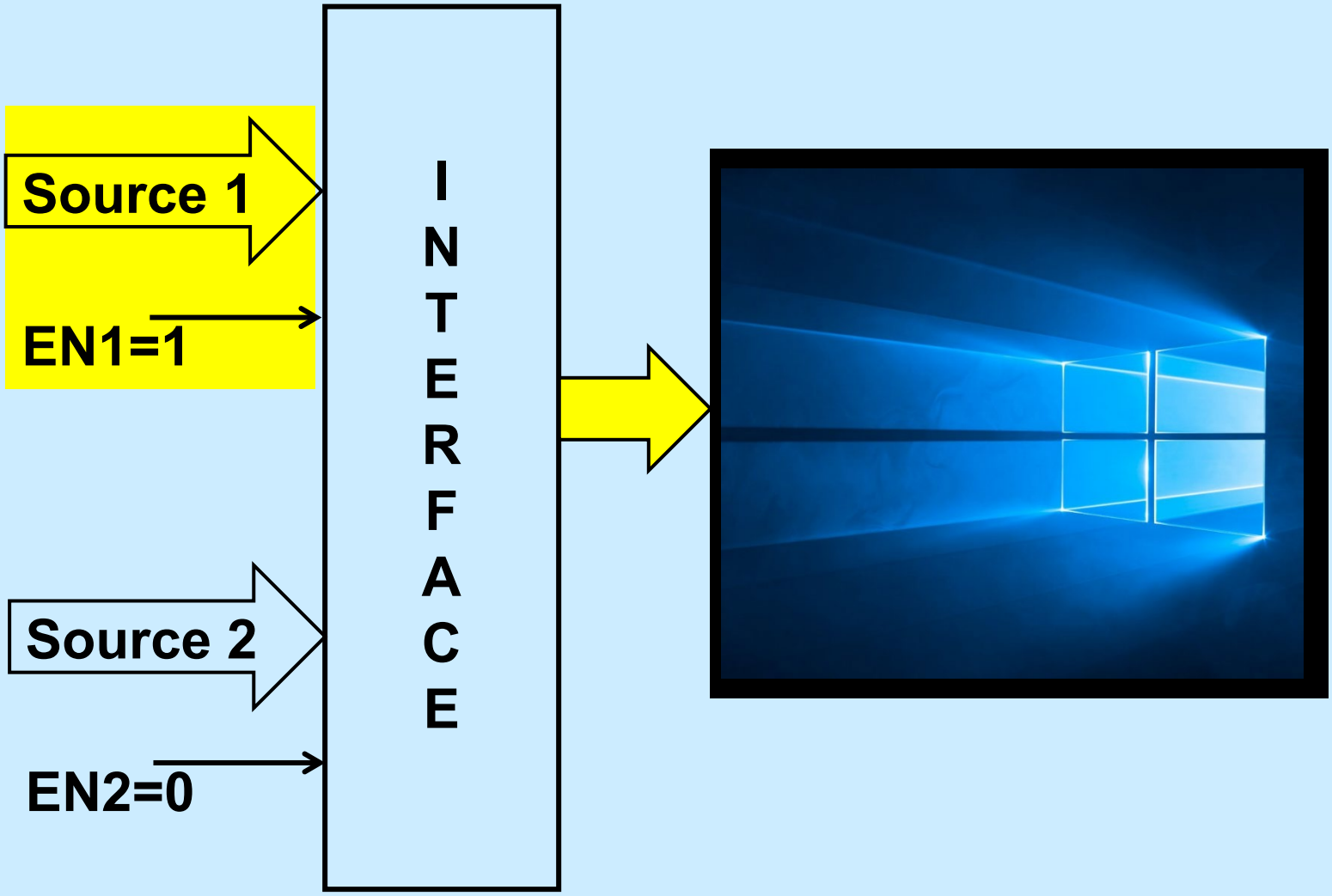
Enable/Disable Circuits (summary)

Gate	Enable input	Enabled output	Disabled output
AND	Active Hi	Non-inverted	0
NAND	Active Hi	Inverted	1
OR	Active Lo	Non-inverted	1
NOR	Active Lo	Inverted	0

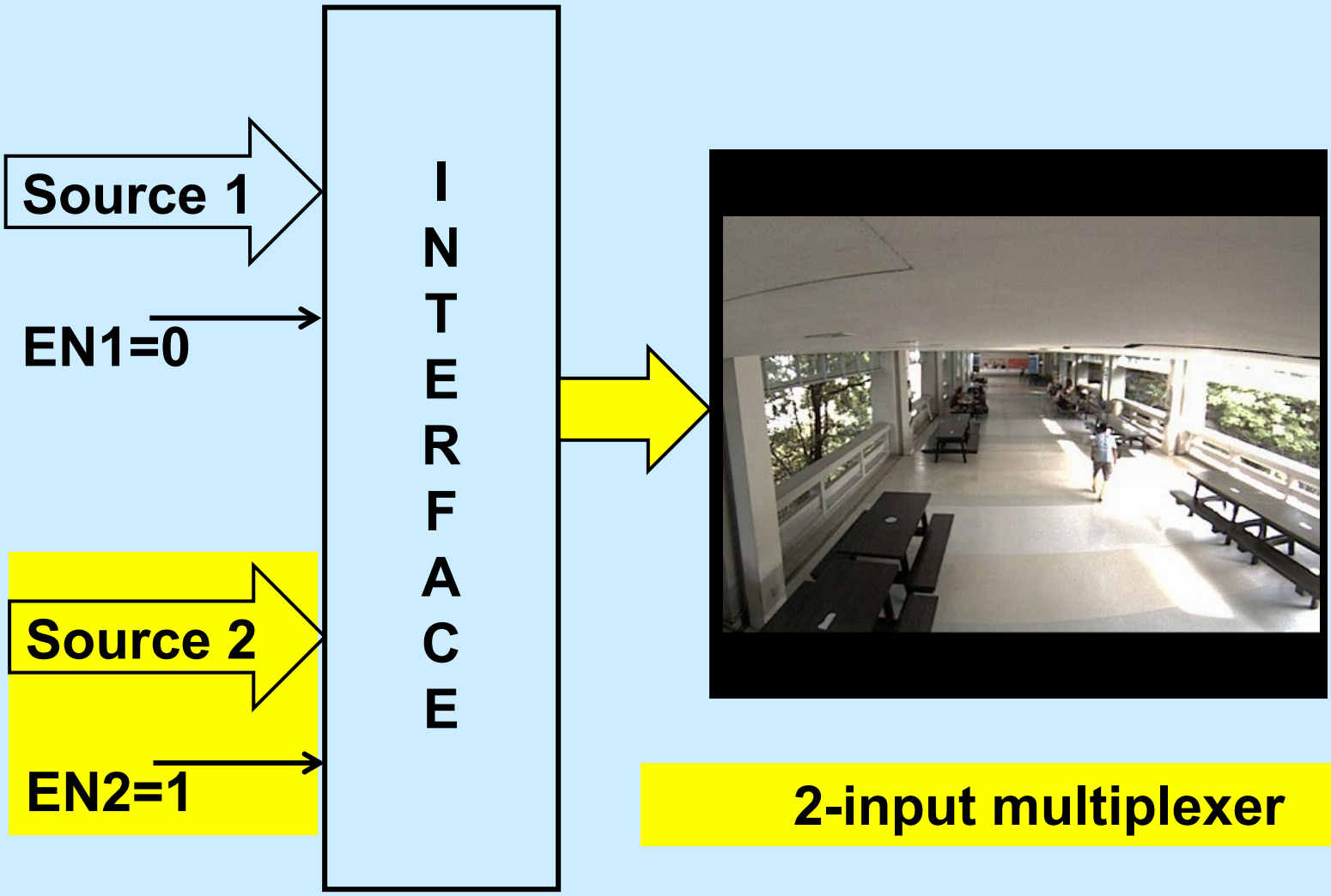
Enable/Disable application



Only source 1 enabled



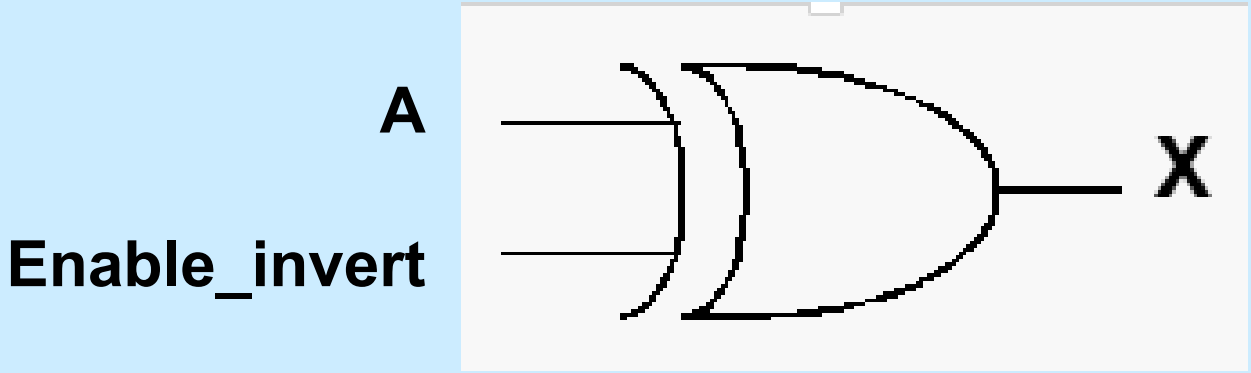
Only source 2 enabled



2-input multiplexer

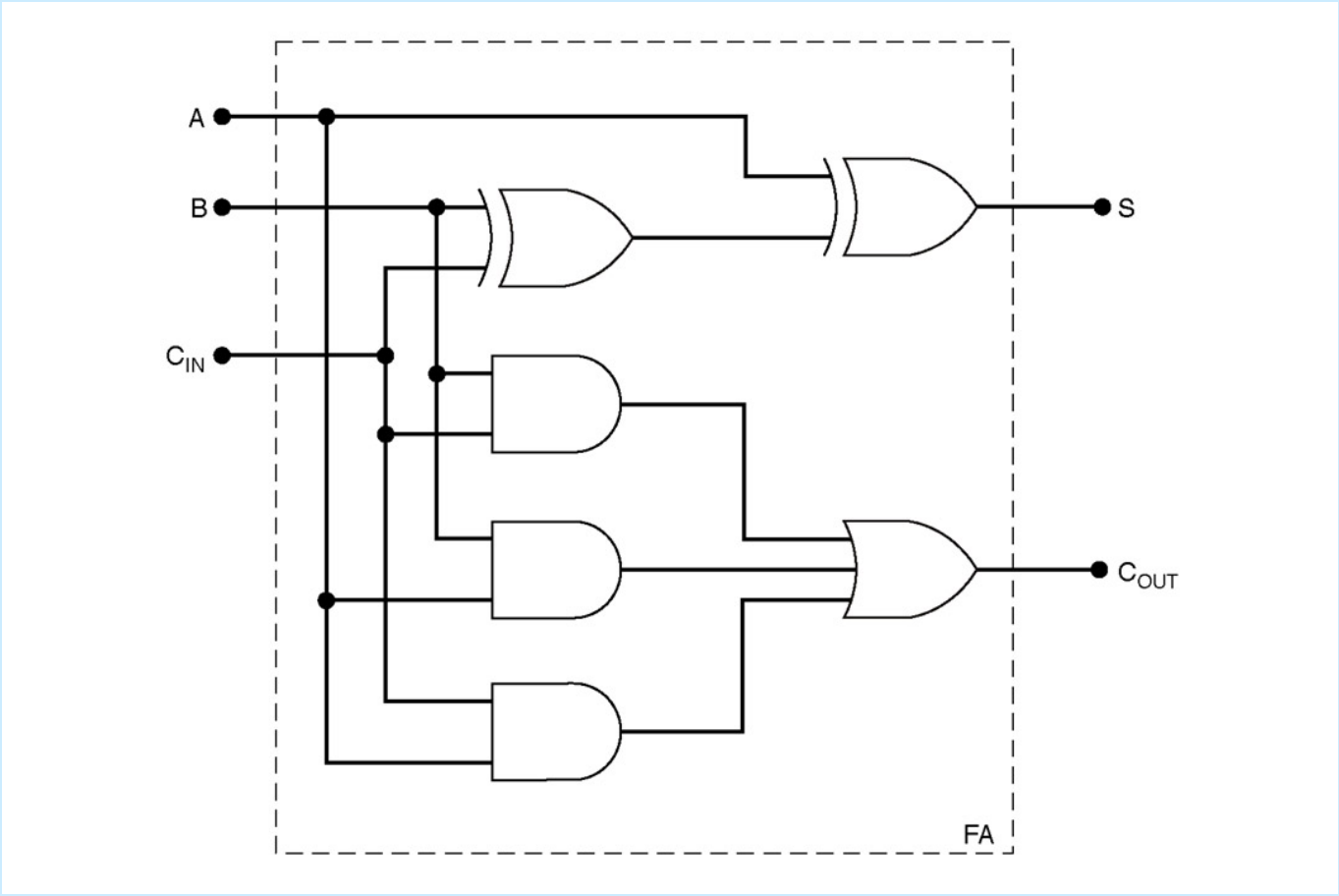
Example:
XOR as a controlled inverter

Enable_invert	Output X	X is inverted A
0 (Low)	**A**	False
1 (High)	**A'**	**True**

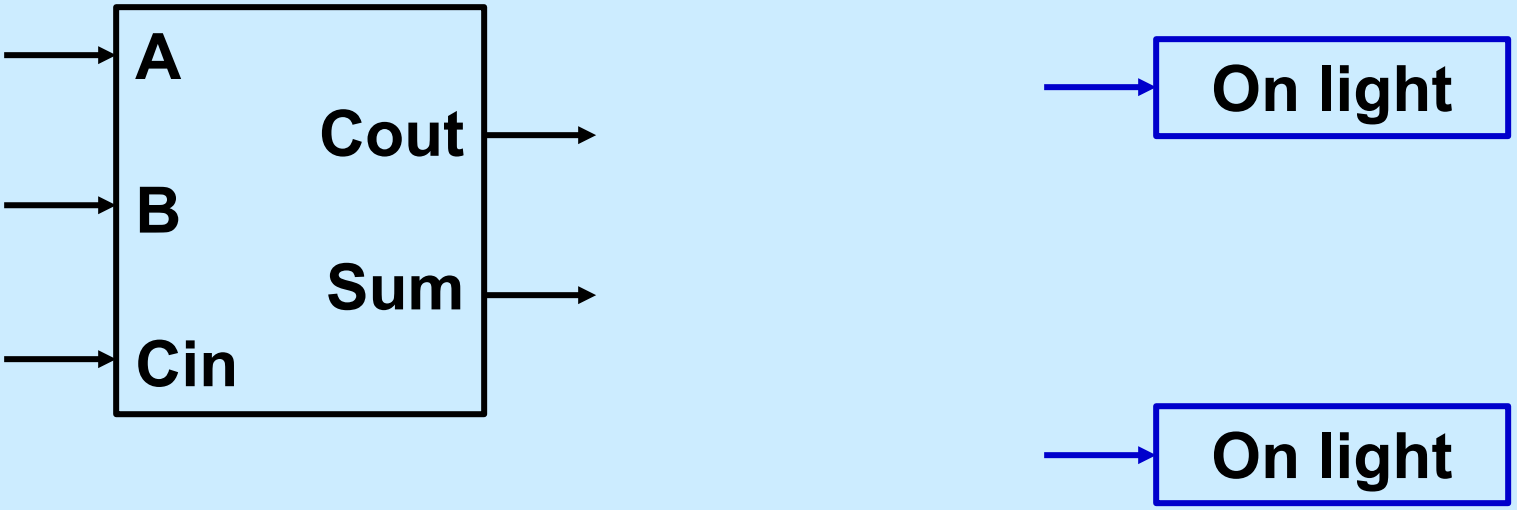


Enable_invert is said to be **active high**

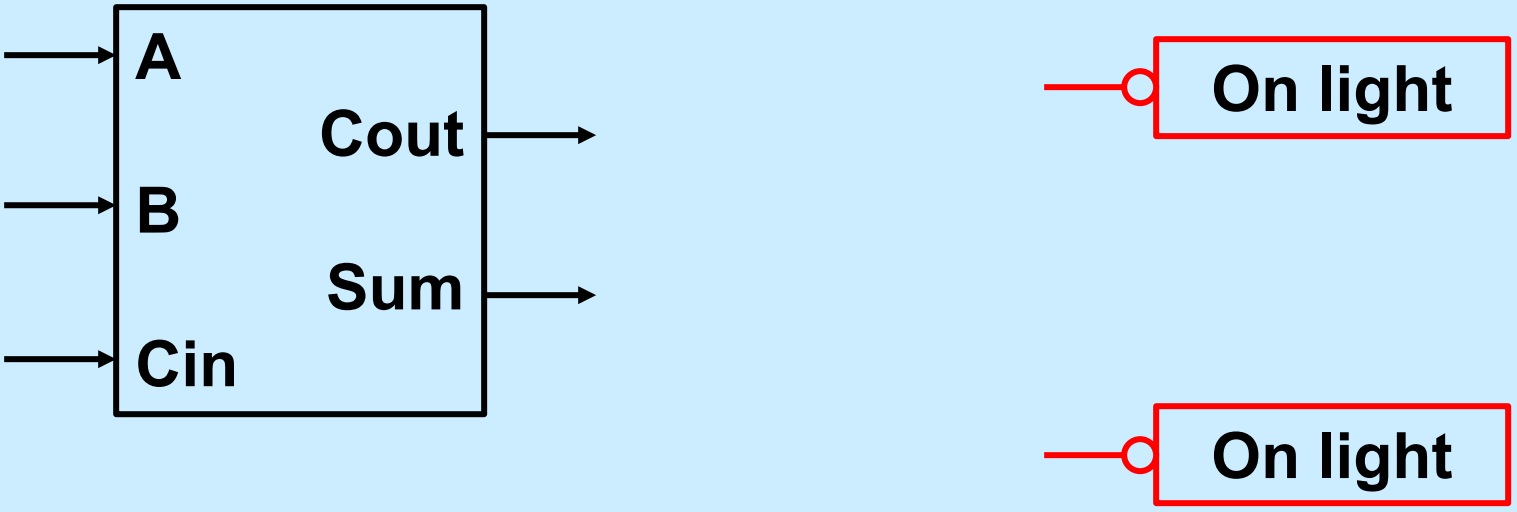
Task: How to enable/disable a FA output?



Case(a) active-Hi enable + light with active-Hi input



Case(b) active-Lo enable + light with active-Lo input



Questions to think about

- Can different canonical sum-of-minterm (or product-of-maxterm) expressions be written for a given truth table?
- What is meant by minimum-cost?
- Can different minimum-cost sum-of-product (or product-of-sum) expressions be written for a given truth table?

More questions to think about

- Does a sum-of-minterm (or sum-of-product) expression always produce 1 in the output?
- Does a product-of-maxterm (or product-of-sum) expression always produce 0 in the output?
- Why do we need enable/disable? Why not simply turn off the power supply of the circuit that needs to be disabled?

End of L8 summary