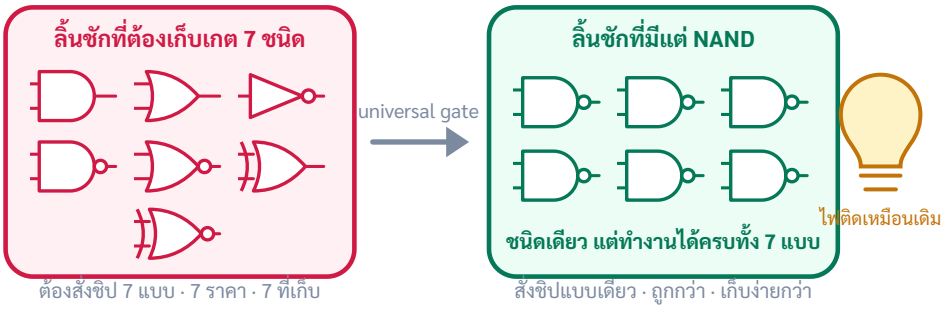
08 EP

เกตชนิดเดียวสร้างได้ทั้งโลก

วิเคราะห์วงจร · gate conversion · universal gates

ไล่นิพจน์จากอินพุตไปเอาต์พุตทีละเกต · AND-OR · AND-OR-Invert · XOR/XNOR จากเกตพื้นฐาน · NAND และ NOR เป็น universal gate · แปลง SOP เป็น NAND ล้วนด้วยเทคนิค bubble คู่ · dual symbol ที่ทำให้วงจรอ่านออกในสามวินาที



โรงงานจริงชอบแบบขวาครับ เพราะผลิตชิปชนิดเดียวพอ

พี่สอนน้องแบบไม่ต้องมีพื้นฐานมาก่อนเลยครับ · อ้างอิงสไลด์ EES271 Week 4 (หน้า 31–69) + Assignment #4 ข้อ 4, 5, 6 · **เล่มสุดท้ายของชุด Week 1–4**

ตอนนี้เราจะเดินไปทางไหนกัน

EP นี้แบ่งเป็นสามช่วงครับ — **ช่วงแรก (หัวข้อ 1–3) คือ "อ่านวงจรให้ออก"** คือไล่นิพจน์จากอินพุตไปเอาต์พุตทีละเกต · **ช่วงที่สอง (หัวข้อ 4–8) คือหัวใจของ EP** — เกตชนิดเดียวสร้างเกตอื่นได้ทั้งหมด และวิธีแปลงวงจรทั้งวงให้เป็น NAND ล้วนหรือ NOR ล้วน · **ช่วงที่สาม (หัวข้อ 9–11) คือเฉลย Assignment #4 ข้อ 4, 5, 6 (a) (b) ครบทุกข้อย่อยแบบไม่ย่อ**

1	Combinational logic คืออะไร และทำไมเกตชนิดเดียวก็พอ	ภาพรวม
2	วิเคราะห์วงจร — เขียนนิพจน์กำกับบนสายทุกเส้น	ทักษะพื้นฐาน
3	AND-OR · AND-OR-Invert · XOR / XNOR จากเกตพื้นฐาน	4 ทรงมาตรฐาน
4	Gate conversion ด้วย NOT — เติมอินเวอร์เตอร์แล้วเกตเปลี่ยนชนิด	ปูทางข้อ 5
5	★ NAND เป็น universal gate — สร้าง NOT / AND / OR / NOR / XOR / XNOR	หัวใจของ EP
6	★ NOR เป็น universal gate — สร้างครบเหมือนกัน	หัวใจของ EP
7	SOP เป็น NAND ล้วน · POS เป็น NOR ล้วน — เทคนิค bubble คู่	ทำจริงในโรงงาน
8	Dual symbol ช่วยอ่านวงจร — bubble ชนกันแล้วหายไป	ข้อ 6 ถามตรง ๆ
9	เฉลย Assignment #4 ข้อ 4 — majority ของสามอินพุต	เฉลยเต็ม
10	เฉลย Assignment #4 ข้อ 5 — จำนวน HIGH ไม่เท่ากับ LOW	เฉลยเต็ม
11	เฉลย Assignment #4 ข้อ 6 (a) และ (b) — redraw ด้วย dual symbol	ข้อยากที่สุด
12	กับดักรวม · สรุปทั้ง 8 EP · เช็คลิสต์ Midterm	ปิดชุด

วิธีใช้ชีทนี้ — I DO / WE DO / YOU DO

ทุกหัวข้อสอนจบด้วยลำดับเดียวกันสามขั้นครับ อย่าข้ามขั้นไหนนะครับ

I DO — พี่ทำให้ดู
พี่ไล่วงจรให้ดูทีละเกต **ทุกขั้นระบายสีเกตที่กำลังทำ** และบอกด้วยว่าทำไมถึงเขียนนิพจน์แบบนั้นครับ

WE DO — ทำด้วยกัน
พี่เว้นช่องบนสายไว้ให้น้องเติม **ลองเติมเองก่อนแล้วค่อยดูเฉลยข้างล่าง** เฉลยจะอธิบายทุกช่องพร้อมเหตุผลครับ

YOU DO — ตาน้อง
น้องทำเองทั้งข้อ **เฉลยอยู่ติดกันทันที** แต่ขอให้ทำให้จบก่อนค่อยเลื่อนดูนะครับ

ข้อตกลงเรื่องภาษาและสัญลักษณ์

โจทย์ทุกข้อพี่พิมพ์เป็นภาษาอังกฤษตามข้อสอบจริง แล้วอธิบายวิธีคิดเป็นภาษาไทยข้างล่างครับ น้องต้องคุ้นกับคำว่า `Write the output expression` · `universal gate` · `Redraw the circuits below using appropriate dual symbols` · `Express the logic in both SOP and POS forms` ให้ได้

เรื่องขีดบน: วิชานี้ใช้ขีดบน ($\overline{A}$) ล้วนทั้งสไลด์และ Assignment ครับ ในวงจร **"bubble" (วงกลมเล็ก ๆ ที่ขาเกต) ก็คือขีดบนนั่นแหละ** — จำประโยคนี้ไว้ให้ดี เพราะทั้ง EP นี้เล่นกับ bubble ล้วน ๆ

ทำไม EP นี้สำคัญกับคะแนน

Assignment 25% · Midterm 35% · Final 40% ครับ — **หัวข้อ 9 ถึง 11 คือเฉลย Assignment #4 ข้อ 4, 5, 6 ซึ่งกินคะแนน 25% โดยตรง** และ **Midterm วันที่ 23 กันยายน 2026 เวลา 12:00–14:00 (2 ชั่วโมง) ครอบ Week 1–7** แปลว่าเนื้อหาทั้ง 8 EP ที่ผ่านมาอยู่ในข้อสอบหมดครับ — หัวข้อ 12 มีเช็คลิสต์ให้ทวนแบบติ๊กทีละข้อ

1 Combinational logic คืออะไร และทำไมเกตชนิดเดียวถึงพอ

ตะขอเกี่ยว · รับจาก EP07

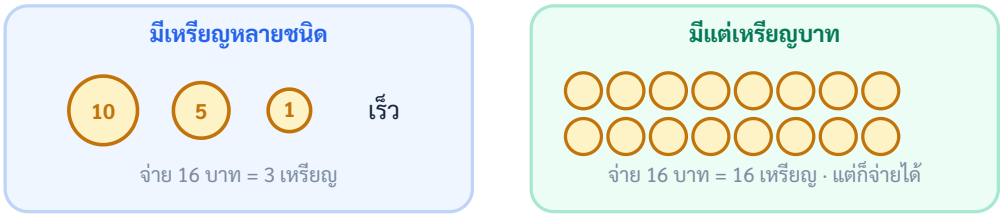
จบ EP07 น้องเดินไปกลับระหว่าง **ตารางค่าความจริง** กับ **นิพจน์** ได้คล่องแล้วครับ — เห็นตารางก็อ่านแถวที่ได้ 1 ออกมาเป็น SOP ได้ เห็นแถวที่ได้ 0 ก็เขียน POS ได้ และเห็นนิพจน์ก็กดตารางกลับได้

ตอนนี้น้องแปลงไปมาระหว่างตารางกับนิพจน์ได้แล้ว แต่ถ้าโรงงานมีแต่เกต NAND อย่างเดียวล่ะครับ จะสร้างวงจรทั้งวงจากเกตชนิดเดียวได้ไหม — คำตอบคือ**ได้ครับ ทุกวงจรในโลกนี้** และนั่นคือเรื่องที่ EP นี้จะสอนทั้งเล่ม พร้อมกับทักษะที่ต้องมาก่อนคือ **“อ่านวงจรที่วางอยู่ตรงหน้าให้ออกว่ามันคำนวณอะไร”**

ของใกล้ตัว: กระเป๋าที่มีแต่เหรียญบาท

สมมติในกระเป๋าน้องมีแต่**เหรียญ 1 บาท** ไม่มีเหรียญ 5 ไม่มีเหรียญ 10 ครับ คำถามคือ **น้องจ่ายเงิน 5 บาทได้ไหม** — ได้ครับ ก็หยิบเหรียญบาท 5 เหรียญ **จ่าย 10 บาทได้ไหม** — ได้ครับ หยิบ 10 เหรียญ

เหรียญบาทจึงเป็น “เหรียญสารพัดประโยชน์” คือมีชนิดเดียวแต่จ่ายเงินจำนวนไหนก็ได้ แลกกับการที่**ต้องใช้จำนวนเหรียญมากกว่า**ครับ



ชนิดเดียวก็ครอบคลุมทุกจำนวน แลกกับใช้จำนวนมากขึ้น

NAND กับ NOR ก็เป็น “เหรียญบาท” ของโลกดิจิทัลครับ — ชนิดเดียวสร้างได้ทุกเกต แลกกับใช้จำนวนเกตมากขึ้น

นิยามจากสไลด์หน้า 31 · ต้องจำให้ได้

Combinational logic (or combinatorial logic) is a type of digital logic implemented by Boolean circuits in which the output is a pure function of the present input only (with no memory or storage of information).

แปลเป็นภาษาพี่: วงจร combinational คือวงจรที่**เอาต์พุตขึ้นกับอินพุตตอนนี้เท่านั้น** ไม่มีความจำ ไม่เก็บอะไรไว้เลยครับ — ป้อนอินพุตชุดเดิมเมื่อไหร่ก็ได้เอาต์พุตเดิมเสมอ ไม่ว่าจะเคยป้อนอะไรมาก่อนหน้านั้นก็ตาม

ที่มาของชื่อ: `combination` แปลว่า “การรวมกัน” ครับ วงจรพวกนี้แค่**เอาค่าอินพุตมารวมกันตามกฎ** แล้วปล่อยผลออกมาทันที ตรงข้ามกับ `sequential logic` ที่มีความจำ (เช่น flip-flop ที่จะเจอในเทอมหน้า)

ของใกล้ตัวอีกอัน: สวิตช์ไฟ กับ ลิ้นชักที่มีกุญแจ

สวิตช์ไฟ = combinational
กดขึ้นไฟติด กดลงไฟดับ **ไม่ว่าเมื่อกี้จะกดมากี่ครั้ง** ดูที่ตำแหน่งสวิตช์ตอนนี้อย่างเดียวก็รู้ว่าไฟติดหรือดับครับ — นี่คือ “เอาต์พุตขึ้นกับอินพุตตอนนี้เท่านั้น” เป๊ะ ๆ

ลิ้นชักที่ล็อกอยู่ = ไม่ใช่ combinational
ลิ้นชักเปิดได้หรือไม่ **ขึ้นกับว่าเมื่อกี้น้องไขกุญแจไปหรือยัง** คือมันจำสถานะไว้ — แบบนี้เรียก sequential ครับ **EP นี้ไม่แตะเรื่องนี้เลย** ทุกวงจรใน Week 4 เป็น combinational ทั้งหมด

รายการวงจร combinational พื้นฐาน 4 ทรง (สไลด์หน้า 31)

สไลด์บอกไว้ชัดว่าวงจร combinational พื้นฐานมี 4 ทรง — พี่จะสอนครบทั้ง 4 ทรงในหัวข้อ 3 ครับ

ชื่ออังกฤษ	หน้าตานิพจน์	เอาไว้ทำอะไร
AND-OR logic	$AB + CD$	ทรงมาตรฐานของ **SOP** — AND หนึ่งตัวต่อหนึ่งพจน์ แล้ว OR รวมทุกพจน์
AND-OR-Invert logic	$\overline{AB + CD}$	เอาเอาต์พุตของ AND-OR ไปกลับค่า — ได้ทรง **POS** ออกมา
Exclusive-OR logic	$\overline{A}B + A\overline{B}$	XOR ที่จริงคือ AND 2 ตัว + OR 1 ตัว + อินเวอร์เตอร์ 2 ตัว
Exclusive-NOR logic	$\overline{A}\,\overline{B} + AB$	XNOR สร้างได้ 2 วิธี พี่จะวาดให้ดูทั้งสอง

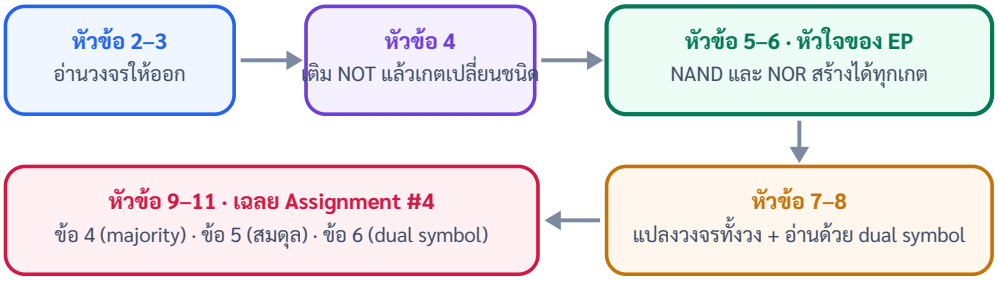
และสไลด์ยังยกตัวอย่างวงจรจริงที่สร้างจาก combinational logic ไว้ด้วยครับ — Arithmetic logic unit (ALU) · Half and full adders · Half and full subtractors · Multiplexers and demultiplexers · Encoders and decoders — **ทั้งหมดนี้คือเนื้อหา Week 6–7 ครับ ยังไม่ต้องรู้ตอนนี้ แค่รู้ว่ามันสร้างจากเกตที่น้องกำลังจะเรียนวิธีต่อ**

เทียบของเก่า · EP05 กับ EP06 ให้อะไรมาแล้วบ้าง

จาก EP05 น้องได้**สัญลักษณ์เกต 7 ตัว** กับ**ความคิดเรื่อง dual symbol**ครับ — ว่า NAND วาดได้สองหน้าตา คือหน้าตา AND-มี-bubble-ที่ขาออก กับหน้าตา OR-มี-bubble-ที่ขาเข้า **EP นี้จะเอาความคิดนั้นมาใช้จริงในหัวข้อ 8 และในเฉลย Assignment ข้อ 6**

จาก EP06 น้องได้ **De Morgan** ครับ — $\overline{AB} = \overline{A} + \overline{B}$ และ $\overline{A + B} = \overline{A}\,\overline{B}$ **ทุกครั้งที่ EP นี้พิสูจน์ว่า "NAND ทำเป็น OR ได้" จริง ๆ แล้วก็คือการใช้ De Morgan หนึ่งบรรทัดเท่านั้นเองครับ** ถ้าน้องยังไม่แม่นสองสูตรนี้ ให้ย้อนไปทวน EP06 หัวข้อ 3 ก่อน

แผนที่ของ EP นี้ — จากอ่านวงจร ไปจนถึงสร้างวงจรจากเกตชนิดเดียว



ทุกหัวข้อต่อยอดจากหัวข้อก่อนหน้าตรง ๆ ครับ ห้ามข้าม

เส้นทางของ EP08 — จบแล้วน้องจะอ่านวงจร NAND ล้วนออกในสามวินาที

ศัพท์สอบของ EP นี้ (ชุดที่ 1)

English	แปลว่า	ข้อสอบเขียนมาแบบนี้
combinational logic	ลอจิกเชิงผสม (ไม่มีความจำ)	*"...is a type of digital logic in which the output is a pure function of the present input only"*
output expression	นิพจน์ที่ปลายทางเอาต์พุต	*"Write the output expression."* = ให้เขียนนิพจน์ที่ขาออก
universal gate	เกตสารพัดประโยชน์	*"The NAND gate is a universal gate because it can be used in place of any other gates."*
in place of	ใช้แทน	เจอคำนี้แปลว่า "เอาไปใช้แทนของเดิมได้เลย"
gate conversion	การแปลงชนิดเกต	*"NOT gates can be used to convert some gates into other equivalent logic."*

English	แปลว่า	ข้อสอบเขียนมาแบบนี้
dual symbol	สัญลักษณ์คู่ (สองหน้าตาของเกตเดียวกัน)	*“The NAND symbol and the negative-OR symbol are dual symbols.”*

ทางลัดที่จะใช้ตลอดทั้ง EP

“bubble = ขีดบน” ครับ — จำประโยคเดียวนี้แล้วทุกอย่างใน EP นี้จะง่ายขึ้นทันที

เห็น bubble ที่ **ขาออก** ของเกต → เอานิพจน์ที่ได้มาครอบขีดบน (เช่น AND มี bubble = $\overline{AB}$)

เห็น bubble ที่ **ขาเข้า** ของเกต → ครอบขีดบน**เฉพาะตัวแปรที่เข้าขานั้น**ก่อนคำนวณ (เช่น OR ที่มี bubble ทั้งสองขาเข้า = $\overline{A} + \overline{B}$)

กับดักแรกของ EP นี้

อย่ารีบเดาว่าวงจรทำอะไรจากรูปทรงรวม ๆ ครับ — น้องหลายคนเห็นวงจรมี AND กับ OR ก็เดาว่าเป็น $AB + CD$ ทั้งที่จริง ๆ อาจมี bubble แอบอยู่ที่ขาไหนสักขา แล้วคำตอบเปลี่ยนไปทั้งข้อ

วิธีกันพลาด: ใช้ปากกาไล่**ทีละเกตจากซ้ายไปขวา** และ**เขียนนิพจน์กำกับลงบนสายทุกเส้น** ห้ามคิดในหัวข้ามเกต — หัวข้อถัดไปคือวิธีทำแบบนั้นครับ

2 วิเคราะห์วงจร — เขียนนิพจน์กำกับบนสายทุกเส้น

ตะขอเกี่ยว · ต่อจากหัวข้อ 1

EP07 น้องได้**ตาราง**มาแล้วเขียน**นิพจน์**ครับ แต่ข้อสอบวิชานี้ชอบให้**รูปวงจร**มาแทน แล้วสั่งว่า **"Write the output expression"** — **สมมติว่าถ้าน้องเจอรูปวงจรที่มีเกต 5 ตัวต่อกันเป็นทอด ๆ ละครับ จะเริ่มจากตรงไหนดี** คำตอบคือเริ่มจาก**ซ้ายสุด** แล้วเดินไปขวาทีละเกต และ**เขียนคำตอบกำกับลงบนสายทุกเส้นที่ผ่าน**ครับ

ของใกล้ตัว: ติดสติกเกอร์ชื่อของบนลิ้นชักทุกใบ

ลองนึกถึงตู้ที่มีลิ้นชัก 5 ใบเรียงกันครับ ถ้าน้องอยากรู้ว่าใบสุดท้ายมีอะไร วิธีที่ช้าแต่**ไม่มีวันพลาด**คือ **เปิดใบที่ 1 → เขียนสติกเกอร์แปะว่ามีอะไร → เปิดใบที่ 2 → แปะ → ...** ไล่ไปจนถึงใบสุดท้าย

วงจรก็เหมือนกันเป๊ะครับ — **สายไฟแต่ละเส้นคือลิ้นชักหนึ่งใบ** และ**นิพจน์ที่เราเขียนกำกับบนสายก็คือสติกเกอร์** ถ้าน้องแปะครบทุกเส้นพอถึงเส้นสุดท้ายคำตอบจะโผล่มาเองโดยไม่ต้องคิดเลยครับ

วิธีทำ 4 ขั้น · ใช้ได้กับทุกวงจรในข้อสอบ

ขั้น 1 — หาอินพุตทั้งหมดที่ขอบซ้าย แล้วเขียนชื่อตัวแปรกำกับไว้ (A, B, C, ...)

ขั้น 2 — หาเกตที่ "ขาเข้าทุกขารู้ค่าแล้ว" — ตอนเริ่มต้นคือเกตที่ติดกับอินพุตโดยตรง

ขั้น 3 — เขียนนิพจน์ที่ขาออกของเกตนั้นลงบนสาย ตามชนิดเกต (AND = คูณ · OR = บวก · bubble = ครอบขีดบน) แล้ววนกลับไปขั้น 2

ขั้น 4 — พอถึงเกตขวาสุด นิพจน์บนสายเส้นนั้นคือคำตอบ แล้วค่อยย่อด้วยกฎบูลีนจาก EP06 ถ้าโจทย์ขอ

ทางลัด · ตารางแปลงเกตเป็นนิพจน์

ท่องตารางนี้ให้ขึ้นใจแล้วน้องจะไล่วงจรได้เร็วมากครับ

เกต	ขาเข้า P, Q ให้อะไร	วิธีจำ
AND	PQ	เอามาคูณกันเฉย ๆ
OR	$P + Q$	เอามาบวกกันเฉย ๆ
NOT	$\overline{P}$	ครอบขีดบนตัวเดียว
NAND	$\overline{PQ}$	คูณก่อน **แล้วค่อยครอบขีดทั้งก้อน**
NOR	$\overline{P + Q}$	บวกก่อน **แล้วค่อยครอบขีดทั้งก้อน**
XOR	$P \oplus Q$ หรือ $\overline{P}Q + P\overline{Q}$	"ต่างกันได้ 1"
XNOR	$\overline{P \oplus Q}$ หรือ $\overline{P}\,\overline{Q} + PQ$	"เหมือนกันได้ 1"

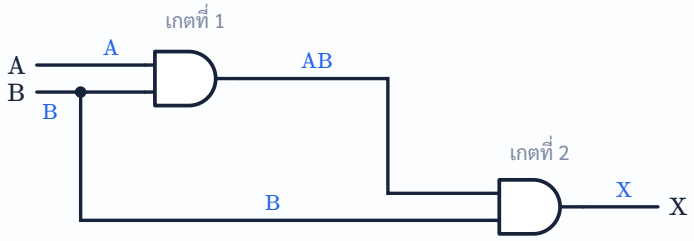
กับดัก · วงเล็บหายตอนผ่านเกตที่มี bubble

สมมติสายที่เข้า NAND เป็น $A + B$ กับ C ครับ — เอาต์พุตคือ $\overline{(A+B)C}$ **ไม่ใช่** $\overline{A+B}\,C$ เพราะขีดบนของ NAND **คลุมผลคูณทั้งก้อน**

วิธีกันพลาด: เขียนวงเล็บครอบนิพจน์ที่รับมาจากสายเสมอ**ก่อน** แล้วค่อยลากขีดบนคลุมทั้งหมดครับ

I DO — พี่ไล่ให้ดูทีละเกต

วงจรจากสไลด์หน้า 33 อันที่หนึ่ง — Write the output expression



ตัวหนังสือสีน้ำเงินคือ “สติกเกอร์” ที่พี่แปะไว้บนสายทุกเส้นครับ · จุดดำคือจุดที่สาย B แยกออกเป็นสองทาง

1

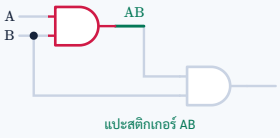


ขั้น 1 — เขียนชื่ออินพุตให้ครบก่อน

วงจรนี้มีสายเข้าจากขอบซ้ายสองเส้นคือ A กับ B **เพราะ** อะไรที่ไม่ได้ออกมาจากเกตตัวไหนเลย คืออินพุตครับ

สังเกตว่าสาย B มี**จุดดำ**อยู่ — จุดดำแปลว่า **สายแยกออกเป็นสองทาง** และทั้งสองทาง**มีค่าเท่ากันคือ** B ครับ

2

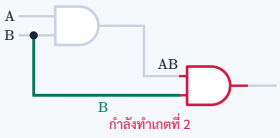


ขั้น 2 — เกตที่ 1 เป็น AND รับ A กับ B

ขาเข้าทั้งสองขารู้ค่าแล้ว จึงลงมือได้เลยครับ **เพราะ** AND คือ “คูณ” สายที่ออกจากเกตนี้จึงมีค่า AB

แปะสติกเกอร์ AB ลงบนสายเส้นนั้นทันที ห้ามจำไว้ในหัวเฉย ๆ ครับ เพราะพอวงจรใหญ่ขึ้นจะจำสลับกันแน่นอน

3

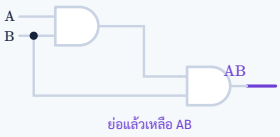


ขั้น 3 — เกตที่ 2 รับ AB (จากบน) กับ B (จากสายที่แยกมา)

ขาบนได้ AB จากขั้น 2 · ขาล่างได้ B ตรง ๆ จากจุดแยก **เพราะ** สายที่แยกจากจุดดำ**ไม่ได้ผ่านเกตอะไรเลย** ค่าจึงยังเป็น B เท่าเดิมครับ

เกตที่ 2 เป็น AND อีกตัว จึงได้ $X = (AB) \cdot B$

4



ขั้น 4 — ย่อด้วยกฎบูลีนจาก EP06

$X = (AB)B = A(BB)$ ด้วย Associative Law II แล้ว $BB = B$ ด้วย Idempotent Law II **เพราะ** ตัวแปรคูณตัวมันเองได้ตัวเดิมครับ

ตอบ: $X = AB$ — เกตที่ 2 ในวงจรนี้**ไม่ได้ทำอะไรเลย ตัดทิ้งได้** นี่คือเหตุผลว่าทำไมวิศวกรต้องย่อนิพจน์ครับ

$$X = (AB) \cdot B = A(BB) = AB$$

Associative II แล้วต่อด้วย Idempotent II

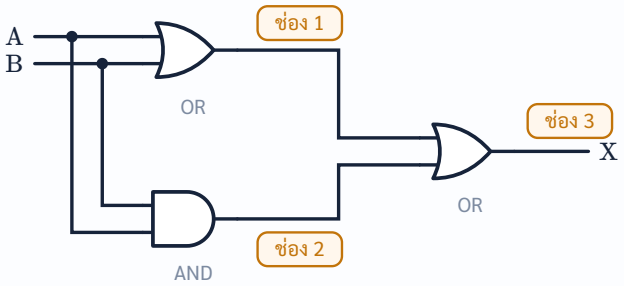
ตรวจด้วยตารางค่าความจริง — คอลัมน์ X กับคอลัมน์ AB ต้องเหมือนกันทุกแถวครับ ถ้าไม่เหมือนแปลว่าย่อผิด

A	B	AB	$X = (AB)B$
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	1

เหมือนกันทั้ง 4 แถว → ย่อถูกครับ (ตรวจด้วย `same_logic("(A&B)&B","A&B","AB")` ใน `tools/dig.py` ได้ True)

WE DO — เติมสติกเกอร์ด้วยกัน

วงจรจากสไลด์หน้า 33 อันที่สอง — พี่เว้นช่องบนสายไว้ให้น้องเติมครับ



เติมนิพจน์ลงช่อง 1, 2, 3 ครับ · จุดดำสองจุดแปลว่าทั้ง A และ B แยกไปเข้า AND ข้างล่างด้วย

เฉลย WE DO — ทีละช่องพร้อมเหตุผล

ช่อง 1 = $A + B$ เพราะ เกตบนเป็น OR และขาเข้าทั้งสองขาคือ A กับ B ตรง ๆ (ไม่ได้ผ่านเกตอะไรก่อน) เกต OR แปลว่า “บวก” จึงได้ $A + B$ ครับ

ช่อง 2 = AB เพราะ เกตล่างเป็น AND และขาเข้าคือสายที่แยกมาจากจุดดำ ซึ่ง**ยังมีค่า A และ B เท่าเดิม** (สายที่แยกออกไม่เปลี่ยนค่าครับ) AND แปลว่า “คูณ” จึงได้ AB

ช่อง 3 = $X = (A + B) + AB$ เพราะ เกตขวาสุดเป็น OR รับช่อง 1 กับช่อง 2 มาบวกกัน — **อย่าลืมวงเล็บครอบ $A + B$** ตอนเขียนบรรทัดแรกครับ จะได้ไม่สับสนว่าอะไรมาจากไหน

ย่อต่อ: $(A + B) + AB = A + B + AB$ ครับ ใช้ Absorption Law I ที่ว่า $B + AB = B$ (มองว่า B คือ “ตัวสั้น” ที่กลืน AB ซึ่งเป็น “ตัวยาวที่มี B อยู่ข้างใน”) เหลือ $A + B$

ตอบ: $X = A + B$ — วงจรนี้ใช้เกต 3 ตัว ทั้งที่ความจริงใช้ OR ตัวเดียวก็พอ

“X = (A + B) + AB = A + B by the absorption law.”

A	B	$A + B$	AB	X
0	0	0	0	0
0	1	1	0	1

A	B	$A+B$	AB	X
1	0	1	0	1
1	1	1	1	1

คอลัมน์ X เหมือนคอลัมน์ $A+B$ ทุกแถว → ย่อถูกครับ

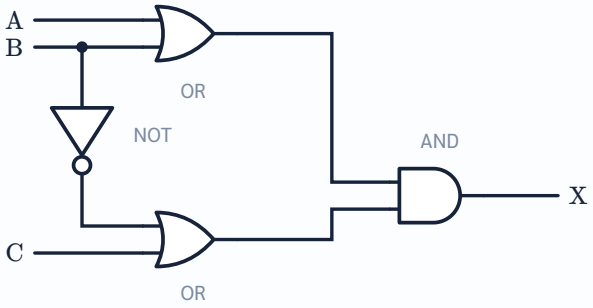
YOU DO — ตาน้องแล้วครับ

สามข้อ ทำให้จบก่อนค่อยเลื่อนดูเฉลยที่อยู่ติดกันข้างล่างนะครับ

Y2.1 สไลด์ Week 4 หน้า 33 อันที่สาม

> Write the output expression for the circuit below.

ระวังอินเวอร์เตอร์ที่แอบอยู่ตรงกลางนะครับ — สาย B แยกไปเข้าอินเวอร์เตอร์ก่อนแล้วค่อยลงไปข้างล่าง



สาย B แยกที่จุดดำ ทางหนึ่งเข้า OR ตัวบน อีกทางลงไปเข้าอินเวอร์เตอร์แล้วค่อยเข้า OR ตัวล่าง

เฉลย Y2.1

ขั้น 1 — แปะสติกเกอร์อินพุตก่อน วงจรนี้มี 3 อินพุตคือ A, B, C **เพราะ** มีสายเข้าจากขอบซ้ายสามเส้นครับ และสาย B มีจุดดำ แปลว่าแยกเป็นสองทาง

ขั้น 2 — อินเวอร์เตอร์ก่อน เพราะขาเข้ารู้ค่าแล้ว อินเวอร์เตอร์รับ B จึงให้ $\overline{B}$ ออกมา **เพราะ** bubble ที่ปลายสามเหลี่ยมคือขีดบนครับ **แปะสติกเกอร์ $\overline{B}$ บนสายที่ลงไปข้างล่าง**

ขั้น 3 — OR ตัวบนรับ A กับ B ได้ $A+B$ เพราะ สาย B ทางบน**ไม่ได้ผ่านอินเวอร์เตอร์** ค่าจึงยังเป็น B เปล่า ๆ — **นี่คือจุดที่คนพลาดมากที่สุดของข้อนี้**ครับ

ขั้น 4 — OR ตัวล่างรับ $\overline{B}$ กับ C ได้ $\overline{B}+C$ เพราะ ขาบนของเกตนี้มาจากอินเวอร์เตอร์ ขาล่างมาจาก C ตรง ๆ

ขั้น 5 — AND ขวาสุดรับผลของ OR ทั้งสองตัว ได้ $X=(A+B)(\overline{B}+C)$ **เพราะ** AND แปลว่าคูณ และ**ต้องใส่วงเล็บครอบทั้งสองก้อน**ไม่งั้นจะกลายเป็น $A+B\overline{B}+C$ ซึ่งผิดคนละเรื่องครับ

ขั้น 6 — ตรวจว่ายังย่อได้อีกไหม กระจายวงเล็บ: $(A+B)(\overline{B}+C)=A\overline{B}+AC+B\overline{B}+BC$ แล้ว $B\overline{B}=0$ (Complementary Law II) เหลือ $A\overline{B}+AC+BC$ ซึ่ง**ใช้เกตมากกว่ารูปวงเล็บเดิม** จึงไม่ต้องย่อครับ

ตอบ: $X=(A+B)(\overline{B}+C)$

“X = (A + B)(B̄ + C).”

A	B	C	$\overline{B}$	$A+B$	$\overline{B}+C$	X
0	0	0	1	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	1	0	1	1	1
1	0	0	1	1	1	1
1	0	1	1	1	1	1
1	1	0	0	1	0	0
1	1	1	0	1	1	1

ตารางเต็มของ Y2.1 · ตรวจแล้วด้วย `tools/dig.py` — คอลัมน์ X ตรงกับ $A\overline{B}+AC+BC$ ทุกแถว

Y2.2 พี่แต่งเอง

> A NAND gate receives A and B. Its output is fed into an OR gate together with C. Write the output expression X and simplify it if possible.

ข้อนี้ฝึกเรื่อง “ขีดบนคลุมทั้งก้อน” ครับ

เฉลย Y2.2

ขั้น 1 — เกตแรกคือ NAND รับ A กับ B ได้ $\overline{AB}$ เพราะ NAND = AND แล้วครอบขีด **ขีดต้องคลุม AB ทั้งก้อน ไม่ใช่คลุมแค่ A** ครับ

ขั้น 2 — เกตที่สองคือ OR รับ $\overline{AB}$ กับ C ได้ $X=\overline{AB}+C$ เพราะ OR แปลว่าบวก และ **ขีดบนของก้อนแรกยังอยู่เหมือนเดิม** ไม่มีอะไรมาลบมันครับ

ขั้น 3 — ย่อด้วย De Morgan (ถ้าโจทย์อยากได้รูปที่แตกขีดแล้ว) $\overline{AB}=\overline{A}+\overline{B}$ จึงได้ $X=\overline{A}+\overline{B}+C$ เพราะ ขีดคลุมผลคูณแตกออกมาเป็นผลบวกของตัวที่มีขีดครับ (callback EP06)

ตอบ: $X=\overline{AB}+C=\overline{A}+\overline{B}+C$

“X = $\bar{A}\bar{B}$ + C = $\bar{A}$ + $\bar{B}$ + C by De Morgan's theorem.”

A	B	C	AB	$\overline{AB}$	X
0	0	0	0	1	1
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	0	1	1
1	0	0	0	1	1
1	0	1	0	1	1
1	1	0	1	0	0
1	1	1	1	0	1

คอลัมน์สุดท้ายคือ X · ตรวจแล้วว่า $\overline{AB}+C$ กับ $\overline{A}+\overline{B}+C$ ให้ค่าเท่ากันทั้ง 8 แถว

Y2.3 พี่แต่งเอง

> An OR gate receives A and B. Its output goes to an AND gate whose other input is $\overline{C}$ (taken from C through an inverter). Write X in SOP form.

ข้อนี้ฝึกการกระจายวงเล็บให้เป็นรูป SOP ครับ (callback EP07)

เฉลย Y2.3

ขั้น 1 — OR รับ A กับ B ได้ $A+B$ เพราะ OR คือบวกครับ

ขั้น 2 — อินเวอร์เตอร์รับ C ได้ $\overline{C}$ เพราะ bubble คือขีดบน

ขั้น 3 — AND รับ $(A+B)$ กับ $\overline{C}$ ได้ $X=(A+B)\overline{C}$ เพราะ AND คือคูณ และต้องมีวงเล็บครอบ $A+B$ ไม่งั้นจะอ่านผิดเป็น $A+B\overline{C}$ ครับ

ขั้น 4 — โจทย์ขอรูป SOP จึงต้องกระจายวงเล็บ ใช้ Distributive Law I : $(A+B)\overline{C}=A\overline{C}+B\overline{C}$ **เพราะ** SOP แปลว่า “ผลบวกของผลคูณ” ต้องไม่มีวงเล็บเหลือครับ

ตอบ: $X=(A+B)\overline{C}=A\overline{C}+B\overline{C}$

“X = (A + B)C̄ = AC̄ + BC̄ in SOP form.”

A	B	C	$A+B$	$\overline{C}$	X
0	0	0	0	1	0
0	0	1	0	0	0
0	1	0	1	1	1
0	1	1	1	0	0
1	0	0	1	1	1
1	0	1	1	0	0
1	1	0	1	1	1
1	1	1	1	0	0

กับดักรวมของหัวข้อนี้ — 3 อย่างที่ทำให้ตกม้าตาย

① จุดดำ (สายแยก) ไม่เปลี่ยนค่า — สายที่แยกจากจุดดำมีค่าเท่ากับสายต้นทางเป๊ะ ๆ ครับ คนชอบเผลอเขียนตัวแปรใหม่ให้มัน

② สายที่ดูเหมือนผ่านอินเวอร์เตอร์ อาจไม่ได้ผ่าน — ใน Y2.1 สาย B ทางบนเข้า OR ตรง ๆ ส่วนทางล่างถึงจะผ่านอินเวอร์เตอร์ **ไล่นิ้วตามเส้นจริง ๆ อย่าดูด้วยตาเฉย ๆ**

③ ลืมวงเล็บตอนสายที่เป็นผลบวกเข้า AND — $(A+B)C \neq A+BC$ ครับ ผิดคนละเรื่องเลย

3 สี่ทรงมาตรฐาน — AND-OR · AND-OR-Invert · XOR · XNOR

ตะขอเกี่ยว · รับจาก EP07 โดยตรง

EP07 จบด้วยการเขียน SOP จากตารางครับ เช่นได้ $X = AB + CD$ ออกมา — **สมมติว่าถ้าน้องต้องเอานิพจน์นั้นไปสร้างเป็นวงจรจริงบนแผงล่ะครับ จะวางเกตยังไง** คำตอบง่ายกว่าที่คิดมากครับ: **พจน์คูณหนึ่งพจน์ = AND หนึ่งตัว · เครื่องหมายบวกทั้งหมด = OR หนึ่งตัว** แค่นั้นเลย และทรงนี้มีชื่อเรียกว่า AND-OR logic

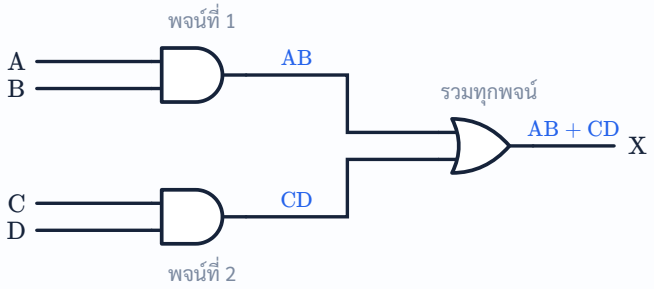
ทรงที่ 1 — AND-OR logic (สไลด์หน้า 37)

นิยามจากสไลด์ · คำต่อคำ

Any SOP expression can be implemented using an AND gate for each product term and one OR gate for summing all of the product terms.

แปล: นิพจน์ SOP อะไรก็ตาม สร้างได้ด้วย **AND หนึ่งตัวต่อหนึ่งพจน์คูณ** แล้วใช้ **OR ตัวเดียว** รวมทุกพจน์ครับ

ที่มา — ทำไมถึงเป็นแบบนั้น: เพราะ SOP แปลตรงตัวว่า Sum of Products คือ “ผลบวกของผลคูณ” ครับ **ผลคูณ = AND · ผลบวก = OR** · มีกี่พจน์ก็ AND กี่ตัว · บวกกันทั้งหมดครั้งเดียวก็ OR ตัวเดียว ไม่มีอะไรต้องจำเลย แค่แปลคำว่า Sum กับ Product ให้ถูกครับ



วงจร AND-OR ของ $X = AB + CD$ — สองพจน์จึงใช้ AND สองตัว บวกกันครั้งเดียวจึงใช้ OR ตัวเดียวครับ

A	B	C	D	AB	CD	X
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	0	0
0	0	1	1	0	1	1
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	0	0
0	1	1	1	0	1	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	0	0	0

อ่านตารางยังไง

แถวเขียวคือแถวที่ $X = 1$ ครับ สังเกตว่า**แถวเขียวคือแถวที่ AB เป็น 1 หรือ CD เป็น 1** (หรือทั้งสองอย่าง) ตรงกับความหมายของ OR เป๊ะ ๆ

สไลด์ยังบอกอีกว่า any truth table or logic can be expressed in the SOP form แปลว่า**ทรง AND-OR สร้างฟังก์ชันลอจิกได้ทุกฟังก์ชันที่ไม่ต้องใช้ความจำ**ครับ นี่คือประโยคที่ออกสอบบ่อย

A	B	C	D	AB	CD	X
1	0	1	1	0	1	1
1	1	0	0	1	0	1
1	1	0	1	1	0	1
1	1	1	0	1	0	1
1	1	1	1	1	1	1

ทรงที่ 2 — AND-OR-Invert logic (AOI) (สไลด์หน้า 39)

นิยามจากสไลด์ · คำต่อคำ

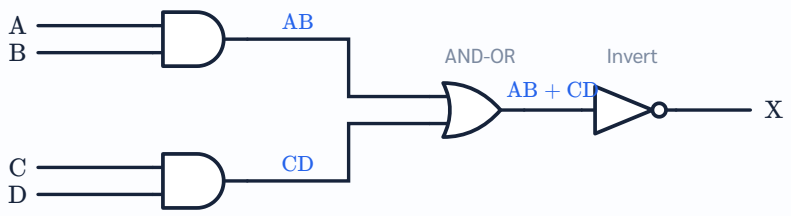
When the output of an AND-OR circuit is complemented (inverted), it results in an AND-OR-Invert (AOI) circuit, which can be used to implement any POS expressions.

แปล: เอาเอาต์พุตของวงจร AND-OR ไป**กลับค่า** ก็ได้วงจร AOI ซึ่งใช้สร้างนิพจน์ทรง **POS** ได้ทุกอันครับ

สไลด์ให้สูตรทำสองขั้นไว้ด้วย:

(1) Construct $\overline{X}$ in the SOP form using AND-OR logic — สร้าง**ตัวกลับค่าของคำตอบ**ในรูป SOP ก่อน

(2) Pass it in to the NOT gate to produce X — แล้วส่งเข้าอินเวอร์เตอร์ตัวเดียวเพื่อได้ X



วงจร AOI — เหมือน AND-OR ทุกอย่าง แค่เติมอินเวอร์เตอร์ตัวเดียวที่ปลายทางครับ

$$X = \overline{AB + CD} = \overline{AB} \cdot \overline{CD} = (\overline{A} + \overline{B})(\overline{C} + \overline{D})$$

De Morgan I แล้วต่อด้วย De Morgan II — callback EP06

1 $\overline{\text{AB} + \text{CD}}$

ขั้น 1 — เริ่มจากนิพจน์ที่ปลายอินเวอร์เตอร์

สายก่อนเข้าอินเวอร์เตอร์คือ $AB + CD$ พออินเวอร์เตอร์ครอบขีด ก็ได้ $\overline{AB + CD}$ **เพราะ** bubble คือขีดบน และมัน**คลุมทุกอย่างที่เข้ามาทางขานั้น**ครับ

ตอนนี้เรามีขีดยาวคลุมผลบวกอยู่ 1 ชั้น

2 $\overline{\text{AB}} \cdot \overline{\text{CD}}$

ขั้น 2 — ขีดคลุมผลบวก แตกเป็นผลคูณ (De Morgan I)

$\overline{AB + CD} = \overline{AB} \cdot \overline{CD}$ **เพราะ** คาถา “ขีดแตก เครื่องหมายพลิก” ครับ — เครื่องหมาย $+$ พลิกเป็น $\cdot$ แล้วขีดยาวหนึ่งเส้นแตกเป็นขีดสั้นสองเส้นคลุมของแต่ละก้อน

ระวัง: ต้องแตกทีละชั้น ห้ามแตกสองชั้นพร้อมกันในบรรทัดเดียว

3	$(\overline{\mathrm{A}}+\overline{\mathrm{B}})$ $(\overline{\mathrm{C}}+\overline{\mathrm{D}})$	**ขั้น 3 — แตกอีกชั้น ได้ทรง POS ออกมา (De Morgan II)** $\overline{AB}=\overline{A}+\overline{B}$ และ $\overline{CD}=\overline{C}+\overline{D}$ **เพราะ** คราวนี้ขีดคลุมผลคูณ จึงพลิกเป็นผลบวกครับ ได้ $X=(\overline{A}+\overline{B})(\overline{C}+\overline{D})$ ซึ่งเป็น **ผลคูณของผลบวก = POS** เป๊ะตามที่สไลด์บอกไว้

A	B	C	D	AB	CD	$AB+CD$	X
0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	1
0	0	1	0	0	0	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	0	1
0	1	1	0	0	0	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	0	0	1
1	0	0	1	0	0	0	1
1	0	1	0	0	0	0	1
1	0	1	1	0	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	0
1	1	1	0	1	0	1	0
1	1	1	1	1	1	1	0

คอลัมน์ X คือค่ากลับของคอลัมน์ $AB+CD$ ทุกแถว · ตรวจแล้วว่า $\overline{AB+CD}$ เท่ากับ $(\overline{A}+\overline{B})(\overline{C}+\overline{D})$ ทั้ง 16 แถวครับ

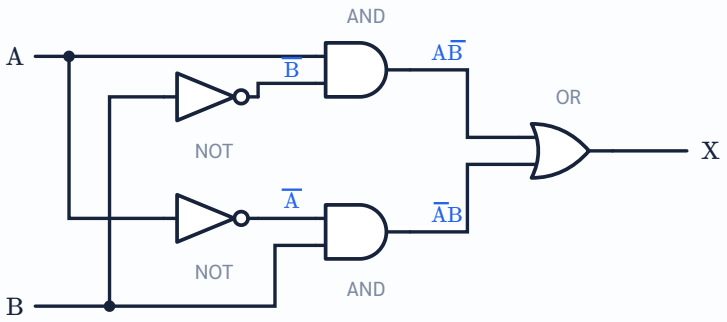
เทียบของเก่า · EP07 บอกว่า POS มาจากแถวที่ได้ 0

EP07 สอนว่า POS อ่านจาก**แถวที่เอาต์พุตเป็น 0** ครับ — ในตารางข้างบนแถวสีแดง (X=0) มีอยู่ 7 แถว จึงได้ maxterm 7 ตัว ซึ่งพอคูณกันแล้วจะเท่ากับ $(\overline{A}+\overline{B})(\overline{C}+\overline{D})$ ที่เราได้จาก De Morgan

เห็นไหมครับว่าสองทางมาบรรจบกัน — ทางหนึ่งคืออ่านจากตาราง (EP07) อีกทางคือลากขีดแล้วใช้ De Morgan (EP06) **ตอนสอบเลือกทางที่เร็วกว่าได้เลย** ถ้าโจทย์ให้วงจรมาให้ใช้ De Morgan · ถ้าโจทย์ให้ตารางมาให้อ่านแถว 0

ทรงที่ 3 — XOR สร้างจากเกตพื้นฐาน (สไลด์หน้า 43)

สไลด์บอกว่า Although XOR gate is considered a type of logic gate with its own unique symbol ... it is actually a combination of two AND gates, one OR gate, and two inverters. แปลว่า **XOR มีสัญลักษณ์เป็นของตัวเองก็จริง แต่ข้างในคือ AND 2 ตัว + OR 1 ตัว + อินเวอร์เตอร์ 2 ตัว**ครับ



XOR ข้างใน — บนคือ $A\overline{B}$ ล่างคือ $\overline{A}B$ แล้ว OR รวมกัน ได้ $X = A\overline{B} + \overline{A}B$

A	B	$\overline{A}B$	$A\overline{B}$	X
0	0	0	0	0
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

วิธีจำ XOR แบบไม่ต้องท่อง

XOR = “ต่างกันได้ 1” ครับ — ดูตารางแล้วจะเห็นว่าเขียวอยู่ที่แถว 01 กับ 10 คือแถวที่ A กับ B **ไม่เหมือนกัน**

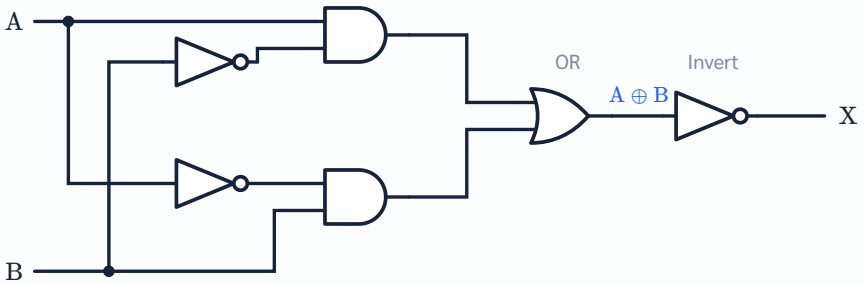
และ**นิพจน์ก็บอกเรื่องเดียวกัน**: $\overline{A}B$ แปลว่า “A เป็น 0 แต่ B เป็น 1” · $A\overline{B}$ แปลว่า “A เป็น 1 แต่ B เป็น 0” — สองกรณีนี้คือ**ทุกวิธีที่จะต่างกันได้**ครับ จึง OR รวมกัน

สัญกรณ์: $A \oplus B$ อ่านว่า **A XOR B**

ทรงที่ 4 — XNOR สร้างได้ 2 วิธี (สไลด์หน้า 44)

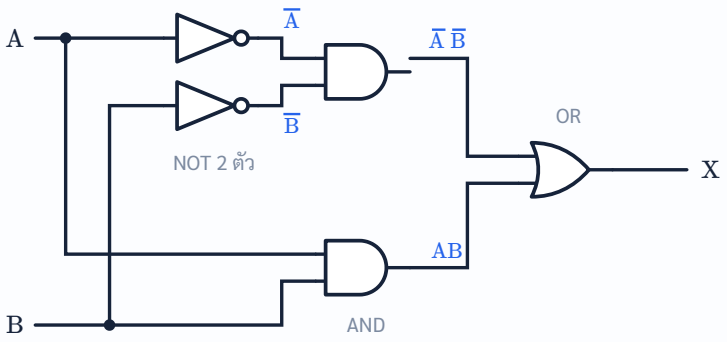
สไลด์เขียนว่า **XNOR logic can be constructed in 2 different ways** และวาดมาให้สองวงจรครับ พี่จะวาดทั้งสองแบบให้ดูพร้อมเหตุผลว่าทำไมได้ผลเหมือนกัน

วิธีที่ 1 — สร้าง XOR ก่อน แล้วกลับค่าที่ปลายทาง



วิธีที่ 1 — $X = \overline{A\overline{B} + \overline{A}B} = \overline{A \oplus B}$ ครับ

วิธีที่ 2 — สร้างตรง ๆ จากแถวที่เอาต์พุตเป็น 1



วิธีที่ 2 — AND ตัวบนรับ $\overline{A}$ กับ $\overline{B}$ · ตัวล่างรับ A กับ B ตรง ๆ ได้ $X = \overline{A}\,\overline{B} + AB$

A	B	$\overline{A}B + A\overline{B}$	$\overline{A}\,\overline{B}$	AB	X
0	0	0	1	0	1
0	1	1	0	0	0
1	0	1	0	0	0
1	1	0	0	1	1

ทำไมสองวิธีให้ผลเท่ากัน — พิสูจน์ด้วยพีชคณิต

วิธีที่ 1 ให้ $X = \overline{A\overline{B} + \overline{A}B}$ ครับ ลองแตกดู:

$\overline{A\overline{B} + \overline{A}B} = \overline{A\overline{B}} \cdot \overline{\overline{A}B}$ (De Morgan I)

$= (\overline{A} + B)(A + \overline{B})$ (De Morgan II สองครั้ง + involution)

$= \overline{A}A + \overline{A}\,\overline{B} + BA + B\overline{B}$ (Distributive I)

$= 0 + \overline{A}\,\overline{B} + AB + 0 = \overline{A}\,\overline{B} + AB$ (Complementary II แล้ว Identity I)

ได้เท่ากับวิธีที่ 2 พอดีครับ — ตรวจด้วย `same_logic()` แล้วได้ True เหมือนกันทั้ง 4 แถว

กับดักของหัวข้อนี้

① คิดว่า AOI คือ NAND ตัวใหญ่ — ไม่ใช่ครับ AOI มีอินเวอร์เตอร์อยู่ที่**เอาต์พุตของ OR** ไม่ใช่ที่เอาต์พุตของ AND ถ้าไปกลับค่าที่ AND แต่ละตัวแทน จะได้ $\overline{AB} + \overline{CD}$ ซึ่ง**คนละฟังก์ชันเลย**

② สลับ $\overline{A}B$ กับ $A\overline{B}$ ใน XOR — ไม่ผิดครับเพราะบวกกันสลับที่ได้ (commutative) แต่**ห้ามเขียนเป็น $\overline{AB}$** เพราะขีดจะกลายเป็นคลุมทั้งก้อน กลายเป็น NAND ไปเลย

③ จำ XNOR สลับกับ XOR — ท่องว่า **XOR = ต่างกันได้ 1 · XNOR = เหมือนกันได้ 1** ครับ ดูตัว N ที่แปลว่า NOT แล้วนึกว่า “กลับค่าจาก XOR”

ศัพท์สอบของหัวข้อนี้

English	แปลว่า	ข้อสอบเขียนมาแบบนี้
product term	พจน์คูณ (เช่น AB)	*“an AND gate for each product term”*

English	แปลว่า	ข้อสอบเขียนมาแบบนี้
summing	การบวกรวม (= OR)	*“one OR gate for summing all of the product terms”*
complemented / inverted	ถูกกลับค่า	*“When the output of an AND-OR circuit is complemented …”*
AND-OR-Invert (AOI)	วงจร AND-OR ที่มีอินเวอร์เตอร์ปลายทาง	*“… it results in an AND-OR-Invert (AOI) circuit”*
Exclusive-OR / Exclusive-NOR	XOR / XNOR	ข้อสอบเขียนเต็มบ้าง ย่อบ้าง ต้องรู้ทั้งสองแบบ

4 Gate conversion ด้วย NOT — เติมอินเวอร์เตอร์แล้วเกตเปลี่ยนชนิด

ตะขอเกี่ยว · ต่อจากหัวข้อ 3

หัวข้อ 3 น้องสร้างวงจรจากนิพจน์ได้แล้วครับ แต่โลกจริงมีปัญหาอีกอย่างหนึ่ง — **สมมติว่าถ้าน้องออกแบบวงจรเสร็จแล้วต้องใช้ AND แต่ในกล่องเหลือแต่ NOR กับอินเวอร์เตอร์ล่ะครับ ต้องไปซื้อ AND ใหม่ไหม** คำตอบคือ**ไม่ต้องครับ** เพราะ NOR กับอินเวอร์เตอร์ประกอบกันเป็น AND ได้ และสไลด์หน้า 45–46 สอนวิธีนี้ไว้ทั้ง 4 คู่

นิยามจากสไลด์หน้า 45

NOT gates can be used to convert some gates into other equivalent logic. Useful when some gates are not available for implementation of logic operation.

แปล: เอาอินเวอร์เตอร์มาช่วย ทำให้เกตชนิดหนึ่ง**ทำงานเหมือน**เกตอีกชนิดได้ครับ มีประโยชน์ตอน**ที่ไม่มีเกตชนิดที่ต้องการอยู่ในมือ**

ของใกล้ตัว: กบเหลาที่ใช้แทนมีดได้

ถ้าน้องต้องเหลาปลายดินสอแต่ไม่มีมีดครับ — เอากบเหลามาก็ได้ผลเดียวกัน **เครื่องมือคนละชนิด แต่ผลลัพธ์บนดินสอเหมือนกัน**

เกตก็เหมือนกันครับ — **ถ้าตารางค่าความจริงออกมาเท่ากันทุกแถว ก็ใช้แทนกันได้ 100%** วงจรไม่มีทางรู้เลยว่าข้างในเป็นเกตชนิดไหน มันเห็นแค่ค่า 0 กับ 1 ที่ออกมาครับ

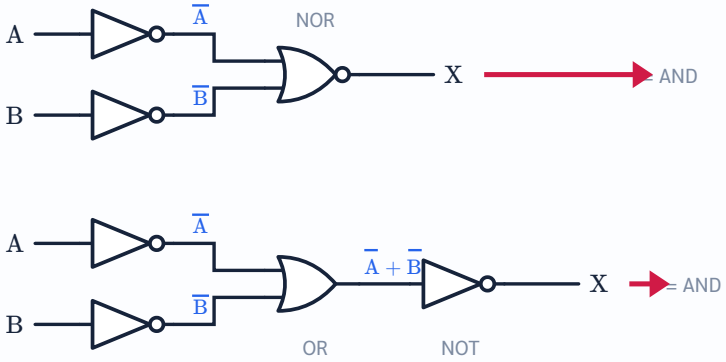
ทางลัด · หลักคิดเดียวที่ใช้ได้ทั้ง 4 คู่

ทุกการแปลงในหัวข้อนี้คือ De Morgan บรรทัดเดียวครับ — ใส่อินเวอร์เตอร์ที่**ขาเข้าทั้งสองขา** แล้วเขียนนิพจน์ออกมา จะเห็นว่ามันคือด้านหนึ่งของสูตร De Morgan พอดี

$\overline{\overline{A}+\overline{B}} = \overline{\overline{A}}\cdot\overline{\overline{B}} = AB$ → **NOR ที่ใส่อินเวอร์เตอร์ขาเข้า = AND**

$\overline{\overline{A}\cdot\overline{B}} = \overline{\overline{A}}+\overline{\overline{B}} = A+B$ → **NAND ที่ใส่อินเวอร์เตอร์ขาเข้า = OR**

คู่ที่ 1 — OR / NOR + อินเวอร์เตอร์ ใช้แทน AND ได้



สไลด์หน้า 45 บนสุด — สองวงจรนี้ให้ผลเท่ากับ AND ทุกแถว · เส้นสีน้ำเงินคือขีดบนที่พี่วาดกำกับไว้ให้เห็น

A	B	$\overline{A}$	$\overline{B}$	$\overline{\overline{A}+\overline{B}}$	AB
0	0	1	1	0	0
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	1	1

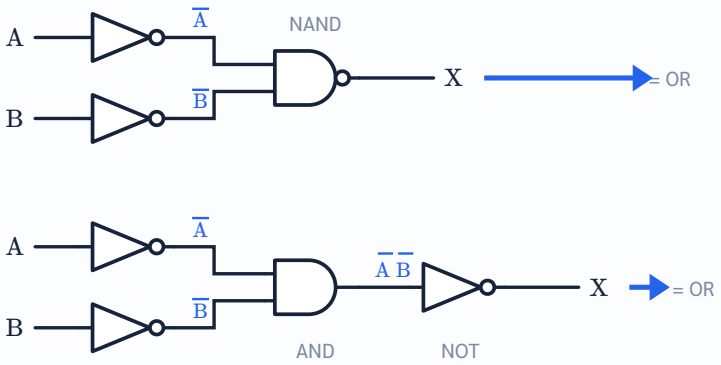
พิสูจน์ด้วยพีชคณิต

วงจรบน (NOR): ขาเข้าเป็น $\overline{A}$ กับ $\overline{B}$ NOR แปลว่า “บวกแล้วครอบขีด” จึงได้

$$X = \overline{\overline{A}+\overline{B}}$$
$$= \overline{\overline{A}} \cdot \overline{\overline{B}} \quad \text{(De Morgan I)}$$
$$= A \cdot B \quad \text{(Involution สองครั้ง)}$$

วงจรล่าง (OR แล้ว NOT): $X = \overline{\overline{A}+\overline{B}}$ ตัวเดียวกันเป๊ะครับ เพราะ “OR แล้วครอบขีดทีหลัง” กับ “NOR” คือสิ่งเดียวกัน

คู่ที่ 2 — AND / NAND + อินเวอร์เตอร์ ใช้แทน OR ได้



สไลด์หน้า 45 ล่างสุด — NAND ที่มีอินเวอร์เตอร์ที่ขาเข้าทั้งสองขา ทำงานเหมือน OR เป๊ะ

A	B	$\overline{A}$	$\overline{B}$	$\overline{\overline{A}\,\overline{B}}$	$A+B$
0	0	1	1	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	1	1

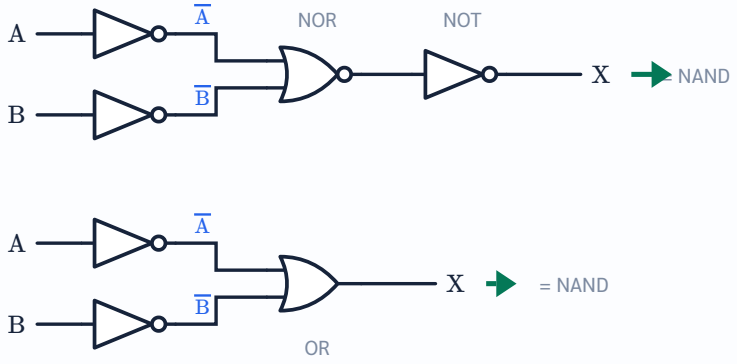
พิสูจน์ด้วยพีชคณิต

วงจรบน (NAND):

$$X = \overline{\overline{A} \cdot \overline{B}}$$
$$= \overline{\overline{A}} + \overline{\overline{B}} \quad \text{(De Morgan II)}$$
$$= A + B \quad \text{(Involution)}$$

สังเกตแพตเทิร์นครับ — ทั้งสองคู่ที่ผ่านมา **ใส่อินเวอร์เตอร์ที่ขาเข้าแล้วเกตจะ “พลิก” เป็นชนิดตรงข้าม** คือ NOR กลายเป็น AND · NAND กลายเป็น OR

คู่ที่ 3 — OR / NOR + อินเวอร์เตอร์ ใช้แทน NAND ได้ (สไลด์หน้า 46)



วงจรล่างสั้นกว่า — **OR ที่มีอินเวอร์เตอร์ที่ขาเข้าคือ NAND นั่นเอง** และนี่คือที่มาของสัญลักษณ์ negative-OR ในหัวข้อ 8 ครับ

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$	$\overline{AB}$
0	0	1	1	1	1
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	0	0

พิสูจน์ด้วยพีชคณิต

วงจรล่าง (OR เปล่า ๆ): ขาเข้าเป็น $\overline{A}$ กับ $\overline{B}$ ผ่าน OR ได้

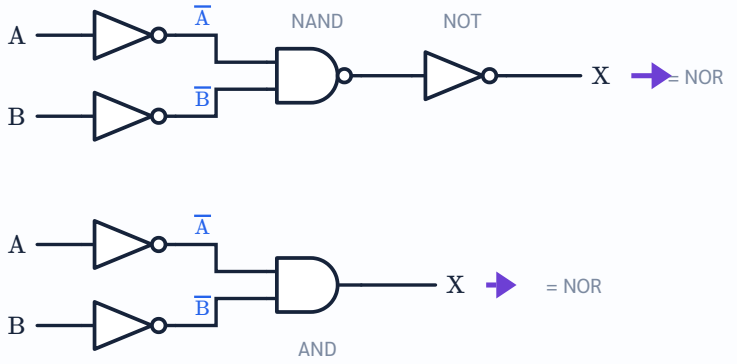
$$X = \overline{A} + \overline{B}$$

$$= \overline{AB}$$ (De Morgan II อ่านย้อนจากขวาไปซ้าย)

วงจรบน (NOR แล้ว NOT): $X = \overline{\overline{\overline{A} + \overline{B}}} = \overline{A} + \overline{B}$ เพราะขีดสองชั้นหักล้างกัน (involution) ครับ — ได้ตัวเดียวกัน

ข้อสังเกตสำคัญมาก: วงจรล่างใช้เกตแค่ 3 ตัว ส่วนวงจรบนใช้ 4 ตัว **ทั้งที่ทำงานเหมือนกัน** — เลือกอันที่ประหยัดกว่าเสมอครับ

คู่ที่ 4 — AND / NAND + อินเวอร์เตอร์ ใช้แทน NOR ได้



วงจรล่าง — **AND ที่มีอินเวอร์เตอร์ที่ขาเข้าคือ NOR นั่นเอง** คือที่มาของสัญลักษณ์ negative-AND ครับ

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\,\overline{B}$	$\overline{A+B}$
0	0	1	1	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	0	0

พิสูจน์ด้วยพีชคณิต

วงจรล่าง (AND เปล่า ๆ):

$$X = \overline{A} \cdot \overline{B}$$

$$= \overline{A+B}$$ (De Morgan I อ่านย้อนจากขวาไปซ้าย)

วงจรบน (NAND แล้ว NOT): $X = \overline{\overline{\overline{A} \cdot \overline{B}}} = \overline{A} \cdot \overline{B}$ ครับ ได้ตัวเดียวกัน

สรุปตารางแปลงเกต 4 คู่ — จำแค่ตารางนี้ก็พอ

มีเกตอะไรอยู่ในมือ	ทำอะไรเพิ่ม	ได้เกตอะไร	สูตรที่ใช้
NOR	ใส่อินเวอร์เตอร์ที่ขาเข้าทั้งสองขา	**AND**	$\overline{\overline{A}+\overline{B}} = AB$
NAND	ใส่อินเวอร์เตอร์ที่ขาเข้าทั้งสองขา	**OR**	$\overline{\overline{A}\,\overline{B}} = A+B$
OR	ใส่อินเวอร์เตอร์ที่ขาเข้าทั้งสองขา	**NAND**	$\overline{A}+\overline{B} = \overline{AB}$
AND	ใส่อินเวอร์เตอร์ที่ขาเข้าทั้งสองขา	**NOR**	$\overline{A}\,\overline{B} = \overline{A+B}$

กฎเดียวที่ต้องจำ: ใส่อินเวอร์เตอร์ที่**ขาเข้าทุกขา** แล้วเกตจะกลายเป็น**เกตที่ "สลับ AND↔OR และสลับมี/ไม่มี bubble ที่ขาออก"** — ทั้งหมดนี้คือ De Morgan บรรทัดเดียวครับ

YOU DO — ตาน้องแล้วครับ

เฉลยอยู่ติดกันทันที ทำให้จบก่อนนะครับ

Y4.1 พี่แต่งเอง

> You have only NOR gates and inverters in the drawer. Show how to build a 3-input AND gate ($X = ABC$). Draw the circuit and write the expression at every wire.

ใบ้: ทำทีละคู่ก่อน แล้วค่อยเอาผลไป AND กับตัวที่สาม

เฉลย Y4.1

ขั้น 1 — ตัดปัญหาให้เล็กลงก่อน $ABC = (AB) \cdot C$ **เพราะ** Associative Law II บอกว่าจับคู่คูณก่อนหลังยังไงก็ได้ครับ ตอนนี้เราต้องการ AND สองขาสองตัว

ขั้น 2 — สร้าง AND ตัวแรกจาก NOR ใส่อินเวอร์เตอร์ที่ A กับ B แล้วป้อนเข้า NOR: ได้ $\overline{\overline{A}+\overline{B}} = AB$ **เพราะ** คู่ที่ 1 ในตารางข้างบนครับ **สายที่ออกมามีค่า** AB

ขั้น 3 — สร้าง AND ตัวที่สองจาก NOR อีกตัว คราวนี้ขาเข้าคือ AB กับ C ต้องใส่อินเวอร์เตอร์ทั้งสองขาอีกครั้ง ได้ $X = \overline{\overline{AB}+\overline{C}} = (AB) \cdot C = ABC$ **เพราะ** ใช้กฎเดิมซ้ำครับ

ขั้น 4 — นับของที่ใช้ NOR 2 ตัว + อินเวอร์เตอร์ 4 ตัว = 6 ตัว **เพราะ** ขาเข้าของ NOR แต่ละตัวต้องมีอินเวอร์เตอร์ครบทั้งสองขา

ตอบ: $X = \overline{\overline{\overline{A}+\overline{B}}+\overline{C}} = ABC$ (ใช้ NOR 2 ตัว + inverter 4 ตัว)

"Invert both inputs of a NOR gate to obtain an AND function, then repeat for the third input."

หมายเหตุ: ในหัวข้อ 6 น้องจะได้เห็นว่า **อินเวอร์เตอร์พวกนี้ก็สร้างจาก NOR ได้อีก** แปลว่าวงจรนี้ทำจาก NOR ล้วน ๆ ได้เลยครับ

Y4.2 พี่แต่งเอง

> An AND gate has an inverter on each of its two inputs and another inverter on its output. What single gate does the whole thing behave like?

ใบ้: ทำทีละชั้น อย่าคิดรวบ

เฉลย Y4.2

ขั้น 1 — เขียนสิ่งที่เข้า AND ก่อน ขาเข้าผ่านอินเวอร์เตอร์แล้ว จึงเป็น $\overline{A}$ กับ $\overline{B}$ **เพราะ** อินเวอร์เตอร์ครอบขีดให้ตัวแปรที่วิ่งผ่านครับ

ขั้น 2 — ผ่าน AND ได้ $\overline{A}\,\overline{B}$ **เพราะ** AND คือคูณ

ขั้น 3 — ผ่านอินเวอร์เตอร์ตัวที่ขาออก ได้ $\overline{\overline{A}\,\overline{B}}$ **เพราะ** ครอบขีดคลุมทั้งก้อน

ขั้น 4 — ใช้ De Morgan II แตกขีด $\overline{\overline{A}\,\overline{B}} = \overline{\overline{A}} + \overline{\overline{B}} = A + B$ **เพราะ** ขีดคลุมผลคูณ แตกเป็นผลบวก แล้วขีดสองชั้นหักล้างกัน

ตอบ: ทั้งก้อนทำงานเหมือน OR gate ธรรมดาตัวเดียว (ใช้เกต 4 ตัวแทนที่จะใช้ตัวเดียว — เปลืองมากครับ)

"The circuit is equivalent to a single 2-input OR gate."

กับดักของหัวข้อนี้

① **ใส่อินเวอร์เตอร์แค่ขาเดียว** — กฎในตารางใช้ได้ก็ต่อเมื่อ **ใส่ครบทุกขาเข้า**ครับ ถ้าใส่ขาเดียวจะได้ฟังก์ชันใหม่ที่ไม่ใช่เกตมาตรฐานเลย เช่น NOR ที่มีอินเวอร์เตอร์ขาเดียวให้ $\overline{\overline{A} + B} = A\overline{B}$

② **สับสนระหว่าง "bubble ที่ขาเข้า" กับ "bubble ที่ขาออก"** — bubble ที่ขาออกครอบทั้งก้อน · bubble ที่ขาเข้าครอบแค่ตัวแปรที่วิ่งเข้าขานั้น **ตำแหน่งต่างกันคนละความหมายเลยครับ**

③ **ลืมนับอินเวอร์เตอร์ตอนตอบว่า "ใช้กี่เกต"** — ข้อสอบชอบถามจำนวนเกตทั้งหมด อินเวอร์เตอร์ก็นับเป็นเกตนะครับ

5 NAND เป็น universal gate — เกตชนิดเดียวสร้างได้ทั้งหมด

ตะขอเกี่ยว · รับจากหัวข้อ 4

หัวข้อ 4 น้องเห็นแล้วว่า**เกตหนึ่งชนิด + อินเวอร์เตอร์ ทำงานแทนอีกชนิดได้**ครับ แต่ยังต้องพึ่งอินเวอร์เตอร์อยู่ดี

สมมติว่าถ้าลิ้นชักน้องเหลือแต่ NAND อย่างเดียว ไม่มีแม้แต่อินเวอร์เตอร์ล่ะครับ ยังทำงานได้ไหม — คำตอบคือ**ได้ครับ ทุกอย่าง** เพราะ **NAND สร้างอินเวอร์เตอร์ได้ด้วยตัวมันเอง** แล้วพอมีอินเวอร์เตอร์ ก็ย้อนกลับไปใช้ทุกอย่างในหัวข้อ 4 ได้หมดเลย

นิยามจากสไลด์หน้า 50 · ประโยคนี้ออกสอบแน่นอน

The NAND gate is a universal gate because it can be used in place of any other gates.

แปล: NAND เป็น **universal gate** เพราะ**เอาไปใช้แทนเกตอื่นได้ทุกตัว**ครับ

ที่มา — ทำไม NAND ถึงทำได้: เพราะ NAND มีของสองอย่างครบในตัวเดียว คือ ① **ตัวคูณ (AND)** และ ② **ตัวกลับค่า (bubble)** ครับ — ในพีชคณิตบูลีนถ้ามีทั้ง “คูณ” และ “กลับค่า” ก็สร้าง “บวก” ได้ด้วย De Morgan ($A + B = \overline{\overline{A}\,\overline{B}}$) แปลว่ามีครบทั้ง $+$, $\cdot$ และขีดบน ซึ่งเป็น**ทุกเครื่องหมายที่พีชคณิตบูลีนมี** — จบครับ ไม่ต้องมีอะไรอีกแล้ว

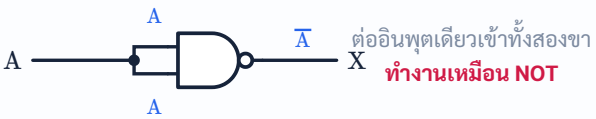
ทำไมโรงงานถึงชอบ — เหตุผลที่ต้องตอบให้ได้ในข้อสอบ

ผลิตชิปชนิดเดียวพอ
ถ้าออกแบบวงจรทั้งวงด้วย NAND ล้วน โรงงานก็สั่งชิป NAND แบบเดียว **ไม่ต้องสั่ง 7 แบบ** — สายการผลิตเดียว คลังเดียว ราคาถูกกว่าครับ

ชิปหนึ่งตัวมีหลายเกต
ชิป NAND เบอร์มาตรฐานหนึ่งตัวมี NAND 4 เกตอยู่ข้างใน ถ้าวงจรใช้ NAND ล้วน **ก็ใช้ทุกเกตในชิปให้คุ้มได้** ต่างจากวงจรผสมที่มักเหลือเกตว่าง ๆ ในชิปทิ้งไว้

ผลิตง่ายกว่าในระดับทรานซิสเตอร์
ในเทคโนโลยี CMOS **NAND กับ NOR ใช้ทรานซิสเตอร์น้อยกว่า AND กับ OR**ครับ (เพราะ AND ที่จริงคือ NAND ต่อด้วยอินเวอร์เตอร์อยู่แล้ว)

① NAND 1 ตัว = อินเวอร์เตอร์ (สไลด์หน้า 50 บน)



เอาสาย A เส้นเดียวแยกเข้าทั้งสองขาของ NAND ครับ (จุดดำคือจุดแยก)

A	$X = \overline{A \cdot A}$	$\overline{A}$
0	1	1
1	0	0

พิสูจน์ด้วยพีชคณิต

$X = \overline{A \cdot A}$ (นิยามของ NAND)

$= \overline{A}$ (Idempotent Law II: $AA = A$)

ตารางยืนยัน: คอลัมน์ X กับคอลัมน์ $\overline{A}$ เหมือนกันทั้ง 2 แถวครับ

วิธีที่สอง: ต่อขาที่สองไว้ที่ HIGH ถาวรก็ได้ครับ เพราะ $\overline{A \cdot 1} = \overline{A}$ (Identity Law II) — สไลด์วาดแบบต่อทั้งสองขาเข้าด้วยกัน พี่แนะนำให้ใช้แบบสไลด์ตอนสอบ

② NAND 2 ตัว = AND (สไลด์หน้า 50 ล่าง)



NAND ตัวแรกให้ $\overline{AB}$ · ตัวที่สองต่อแบบอินเวอร์เตอร์เพื่อลบขีดออก เหลือ AB ครับ

A	B	$\overline{AB}$	$X = \overline{\overline{AB} \cdot \overline{AB}}$	AB
0	0	1	0	0
0	1	1	0	0
1	0	1	0	0
1	1	0	1	1

พิสูจน์ด้วยพีชคณิต

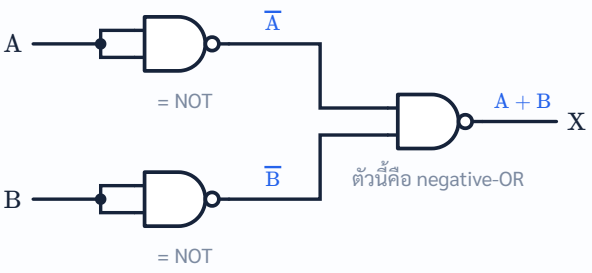
$X = \overline{\overline{AB} \cdot \overline{AB}}$ (NAND ตัวที่ 2 รับ $\overline{AB}$ เข้าทั้งสองขา)

$= \overline{\overline{AB}}$ (Idempotent II)

$= AB$ (Involution — ขีดสองชั้นหักล้างกัน)

วิธีคิดแบบสั้น: NAND = AND + ขีด ถ้าอยากได้ AND เปล่า ๆ ก็แค่**ลบขีดออกด้วยอินเวอร์เตอร์อีกตัว** และอินเวอร์เตอร์ตัวนั้นก็คือ NAND ที่ต่อขาเข้าด้วยกัน (จากข้อ ①) ครับ

③ NAND 3 ตัว = OR (สไลด์หน้า 51 บน)



สองตัวแรกทำหน้าที่เป็นอินเวอร์เตอร์ · ตัวที่สามรับ $\overline{A}$ กับ $\overline{B}$ แล้ว De Morgan ทำงานให้เราครับ

A	B	$\overline{A}$	$\overline{B}$	$X = \overline{\overline{A} \cdot \overline{B}}$	$A + B$
0	0	1	1	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	1	0	0	1	1

พิสูจน์ด้วยพีชคณิต — บรรทัดสำคัญที่สุดของ EP นี้

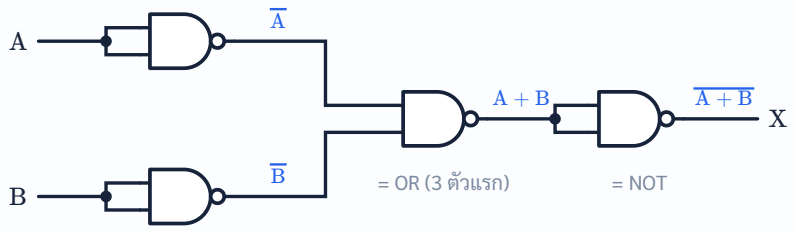
$X = \overline{\overline{A} \cdot \overline{B}}$ (NAND ตัวที่ 3)

$= \overline{\overline{A}} + \overline{\overline{B}}$ **De Morgan II** — callback EP06

$= A + B$ (Involution สองครั้ง)

บรรทัดที่ 2 คือหัวใจครับ — De Morgan คือเหตุผลทั้งหมดว่าทำไม NAND ถึงเป็น universal gate ถ้าไม่มี De Morgan เราจะสร้าง OR จาก AND ไม่ได้เลย

④ NAND 4 ตัว = NOR (สไลด์หน้า 51 ล่าง)



เอาวงจร OR จากข้อ ③ มาต่ออินเวอร์เตอร์ (NAND ตัวที่ 4) ที่ปลายทาง ได้ NOR ครับ

A	B	$A + B$	X	$\overline{A + B}$
0	0	0	1	1
0	1	1	0	0
1	0	1	0	0
1	1	1	0	0

พิสูจน์ด้วยพีชคณิต

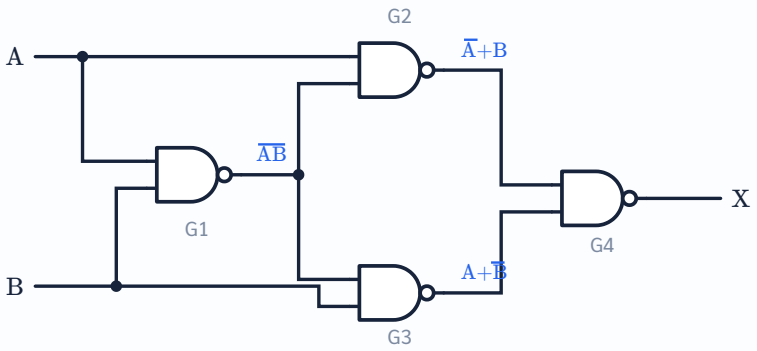
จากข้อ ③ สายก่อนเข้า NAND ตัวที่ 4 มีค่า $A + B$ อยู่แล้วครับ

$X = \overline{(A + B) \cdot (A + B)}$ (NAND ตัวที่ 4 ต่อขาเข้าด้วยกัน)

$= \overline{A + B}$ (Idempotent II)

ทางลัดคิด: “ได้ OR แล้วเติม NOT = NOR” ครับ และเราสร้าง NOT จาก NAND ได้อยู่แล้วจากข้อ ① — **ทุกอย่างต่อยอดกันเป็นทอด ๆ**

⑤ NAND 4 ตัว = XOR (สไลด์หน้า 52 บน)



โครงคลาสสิก 4 NAND — G_1 อยู่กลาง ป้อนกลับเข้าทั้ง G_2 และ G_3 · จุดดำ 3 จุดคือจุดแยกสาย

A	B	$G_1 = \overline{AB}$	$G_2 = \overline{A\,G_1}$	$G_3 = \overline{B\,G_1}$	$X = \overline{G_2 G_3}$	$A \oplus B$
0	0	1	1	1	0	0
0	1	1	1	0	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0

พิสูจน์ด้วยพีชคณิต — ทีละเกต

$G_1 = \overline{AB}$ (NAND ธรรมดา)

$G_2 = \overline{A \cdot G_1} = \overline{A \cdot \overline{AB}} = \overline{A} + \overline{\overline{AB}} = \overline{A} + AB$ (De Morgan II แล้ว involution)

$\overline{A} + AB = (\overline{A} + A)(\overline{A} + B) = 1 \cdot (\overline{A} + B) = \overline{A} + B$ (Distributive II, Complementary I, Identity II)

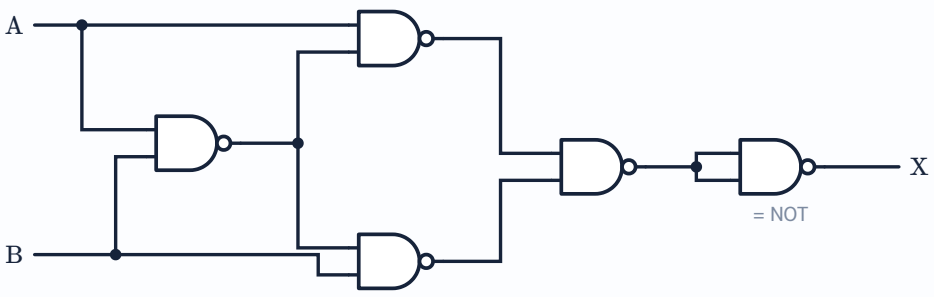
$G_3 = \overline{B \cdot G_1} = \overline{B} + A$ (ทำเหมือน G_2 ทุกขั้น แค่สลับ A กับ B)

$X = \overline{G_2 \cdot G_3} = \overline{(\overline{A} + B)(A + \overline{B})}$

$= \overline{\overline{A}A + \overline{A}\,\overline{B} + BA + B\overline{B}} = \overline{\overline{A}\,\overline{B} + AB}$ (กระจายแล้วตัดพจน์ที่เป็น 0)

$= \overline{\overline{A \oplus B}} = A \oplus B$ (เพราะ $\overline{A}\,\overline{B} + AB$ คือ XNOR จากหัวข้อ 3) **ได้ XOR ครับ**

⑥ NAND 5 ตัว = XNOR (สไลด์หน้า 52 ล่าง)



วงจร XOR 4 ตัวจากข้อ ⑤

เอาวงจร XOR มาต่ออินเวอร์เตอร์ (NAND ตัวที่ 5) — ได้ XNOR ครับ

A	B	XOR	X	$\overline{A \oplus B}$
0	0	0	1	1
0	1	1	0	0
1	0	1	0	0
1	1	0	1	1

พิสูจน์ด้วยพีชคณิต

$X = \overline{(A \oplus B) \cdot (A \oplus B)} = \overline{A \oplus B}$ (Idempotent II)

จำนวนเกตที่ต้องใช้สรุปได้เป็นตารางข้างล่างครับ ข้อสอบชอบถามว่า “ต้องใช้ NAND กี่ตัวถึงจะได้ ...” ต้องตอบได้ทันที

สรุป — ต้องใช้ NAND กี่ตัวถึงจะได้เกตแต่ละชนิด

อยากได้เกต	ใช้ NAND กี่ตัว	ต่อยังไง	นิพจน์
NOT	**1**	ต่ออินพุตเดียวเข้าทั้งสองขา	$\overline{A \cdot A} = \overline{A}$
AND	**2**	NAND แล้วต่ออินเวอร์เตอร์	$\overline{\overline{AB}} = AB$
OR	**3**	อินเวอร์เตอร์ 2 ตัวที่ขาเข้า แล้ว NAND	$\overline{\overline{A}\,\overline{B}} = A + B$
NOR	**4**	วงจร OR แล้วต่ออินเวอร์เตอร์	$\overline{A + B}$
XOR	**4**	โครงป้อนกลับ G_1 เข้า G_2 และ G_3	$A \oplus B$
XNOR	**5**	วงจร XOR แล้วต่ออินเวอร์เตอร์	$\overline{A \oplus B}$

YOU DO — ตาน้องแล้วครับ

เฉลยอยู่ติดกันทันที

Y5.1 พี่แต่งเอง

> Using NAND gates only, build a circuit for $X = A + \overline{B}$. How many NAND gates are needed?

ใบ้: $\overline{B}$ ก็คือ NOT ที่สร้างจาก NAND ได้ครับ แล้วค่อยเอาไป OR

เฉลย Y5.1

ขั้น 1 — แยกงานออกเป็นชิ้น นิพจน์นี้ต้องการสองอย่างคือ ① $\overline{B}$ และ ② การ OR **เพราะ** $A + \overline{B}$ อ่านว่า “A หรือ $\overline{B}$” ครับ

ขั้น 2 — สร้าง OR ก่อน (ใช้ 3 NAND) สูตร OR คือ $P + Q = \overline{\overline{P}\,\overline{Q}}$ **เพราะ** ต้องกลับค่าขาเข้าทั้งสองก่อนแล้วค่อย NAND (ข้อ ③) ในที่นี้ $P = A, Q = \overline{B}$

ขั้น 3 — แทน $Q = \overline{B}$ ลงไป $X = \overline{\overline{A} \cdot \overline{\overline{B}}} = \overline{\overline{A} \cdot B}$ **เพราะ** ขีดสองชั้นบน B หักล้างกัน — **ข้อดีคือเราไม่ต้องสร้าง $\overline{B}$ เลย ใช้ B ตรง ๆ ได้**

ขั้น 4 — นับเกต ต้องมี ① NAND ที่ทำเป็นอินเวอร์เตอร์ของ A (1 ตัว) ② NAND ตัวสุดท้ายที่รับ $\overline{A}$ กับ B (1 ตัว) — **รวม 2 ตัวเท่านั้น** **เพราะ** ขาที่ต้องกลับค่าคือขาของ A ขาเดียว ส่วน B เข้าตรง ๆ ได้เลย

ตอบ: ใช้ NAND 2 ตัว · $X = \overline{\overline{A} \cdot B} = A + \overline{B}$

“One NAND wired as an inverter for A, then a second NAND with inputs $\bar{A}$ and B.”

ตรวจแล้วด้วย `same_logic("~( ~(A&A) & ~(~B & ~B) )","A|~B","AB")` **และเวอร์ชันย่อ 2 เกต ได้ True ทั้งคู่ครับ**

Y5.2 พี่แต่งเอง

> Explain in one sentence why a NAND gate can replace every other gate, referring to De Morgan's theorem.

ข้อนี้เป็นข้อเขียนอธิบาย — ข้อสอบชอบถามแบบนี้ครับ

เฉลย Y5.2

ขั้น 1 — ระบุว่าพีชคณิตบูลีนต้องการอะไรบ้าง ต้องการแค่ 3 อย่างคือ **ขีดบน (NOT) · การคูณ (AND) · การบวก (OR)** **เพราะ** ทุกนิพจน์บูลีนเขียนได้ด้วยเครื่องหมายสามตัวนี้เท่านั้นครับ (EP06)

ขั้น 2 — แสดงว่า NAND ให้ครบทั้งสาม NOT ได้จาก $\overline{A \cdot A}$ · **AND** ได้จากการเติม NOT ที่ขาออกของ NAND · **OR** ได้จาก De Morgan $\overline{\overline{A}\,\overline{B}} = A + B$

ขั้น 3 — สรุป เมื่อสร้างได้ครบทั้งสามเครื่องหมาย ก็สร้างนิพจน์อะไรก็ได้ จึงสร้างเกตอะไรก็ได้ **เพราะ** เกตทุกตัวก็คือนิพจน์บูลีนตัวหนึ่งครับ

ตอบ (ประโยคที่ลอกลงกระดาษได้เลย):

“A NAND gate is universal because De Morgan's theorem lets it produce NOT, AND and OR — the only three operations Boolean algebra needs — so any Boolean expression, and therefore any gate, can be built from NAND gates alone.”

กับดักของหัวข้อนี้

① **คิดว่า “NAND 2 ตัวต่อกัน = NAND ที่แรงขึ้น”** — ไม่ใช่ครับ NAND ต่อกันสองตัว**กลายเป็น AND** เพราะขีดถูกลบไปชั้นหนึ่ง

② **สับสนว่า OR ใช้ 3 หรือ 2 ตัว** — ต้อง **3 ตัว**ครับ เพราะต้องกลับค่าขาเข้า**ทั้งสองขา**ก่อน (2 ตัว) แล้วค่อย NAND รวม (อีก 1 ตัว)

③ **ลืมว่าอินพุตที่ “มีขีดอยู่แล้ว” ไม่ต้องกลับค่าซ้ำ** — เหมือนใน Y5.1 ที่ $\overline{B}$ ต้องกลับค่า แต่พอเจอ $\overline{\overline{B}}$ ก็เหลือ B **ประหยัดเกตไปได้ตั้งตัวหนึ่ง** ข้อสอบชอบซ่อนตรงนี้ครับ

6 NOR ก็เป็น universal gate เหมือนกัน

ตะขอเกี่ยว · รับจากหัวข้อ 5

หัวข้อ 5 น้องเห็นแล้วว่า NAND สร้างได้ทุกเกต เพราะมันมี “คูณ + ขีด” ครบในตัว — **สมมติว่าถ้าลิ้นชักน้องเหลือแต่ NOR แทนล่ะครับ ยังรอดไหม** รอดครับ และเหตุผลก็คู่ขนานกันเป๊ะ: **NOR มี “บวก + ขีด” ครบในตัว** แล้ว De Morgan อีกข้อหนึ่งก็เปลี่ยน “บวก” ให้เป็น “คูณ” ให้เรา

นิยามจากสไลด์หน้า 53

NOR gate is also considered as a universal gate. Like the NAND gate, the NOR gate can be used in place of any other gates.

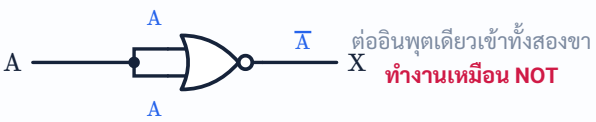
ที่มา: NOR ให้ $\overline{A+B}$ ครับ — มี**ผลบวก**กับ**ขีดบน** เมื่อมีขีดบน ก็ทำ NOT ได้ · เมื่อมี “บวก + ขีด” ก็ทำ OR ได้ · และ **De Morgan I** ($\overline{A}\,\overline{B} = \overline{A+B}$) เปลี่ยนบวกเป็นคูณให้ จึงได้ AND ครบสามเครื่องหมาย เหมือนเดิมเป๊ะครับ

เทียบของเก่า · NAND กับ NOR เป็นภาพสะท้อนกัน

ทุกอย่างในหัวข้อ 5 มีคู่แฝดในหัวข้อ 6 ครับ แค่**สลับ AND กับ OR** ทุกที่ที่เจอ

ฝั่ง NAND (หัวข้อ 5)	ฝั่ง NOR (หัวข้อ 6)	ต่างกันตรงไหน
1 NAND = NOT	1 NOR = NOT	เหมือนกัน
2 NAND = **AND**	2 NOR = **OR**	สลับ
3 NAND = **OR**	3 NOR = **AND**	สลับ
4 NAND = **NOR**	4 NOR = **NAND**	สลับ
4 NAND = **XOR** · 5 NAND = XNOR	5 NOR = **XNOR** · 6 NOR = XOR	**ฝั่ง NOR ได้ XNOR ก่อน** — จุดที่คนจำสลับบ่อยที่สุด

① NOR 1 ตัว = อินเวอร์เตอร์ (สไลด์หน้า 53 บน)



ต่ออินพุตเดียวเข้าทั้งสองขา
ทำงานเหมือน NOT

เหมือนฝั่ง NAND เป๊ะครับ ต่างแค่รูปทรงเกต

A	$X = \overline{A + A}$	$\overline{A}$
0	1	1
1	0	0

พิสูจน์ด้วยพีชคณิต

$X = \overline{A + A}$ (นิยามของ NOR)

$= \overline{A}$ (Idempotent Law I: $A + A = A$)

วิธีที่สอง: ต่อขาที่สองไว้ที่ LOW ถาวรก็ได้ เพราะ $\overline{A + 0} = \overline{A}$ (Identity Law I) ครับ

สังเกต: ฝั่ง NAND ใช้ $AA = A$ ส่วนฝั่ง NOR ใช้ $A + A = A$ — คนละกฎแต่ชื่อเดียวกันคือ idempotent ครับ

② NOR 2 ตัว = OR (สไลด์หน้า 53 ล่าง)



NOR ตัวแรกให้ $\overline{A + B}$ · ตัวที่สองลบขีดออก เหลือ $A + B$ ครับ

A	B	$\overline{A + B}$	$X = \overline{\overline{A + B} + \overline{A + B}}$	$A + B$
0	0	1	0	0
0	1	0	1	1
1	0	0	1	1
1	1	0	1	1

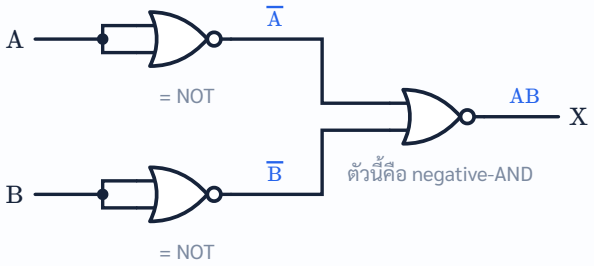
พิสูจน์ด้วยพีชคณิต

$X = \overline{\overline{A + B} + \overline{A + B}}$

$= \overline{\overline{A + B}}$ (Idempotent I)

$= A + B$ (Involution)

③ NOR 3 ตัว = AND (สไลด์หน้า 54 บน)



สองตัวแรกเป็นอินเวอร์เตอร์ · ตัวที่สามรับ $\overline{A}$ กับ $\overline{B}$ แล้ว De Morgan I ทำงานให้ครับ

A	B	$\overline{A}$	$\overline{B}$	$X = \overline{\overline{A} + \overline{B}}$	AB
0	0	1	1	0	0
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	1	1

พิสูจน์ด้วยพีชคณิต — บรรทัดสำคัญของฝั่ง NOR

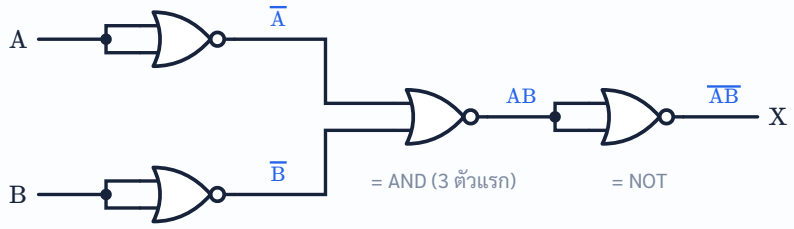
$X = \overline{\overline{A} + \overline{B}}$ (NOR ตัวที่ 3)

$= \overline{\overline{A}} \cdot \overline{\overline{B}}$ **De Morgan I** — callback EP06

$= A \cdot B$ (Involution สองครั้ง)

เทียบกับฝั่ง NAND: ตรงนั้นใช้ De Morgan **II** ($\overline{AB} = \overline{A} + \overline{B}$) แต่ตรงนี้ใช้ De Morgan **I** ($\overline{A + B} = \overline{A}\,\overline{B}$) ครับ — **คนละข้อของสูตรเดียวกัน**

④ NOR 4 ตัว = NAND (สไลด์หน้า 54 ล่าง)



เอาวงจร AND จากข้อ ③ มาต่ออินเวอร์เตอร์ (NOR ตัวที่ 4) ได้ NAND ครับ

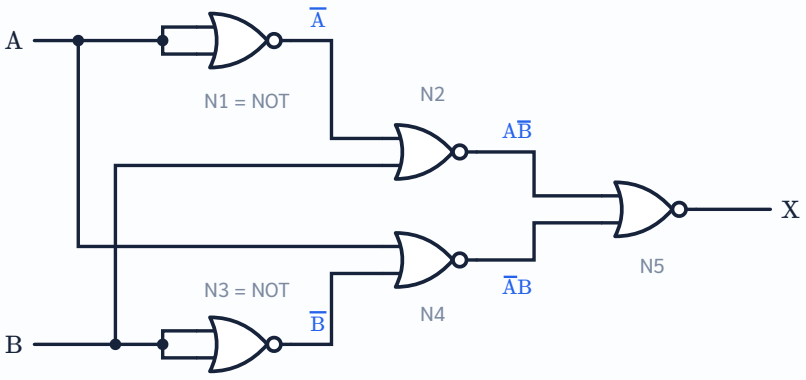
A	B	AB	X	$\overline{AB}$
0	0	0	1	1
0	1	0	1	1
1	0	0	1	1
1	1	1	0	0

พิสูจน์ด้วยพีชคณิต

$X = \overline{AB + AB} = \overline{AB}$ (Idempotent I แล้วเหลือขีดเดียว)

สรุปแพตเทิร์นของทั้งสองหัวข้อ: “ได้เกตที่ต้องการแล้ว อยากได้เวอร์ชันมี bubble ก็ต่ออินเวอร์เตอร์เพิ่มอีกตัว” ครับ ค่าใช้จ่ายคือเกตเพิ่ม 1 ตัวทุกครั้ง

⑤ NOR 5 ตัว = XNOR (สไลด์หน้า 55 ล่าง)



N_2 รับ $\overline{A}$ กับ B ได้ $A\overline{B}$ · N_4 รับ A กับ $\overline{B}$ ได้ $\overline{A}B$ · N_5 รวมแล้วกลับค่า

A	B	$N_1 = \overline{A}$	$N_3 = \overline{B}$	$N_2 = \overline{\overline{A} + B}$	$N_4 = \overline{A + \overline{B}}$	$X = \overline{N_2 + N_4}$	$\overline{A \oplus B}$
0	0	1	1	0	0	1	1
0	1	1	0	0	1	0	0
1	0	0	1	1	0	0	0
1	1	0	0	0	0	1	1

พิสูจน์ด้วยพีชคณิต — ทีละเกต

$N_1 = \overline{A + A} = \overline{A}$ (อินเวอร์เตอร์)

$N_3 = \overline{B + B} = \overline{B}$ (อินเวอร์เตอร์)

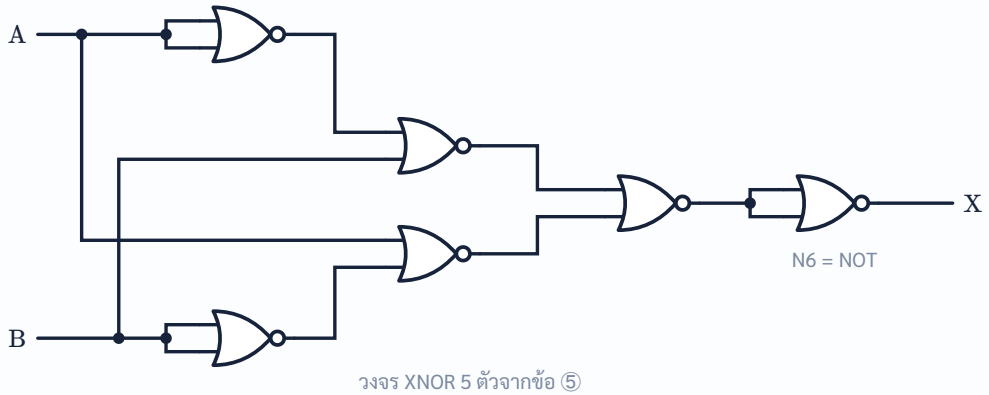
$N_2 = \overline{\overline{A} + B} = \overline{\overline{A}} \cdot \overline{B} = A\overline{B}$ (De Morgan I แล้ว involution)

$N_4 = \overline{A + \overline{B}} = \overline{A} \cdot \overline{\overline{B}} = \overline{A}B$ (De Morgan I แล้ว involution)

$X = \overline{N_2 + N_4} = \overline{A\overline{B} + \overline{A}B} = \overline{A \oplus B}$ (ข้างในวงเล็บคือ XOR จากหัวข้อ 3 พอดี)

ได้ XNOR ครับ — จุดสังเกตคือ**ฝั่ง NOR ได้ XNOR ก่อนแล้วค่อยเติมอินเวอร์เตอร์ถึงจะได้ XOR** ตรงข้ามกับฝั่ง NAND ที่ได้ XOR ก่อน

⑥ NOR 6 ตัว = XOR (สไลด์หน้า 55 บน)



วงจร XNOR 5 ตัวจากข้อ ⑤

เติม NOR ตัวที่ 6 เป็นอินเวอร์เตอร์ที่ปลายทาง — ได้ XOR ครับ

A	B	$XNOR$	X	$A \oplus B$
0	0	1	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	0	0

พิสูจน์ด้วยพีชคณิต

$X = \overline{\overline{A \oplus B} + \overline{A \oplus B}} = \overline{\overline{A \oplus B}} = A \oplus B$

ข้อสังเกตเรื่องต้นทุน: XOR ทำจาก NAND ใช้ **4 ตัว** แต่ทำจาก NOR ใช้ถึง **6 ตัว**ครับ — **เกตชนิดเดียวกันไม่ได้ประหยัดเท่ากันทุกงาน** เวลาออกแบบจริงจึงต้องดูว่าวงจรของเราเป็นทรง SOP (เหมาะกับ NAND) หรือทรง POS (เหมาะกับ NOR)

สรุป — ต้องใช้ NOR กี่ตัวถึงจะได้เกตแต่ละชนิด

อยากได้เกต	ใช้ NOR กี่ตัว	ต่อยังไง	นิพจน์
NOT	**1**	ต่ออินพุตเดียวเข้าทั้งสองขา	$\overline{A+A}=\overline{A}$
OR	**2**	NOR แล้วต่ออินเวอร์เตอร์	$\overline{\overline{A+B}}=A+B$
AND	**3**	อินเวอร์เตอร์ 2 ตัวที่ขาเข้า แล้ว NOR	$\overline{\overline{A}+\overline{B}}=AB$
NAND	**4**	วงจร AND แล้วต่ออินเวอร์เตอร์	$\overline{AB}$
XNOR	**5**	N_1, N_3 เป็น NOT · N_2, N_4 ไขว้กัน · N_5 รวม	$\overline{A\oplus B}$
XOR	**6**	วงจร XNOR แล้วต่ออินเวอร์เตอร์	$A\oplus B$

YOU DO — ตาน้องแล้วครับ

เฉลยติดกันทันที

Y6.1 พี่แต่งเอง

> Using NOR gates only, build $X = AB$ where A is already available in complemented form ($\overline{A}$ is given). How many NOR gates do you save?

ใบ้: ปกติ AND ใช้ NOR 3 ตัว แต่ถ้ามี $\overline{A}$ ให้มาแล้วล่ะครับ

เฉลย Y6.1

ขั้น 1 — เขียนสูตร AND จาก NOR ก่อน $AB=\overline{\overline{A}+\overline{B}}$ **เพราะ** ข้อ ③ บอกว่าต้องกลับค่าขาเข้าทั้งสองก่อนแล้วค่อย NOR ครับ

ขั้น 2 — ดูว่าต้องสร้างอะไรบ้าง เราต้องการ $\overline{A}$ กับ $\overline{B}$ **เพราะ** นั่นคือขาเข้าของ NOR ตัวสุดท้าย — แต่โจทย์**ให้** $\overline{A}$ **มาแล้ว**

ขั้น 3 — นับเกตที่ยังต้องสร้าง เหลือแค่ ① NOR ที่ทำเป็นอินเวอร์เตอร์ของ B (1 ตัว) ② NOR ตัวสุดท้าย (1 ตัว) **รวม 2 ตัว**

ขั้น 4 — เทียบกับกรณีปกติ ปกติต้องใช้ 3 ตัว ตอนนี้ใช้ 2 ตัว **ประหยัดไป 1 ตัว**

ตอบ: ใช้ NOR 2 ตัว ประหยัดไป 1 ตัว · $X=\overline{\overline{A}+\overline{B+B}}=AB$

"Since $\bar{A}$ is already available, only one NOR is needed as an inverter for B, plus one NOR to combine them — two NOR gates in total."

Y6.2 พี่แต่งเอง

> A designer must build an XOR function. Which is cheaper in gate count — NAND-only or NOR-only? State the numbers and explain why.

ข้อนี้ตอบเป็นข้อความ ต้องมีตัวเลขและเหตุผลครับ

เฉลย Y6.2

ขั้น 1 — เรียกตัวเลขจากตารางสรุปทั้งสองหัวข้อ XOR จาก NAND ใช้ **4 ตัว** · XOR จาก NOR ใช้ **6 ตัว** **เพราะ** ฝั่ง NOR ต้องผ่าน XNOR ก่อน (5 ตัว) แล้วเติมอินเวอร์เตอร์อีก 1 ตัวครับ

ขั้น 2 — อธิบายเหตุผลเชิงโครงสร้าง XOR เขียนได้ตรง ๆ ในรูป **SOP** คือ $A\overline{B} + \overline{A}B$ **เพราะ** มันคือผลบวกของสองพจน์คูณ และ**ทรง SOP เข้ากับ NAND ได้พอดี** (จะเห็นชัดในหัวข้อ 7) ส่วน NOR ถนัดทรง POS มากกว่า จึงต้องอ้อมกว่า

ตอบ: NAND ถูกกว่า — ใช้ 4 ตัวเทียบกับ NOR ที่ใช้ 6 ตัว

"NAND-only is cheaper: 4 gates versus 6, because XOR is naturally an SOP expression and SOP maps directly onto NAND-NAND logic."

กับดักของหัวข้อนี้

① **จำสลับว่า NOR 5 ตัวได้ XOR — ไม่ใช่ครับ ได้ XNOR** ท่องว่า "**NOR ได้ตัวมี N ก่อน**" (NOR → XNOR) จะจำไม่ผิด

② **คิดว่า NAND กับ NOR ราคาเท่ากันเสมอ** — จำนวนเกตต่างกันตามฟังก์ชันครับ ต้องดูตารางสรุปทั้งสองอันก่อนตอบ

③ **ใช้ De Morgan ผิดข้อ** — ฝั่ง NAND ใช้ข้อ II ($\overline{AB} = \overline{A} + \overline{B}$) · ฝั่ง NOR ใช้ข้อ I ($\overline{A + B} = \overline{A}\,\overline{B}$) **ดูว่าขีดคลุมอะไรอยู่ก่อนเสมอ** ครับ

ศัพท์สอบของสองหัวข้อนี้

English	แปลว่า	ข้อสอบเขียนมาแบบนี้
universal gate	เกตสารพัดประโยชน์	*"NOR gate is also considered as a universal gate."*
used in place of	ใช้แทน	*"... can be used in place of any other gates."*
gate count	จำนวนเกตที่ใช้	*"Which implementation has the lower gate count?"*
NAND-only / NOR-only	ใช้ NAND ล้วน / NOR ล้วน	*"Implement the function using NAND gates only."*
equivalent logic	ลอจิกที่สมมูลกัน	ตารางค่าความจริงเหมือนกันทุกแถว = ใช้แทนกันได้

7 แปลงวงจรทั้งวง — SOP เป็น NAND ล้วน · POS เป็น NOR ล้วน

ตะขอเกี่ยว · รับจากหัวข้อ 5–6

หัวข้อ 5–6 น้องสร้าง**เกตทีละตัว**จาก NAND หรือ NOR ได้แล้วครับ แต่ถ้าทำแบบนั้นทั้งวงจรจะเปลืองมาก — AND-OR ของ $AB + CD$ ที่ใช้ 3 เกต ถ้าแทนทีละตัวจะกลายเป็น NAND ตั้ง $2 + 2 + 3 = 7$ ตัว

สมมติว่าถ้ามีทางลัดที่แปลง AND-OR ทั้งวงเป็น NAND ล้วนโดยใช้เกตเท่าเดิมล่ะครับ — มีครับ และมันสวยมาก ชื่อเทคนิคว่า **“ใส่ bubble คู่”** (double bubble) วงจร $AB + CD$ จะกลายเป็น NAND แค่ **3 ตัว** เท่าเดิมเป๊ะ

ของใกล้ตัว: คูณด้วย 1 แล้วหารด้วย 1

ตอนเรียนเลขน้องเคยเจอกลไหมครับ — อยากทำเศษส่วนให้ตัวส่วนเหมือนกัน ก็**คูณเศษและส่วนด้วยเลขเดียวกัน** ค่าไม่เปลี่ยนเพราะที่จริงเราคูณด้วย 1

bubble คู่ก็คือกลอันเดียวกันครับ — เอาขีดบนมาซ้อนกันสองชั้นบนสายเส้นเดียวกัน ค่าบนสายไม่เปลี่ยนเลย เพราะ $\overline{\overline{P}} = P$ (involution จาก EP06) แต่**พอมี bubble อยู่ เกตข้าง ๆ ก็ “ดูดกลืน” มันเข้าไปเป็นส่วนหนึ่งของตัวเองได้** แล้วเกตก็เปลี่ยนชนิดทันที

กติกาของเทคนิค bubble คู่ · 3 บรรทัด

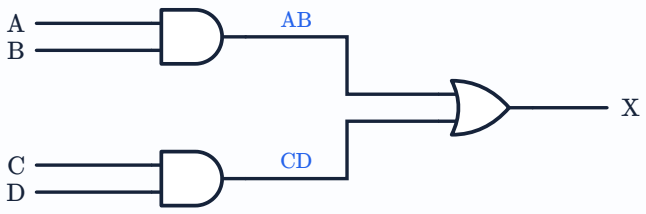
บรรทัดที่ 1 — ใส่ได้เพราะ involution วางวงกลมสองวงติดกันบนสายเส้นเดียว ค่าบนสายเท่าเดิมเป๊ะ ($\overline{\overline{P}} = P$)

บรรทัดที่ 2 — bubble ตัวซ้ายถูกดูดเข้าเกตต้นทาง ทำให้ **AND กลายเป็น NAND** (หรือ **OR กลายเป็น NOR**)

บรรทัดที่ 3 — bubble ตัวขวาถูกดูดเข้าขาเข้าของเกตปลายทาง ทำให้ **OR กลายเป็น negative-OR ซึ่งก็คือ NAND ในอีกหน้าตาหนึ่ง** (De Morgan II) — พอครบสามบรรทัด ทั้งวงจรก็เป็น NAND ล้วนแล้วครับ

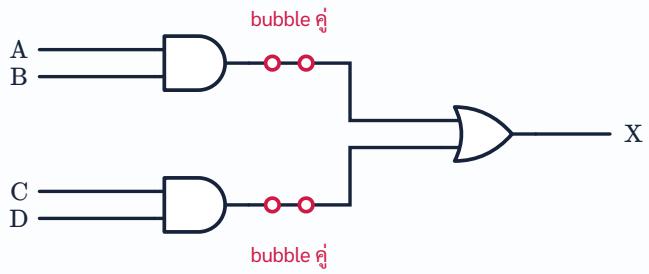
ตัวอย่างที่ 1 — $X = AB + CD$ แปลงเป็น NAND ล้วน

ขั้น 1 — วงจร AND-OR ตั้งต้น (3 เกต)



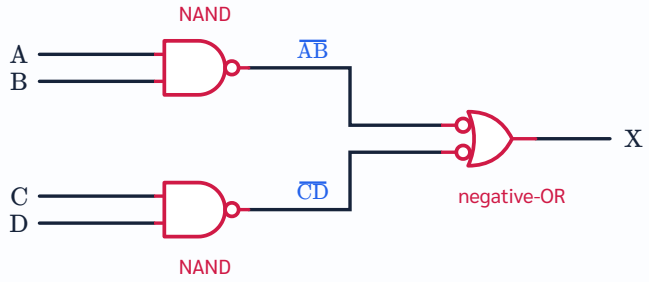
จุดตั้งต้น — AND 2 ตัว + OR 1 ตัว รวม 3 เกต · $X = AB + CD$

ขั้น 2 — ใส่ bubble คู่บนสายที่เชื่อม AND กับ OR (ค่ายังไม่เปลี่ยน)



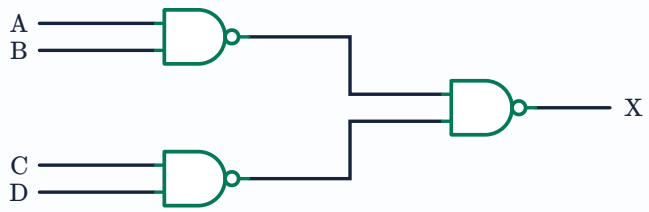
วงกลมสองวงบนสายเดียวกัน = ขีดบนสองชั้น = ไม่เปลี่ยนค่าเลย ($\overline{\overline{AB}} = AB$)

ขั้น 3 — เกตดูดกลืน bubble เข้าไป (AND → NAND · OR → negative-OR)



bubble ตัวซ้ายเข้าไปอยู่ที่ขาออกของ AND (กลายเป็น NAND) · ตัวขวาเข้าไปอยู่ที่ขาเข้าของ OR (กลายเป็น negative-OR)

ขั้น 4 — วาด negative-OR ใหม่ด้วยสัญลักษณ์ NAND (เป็นเกตตัวเดียวกัน)



NAND 3 ตัว · เกตเท่าเดิมกับ AND-OR เป๊ะ

ผลลัพธ์ — **วงจร NAND ล้วน 3 ตัว** ทำงานเหมือน $AB + CD$ ทุกแถวครับ

A	B	C	D	$AB + CD$	$\overline{AB}$	$\overline{CD}$	$X = \overline{\overline{AB} \cdot \overline{CD}}$
0	0	0	0	0	1	1	0
0	0	0	1	0	1	1	0
0	0	1	0	0	1	1	0
0	0	1	1	1	1	0	1
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	0
0	1	1	1	1	1	0	1

A	B	C	D	$AB + CD$	$\overline{AB}$	$\overline{CD}$	$X = \overline{\overline{AB} \cdot \overline{CD}}$
1	0	0	0	0	1	1	0
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	1	1
1	1	0	1	1	0	1	1
1	1	1	0	1	0	1	1
1	1	1	1	1	0	0	1

คอลัมน์ $AB + CD$ กับคอลัมน์ X ที่ได้จากวงจร NAND เหมือนกันทั้ง 16 แถว · ตรวจแล้วด้วย `same_logic("(A&B)|(C&D)", "~( ~(A&B) & ~(C&D) )", "ABCD")` = True

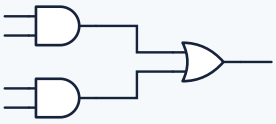
พิสูจน์ด้วยพีชคณิต — ทำไมถึงได้ผลเท่าเดิม

$X = \overline{\overline{AB} \cdot \overline{CD}}$ (นิพจน์ของวงจร NAND 3 ตัว)

$= \overline{\overline{AB}} + \overline{\overline{CD}}$ **De Morgan II**

$= AB + CD$ (Involution สองครั้ง) **ได้ SOP ตัวเดิมกลับมาเป๊ะครับ**

1

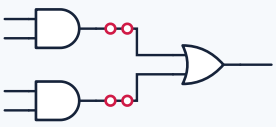


ขั้น 1 — เริ่มจากวงจร AND-OR ที่ได้จาก SOP

เขียน SOP ให้เป็นวงจร AND-OR ตามหัวข้อ 3 ก่อนครับ **เพราะ** เทคนิค bubble คู่ใช้ได้เฉพาะกับวงจรที่มี**โครงสองชั้น: ชั้นแรก AND ชั้นสอง OR** เท่านั้น

ถ้านิพจน์ยังไม่เป็น SOP ต้องกระจายวงเล็บให้เป็น SOP ก่อน (ใช้ Distributive Law I จาก EP06)

2

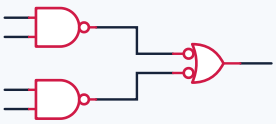


ขั้น 2 — ใส่ bubble สองวงบนสายเชื่อมทุกเส้น

วางวงกลมสองวงติดกันบนสายที่ออกจาก AND แต่ละตัว **เพราะ** $\overline{\overline{P}} = P$ ครับ — **ค่าบนสายไม่ขยับแม้แต่แถวเดียว**

ต้องใส่เป็นคู่เสมอ ถ้าใส่วงเดียววงจรจะเปลี่ยนฟังก์ชันทันที

3

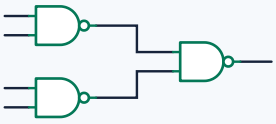


ขั้น 3 — เกตดูดกลืน bubble

bubble ที่ติดกับขาออกของ AND **รวมเข้ากับ AND กลายเป็น NAND** · bubble ที่ติดกับขาเข้าของ OR **รวมเข้ากับ OR กลายเป็น negative-OR เพราะ** สัญลักษณ์เกต**รวม bubble ที่ติดขาไว้ในตัวได้**ครับ

ตอนนี้ไม่มี bubble ลอย ๆ บนสายแล้ว

4



ขั้น 4 — วาด negative-OR ใหม่เป็น NAND

negative-OR กับ NAND **คือเกตตัวเดียวกันคนละหน้าตา** (De Morgan II) **เพราะ** $\overline{A} + \overline{B} = \overline{AB}$ ครับ

ผลสุดท้าย: NAND 3 ตัว ใช้เกตเท่ากับ AND-OR เดิมเป๊ะ — นี่คือเหตุผลที่เทคนิคนี้ดีกว่าการแทนเกตทีละตัว (ซึ่งจะใช้ถึง 7 ตัว)

ทางลัดที่ใช้ตอนสอบได้เลย — ไม่ต้องวาด bubble ก็ได้

“SOP กี่พจน์ = NAND ชั้นแรกกี่ตัว แล้วบวก NAND รวมอีก 1 ตัว” ครับ

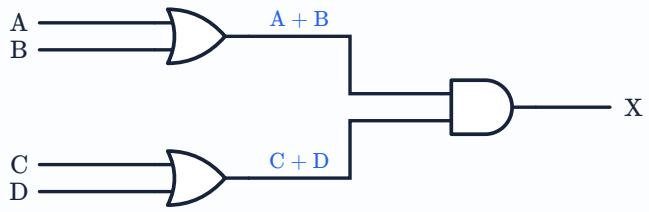
เช่น $X = AB + CD + EF$ (3 พจน์) → NAND ชั้นแรก 3 ตัว + NAND รวม 1 ตัว (3 ขาเข้า) = **4 ตัว**

แต่ตอนพิสูจน์ในข้อสอบ ต้องเขียนขั้น bubble ให้ครบ เพราะเขาให้คะแนนที่กระบวนการครับ — ทางลัดนี้ใช้เช็คคำตอบตอนท้าย

ตัวอย่างที่ 2 — $X = (A + B)(C + D)$ แปลงเป็น NOR ล้วน

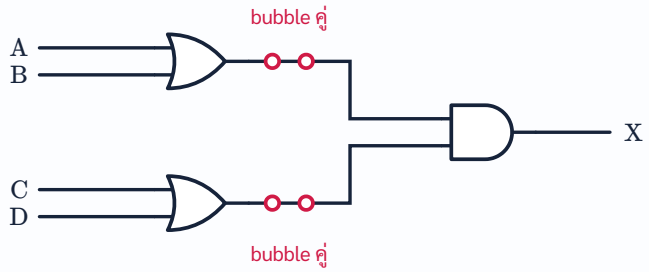
ฝั่ง POS ทำเหมือนกันทุกขั้นครับ แค่**สลับบทบาทของ AND กับ OR**

ขั้น 1 — วงจร OR-AND ตั้งต้น

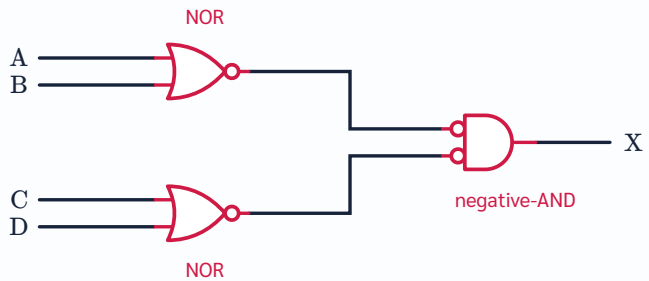


POS ทำให้ชั้นแรกเป็น OR (หนึ่งตัวต่อหนึ่งวงเล็บ) และชั้นสองเป็น AND ตัวเดียวครับ

ขั้น 2 — ใส่ bubble คู่ แล้วให้เกตดูดกลืน

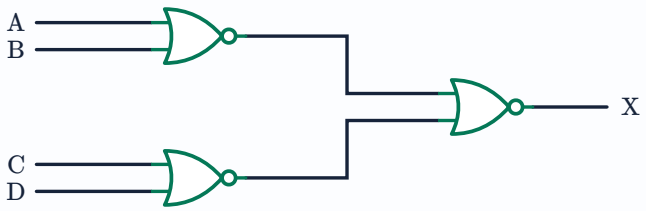


ใส่ bubble คู่บนสายเชื่อม — ค่ายังไม่เปลี่ยนเพราะ involution



OR ดูด bubble ที่ขาออก → NOR · AND ดูด bubble ที่ขาเข้า → negative-AND

ขั้น 3 — วาด negative-AND ใหม่เป็น NOR



NOR 3 ตัว · เกตเท่าเดิมกับ OR-AND เป๊ะ

ผลลัพธ์ — **วงจร NOR ล้วน 3 ตัว** ทำงานเหมือน $(A+B)(C+D)$ ทุกแถวครับ

A	B	C	D	$(A+B)(C+D)$	$\overline{A+B}$	$\overline{C+D}$	$X=\overline{\overline{A+B}+\overline{C+D}}$
0	0	0	0	0	1	1	0
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	1	0
0	1	0	1	1	0	0	1
0	1	1	0	1	0	0	1
0	1	1	1	1	0	0	1
1	0	0	0	0	0	1	0
1	0	0	1	1	0	0	1
1	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1
1	1	0	0	0	0	1	0
1	1	0	1	1	0	0	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	1

พิสูจน์ด้วยพีชคณิต

$X=\overline{\overline{A+B}+\overline{C+D}}$ (นิพจน์ของวงจร NOR 3 ตัว)

$=\overline{\overline{A+B}}\cdot\overline{\overline{C+D}}$ **De Morgan I**

$=(A+B)(C+D)$ (Involution สองครั้ง) **ได้ POS ตัวเดิมครับ**

ตารางเทียบสองฝั่ง — จำเป็นตารางเดียวจบ

เริ่มจาก	โครงวงจร	ใส่ bubble คู่แล้วได้	วาดใหม่เป็น
SOP เช่น $AB+CD$	AND-OR	NAND + negative-OR	**NAND ล้วน** (NAND-NAND)

เริ่มจาก	โครงวงจร	ใส่ bubble คู่แล้วได้	วาดใหม่เป็น
POS เช่น $(A+B)(C+D)$	OR-AND	NOR + negative-AND	**NOR ล้วน** (NOR-NOR)

ท่องประโยคเดียว: “SOP คู่กับ NAND · POS คู่กับ NOR” และเหตุผลคือ De Morgan II คู่กับ NAND ส่วน De Morgan I คู่กับ NOR ครับ

YOU DO — ตาน้องแล้วครับ

เฉลยติดกันทันที ทำให้จบก่อนนะครับ

Y7.1 พี่แต่งเอง · ปูทาง Assignment ข้อ 4

> Convert $X = AB + AC + BC$ into an all-NAND implementation. How many NAND gates are needed, and how many inputs does each one have?

ใบ้: 3 พจน์ → นับตามทางลัดในกล่องข้างบนครับ

เฉลย Y7.1

ขั้น 1 — เช็คว่าเป็น SOP หรือยัง $AB + AC + BC$ เป็นผลบวกของพจน์คูณ 3 พจน์ — **เป็น SOP อยู่แล้ว เพราะ** ไม่มีวงเล็บเหลือและไม่มีขีดคลุมยาวครับ ลงมือได้เลย

ขั้น 2 — วาดวงจร AND-OR ก่อน AND 2 ขาเข้า 3 ตัว (สำหรับ AB, AC, BC) แล้ว OR 3 ขาเข้า 1 ตัว **เพราะ** หนึ่งพจน์คูณ = AND หนึ่งตัว · บวกกันทีเดียว = OR ตัวเดียว

ขั้น 3 — ใส่ bubble คู่บนสายเชื่อมทั้ง 3 เส้น เพราะ ต้องใส่ทุกเส้นที่วิ่งจากชั้น AND ไปชั้น OR ครับ ไม่งั้นจะเหลือเกตชนิดอื่นปน

ขั้น 4 — ดูดกลืน AND ทั้ง 3 ตัวกลายเป็น NAND 2 ขาเข้า · OR 3 ขาเข้ากลายเป็น negative-OR 3 ขาเข้า ซึ่งวาดใหม่เป็น **NAND 3 ขาเข้า**

ขั้น 5 — ตรวจด้วยพีชคณิต $\overline{\overline{AB}\cdot\overline{AC}\cdot\overline{BC}} = \overline{\overline{AB}} + \overline{\overline{AC}} + \overline{\overline{BC}} = AB + AC + BC$ ✓ (De Morgan II กับสามพจน์ แล้ว involution)

ตอบ: ใช้ NAND ทั้งหมด 4 ตัว — NAND 2 ขาเข้า 3 ตัว และ NAND 3 ขาเข้า 1 ตัว · $X = \overline{\overline{AB}\cdot\overline{AC}\cdot\overline{BC}}$

“Three 2-input NAND gates feeding one 3-input NAND gate.”

ตรวจแล้วด้วย `same_logic("(A&B)|(A&C)|(B&C)", "~( ~(A&B) & ~(A&C) & ~(B&C) )", "ABC")` **= True ครับ**

Y7.2 พี่แต่งเอง

> A student inserts only ONE bubble (not a pair) on the wire between an AND gate and the OR gate. What function does the circuit compute now, and why is it wrong?

ข้อนี้ให้เห็นว่าทำไมต้องใส่เป็นคู่ครับ

เฉลย Y7.2

ขั้น 1 — เขียนค่าบนสายหลังใส่ bubble วงเดียว เดิมสายมีค่า AB พอใส่ bubble วงเดียวก็กลายเป็น $\overline{AB}$ **เพราะ** bubble = ขีดบนหนึ่งชั้นครับ

ขั้น 2 — เขียนเอาต์พุตใหม่ OR รับ $\overline{AB}$ กับ CD ได้ $X' = \overline{AB} + CD$ **เพราะ** อีกขาไม่ได้ถูกแตะเลย

ขั้น 3 — เทียบกับของเดิม ของเดิมคือ $AB + CD$ ครับ ลองแทน $A = B = 1, C = D = 0$:
ของเดิมได้ $1 + 0 = \mathbf{1}$ · ของใหม่ได้ $\overline{1} + 0 = 0 + 0 = \mathbf{0}$ **ต่างกันแล้ว**

ขั้น 4 — สรุปเหตุผล bubble วงเดียวคือขีด**ชั้นเดียว** ซึ่ง**ไม่หักล้าง** ต่างจาก bubble คู่ที่เป็นขีดสองชั้นแล้ว involution ลบทิ้งให้ครับ

ตอบ: ได้ $X' = \overline{AB} + CD$ ซึ่งผิดจากเดิม เพราะขีดชั้นเดียวไม่หักล้างตัวเอง ต้องใส่ bubble เป็นคู่เสมอ
"A single bubble inverts the signal; only a pair cancels by the involution law."

กับดักของหัวข้อนี้

① **เอาเทคนิคนี้ไปใช้กับวงจรที่ไม่ใช่สองชั้น** — ถ้าวงจรมี 3 ชั้นขึ้นไป ต้องทำ**ทีละคู่ชั้น**ครับ ไม่ใช่โยน bubble มั่ว ๆ ทั้งวง

② **ลืมใส่ bubble บนสายบางเส้น** — ต้องใส่**ทุกเส้น**ที่วิ่งจากชั้นแรกไปชั้นสอง ถ้าตกไปเส้นหนึ่ง เกตที่ชั้นสองจะกลายเป็นเกตลูกผสมที่มี bubble บางขา ไม่ใช่ NAND

③ **ใส่ bubble บนสายที่มาจาก**อินพุตโดยตรง — ห้ามครับ ถ้าตัวแปรวิ่งจากขอบซ้ายเข้าเกตชั้นสองเลย (ไม่ผ่าน AND) ตัวนั้น**ต้องเติมอินเวอร์เตอร์จริง ๆ ให้มัน** ไม่ใช่ปล่อยผ่าน — เพราะ negative-OR จะกลับค่ามันด้วย **ข้อ 6 ของ Assignment มีเคสแบบนี้พอดี ระวังให้ดีครับ**

8 Dual symbol — วิธีทำให้วงจร NAND ล้วนอ่านออกในสามวินาที

ตะขอเกี่ยว · รับจากหัวข้อ 7 และ EP05

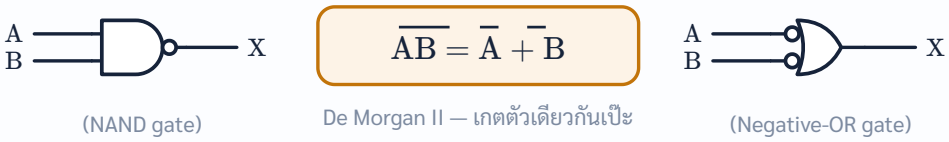
หัวข้อ 7 น้องแปลง $AB + CD$ เป็น NAND 3 ตัวได้แล้วครับ แต่ลองกลับด้านดู — **ถ้ามีใครยื่นวงจร NAND 5 ตัวมาให้แล้วบอกว่า “เขียนนิพจน์เอาต์พุตมา” ล่ะครับ** ถ้าไล่ตรง ๆ จะได้ขีดซ้อนขีดสี่ชั้น แล้วต้องนั่งแตก De Morgan อีกสิบบรรทัด เสียเวลามาก

ทางลัดคือเปลี่ยนสัญลักษณ์ของเกตบางตัวเป็น “หน้าตาที่สอง” ก่อน แล้วนิพจน์จะโผล่มาให้อ่านตรง ๆ โดยไม่ต้องแตกขีดเลยครับ — นี่คือสิ่งที่ **Assignment #4 ข้อ 6 ถามตรง ๆ** และ EP05 เคยเกริ่นไว้แล้วว่า “เกตหนึ่งตัวมีสองหน้าตา”

นิยามจากสไลด์หน้า 60 · dual symbol ของ NAND

Recall: the NAND gate can function as either a NAND or a negative-OR logic (using De Morgan's Theorem). The NAND symbol and the negative-OR symbol are dual symbols.

แปล: เกต NAND ตัวเดียวกัน**วาดได้สองหน้าตา** — หน้าตา NAND (รูป AND มี bubble ที่ขาออก) กับหน้าตา negative-OR (รูป OR มี bubble ที่ขาเข้าทุกขา) ครับ **เป็นชิปตัวเดิม สายเดิม ต่างกันแค่รูปที่เราวาดลงกระดาษ**



(NAND gate) — De Morgan II — เกตตัวเดียวกันเป๊ะ — (Negative-OR gate)

สองรูปนี้คือชิปตัวเดียวกันครับ — เลือกวาดแบบไหนก็ได้ตามที่อ่านง่ายกว่า

A	B	NAND $\overline{AB}$	neg-OR $\overline{A} + \overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

ตารางยืนยัน

คอลัมน์ NAND ($\overline{AB}$) กับคอลัมน์ negative-OR ($\overline{A} + \overline{B}$) **เหมือนกันทั้ง 4 แถว** ครับ — เพราะเป็น De Morgan II ตรง ๆ

วิธีจำหน้าตาที่สอง: “พลิกรูปเกต แล้วย้าย bubble จากขาออกไปไว้ขาเข้าทุกขา” ครับ — AND พลิกเป็น OR · bubble ที่เคยอยู่ขาออกย้ายมาอยู่ขาเข้าทั้งสองขา

นิยามจากสไลด์หน้า 65 · dual symbol ของ NOR

Similarly, a NOR gate can function as either a NOR or a negative-AND logic (again using De Morgan's Theorem), which are dual symbols. — NOR วาดได้สองหน้าตาเช่นกันครับ คือหน้าตา NOR (รูป OR มี bubble ขาออก) กับ negative-AND (รูป AND มี bubble ขาเข้าทุกขา)



(NOR gate)

$$\overline{A + B} = \overline{A}\,\overline{B}$$

De Morgan I — เกตตัวเดียวกันเป๊ะ



(Negative-AND gate)

NOR กับ negative-AND ก็เป็นคู่ dual symbol เหมือนกันครับ

A	B	NOR $\overline{A+B}$	neg-AND $\overline{A}\,\overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

ตารางคู่ dual symbol ที่ต้องจำ

หน้าตาที่ 1	หน้าตาที่ 2	สูตร
NAND	negative-OR	$\overline{AB} = \overline{A} + \overline{B}$
NOR	negative-AND	$\overline{A+B} = \overline{A}\,\overline{B}$

กฎเหล็กในการเลือกหน้าตา — bubble-to-bubble หรือ nonbubble-to-nonbubble

กฎจากสไลด์หน้า 60 · ข้อสอบใช้กฎนี้ตรง ๆ

All logic diagrams using NAND gates should be drawn with each gate represented by either a NAND symbol or the equivalent negative-OR symbol.

Rules: To obtain a logic diagram that is easier to read, analyze, and obtain the output expression directly from the logic diagram — every connection between a gate output and a gate input is either bubble-to-bubble or nonbubble-to-nonbubble.

แปลเป็นภาษาพี่: ทุกจุดที่**ขาออกของเกตหนึ่งไปต่อกับขาเข้าของอีกเกตหนึ่ง** ต้องเป็นอย่างใดอย่างหนึ่งใน 2 แบบนี้เท่านั้นครับ — **มี bubble ทั้งสองฝั่ง** (แล้วมันหักล้างกัน) หรือ **ไม่มี bubble ทั้งสองฝั่ง ห้ามมีฝั่งเดียว** เพราะถ้ามีฝั่งเดียวแปลว่ามีขีดบนเหลือค้าง แล้วจะอ่านนิพจน์ตรง ๆ ไม่ได้

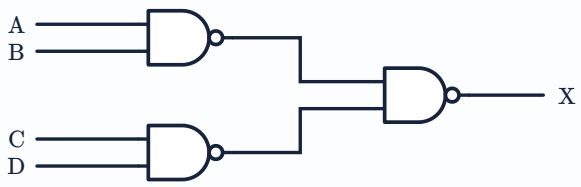
ของใกล้ตัว: กุญแจกับแม่กุญแจต้องมาเป็นคู่

คิดว่า bubble คือ**แม่กุญแจ**ครับ — ถ้าปลายสายฝั่งหนึ่งใส่แม่กุญแจไว้ อีกฝั่งก็ต้องมี**กุญแจ**มาไข ถึงจะเปิดผ่านได้ (bubble ชนกัน = หักล้าง = สายโล่ง)

ถ้าฝั่งหนึ่งใส่แม่กุญแจแต่อีกฝั่งไม่มีอะไร — สายนั้นก็**ยังล็อกอยู่** คือยังมีขีดบนค้างอยู่หนึ่งชั้น แล้วเราจะอ่านนิพจน์ตรง ๆ ไม่ได้ ต้องไปนั่งแตก De Morgan เองครับ

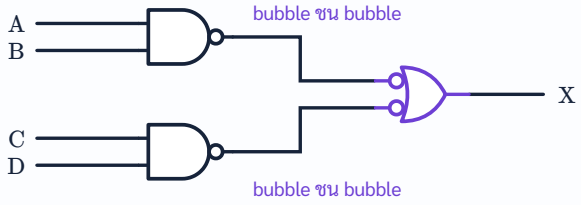
ตัวอย่างสั้น — วงจร NAND 3 ตัวเขียนได้ 3 หน้าตา (สไลด์หน้า 61)

หน้าตาที่ 1 — Original NAND logic diagram (อ่านยาก)



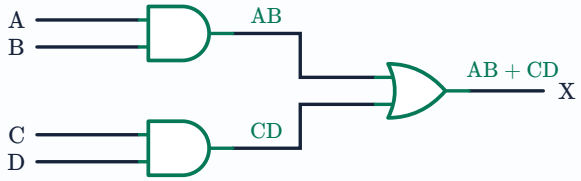
ไล่ตรง ๆ ได้ $X = \overline{\overline{AB} \cdot \overline{CD}}$ — **ขีดสามชั้น อ่านไม่ออกทันที**

หน้าตาที่ 2 — Equivalent NAND / negative-OR (bubble ชนกันพอดี)



วาดเกตขวาเป็น negative-OR แทน — ตอนนี้ bubble ชนกันทุกจุดเชื่อม ตามกฎในสไลด์ครับ

หน้าตาที่ 3 — AND-OR equivalent (bubble หักล้างกันหายไปหมด)



$X = AB + CD$ **อ่านออกทันทีครับ** — วงจร NAND ล้วนที่แท้จริงแล้วเป็น SOP ธรรมดา

สรุปวิธีอ่านวงจร NAND ล้วนใน 3 ขั้น

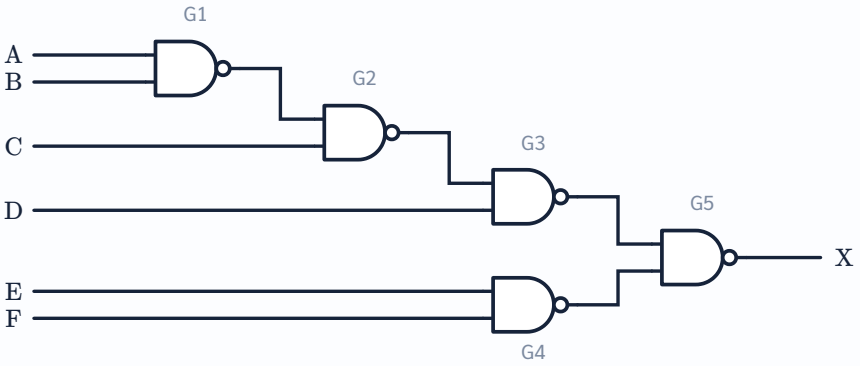
ขั้น 1 เริ่มจากเกตขวาสุด **วาดใหม่เป็น negative-OR** (เพราะเราอยากได้ทรง SOP)

ขั้น 2 ไล่ย้อนเข้ามา — เกตที่ป้อนเข้าขาที่**มี bubble** ให้วาดเป็นแบบ**มี bubble ที่ขาออก** (NAND) · เกตที่ป้อนเข้าขาที่**ไม่มี bubble** ให้วาดเป็นแบบ**ไม่มี bubble ที่ขาออก** (negative-OR)

ขั้น 3 ลบ bubble ที่ชนกันออกทั้งคู่ แล้วอ่านนิพจน์ตรง ๆ — **bubble ที่เหลือค้างจะมีแต่ที่ขาซึ่งรับตัวแปรจากขอบซ้ายโดยตรง** ตัวนั้นให้ใส่ขีดบนให้มันครับ

ตัวอย่างยาว — วงจร NAND 5 ตัวจากสไลด์หน้า 62–64

แบบเดิม — NAND ทั้ง 5 ตัว



สไลด์บอกว่า **"Several Boolean steps are required to arrive at the final output expression"** — ไล่ตรง ๆ จะได้ขีดสี่ชั้นครับ

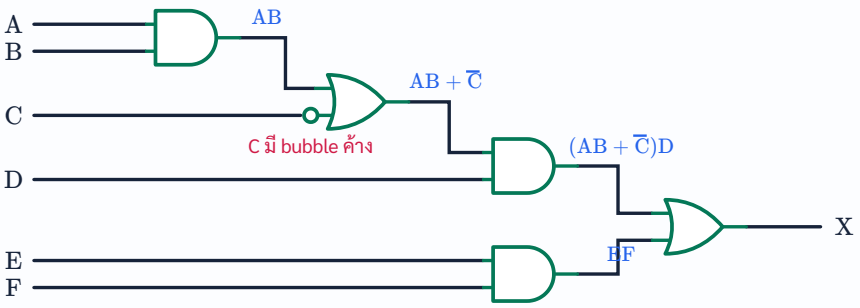
ลองไล่ตรง ๆ ดูก่อน — จะเห็นว่ามันโหดแค่ไหน

$$G_1 = \overline{AB} \cdot G_2 = \overline{\overline{AB}\,C} \cdot G_3 = \overline{\overline{\overline{AB}\,C}\;D} \cdot G_4 = \overline{EF} \cdot X = \overline{G_3 \cdot G_4}$$

แทนกลับเข้าไปจะได้ $X = \overline{\overline{\overline{AB}\,C}\;D \cdot \overline{EF}}$ — **ขีดสี่ชั้นซ้อนกัน** ต้องแตก De Morgan ทีละชั้นอีกหลายบรรทัดครับ

นี่คือเหตุผลที่ต้องใช้ dual symbol — ไม่ใช่เพราะเท่ แต่เพราะเร็วกว่าจริง ๆ

แบบ dual symbol — AND-OR equivalent (สไลด์หน้า 64)



bubble ที่ชนกันหายไปหมด **เหลือ bubble ค้างอยู่ที่ขา C ขาเดียว** เพราะ C มาจากขอบซ้ายโดยตรง ไม่มีคู่ให้ชน

$$X = (AB + \overline{C})D + EF$$

อ่านตรงจากรูป dual symbol · ตรวจด้วย dig.py แล้วตรงกับวงจร NAND เดิมทั้ง 64 แถว

1

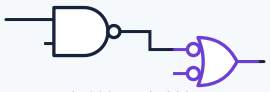


ขั้น 1 — เริ่มจากเกตขวาสุด แล้วเลือกให้มันเป็น negative-OR

เราอยากอ่านนิพจน์ในทรง SOP (บวกกันข้างนอก) **เพราะ** ทรง SOP อ่านง่ายที่สุดครับ ดังนั้นเกตขวาสุดควรทำหน้าที่เป็น **OR**

NAND วาดเป็น negative-OR ได้ → ตัวมันเลยกลายเป็น OR ที่มี bubble ที่ขาเข้าทั้งสองขา

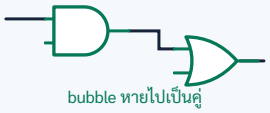
2



ขั้น 2 — ไล่ย้อนเข้าไป ให้ทุกจุดเชื่อม “ชนกันพอดี”

ขาเข้าของ negative-OR **มี bubble** → เกตที่ป้อนเข้ามา (G_3 กับ G_4) ต้องมี bubble ที่ขาออก **เพราะ** กฎบอกว่า bubble ต้องชน bubble ครับ → G_3, G_4 **คงเป็น NAND ไว้เหมือนเดิม** ถัดไป ขาเข้าของ G_3 (แบบ NAND) **ไม่มี bubble** → G_2 ต้องไม่มี bubble ที่ขาออก → G_2 **วาดเป็น negative-OR** · แล้ววนแบบเดียวกันจนถึง G_1

3

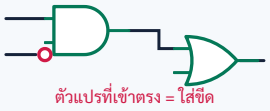


ขั้น 3 — ลบ bubble ที่ชนกันออกทั้งคู่

bubble สองวงที่จ่อกันบนสายเดียวกันคือขีดสองชั้น **เพราะ** involution มันหักล้างกัน จึงลบทิ้งได้ทั้งคู่ครับ (นี่คือขั้นตอนย้อนกลับของหัวข้อ 7 เป๊ะ ๆ)

ผลคือ NAND กลายเป็น AND ธรรมดา และ negative-OR กลายเป็น OR ธรรมดา

4



ขั้น 4 — ดู bubble ที่เหลือค้าง แล้วใส่ขีดบนให้ตัวแปรนั้น

ตัวแปร C วิ่งจากขอบซ้ายเข้าขาของ G_2 ซึ่งตอนนี้เป็น negative-OR ที่**มี bubble ที่ขาเข้าทุกขา** **เพราะ** ไม่มีเกตต้นทางมาชนให้ bubble หักล้าง bubble วงนั้นจึงค้างอยู่ → C **กลายเป็น** $\overline{C}$ **นี่คือจุดที่คนพลาดมากที่สุดของทั้ง EP ครับ** — ต้องไล่ดูทุกขาที่รับตัวแปรตรงจากขอบซ้าย ว่าขานั้นมี bubble ค้างไหม

5

$(\mathrm{AB}+\overline{\mathrm{C}})\mathrm{D}+\mathrm{EF}$

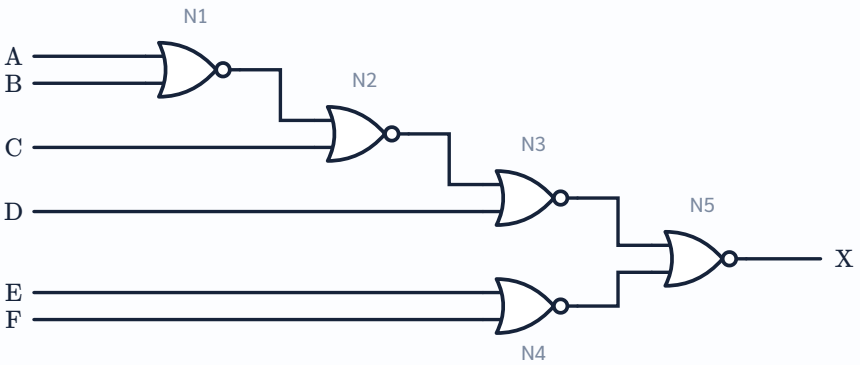
ขั้น 5 — อ่านนิพจน์จากรูปใหม่ตรง ๆ

G_1 (AND) ให้ AB → G_2 (OR ที่ C มี bubble) ให้ $AB+\overline{C}$ → G_3 (AND กับ D) ให้ $(AB+\overline{C})D$ → G_4 (AND) ให้ EF → G_5 (OR) รวมเป็นคำตอบ

$X=(AB+\overline{C})D+EF$ **— จบใน 5 บรรทัด ไม่ต้องแตก De Morgan สักครั้งเดียวครับ**

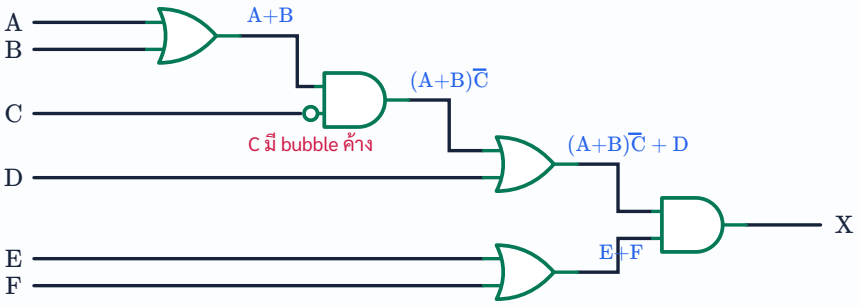
ฝั่ง NOR — วงจร NOR 5 ตัวจากสไลด์หน้า 67–69

แบบเดิม — NOR ทั้ง 5 ตัว



โครงเหมือนฝั่ง NAND ทุกอย่าง แค่เปลี่ยนเกตเป็น NOR ครับ

แบบ dual symbol — OR-AND equivalent (สไลด์หน้า 69)



ฝั่ง NOR ได้ทรง **POS** ออกมาแทน — C ก็ยังมี bubble ค้างเหมือนเดิมครับ

$$X = \big((A+B)\overline{C}+D\big)(E+F)$$

อ่านตรงจากรูป dual symbol · ตรวจด้วย dig.py แล้วตรงกับวงจร NOR เดิมทั้ง 64 แถว

เทียบของเก่า · EP05 เกริ่นไว้ ตอนนี้ได้ใช้จริงแล้ว

EP05 สอนว่า NAND กับ negative-OR เป็นเกตเดียวกัน แล้วพิสูจน์ด้วยตารางค่าความจริง — **ตอนนั้นน้องอาจสงสัยว่ารู้ไปทำไม คำตอบอยู่ตรงนี้ครับ** — มันคือเครื่องมือที่ทำให้อ่านวงจร NAND ล้วนได้ในสามวินาที แทนที่จะแตก De Morgan สิบบรรทัด และ **Assignment #4 ข้อ 6 ถามเรื่องนี้ตรง ๆ ทั้งข้อ** ซึ่งเราจะทำในหัวข้อ 11 ครับ

กับดักของหัวข้อนี้ — 3 อย่างที่ทำให้เฉลยผิดทั้งข้อ

① **ลืม bubble ที่ค้างอยู่บนขาที่รับตัวแปรตรง** — ในตัวอย่างข้างบนคือ C ที่ต้องกลายเป็น $\overline{C}$ ครับ **ถ้าลืม คำตอบจะกลายเป็น** $(AB+C)D+EF$ **ซึ่งผิด**

② **เปลี่ยนสัญลักษณ์ของเกตแล้วคิดว่าวงจรเปลี่ยน** — ไม่เปลี่ยนครับ **ชิปตัวเดิม สายเดิม ค่าเดิมทุกแถว** เปลี่ยนแค่รูปที่เราวาด

③ **วาดเกตสองตัวติดกันให้ bubble ไม่ชนกัน** — ถ้าเจอ “bubble ฝั่งเดียว” แปลว่าเลือกหน้าตาผิด ให้กลับไปสลับหน้าตาของเกตตัวใดตัวหนึ่งใหม่ครับ

ศัพท์สอบของหัวข้อนี้

English	แปลว่า	ข้อสอบเขียนมาแบบนี้
dual symbols	สัญลักษณ์คู่ของเกตเดียวกัน	*“Redraw the circuits below … using appropriate dual symbols.”*
negative-OR / negative-AND	OR ที่มี bubble ขาเข้า / AND ที่มี bubble ขาเข้า	*“(Negative OR gate)”* ใต้รูปในสไลด์
bubble-to-bubble	bubble ชน bubble	*“Every connection … is either bubble-to-bubble or nonbubble-to-nonbubble.”*
develop the output expression	หา/เขียนนิพจน์เอาต์พุต	คำสั่งของ Assignment ข้อ 6 ตรง ๆ
equivalent logic diagram	ผังวงจรที่สมมูลกัน	*“Equivalent NAND/negative-OR logic diagram”*

9 เฉลย Assignment #4 ข้อ 4 — majority ของสามอินพุต

วิธีอ่านเฉลยส่วนนี้

ทุกข้อพี่ทำครบ **6 ชั้น** เรียงเหมือนกันหมดครับ — **① โจทย์ต้นฉบับ + โจทย์ซ่อนอะไร · ② เตรียมก่อนลงมือ · ③ ไล่ทีละขั้นพร้อมภาพ · ④ คำตอบสุดท้าย · ⑤ ตรวจด้วยวิธีที่สอง · ⑥ กับดักของข้อนี้**

ชั้น ① โจทย์ต้นฉบับ

Assignment #4 ข้อ 4 — Kallapapruk Wichianrut · EES271-4

> Design a three variable Boolean expression whose output is HIGH when a majority of the inputs are HIGH. Express the logic in both SOP and POS forms.

แปล: ออกแบบนิพจน์บูลีน 3 ตัวแปร ที่เอาต์พุตเป็น **HIGH** เมื่อ**อินพุตส่วนใหญ่**เป็น HIGH แล้วเขียนคำตอบเป็นทั้งรูป SOP และ POS ครับ

โจทย์ซ่อนอะไรไว้

① **คำว่า "majority" ไม่ได้แปลว่า "ทั้งหมด"** ครับ — majority แปลว่า **"เกินครึ่ง"** ถ้ามี 3 อินพุต ครึ่งหนึ่งคือ 1.5 ดังนั้น majority คือ **ตั้งแต่ 2 ตัวขึ้นไป ไม่ใช่ 3 ตัว** — ถ้าอ่านผิดจะได้แค่ ABC ซึ่งผิดทั้งข้อ

② **โจทย์ไม่ได้ให้ตารางค่าความจริงมา ต้องสร้างเอง** (ในกระดาษโจทย์มีตารางเปล่าให้เติม 8 แถว) — ห้ามข้ามขั้นนี้เด็ดขาดครับ

③ **ต้องตอบทั้งสองรูป** — both SOP and POS forms คือ**สองคำตอบ** ตอบมาอันเดียวได้ครึ่งคะแนน

ชั้น ② เตรียมก่อนลงมือ

เลือกวิธีไหน และทำไม

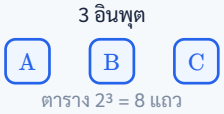
วิธีที่เลือก: สร้างตารางค่าความจริงก่อน แล้วค่อยอ่านออกมาเป็น SOP และ POS

ทำไมเลือกวิธีนี้: เพราะโจทย์อธิบายเงื่อนไขเป็น**ภาษาคน** ("ส่วนใหญ่เป็น HIGH") ไม่ใช่นิพจน์ครับ — **ตารางค่าความจริงคือสะพานระหว่างภาษาคนกับนิพจน์** และเมื่อมีตารางแล้ว SOP กับ POS ก็อ่านออกมาได้ทันทีทั้งคู่โดยไม่ต้องคิดใหม่

กฎที่จะใช้ (จาก EP07):
SOP = อ่านจาก**แถวที่เอาต์พุตเป็น 1** — ตัวแปรที่เป็น 1 เขียนตัวเปล่า · เป็น 0 เขียนมีขีด · แล้วคูณกัน · ทุกแถวบวกกัน
POS = อ่านจาก**แถวที่เอาต์พุตเป็น 0** — **กลับกันเป๊ะ** ตัวแปรที่เป็น 0 เขียนตัวเปล่า · เป็น 1 เขียนมีขีด · แล้วบวกกันในวงเล็บ · ทุกวงเล็บคูณกัน

ชั้น ③ ไล่ทีละขั้นพร้อมภาพ

1



ขั้น 1 — ตั้งตารางค่าความจริง 8 แถว

3 ตัวแปร → $2^3 = 8$ แถว **เพราะ** แต่ละตัวแปรมีได้ 2 ค่า สามตัวจึงมีชุดค่าที่เป็นไปได้ $2 \times 2 \times 2 = 8$ ชุดครับ

เรียงแบบนับเลขฐานสอง 000 → 111 เพื่อไม่ให้ตกแถวไหน (เทคนิคจาก EP01 ครับ)

2

```
011   มี 1 สองตัว
101   มี 1 สองตัว
110   มี 1 สองตัว
111   มี 1 สามตัว
```

ขั้น 2 — หาว่าแถวไหนที่ “ส่วนใหญ่เป็น HIGH”

นับจำนวนขาที่เป็น 1 ในแต่ละแถว แล้วถามว่า**มากกว่า 1.5 ไหม เพราะ** majority ของ 3 คือ มากกว่าครึ่ง และครึ่งของ 3 คือ 1.5 ครับ

ได้ 4 แถวที่เป็น 1: 011, 101, 110, 111 — อีก 4 แถวที่เหลือ (000, 001, 010, 100) มี 1 ไม่ถึงสองตัว จึงเป็น 0

3

แถวเขียว 4 แถว → SOP

แถวแดง 4 แถว → POS

ขั้น 3 — ระบายสีแถวแล้วอ่านสองทาง

แถวเขียว (X=1) มี 4 แถว → ได้ SOP ที่มี 4 พจน์ · **แถวแดง (X=0) มี 4 แถว** → ได้ POS ที่มี 4 วงเล็บ **เพราะ** SOP นับพจน์ตามแถวที่ได้ 1 ส่วน POS นับวงเล็บตามแถวที่ได้ 0 ครับ

เช็คได้ทันทีว่าจำนวนพจน์ SOP + จำนวนวงเล็บ POS ต้องเท่ากับ 8 พอดี $(4+4=8)$ — ถ้าไม่ครบแปลว่าตกแถวครับ

ตารางค่าความจริงเต็ม พร้อม minterm และ maxterm ของทุกแถว

A	B	C	จำนวนขาที่เป็น 1	X	minterm / maxterm
0	0	0	0	0	$(A+B+C)$
0	0	1	1	0	$(A+B+\overline{C})$
0	1	0	1	0	$(A+\overline{B}+C)$
0	1	1	2	1	$\overline{A}BC$
1	0	0	1	0	$(\overline{A}+B+C)$
1	0	1	2	1	$A\overline{B}C$
1	1	0	2	1	$AB\overline{C}$
1	1	1	3	1	ABC

แถวเขียว = majority (มี 1 อย่างน้อย 2 ตัว) · คอลัมน์ขวาสุดคือ minterm ของแถวเขียว และ maxterm ของแถวแดงครับ

อ่าน SOP จากแถวเขียว 4 แถว

แถวที่	ABC	แปลงยังไง	ได้ minterm
m_3	0 1 1	$A=0$ → ใส่ขีด · $B=1$ → ตัวเปล่า · $C=1$ → ตัวเปล่า	$\overline{A}BC$
m_5	1 0 1	$A=1$ → ตัวเปล่า · $B=0$ → ใส่ขีด · $C=1$ → ตัวเปล่า	$A\overline{B}C$
m_6	1 1 0	$A=1 \cdot B=1$ → ตัวเปล่า · $C=0$ → ใส่ขีด	$AB\overline{C}$
m_7	1 1 1	ทุกตัวเป็น 1 → ตัวเปล่าหมด	ABC

$$X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

SOP มาตรฐาน (standard SOP) · เขียนย่อได้เป็น $X = \sum m(3,5,6,7)$

อ่าน POS จากแถวแดง 4 แถว

จุดที่คนสับสนที่สุด — POS ต้องกลับค่าตัวแปร

SOP: เห็น 1 เขียนตัวเปล่า · เห็น 0 เขียนมีขีด → แล้ว**คูณ**กัน

POS: เห็น 0 เขียน**ตัวเปล่า** · เห็น 1 เขียน**มีขีด** → แล้ว**บวก**กันในวงเล็บ — **กลับหมดทั้งสองอย่าง**ครับ

ที่มาว่าทำไมต้องกลับ: maxterm ต้องมีค่าเป็น **0 เฉพาะแถวนั้นแถวเดียว** ผลบวกจะเป็น 0 ก็ต่อเมื่อ**ทุกตัวที่บวกกันเป็น 0 หมด** ดังนั้นถ้าแถวนั้น $A = 1$ เราต้องเขียน $\overline{A}$ เพื่อให้มันกลายเป็น 0 ครับ

แถวที่	ABC	แปลงยังไง (กลับค่า)	ได้ maxterm
M_0	0 0 0	ทุกตัวเป็น 0 → ตัวเปล่าหมด	$(A + B + C)$
M_1	0 0 1	$A = 0, B = 0$ → ตัวเปล่า · $C = 1$ → ใส่ขีด	$(A + B + \overline{C})$
M_2	0 1 0	$A = 0, C = 0$ → ตัวเปล่า · $B = 1$ → ใส่ขีด	$(A + \overline{B} + C)$
M_4	1 0 0	$B = 0, C = 0$ → ตัวเปล่า · $A = 1$ → ใส่ขีด	$(\overline{A} + B + C)$

$$X = (A + B + C)(A + B + \overline{C})(A + \overline{B} + C)(\overline{A} + B + C)$$

POS มาตรฐาน · เขียนย่อได้เป็น $X = \prod M(0, 1, 2, 4)$

รูปย่อ (ถ้าอาจารย์อยากได้วงจรที่ประหยัดเกต)

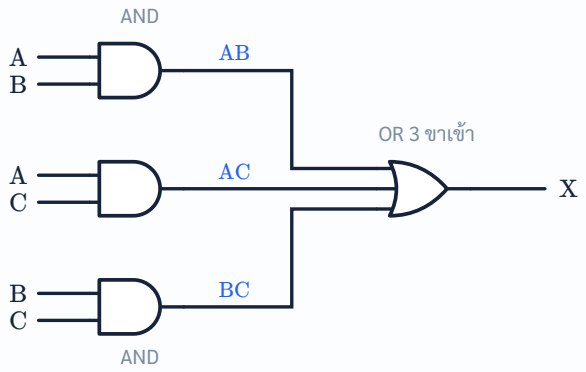
ย่อ SOP ด้วยกฎบูลีนจาก EP06

$X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$

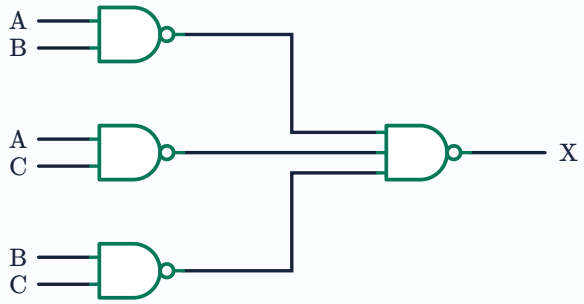
$= \overline{A}BC + ABC + A\overline{B}C + ABC + AB\overline{C} + ABC$ (Idempotent I — ก๊อป ABC ขึ้นมาใช้ซ้ำ 3 ครั้ง)

$= BC(\overline{A} + A) + AC(\overline{B} + B) + AB(\overline{C} + C)$ (Distributive I อ่านย้อน = ดึงตัวร่วม)

$= BC \cdot 1 + AC \cdot 1 + AB \cdot 1 = \mathbf{AB} + \mathbf{AC} + \mathbf{BC}$ (Complementary I แล้ว Identity II)



วงจรของรูปย่อ $X = AB + AC + BC$ — ใช้ AND 2 ขาเข้า 3 ตัว + OR 3 ขาเข้า 1 ตัว รวม 4 เกตครับ



เวอร์ชัน NAND ล้วน 4 ตัว (เทคนิค bubble คู่จากหัวข้อ 7)

ถ้าโรงงานมีแต่ NAND ก็ใช้วงจรนี้ครับ — $X = \overline{\overline{AB} \cdot \overline{AC} \cdot \overline{BC}}$ ให้ผลเท่ากันทุกแถว

ชั้น ④ คำตอบสุดท้าย

คำตอบข้อ 4

SOP form: $\mathbf{X = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC = \sum m(3,5,6,7)}$

POS form: $\mathbf{X = (A+B+C)(A+B+\overline{C})(A+\overline{B}+C)(\overline{A}+B+C) = \prod M(0,1,2,4)}$

รูปย่อ (ถ้าโจทย์ต้องการ): $\mathbf{X = AB + AC + BC}$ — รูปนี้เรียกว่า majority function ครับ

"X = $\bar{A}BC$ + $A\bar{B}C$ + $AB\bar{C}$ + ABC (SOP) = (A+B+C)(A+B+$\bar{C}$)(A+$\bar{B}$+C)($\bar{A}$+B+C) (POS)."

ชั้น ⑤ ตรวจคำตอบด้วยวิธีที่สอง

ตรวจจริงด้วย tools/dig.py

พี่รันจริงก่อนเขียนลงชีททุกครั้งครับ ผลที่ได้:

```
truth_table("(A&B)|(A&C)|(B&C)", "ABC")
  X column  =  [0, 0, 0, 1, 0, 1, 1, 1]      ตรงกับตารางในชีท
minterms(...)  =  [3, 5, 6, 7]               SOP 4 พจน์
maxterms(...)  =  [0, 1, 2, 4]               POS 4 วงเล็บ

same_logic(SOP4, "(A&B)|(A&C)|(B&C)", "ABC")   =  True
same_logic(POS4, "(A&B)|(A&C)|(B&C)", "ABC")   =  True
same_logic(SOP4, POS4, "ABC")                     =  True
same_logic("~( ~(A&B) & ~(A&C) & ~(B&C) )", maj, "ABC") =  True
```

อ่านผลยังไง: บรรทัดที่ 3 บอกว่า SOP ที่เราเขียนตรงกับรูปย่อ · บรรทัดที่ 4 บอกว่า POS ก็ตรงเหมือนกัน · บรรทัดที่ 5 บอกว่า **SOP กับ POS เป็นฟังก์ชันเดียวกันจริง** ซึ่งเป็นการตรวจที่สำคัญที่สุดของข้อนี้ครับ · บรรทัดสุดท้ายยืนยันว่าเวอร์ชัน NAND ล้วนก็ถูก

ตรวจด้วยมืออีกทาง — แทนค่าสองแถวที่เสี่ยงที่สุด

แถว $A=0, B=1, C=1$ (majority = ใช่ ต้องได้ 1):

SOP: $\overline{0}\cdot 1\cdot 1 + 0\cdot\overline{1}\cdot 1 + 0\cdot 1\cdot\overline{1} + 0\cdot 1\cdot 1 = 1+0+0+0 = \mathbf{1}$ ✓

POS: $(0+1+1)(0+1+\overline{1})(0+\overline{1}+1)(\overline{0}+1+1) = (1)(1)(1)(1) = \mathbf{1}$ ✓

แถว $A=1, B=0, C=0$ (majority = ไม่ใช่ ต้องได้ 0):

SOP: $\overline{1}\cdot 0\cdot 0 + 1\cdot\overline{0}\cdot 0 + 1\cdot 0\cdot\overline{0} + 1\cdot 0\cdot 0 = 0+0+0+0 = \mathbf{0}$ ✓

POS: วงเล็บสุดท้าย $(\overline{A}+B+C) = (\overline{1}+0+0) = (0+0+0) = 0$ ทำให้ผลคูณทั้งหมดเป็น $\mathbf{0}$ ✓ — **เห็นชัดว่า maxterm ตัวไหน "ฆ่า" แถวไหน**ครับ

ชั้น ⑥ กับดักของข้อนี้

4 อย่างที่ทำให้เสียคะแนนข้อนี้

① **ตอบ $X = ABC$ อย่างเดียว** — เข้าใจว่า majority = ทุกตัว **คำตอบผิดที่จะได้:** $X = ABC$ ซึ่งให้ 1 แค่แถวเดียวจาก 4 แถวที่ควรได้ 1 **เสียคะแนนทั้งข้อครับ**

② **เขียน POS แต่ไม่กลับค่าตัวแปร — คำตอบผิดที่จะได้:** $(\overline{A}+\overline{B}+\overline{C})(\overline{A}+\overline{B}+C)\ldots$ ซึ่งเป็นการเอาแถว 0 มาเขียนแบบ SOP ครับ **ต้องกลับค่าทุกตัวเสมอ**

③ **ตอบ SOP มาแล้วเอาไปย่อจนเหลือ $AB+AC+BC$ แล้วไม่ตอบรูปเต็ม** — โจทย์เขียนว่า `express the logic in both SOP and POS forms` **พี่แนะนำให้ตอบรูปมาตรฐาน (standard form) ก่อน แล้วค่อยเขียนรูปย่อต่อท้ายเป็นโบนัส**ครับ ปลอดภัยที่สุด

④ **สลับลำดับตัวแปรตอนเขียน minterm** — ต้องเรียง A, B, C ตามหัวตารางเสมอครับ ถ้าเขียน $CB\overline{A}$ อาจารย์อาจอ่านว่าคนละพจน์

หัวใจของข้อนี้ — 1 บรรทัด

"majority ของ 3 อินพุต = มี 1 อย่างน้อย 2 ตัว = 4 แถวจาก 8 แถว → SOP 4 พจน์ · POS 4 วงเล็บ · รูปย่อคือ $AB+AC+BC$"

10 เฉลย Assignment #4 ข้อ 5 — จำนวน HIGH ไม่เท่ากับจำนวน LOW

ชั้น ① โจทย์ต้นฉบับ

Assignment #4 ข้อ 5 — Kallapapruk Wichianrut · EES271-4

> Derive a four variable Boolean expression whose output is HIGH when the number of HIGH inputs and the number of LOW inputs are NOT equal. Express the logic in both SOP and POS forms.

แปล: หานิพจน์บูลีน 4 ตัวแปร ที่เอาต์พุตเป็น HIGH เมื่อ **จำนวนอินพุตที่เป็น HIGH ไม่เท่ากับจำนวนที่เป็น LOW** แล้วเขียนทั้งรูป SOP และ POS ครับ

โจทย์ซ่อนอะไรไว้ — ข้อนี้ซ่อนเยอะที่สุดในชุด

① **คำว่า NOT equal แปลว่าเราต้องหา "แถวที่เท่ากัน" ก่อน แล้วกลับค่า** — มันง่ายกว่าไล่หาแถวที่ไม่เท่ากันทั้ง 10 แถวมากครับ

② **มี 4 อินพุต แปลว่า "เท่ากัน" มีได้แบบเดียวคือ 2 ต่อ 2** — เพราะจำนวน HIGH กับ LOW ต้องรวมกันได้ 4 เสมอ ถ้าเท่ากันก็ต้องเป็น $2+2$ พอดี **ไม่มีทางเป็น 1 ต่อ 1 หรือ 3 ต่อ 3 ได้** เพราะรวมแล้วไม่ครบ 4

③ **ดังนั้น $X=0$ เฉพาะแถวที่มี 1 อยู่พอดี 2 ตัว** ซึ่งมี $\binom{4}{2}=6$ แถว · **อีก 10 แถวเป็น 1 ทั้งหมด**

④ **SOP จะยาว 10 พจน์ แต่ POS สั้นแค่ 6 วงเล็บ** — ข้อนี้เป็นตัวอย่างที่ดีที่สุดว่า**ทำไมต้องมี POS ในโลกนี้**ครับ

ชั้น ② เตรียมก่อนลงมือ

เลือกวิธีไหน และทำไม

วิธีที่เลือก: หาแถวที่เอาต์พุตเป็น 0 ก่อน (แถวที่มี 1 สองตัวพอดี) แล้วเติม 1 ให้แถวที่เหลือทั้งหมด

ทำไมเลือกวิธีนี้: เพราะแถวที่เป็น 0 มีแค่ 6 แถว ส่วนแถวที่เป็น 1 มีตั้ง 10 แถวครับ — **ไล่ของน้อยกว่าย่อมพลาดยากกว่า** และเรารู้วิธีนับด้วยว่ามีกี่แถว (เลือก 2 ตำแหน่งจาก 4 ตำแหน่ง = $\binom{4}{2}=6$) จึงเช็คได้ว่าครบหรือยัง

สูตรที่จะใช้: $2^4=16$ แถว · **จำนวนพจน์ SOP + จำนวนวงเล็บ POS ต้องเท่ากับ 16** ($10+6=16$ ✓) — ใช้เช็คตอนท้ายว่าตกแถวไหนไปหรือเปล่าครับ

ชั้น ③ ไล่ทีละขั้นพร้อมภาพ

1



ขั้น 1 — แปลเงื่อนไขให้เป็น "มี 1 กี่ตัว"

จำนวน HIGH บวกจำนวน LOW ต้องได้ 4 เสมอ **เพราะ** ทุกขาต้องเป็นอย่างใดอย่างหนึ่งครับ ดังนั้นถ้าให้จำนวน HIGH เป็น k จำนวน LOW ก็คือ $4-k$

"เท่ากัน" คือ $k=4-k \rightarrow 2k=4 \rightarrow k=2$ **แปลว่า $X=0$ เฉพาะแถวที่มี 1 พอดีสองตัว** นอกนั้น $X=1$ หมดครับ

2

6 แถวนี้เท่านั้นที่ X = 0

ขั้น 2 — ไล่แถวที่มี 1 สองตัวให้ครบ 6 แถว

วิธีไล่ให้ไม่ตก: **ไล่ตำแหน่งของเลข 1 คู่แรก** ครับ — เลือก 2 ตำแหน่งจาก 4 ตำแหน่ง ได้ $\binom{4}{2} = 6$ แบบพอดี **เพราะ** ตำแหน่งที่เหลือต้องเป็น 0 อัตโนมัติ

ได้ $0011(m_3), 0101(m_5), 0110(m_6), 1001(m_9), 1010(m_{10}), 1100(m_{12})$ — **ครบ 6 แถวพอดี** ✓

3

แถวเขียว 10 → SOP 10 พจน์

แถวแดง 6 → POS 6 วงเล็บ

ขั้น 3 — เติมตารางแล้วนับ

16 แถวลบ 6 แถวที่เป็น 0 เหลือ **10 แถวที่เป็น 1** **เพราะ** ทุกแถวต้องเป็น 0 หรือ 1 อย่างใดอย่างหนึ่งครับ

เช็คทันที: $10 + 6 = 16$ ✓ **ครบทุกแถว** ถ้าบวกแล้วไม่ได้ 16 แปลว่าไล่ตกครับ กลับไปนับใหม่

ตารางค่าความจริงเต็ม 16 แถว

A	B	C	D	จำนวน 1	จำนวน 0	X	minterm / maxterm
0	0	0	0	0	4	1	$\overline{A}\overline{B}\overline{C}\overline{D}$
0	0	0	1	1	3	1	$\overline{A}\overline{B}\overline{C}D$
0	0	1	0	1	3	1	$\overline{A}\overline{B}C\overline{D}$
0	0	1	1	2	2	0	$(A+B+\overline{C}+\overline{D})$
0	1	0	0	1	3	1	$\overline{A}B\overline{C}\overline{D}$
0	1	0	1	2	2	0	$(A+\overline{B}+C+\overline{D})$
0	1	1	0	2	2	0	$(A+\overline{B}+\overline{C}+D)$
0	1	1	1	3	1	1	$\overline{A}BCD$
1	0	0	0	1	3	1	$A\overline{B}\overline{C}\overline{D}$
1	0	0	1	2	2	0	$(\overline{A}+B+C+\overline{D})$
1	0	1	0	2	2	0	$(\overline{A}+B+\overline{C}+D)$
1	0	1	1	3	1	1	$A\overline{B}CD$
1	1	0	0	2	2	0	$(\overline{A}+\overline{B}+C+D)$
1	1	0	1	3	1	1	$AB\overline{C}D$
1	1	1	0	3	1	1	$ABC\overline{D}$
1	1	1	1	4	0	1	$ABCD$

คอลัมน์ “จำนวน 1” กับ “จำนวน 0” ทำให้เห็นทันทีว่าแถวไหนเท่ากัน (2 ต่อ 2) — **แถวแดงทั้ง 6 แถวคือแถวที่เท่ากันพอดี**ครับ

อ่าน SOP จากแถวเขียว 10 แถว (แถวที่ 1 ไม่เท่ากับ 0)

แถว	$ABCD$	จำนวน 1	minterm
m_0	0 0 0 0	0	$\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$
m_1	0 0 0 1	1	$\overline{A}\,\overline{B}\,\overline{C}D$
m_2	0 0 1 0	1	$\overline{A}\,\overline{B}C\overline{D}$
m_4	0 1 0 0	1	$\overline{A}B\overline{C}\,\overline{D}$

แถว	$ABCD$	จำนวน 1	minterm
m_7	0 1 1 1	3	$\overline{A}BCD$
m_8	1 0 0 0	1	$A\overline{B}\,\overline{C}\,\overline{D}$
m_{11}	1 0 1 1	3	$A\overline{B}CD$
m_{13}	1 1 0 1	3	$AB\overline{C}D$
m_{14}	1 1 1 0	3	$ABC\overline{D}$
m_{15}	1 1 1 1	4	$ABCD$

$$X = \overline{A}BCD + A\overline{B}CD + A\overline{B}C\overline{D} + \overline{A}B\overline{C}D + \overline{A}BC\overline{D}$$
$$+ A\overline{B}\overline{C}D + A\overline{B}CD + AB\overline{C}D + ABC\overline{D} + ABCD$$

SOP มาตรฐาน 10 พจน์ · เขียนย่อได้เป็น $X = \sum m(0, 1, 2, 4, 7, 8, 11, 13, 14, 15)$

อ่าน POS จากแถวแดง 6 แถว (กลับค่าตัวแปรทุกตัว)

แถว	$ABCD$	แปลงยังไง (0 → ตัวเปล่า · 1 → ใส่ขีด)	maxterm
M_3	0 0 1 1	A, B เป็น 0 → เปล่า · C, D เป็น 1 → ใส่ขีด	$(A + B + \overline{C} + \overline{D})$
M_5	0 1 0 1	A, C เป็น 0 → เปล่า · B, D เป็น 1 → ใส่ขีด	$(A + \overline{B} + C + \overline{D})$
M_6	0 1 1 0	A, D เป็น 0 → เปล่า · B, C เป็น 1 → ใส่ขีด	$(A + \overline{B} + \overline{C} + D)$
M_9	1 0 0 1	B, C เป็น 0 → เปล่า · A, D เป็น 1 → ใส่ขีด	$(\overline{A} + B + C + \overline{D})$
M_{10}	1 0 1 0	B, D เป็น 0 → เปล่า · A, C เป็น 1 → ใส่ขีด	$(\overline{A} + B + \overline{C} + D)$
M_{12}	1 1 0 0	C, D เป็น 0 → เปล่า · A, B เป็น 1 → ใส่ขีด	$(\overline{A} + \overline{B} + C + D)$

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

POS มาตรฐาน 6 วงเล็บ · เขียนย่อได้เป็น $X = \prod M(3, 5, 6, 9, 10, 12)$

★ ทำไม POS ถึงสั้นกว่า SOP มากในข้อนี้

เหตุผลข้อเดียวที่ต้องตอบให้ได้

เพราะ SOP นับตามแถวที่ได้ 1 ส่วน POS นับตามแถวที่ได้ 0 ครับ — ข้อนี้มีแถว 1 อยู่ 10 แถว แต่แถว 0 มีแค่ 6 แถว

ที่มาว่าทำไมแถว 0 ถึงน้อย: เงื่อนไข “เท่ากัน” เป็นเงื่อนไข**ที่เจาะจงมาก** (ต้องเป็น 2 ต่อ 2 เป๊ะ ๆ) จึงมีแถวที่ผ่านเงื่อนไขนี้น้อย ส่วนเงื่อนไข “ไม่เท่ากัน” เป็นเงื่อนไข**หลวม** มีหลายทางที่จะไม่เท่ากัน (0-4, 1-3, 3-1, 4-0) จึงมีแถวเยอะครับ

กฎที่ใช้ตัดสินใจตอนสอบ: “นับแถว 1 กับแถว 0 ก่อนเสมอ ฝั่งไหนน้อยกว่าให้เขียนรูปนั้น” — ถ้าแถว 1 น้อยกว่าให้ใช้ SOP · ถ้าแถว 0 น้อยกว่าให้ใช้ POS (แต่ถ้าโจทย์สั่งให้เขียนทั้งสองรูปก็ต้องเขียนทั้งคู่นะครับ)

เทียบต้นทุน	SOP	POS
จำนวนพจน์ / วงเล็บ	**10**	**6**
เกตชั้นแรก	AND 4 ขาเข้า **10 ตัว**	OR 4 ขาเข้า **6 ตัว**
เกตชั้นสอง	OR 10 ขาเข้า 1 ตัว	AND 6 ขาเข้า 1 ตัว
อินเวอร์เตอร์	4 ตัว	4 ตัว
รวมเกตชั้นแรก+ชั้นสอง	**11 ตัว**	**7 ตัว**

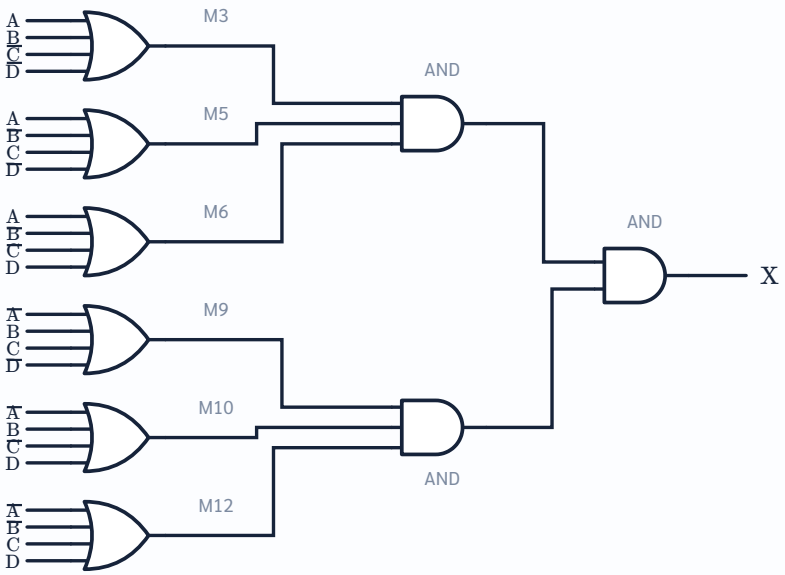
อ่านตารางนี้ยังไง

POS ประหยัดกว่าเกือบครึ่งครับ — 7 เกตเทียบกับ 11 เกต และยัง**ประหยัดขาเข้า**ด้วย เพราะ OR 10 ขาเข้าหาซื้อยากกว่า AND 6 ขาเข้ามาก

ในทางปฏิบัติ เกตขาเข้าเยอะ ๆ ต้องต่อจากเกตหลายตัว เช่น OR 10 ขาเข้าต้องใช้ OR 4 ขาเข้า 3 ตัว + OR 3 ขาเข้าอีก 1 ตัว — **ยิ่งทำให้ SOP แพงขึ้นไปอีก**ครับ

ประโยคที่เขียนลงกระดาษได้เลย: *“The POS form is much shorter here because only 6 of the 16 rows give a 0, whereas 10 rows give a 1.”*

วงจรของรูป POS — 6 วงเล็บ = OR 6 ตัว แล้ว AND รวม



หนึ่งวงเล็บ = OR 4 ขาเข้า 1 ตัว · แล้วคูณกันทั้งหมดด้วย AND (ในของจริงใช้ AND 3 ขาเข้า 2 ตัว + AND 2 ขาเข้า 1 ตัว เพราะ AND 6 ขาเข้าหาซื้อยากครับ)

ชั้น ④ คำตอบสุดท้าย

คำตอบข้อ 5

SOP form (10 พจน์):

$$\mathbf{X = \overline{A}BC\overline{D} + \overline{A}BCD + \overline{A}B\overline{C}\overline{D} + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}C\overline{D} + A\overline{B}CD + AB\overline{C}D + ABC\overline{D} + ABCD}$$

$$= \sum m(0, 1, 2, 4, 7, 8, 11, 13, 14, 15)$$

POS form (6 วงเล็บ):

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

$$= \prod M(3, 5, 6, 9, 10, 12)$$ — **รูป POS สั้นกว่าเกือบครึ่ง เพราะแถวที่ให้ 0 มีแค่ 6 แถว**

"X is LOW only when exactly two inputs are HIGH, so the POS form needs just six maxterms while the SOP form needs ten minterms."

ชั้น ⑤ ตรวจคำตอบด้วยวิธีที่สอง

ตรวจจริงด้วย tools/dig.py

```
X column (สร้างจากเงื่อนไขโจทย์ตรง ๆ)
  = [1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1]

X = 0 ที่ m = [3, 5, 6, 9, 10, 12]      count = 6   ตรงกับ POS 6 วงเล็บ
X = 1 ที่ m = [0,1,2,4,7,8,11,13,14,15] count = 10  ตรงกับ SOP 10 พจน์

truth_table(SOP5, "ABCD") -> ตรงกับ X column ทุกแถว   =  True
truth_table(POS5, "ABCD") -> ตรงกับ X column ทุกแถว   =  True
same_logic(SOP5, POS5, "ABCD")                          =  True
minterms(SOP5,"ABCD") = [0,1,2,4,7,8,11,13,14,15]     ครบ
maxterms(SOP5,"ABCD") = [3,5,6,9,10,12]               ครบ
```

อ่านผลยังไง: สองบรรทัดกลางบอกว่าทั้ง SOP และ POS ที่เราเขียน ให้ค่าตรงกับเงื่อนไขโจทย์ทุกแถว · บรรทัด `same_logic` ยืนยันว่า **SOP กับ POS เป็นฟังก์ชันเดียวกันจริง** ทั้งที่หน้าตาต่างกันมากครับ

ตรวจด้วยมือ — สุ่มสามแถวที่เสี่ยง

แถว $ABCD = 0011$ (มี 1 สองตัว → ต้องได้ 0): ดู maxterm $M_3 = (A + B + \overline{C} + \overline{D}) = (0 + 0 + \overline{1} + \overline{1}) = (0 + 0 + 0 + 0) = 0$ → ผลคูณทั้งหมดเป็น **0** ✓ **เห็นชัดว่า M_3 คือตัวที่ "ฆ่า" แถวนี้พอดี**

แถว $ABCD = 0111$ (มี 1 สามตัว → ต้องได้ 1): เช็ค maxterm ทั้ง 6 ตัวว่ามีตัวไหนเป็น 0 ไหม — $M_3 = (0 + 1 + 0 + 0) = 1 \cdot M_5 = (0 + 0 + 1 + 0) = 1 \cdot M_6 = (0 + 0 + 0 + 1) = 1 \cdot M_9 = (1 + 1 + 1 + 0) = 1 \cdot M_{10} = (1 + 1 + 0 + 1) = 1 \cdot M_{12} = (1 + 0 + 1 + 1) = 1$ → ไม่มีตัวไหนเป็น 0 ดังนั้น $X = \mathbf{1}$ ✓

แถว $ABCD = 1111$ (มี 1 สี่ตัว → ต้องได้ 1): ทาง SOP เร็วกว่า — พจน์สุดท้ายคือ $ABCD = 1 \cdot 1 \cdot 1 \cdot 1 = 1$ → บวกกับอะไรก็ได้ **1** ✓ (Null Law I)

ชั้น ⑥ กับดักของข้อนี้

5 อย่างที่ทำให้เสียคะแนนข้อนี้

① **ไล่แถวที่เป็น 1 ทั้ง 10 แถวเอง แล้วตกไป 1–2 แถว** — **คำตอบผิดที่จะได้:** SOP 8 หรือ 9 พจน์ **วิธีกัน:** ไล่ฝั่งที่น้อยกว่า (6 แถว) แล้วเช็ค $10 + 6 = 16$ ครับ

② **คิดว่า "เท่ากัน" รวมกรณี 0 ต่อ 0 หรือ 1 ต่อ 1 ด้วย** — ไม่มีทางครับ เพราะจำนวนขาต้องรวมได้ 4 เสมอ **2 ต่อ 2 เป็นแบบเดียวที่เป็นไปได้**

③ **เขียน POS โดยไม่กลับค่าตัวแปร** — **คำตอบผิดที่จะได้:** M_3 กลายเป็น $(\overline{A} + \overline{B} + C + D)$ ซึ่งที่จริงคือ M_{12} **คนละแถวกันเลยครับ**

④ **ตอบมาแค่รูปเดียว** — โจทย์เขียน `both SOP and POS forms` ชัดเจน ต้องมีสองคำตอบครับ

⑤ **พยายามย่อ SOP 10 พจน์ด้วยมือแล้วเพี้ยน** — ข้อนี้**ไม่ต้องย่อ**ครับ โจทย์ขอแค่รูปมาตรฐาน และการย่อ 4 ตัวแปรด้วยมือ เป็นงานของ **Karnaugh Map ซึ่งเป็นเนื้อหา Week 5** ที่ยังไม่ได้เรียน

หัวใจของข้อนี้ — 1 บรรทัด

"4 อินพุต → เท่ากันได้แบบเดียวคือ 2 ต่อ 2 → $X = 0$ แค่ 6 แถว → POS 6 วงเล็บสั้นกว่า SOP 10 พจน์ชัดเจน"

11 เฉลย Assignment #4 ข้อ 6 (a) — วงจร NAND 5 ตัว

ชั้น ① โจทย์ต้นฉบับ

Assignment #4 ข้อ 6 — Kallapapruk Wichianrut · EES271-4

> Redraw the circuits below and develop the output expressions using appropriate dual symbols.

แปล: วาดวงจรข้างล่างใหม่ และหานิพจน์เอาต์พุต โดยใช้**สัญลักษณ์คู่ (dual symbol) ที่เหมาะสม**ครับ — คำว่า `appropriate` สำคัญมาก แปลว่า**ต้องเลือกหน้าตาให้ bubble ชนกันพอดีทุกจุด**

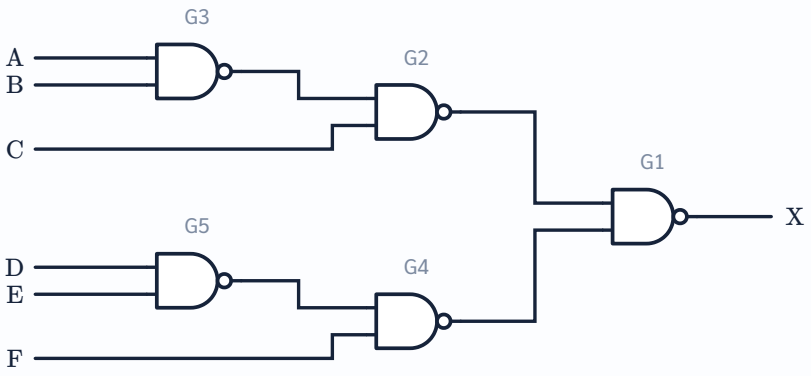
โจทย์ซ่อนอะไรไว้

① **คำสั่งมีสองงานในประโยคเดียว** — `redraw` (วาดใหม่) และ `develop the output expressions` (หานิพจน์) **ต้องทำทั้งสองอย่าง** ตอบมาแต่นิพจน์โดยไม่วาดรูปได้ครึ่งคะแนนครับ

② **ตัวแปร C กับ F วิ่งเข้าเกตชั้นในโดยตรง** — ไม่ได้ผ่านเกตตัวไหนมาก่อน ดังนั้น**ไม่มี bubble มาชนให้หักล้าง** นี่คือจุดที่ต้องระวังที่สุดของทั้งข้อ (กับดักข้อ ③ ของหัวข้อ 7 ที่พี่เตือนไว้)

③ **วงจรมี 5 เกต ไล่ตรง ๆ จะได้ขีดซ้อน 3 ชั้น** — ถ้าไม่ใช้ dual symbol จะเสียเวลาแตก De Morgan หลายบรรทัดครับ

วงจรในโจทย์ — NAND ทั้ง 5 ตัว (พี่วาดใหม่ด้วยไลบรารีเกต)



วงจรข้อ 6 (a) ตามต้นฉบับ — เกตทั้ง 5 ตัวเป็น NAND (รูป AND มี bubble ที่ขาออก) และ**ไม่มี bubble ที่ขาเข้าเลย**ครับ

อ่านวงจรให้ชัดก่อนลงมือ — ใครต่อกับใคร

G_3 รับ A กับ B · G_2 รับ**เอาต์พุตของ** G_3 (ขาบน) กับ C (ขาล่าง) · G_5 รับ D กับ E · G_4 รับ**เอาต์พุตของ** G_5 (ขาบน) กับ F (ขาล่าง) · G_1 รับเอาต์พุตของ G_2 (ขาบน) กับ G_4 (ขาล่าง)

ทุกเกตเป็น NAND ทั้ง 5 ตัว · bubble อยู่ที่ขาออกอย่างเดียว · ไม่มี bubble ที่ขาเข้าเลย — ยืนยันจากรูปในโจทย์แล้วครับ

ชั้น ② เตรียมก่อนลงมือ

เลือกวิธีไหน และทำไม

วิธีที่เลือก: เริ่มจากเกตขวาสุด (G_1) วาดเป็น negative-OR แล้วไล่ย้อนเข้าไปให้ bubble ชนกันทุกจุด

ทำไมเริ่มจาก G_1: เพราะ**เอาต์พุตคือสิ่งที่เราอยากอ่าน**ครับ ถ้าให้เกตสุดท้ายทำหน้าที่ OR เราก็จะได้นิพจน์ทรง SOP ซึ่งอ่านง่ายที่สุด (ทรง SOP คือ “ก้อน ๆ บวกกัน” ซึ่งตรงกับที่ตาเราอ่านจากซ้ายไปขวาพอดี)

กฎที่จะใช้ (สไลด์หน้า 60): ทุกจุดเชื่อมต้องเป็น **bubble-to-bubble** หรือ **nonbubble-to-nonbubble** · และ **De Morgan II** ($\overline{AB} = \overline{A} + \overline{B}$) คือเหตุผลว่าทำไมสลับหน้าตาได้

ชั้น ③ ไล่ทีละขั้นพร้อมภาพ

1



ขั้น 1 — G_1 วาดใหม่เป็น negative-OR

NAND กับ negative-OR เป็นเกตตัวเดียวกัน **เพราะ** De Morgan II ครับ เลือกหน้าตา negative-OR เพราะเราอยากให้เกตสุดท้ายอ่านว่า “บวก”

ตอนนี้ G_1 **มี bubble ที่ขาเข้าทั้งสองขา และไม่มี bubble ที่ขาออก**

2

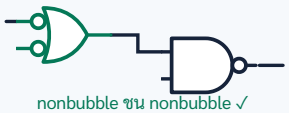


ขั้น 2 — G_2 และ G_4 ต้องคง bubble ที่ขาออกไว้

ขาเข้าของ G_1 ตอนนี้**มี bubble** ดังนั้นเกตที่ป้อนเข้ามาต้อง**มี bubble ที่ขาออก** **เพราะ** กฎบอกว่า bubble ต้องชน bubble ครับ

G_2 **กับ** G_4 **จึงวาดเป็น NAND เหมือนเดิม** ไม่ต้องเปลี่ยนอะไร

3



ขั้น 3 — G_3 และ G_5 ต้องไม่มี bubble ที่ขาออก

ขาเข้าของ G_2 (แบบ NAND) **ไม่มี bubble** ดังนั้นเกตที่ป้อนเข้ามา ต้อง**ไม่มี bubble ที่ขาออก** **เพราะ** กฎอีกครึ่งหนึ่ง คือ nonbubble ชน nonbubble ครับ

G_3 **กับ** G_5 **จึงต้องวาดเป็น negative-OR** คือรูป OR ที่มี bubble ย้ายไปอยู่ที่ขาเข้าทั้งสองขาแทน

4



ขั้น 4 — เช็คขาที่รับตัวแปรตรงจากขอบซ้าย

C วิ่งเข้าขาล่างของ G_2 ซึ่งวาดเป็น **NAND** — ขานั้น**ไม่มี bubble** **เพราะฉะนั้น** C **เข้าไปตรง ๆ ไม่ต้องกลับค่า** ✓

F ก็เหมือนกัน เข้าขาล่างของ G_4 (NAND) ตรง ๆ ครับ — **ข้อนี้โชคดีที่** C **กับ** F **ไม่โดน bubble** ต่างจากตัวอย่างในหัวข้อ 8 **แต่ต้องเช็คทุกครั้ง ห้ามเดา**

5



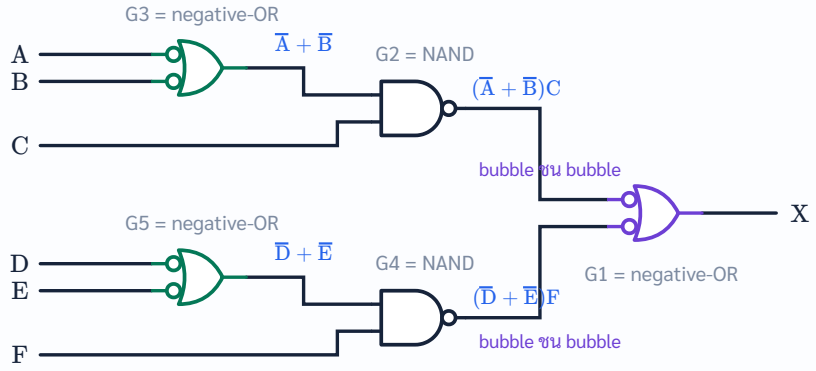
ขั้น 5 — อ่านนิพจน์จากรูปใหม่ ทีละเกต

G_3 (negative-OR) รับ A, B ที่ขาซึ่ง**มี bubble** → กลับค่าก่อนแล้วบวกกัน ได้ $\overline{A} + \overline{B}$ **เพราะ** bubble ที่ขาเข้า = ครอบขีดเฉพาะตัวแปรที่เข้าขานั้นครับ

G_5 ทำแบบเดียวกัน ได้ $\overline{D} + \overline{E}$

6	$(\overline{\mathrm{A}}+\overline{\mathrm{B}})\mathrm{C}$ $(\overline{\mathrm{D}}+\overline{\mathrm{E}})\mathrm{F}$	**ขั้น 6 — รวมทีละชั้นจนถึงเอาต์พุต** G_2 เป็น NAND — **ส่วนที่เป็น AND** ให้ $(\overline{A}+\overline{B})\cdot C$ ส่วน**bubble ที่ขาออกของมันหักล้างกับ bubble ที่ขาเข้าของ** G_1 **เพราะ** ขีดสองชั้นบนสายเดียวกันหายไป (involution) ครับ G_4 ให้ $(\overline{D}+\overline{E})\cdot F$ ในทำนองเดียวกัน แล้ว G_1 (ที่ตอนนี้อ่านเป็น OR) บวกสองก้อนนี้เข้าด้วยกัน

วงจรที่วาดใหม่ด้วย dual symbol — คำตอบของข้อ 6 (a)



ทุกจุดเชื่อมชนกันพอดี — $G_3 \to G_2$ และ $G_5 \to G_4$ เป็น nonbubble-to-nonbubble · $G_2 \to G_1$ และ $G_4 \to G_1$ เป็น bubble-to-bubble ครับ

ชั้น ④ คำตอบสุดท้ายของข้อ 6 (a)

คำตอบข้อ 6 (a)

$$\mathbf{X = (\overline{A}+\overline{B})\,C \;+\; (\overline{D}+\overline{E})\,F}$$

เขียนแบบเทียบเท่าได้อีกรูป (ใช้ De Morgan II ย้อนกลับ): $X = \overline{AB}\,C + \overline{DE}\,F$ — **สองรูปนี้ถูกทั้งคู่ครับ** แต่รูปแรกคือรูปที่**อ่านออกมาจากภาพ dual symbol ตรง ๆ** ซึ่งเป็นสิ่งที่โจทย์ต้องการ

สรุปการวาดใหม่: G_3 และ G_5 วาดเป็น **negative-OR** · G_2 และ G_4 คงเป็น **NAND** · G_1 วาดเป็น **negative-OR**

“Redrawing G3 and G5 as negative-OR and G1 as negative-OR gives X = (Ā + B̄)C + (D̄ + Ē)F directly from the diagram.”

ชั้น ⑤ ตรวจคำตอบด้วยวิธีที่สอง

ตรวจจริงด้วย tools/dig.py — เทียบกับวงจร NAND เดิมทั้ง 64 แถว

```
Q6A = "~( ~(~(A&B) & C) & ~(~(D&E) & F) )"      นิพจน์ของวงจร NAND ต้นฉบับ

same_logic(Q6A, "((~A|~B)&C) | ((~D|~E)&F)", "ABCDEF")   =  True
same_logic(Q6A, "(~(A&B)&C) | (~(D&E)&F)",   "ABCDEF")   =  True
sum of X over all 64 rows                                =  39
```

อ่านผลยังไง: บรรทัดแรกคือนิพจน์ที่ได้จากการไล่วงจรต้นฉบับตรง ๆ (NAND ซ้อนกัน 3 ชั้น) · สองบรรทัดถัดมายืนยันว่า**คำตอบที่เราอ่านจากภาพ dual symbol ให้ค่าเท่ากับวงจรเดิมทุกแถวจาก 64 แถว**ครับ

ตรวจด้วยมือ — สุ่ม 8 แถวจาก 64 แถว

A	B	C	D	E	F	วงจรเดิม (ไล่ทีละเกต)	นิพจน์ที่ได้
0	0	0	0	0	0	**0**	0
1	1	1	0	0	0	**0**	0
1	0	1	0	0	0	**1**	1
0	1	1	1	1	1	**1**	1
1	1	0	1	1	1	**0**	0
1	1	1	1	1	1	**0**	0
0	0	1	1	1	0	**1**	1
1	0	0	0	1	1	**1**	1

คอลัมน์ซ้ายของช่องคู่ = ไล่จากวงจร NAND ต้นฉบับ · คอลัมน์ขวา = แทนค่าลงนิพจน์ $(\overline{A}+\overline{B})C+(\overline{D}+\overline{E})F$ — **ตรงกันทุกแถว**ครับ

ตรวจด้วยพีชคณิตอีกทาง — ไล่จากวงจรเดิมแล้วแตก De Morgan

$G_3=\overline{AB}\ \cdot\ G_2=\overline{G_3\cdot C}=\overline{\overline{AB}\cdot C}\ \cdot\ G_5=\overline{DE}\ \cdot\ G_4=\overline{\overline{DE}\cdot F}$

$X=\overline{G_2\cdot G_4}$

$=\overline{G_2}+\overline{G_4}$ (De Morgan II)

$=\overline{\overline{\overline{AB}\cdot C}}+\overline{\overline{\overline{DE}\cdot F}}$ (แทนค่ากลับ)

$=\overline{AB}\,C+\overline{DE}\,F$ (Involution ทั้งสองก้อน)

$=(\overline{A}+\overline{B})C+(\overline{D}+\overline{E})F$ (De Morgan II อีกครั้ง) **ตรงกับที่อ่านจากภาพเป๊ะครับ** — แต่ใช้ถึง 5 บรรทัด เทียบกับการอ่านจากภาพที่ใช้ 0 บรรทัด

ชั้น ⑥ กับดักของข้อ 6 (a)

4 อย่างที่ทำให้เฉลยข้อนี้ผิด

① **วาด G_3, G_5 เป็น negative-OR แล้วลืมว่าขาเข้ามี bubble — คำตอบผิดที่จะได้:** $(A+B)C+(D+E)F$ ซึ่งไม่มีขีดเลย **ผิดทั้งข้อครับ**

② **เปลี่ยนหน้าตาของ G_2, G_4 เป็น negative-OR ด้วย** — ถ้าทำแบบนั้น bubble ที่ G_1 จะไม่มีคู่ชน **คำตอบผิดที่จะได้:** $\overline{(\overline{A}+\overline{B})+\overline{C}}+\ldots$ ซึ่งมั่วไปหมด **ต้องยึดกฎ bubble-to-bubble เป็นหลักเสมอ**

③ **ใส่ขีดให้ C กับ F ทั้งที่ขานั้นไม่มี bubble — คำตอบผิดที่จะได้:** $(\overline{A}+\overline{B})\overline{C}+\ldots$ **วิธีกัน:** ดูที่รูปว่าขานั้นมีวงกลมไหม ถ้าไม่มีก็ไม่ใส่ขีดครับ

④ **ตอบมาแต่นิพจน์ ไม่วาดรูปใหม่** — โจทย์เขียน `redraw the circuits` ไว้เป็นคำสั่งแรกเลยครับ ต้องวาด

หัวใจของข้อ 6 (a) — 1 บรรทัด

“ไล่ย้อนจากขวาไปซ้าย สลับหน้าตาเกตให้ bubble ชนกันทุกจุด แล้วนิพจน์ทรง SOP จะอ่านออกจากภาพได้ทันที”

11 เฉลย Assignment #4 ข้อ 6 (b) — วงจร NOR 5 ตัว

ชั้น ① โจทย์ต้นฉบับ

Assignment #4 ข้อ 6 (b) — Kallapapruk Wichianrut · EES271-4

> Redraw the circuits below and develop the output expressions using appropriate dual symbols.

โครงเหมือนข้อ (a) เป๊ะทุกสาย ต่างกันแค่**เกตทั้ง 5 ตัวเป็น NOR แทน NAND** ครับ (รูป OR มี bubble ที่ขาออก)

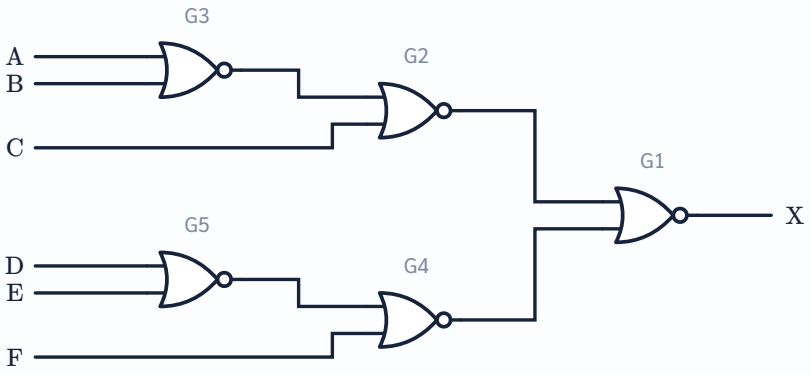
โจทย์ซ่อนอะไรไว้ในข้อ (b)

① NOR ใช้ dual symbol คนละตัวกับ NAND — NAND คู่กับ negative-**OR** แต่ NOR คู่กับ negative-**AND ถ้าจำสลับ คำตอบจะกลายเป็นทรง SOP ทั้งที่ควรเป็น POS ครับ**

② คำตอบข้อ (b) จะออกมาเป็นทรง POS (ผลคูณของผลบวก) ไม่ใช่ SOP เหมือนข้อ (a) — **ถ้าได้ทรง SOP ออกมา แปลว่าทำผิดแน่นอน**

③ ข้อ (a) กับข้อ (b) ให้คนละฟังก์ชันกัน — ตรวจด้วย dig.py แล้ว `same_logic` ได้ **False** ครับ อย่าเผลอลอกคำตอบข้อ (a) มาแล้วแค่เปลี่ยนเครื่องหมาย

วงจรในโจทย์ — NOR ทั้ง 5 ตัว



วงจรข้อ 6 (b) — สายไฟทุกเส้นเหมือนข้อ (a) เป๊ะ แต่เกตเป็น NOR (รูป OR มี bubble ที่ขาออก) ทั้ง 5 ตัว

ชั้น ② เตรียมก่อนลงมือ

เลือกวิธีไหน และทำไม

วิธีที่เลือก: เริ่มจาก G_1 วาดเป็น negative-AND แล้วไล่ย้อนเข้าไปตามกฎ bubble เดิม

ทำไมเป็น negative-AND: เพราะ dual symbol ของ NOR คือ negative-AND ครับ (De Morgan I: $\overline{A+B} = \overline{A}\,\overline{B}$) — **ไม่มีทางเลือกอื่น NOR มีแค่สองหน้าตานี้**

ผลที่คาดไว้ล่วงหน้า: เกตสุดท้ายอ่านเป็น AND → คำตอบจะเป็น**ผลคูณของก้อน = ทรง POS** ครับ ใช้เช็คตอนท้ายได้ว่าเราทำถูกทางไหม

ชั้น ③ ไล่ทีละขั้นพร้อมภาพ

1

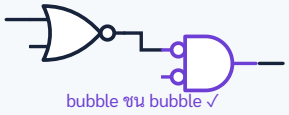


ขั้น 1 — G_1 วาดใหม่เป็น negative-AND

NOR กับ negative-AND เป็นเกตตัวเดียวกัน **เพราะ** De Morgan I ครับ — รูป AND ที่มี bubble ที่ขาเข้าทั้งสองขา และไม่มี bubble ที่ขาออก

เกตสุดท้ายทำหน้าที่ **AND** → คำตอบจะเป็นทรง POS

2

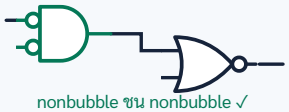


ขั้น 2 — G_2 และ G_4 คงเป็น NOR ไว้

ขาเข้าของ G_1 มี bubble → เกตที่ป้อนเข้ามาต้องมี bubble ที่ขาออก **เพราะ** กฎ bubble-to-bubble ครับ → **G_2, G_4 วาดเป็น NOR เหมือนเดิม**

bubble คู่นี้จะหักล้างกันตอนอ่านนิพจน์

3



ขั้น 3 — G_3 และ G_5 วาดเป็น negative-AND

ขาเข้าของ G_2 (แบบ NOR) ไม่มี bubble → เกตที่ป้อนเข้ามาต้องไม่มี bubble ที่ขาออก **เพราะ** กฎ nonbubble-to-nonbubble ครับ

G_3 อ่านได้ว่า $\overline{A} \cdot \overline{B}$ และ **G_5 อ่านได้ว่า $\overline{D} \cdot \overline{E}$**

4



ขั้น 4 — เช็คขาที่รับ C และ F

C เข้าขาล่างของ G_2 ซึ่งวาดเป็น **NOR** — ขาเข้าของ NOR **ไม่มี bubble เพราะฉะนั้น C เข้าไปตรง ๆ ไม่ต้องกลับค่า** ✓ · F ก็เหมือนกันครับ

ต้องเช็คทุกครั้งจริง ๆ เพราะถ้าเราเลือกวาด G_2 เป็น negative-AND แทน C จะกลายเป็น $\overline{C}$ ทันที — **การเลือกหน้าตาเกตมีผลกับคำตอบโดยตรง**

5

$\overline{\mathrm{A}}\,\mathrm{B} + \mathrm{C}$

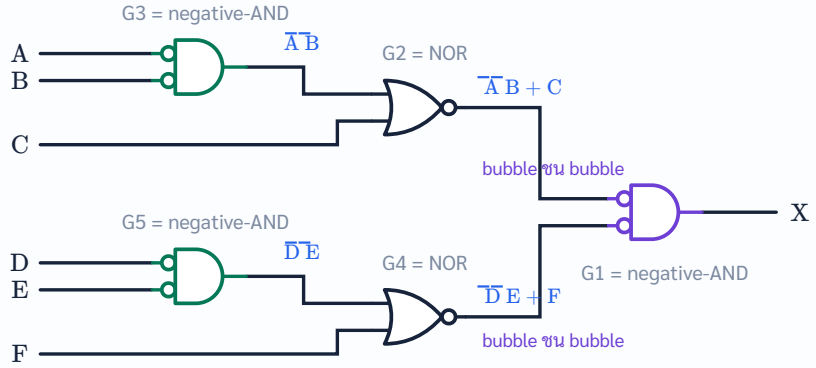
$\overline{\mathrm{D}}\,\mathrm{E} + \mathrm{F}$

ขั้น 5 — รวมทีละชั้น

G_2 เป็น NOR — **ส่วนที่เป็น OR** ให้ $\overline{A}\,\overline{B} + C$ ส่วน bubble ที่ขาออกหักล้างกับ bubble ที่ขาเข้าของ G_1 **เพราะ** involution ครับ

G_4 ให้ $\overline{D}\,\overline{E} + F$ แล้ว G_1 (ที่ตอนนี้อ่านเป็น AND) **คูณ**สองก้อนนี้เข้าด้วยกัน

วงจรที่วาดใหม่ด้วย dual symbol — คำตอบของข้อ 6 (b)



เกตสุดท้ายอ่านเป็น AND → คำตอบเป็น**ผลคูณของสองก้อน = ทรง POS** ตามที่คาดไว้ครับ

ชั้น ④ คำตอบสุดท้ายของข้อ 6 (b)

คำตอบข้อ 6 (b)

$$\mathbf{X = (\overline{A}\,\overline{B} + C)\,(\overline{D}\,\overline{E} + F)}$$

เขียนแบบเทียบเท่าได้อีกรูป (ใช้ De Morgan I ย้อนกลับ): $X = (\overline{A+B}+C)(\overline{D+E}+F)$ — ถูกทั้งคู่ครับ แต่รูปแรกคือรูปที่อ่านออกมาจากภาพ dual symbol ตรง ๆ

สรุปการวาดใหม่: G_3 และ G_5 วาดเป็น **negative-AND** · G_2 และ G_4 คงเป็น **NOR** · G_1 วาดเป็น **negative-AND**

"Redrawing G3, G5 and G1 as negative-AND gives X = (ĀB̄ + C)(D̄Ē + F) directly from the diagram."

ชั้น ⑤ ตรวจคำตอบด้วยวิธีที่สอง

ตรวจจริงด้วย tools/dig.py

```
Q6B = "~( ~(~(A|B) | C) | ~(~(D|E) | F) )"      นิพจน์ของวงจร NOR ต้นฉบับ

same_logic(Q6B, "((~A&~B)|C) & ((~D&~E)|F)", "ABCDEF")  =  True
same_logic(Q6B, "(~(A|B)|C) & (~(D|E)|F)",    "ABCDEF")  =  True
same_logic(Q6A, Q6B, "ABCDEF")                           =  False  (a) กับ (b) คนละฟังก์ชัน
sum of X over all 64 rows                                =  25
```

บรรทัดที่ 3 สำคัญมาก — ยืนยันว่าข้อ (a) กับ (b) **ไม่ใช่ฟังก์ชันเดียวกัน** ทั้งที่สายไฟเหมือนกันเป๊ะ ต่างแค่ชนิดเกตครับ

ตรวจด้วยมือ — สุ่ม 8 แถวเดียวกับข้อ (a) เพื่อเทียบกันได้

A	B	C	D	E	F	วงจรเดิม (ไล่ทีละเกต)	นิพจน์ที่ได้
0	0	0	0	0	0	1	1
1	1	1	0	0	0	1	1
1	0	1	0	0	0	1	1
0	1	1	1	1	1	1	1
1	1	0	1	1	1	0	0
1	1	1	1	1	1	1	1
0	0	1	1	1	0	0	0
1	0	0	0	1	1	0	0

ลองเทียบกับตารางของข้อ (a) ดูครับ — **แถวเดียวกันให้ผลต่างกันหลายแถว** ยืนยันว่าเป็นคนละฟังก์ชัน

ตรวจด้วยพีชคณิตอีกทาง

$G_3 = \overline{A+B}\ \cdot\ G_2 = \overline{G_3 + C}\ \cdot\ G_5 = \overline{D+E}\ \cdot\ G_4 = \overline{G_5 + F}$

$X = \overline{G_2 + G_4} = \overline{G_2} \cdot \overline{G_4}$ (De Morgan I)

$= (G_3 + C)(G_5 + F)$ (Involution — ขีดสองชั้นหักล้าง)

$= (\overline{A+B}+C)(\overline{D+E}+F) = (\overline{A}\,\overline{B}+C)(\overline{D}\,\overline{E}+F)$ (De Morgan I อีกครั้ง) **ตรงกับที่อ่านจากภาพครับ**

ชั้น ⑥ กับดักของข้อ 6 (b)

4 อย่างที่ทำให้เฉลยข้อนี้ผิด

① **ใช้ negative-OR กับ NOR — ผิดครับ** NOR คู่กับ negative-**AND** เท่านั้น **คำตอบผิดที่จะได้:** ทรง SOP ซึ่งผิดตั้งแต่หน้าตาแล้ว **วิธีกัน:** ท่องว่า “**NAND ↔ neg-OR · NOR ↔ neg-AND**” — **ตัวอักษรตรงข้ามกันเสมอ**

② **ลืมว่า negative-AND มี bubble ที่ขาเข้า — คำตอบผิดที่จะได้:** $(AB + C)(DE + F)$ ซึ่งไม่มีขีดเลย

③ **เขียน $\overline{A}\,\overline{B}$ เป็น $\overline{AB}$ — คนละความหมายครับ** $\overline{A}\,\overline{B}$ คือ “กลับค่าแยกกันแล้วคูณ” ส่วน $\overline{AB}$ คือ “คูณก่อนแล้วกลับค่า” (อันหลังเท่ากับ $\overline{A} + \overline{B}$) — **ต้องเขียนขีดสองเส้นแยกกันให้ชัด**

④ **ลอกคำตอบข้อ (a) มาแล้วเปลี่ยนแค่ $+$ เป็น $\cdot$** — บังเอิญข้อนี้ทำแบบนั้นแล้ว**ได้คำตอบผิด** เพราะขีดอยู่คนละตำแหน่งกันครับ ต้องไล่ใหม่ทั้งข้อ

หัวใจของข้อ 6 (b) — 1 บรรทัด

“NOR คู่กับ negative-AND → เกตสุดท้ายอ่านเป็น AND → คำตอบเป็นทรง POS $X = (\overline{A}\,\overline{B} + C)(\overline{D}\,\overline{E} + F)$”

เทียบสองข้อข้างกัน — สายไฟเหมือนกัน แต่คนละฟังก์ชัน

หัวข้อ	ข้อ 6 (a) — NAND	ข้อ 6 (b) — NOR
dual symbol ที่ใช้	negative-OR	negative-AND
สูตรที่รองรับ	De Morgan II: $\overline{AB} = \overline{A} + \overline{B}$	De Morgan I: $\overline{A + B} = \overline{A}\,\overline{B}$
G_3, G_5 **วาดเป็น**	negative-OR	negative-AND
G_2, G_4 **วาดเป็น**	NAND (คงเดิม)	NOR (คงเดิม)
G_1 **วาดเป็น**	negative-OR (อ่านเป็น OR)	negative-AND (อ่านเป็น AND)
ทรงของคำตอบ	**SOP** (ก้อนบวกกัน)	**POS** (ก้อนคูณกัน)
คำตอบ	$X = (\overline{A} + \overline{B})C + (\overline{D} + \overline{E})F$	$X = (\overline{A}\,\overline{B} + C)(\overline{D}\,\overline{E} + F)$
C, F **มีขีดไหม**	ไม่มี	ไม่มี
จำนวนแถวที่ได้ 1 (จาก 64)	39	25

เทียบของเก่า · นี่คือ EP05 + EP06 + EP08 มารวมกันในข้อเดียว

EP05 ให้สัญลักษณ์เกตและความคิดเรื่อง dual symbol · **EP06** ให้ De Morgan ทั้งสองข้อและ involution · **EP08 หัวข้อ 8** ให้กฎ bubble-to-bubble

ข้อ 6 นี้ใช้ทั้งสามอย่างพร้อมกันครับ — ถ้าน้องทำข้อนี้ได้เอง แปลว่าน้องเข้าใจ Week 3 กับ Week 4 ทั้งหมดแล้วจริง ๆ

12 กับดักรวม · สรุปทั้ง 8 EP · เช็คลิสต์เตรียม Midterm

กับดักรวมของ EP08 — 10 ข้อที่ทำให้เสียคะแนนมากที่สุด

#	ทำผิดแบบนี้	ที่ถูกต้องคือ	วิธีกันพลาด
1	ไล่วงจรในหัว ไม่เขียนกำกับบนสาย	เขียนนิพจน์ลงบนสายทุกเส้น	ใช้ดินสอเขียนจริงบนกระดาษข้อสอบ — **สายที่ยังไม่มีป้ายคือสายที่ยังไม่ได้คิด**
2	ลืมวงเล็บตอนผลบวกเข้า AND	$(A+B)C \neq A+BC$	ครอบวงเล็บให้นิพจน์ที่รับมาจากสายก่อนเสมอ แล้วค่อยคูณ
3	ขีดของ NAND คลุมแค่ตัวแรก	$\overline{PQ}$ — ขีดคลุมผลคูณทั้งก้อน	ลากขีดให้ยาวคลุมทุกอย่างในวงเล็บก่อนเขียนตัวถัดไป
4	ใส่อินเวอร์เตอร์ที่ขาเข้าแค่ขาเดียวแล้วบอกว่าแปลงเกตแล้ว	ต้องใส่**ครบทุกขาเข้า** กฎถึงจะใช้ได้	นับขาให้ครบก่อนสรุปว่าได้เกตอะไร
5	จำว่า OR สร้างจาก NAND 2 ตัว	ต้อง **3 ตัว** (กลับค่า 2 ขา + รวม 1)	ท่องตารางสรุปหัวข้อ 5 ให้ขึ้นใจ: 1-2-3-4 / 4-5
6	จำว่า NOR 5 ตัวได้ XOR	NOR 5 ตัวได้ **XNOR** · XOR ต้อง 6 ตัว	ท่อง **“NOR ได้ตัวที่มี N ก่อน”**
7	ใส่ bubble วงเดียวตอนแปลงวงจร	ต้องใส่**เป็นคู่**เสมอ (involution)	นับวงกลมบนสายทุกเส้น ต้องเป็นเลขคู่
8	ลืม bubble ที่ค้างบนขาที่รับตัวแปรตรงจากขอบซ้าย	ตัวแปรนั้นต้องใส่ขีดบน	ไล่ดู**ทุกขา**ที่ต่อจากขอบซ้ายว่ามีวงกลมไหม
9	จับคู่ dual symbol ผิด	NAND ↔ neg-**OR** · NOR ↔ neg-**AND**	ท่อง “ตัวอักษรตรงข้ามกันเสมอ” — N**AND** คู่กับ neg-**OR**
10	เขียน POS โดยไม่กลับค่าตัวแปร	แถวที่เป็น 0 → ตัวเปล่า · เป็น 1 → ใส่ขีด	เขียนคำว่า **“กลับ”** ไว้ข้างหัวคอลัมน์ก่อนเริ่มอ่าน POS ทุกครั้ง

สรุปหน้าเดียวของ EP08

EP08 — วิเคราะห์วงจร · gate conversion · universal gates

วิเคราะห์วงจร 4 ขั้น

① หาอินพุตที่ขอบซ้าย → ② หาเกตที่ขาเข้ารู้ค่าครบแล้ว → ③ **เขียนนิพจน์กำกับบนสาย** → ④ วนจนถึงเกตขวาสุด · **AND = คูณ · OR = บวก · bubble = ขีดบน**

สี่ทรงมาตรฐาน

AND-OR $= AB + CD$ (ทรงของ SOP) · **AND-OR-Invert** $= \overline{AB + CD}$ (ทรงของ POS) · **XOR** $= A\overline{B} + \overline{A}B$ · **XNOR** $= \overline{A}\,\overline{B} + AB$

gate conversion ด้วย NOT

ใส่อินเวอร์เตอร์ที่**ขาเข้าทุกขา** → NOR กลายเป็น AND · NAND กลายเป็น OR · OR กลายเป็น NAND · AND กลายเป็น NOR — **ทั้งหมดคือ De Morgan บรรทัดเดียว**

★ NAND universal

NOT 1 · AND 2 · OR 3 · NOR 4 · XOR 4 · XNOR 5 ตัว · เหตุผล: NAND มี “คูณ + ขีด” ครบ แล้ว **De Morgan II** เปลี่ยนคูณเป็นบวกให้

★ NOR universal

NOT 1 · **OR 2** · **AND 3** · **NAND 4** · **XNOR 5** · **XOR 6** ตัว · เหตุผล: NOR มี “บวก + ขีด” ครบ แล้ว **De Morgan I** เปลี่ยนบวกเป็นคูณให้

bubble คู่ (double bubble)

วางวงกลม 2 วงบนสายเดียวกัน → ค่าไม่เปลี่ยน (**involution**) → เกตซ้ายดูดวงซ้าย เกตขวาดูดวงขวา → **SOP กลายเป็น NAND ล้วน · POS กลายเป็น NOR ล้วน โดยใช้เกตเท่าเดิม**

dual symbol

NAND ↔ negative-OR ($\overline{AB} = \overline{A} + \overline{B}$) · **NOR ↔ negative-AND** ($\overline{A + B} = \overline{A}\,\overline{B}$) · กฎ: ทุกจุดเชื่อมต้อง **bubble-to-bubble** หรือ **nonbubble-to-nonbubble**

คำตอบ Assignment #4 ข้อ 4–6

ข้อ 4: $\sum m(3,5,6,7) = \prod M(0,1,2,4)$ · ย่อได้ $AB + AC + BC$

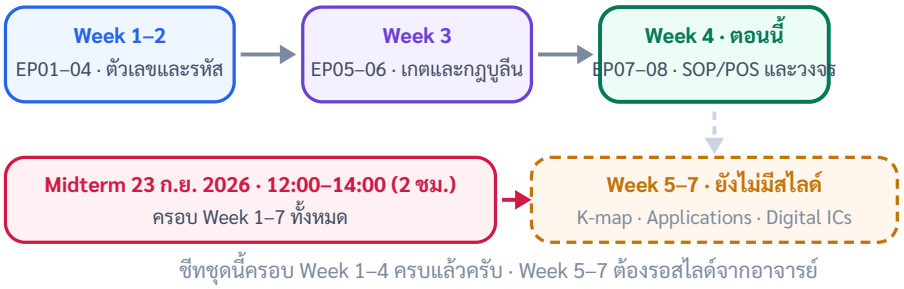
ข้อ 5: $\sum m(0,1,2,4,7,8,11,13,14,15) = \prod M(3,5,6,9,10,12)$ — **POS สั้นกว่า**

ข้อ 6a: $X = (\overline{A} + \overline{B})C + (\overline{D} + \overline{E})F$ · **ข้อ 6b:** $X = (\overline{A}\,\overline{B} + C)(\overline{D}\,\overline{E} + F)$

★ สรุปรวมทั้ง 8 EP — แผนที่ Week 1–4 หน้าเดียว

นี่คือเล่มสุดท้ายของชุด Week 1–4 ครับ พี่สรุปให้ทั้ง 8 EP อยู่ในหน้าเดียว **ใช้ทวนก่อนสอบได้เลย — ถ้าอ่านบรรทัดไหนแล้วนึกวิธีทำไม่ออก ให้ย้อนกลับไปอ่าน EP นั้น**

EP	ชื่อตอน	ต้องทำอะไรได้	สูตร/ตัวเลขที่ต้องจำ
01	**ดิจิทัล + เลขฐาน**	แปลงฐาน 10↔2↔8↔16 · จับกลุ่ม 3 บิต / 4 บิต · เลขทศนิยมฐานสอง	หารสั้นเก็บเศษ (จำนวนเต็ม) · คูณสองเก็บตัวทด (ทศนิยม)
02	**เลขคณิต + เลขมีเครื่องหมาย**	บวก ลบ คูณ หารฐานสอง · 1's / 2's complement · sign-magnitude	2's = 1's + 1 · พิสัย n บิต 2's: -2^{n-1} ถึง $2^{n-1}-1$
03	**Floating point**	IEEE 32 บิต (1 + 8 + 23) · แปลงไป-กลับ · overflow	bias 127 · $(-1)^S \times 1.M \times 2^{E-127}$
04	**รหัส + ตรวจ error**	BCD · Gray code · ASCII · parity · CRC · Hamming	Gray: XOR บิตข้างเคียง · Hamming: พาริตีอยู่ตำแหน่ง 2^k
05	**ลอจิกเกต 7 ตัว**	สัญลักษณ์ · ตารางค่าความจริง · นิพจน์ · timing diagram ของทุกเกต	AND=อนุกรม · OR=ขนาน · XOR=ต่างกันได้ 1
06	**พีชคณิตบูลีน**	ย่อนิพจน์พร้อมชื่อกฎทุกบรรทัด · แตกขีดซ้อนขีด · consensus	**De Morgan I** $\overline{A+B}=\overline{A}\,\overline{B}$ · **II** $\overline{AB}=\overline{A}+\overline{B}$
07	**Minterm + SOP/POS**	ตาราง → SOP (แถว 1) · ตาราง → POS (แถว 0 + กลับค่า) · นิพจน์ → ตาราง	n ตัวแปร = 2^n แถว · จำนวน minterm + maxterm = 2^n
08	**วิเคราะห์วงจร + universal gates**	ไล่นิพจน์ทีละเกต · แปลงชนิดเกต · สร้างทุกเกตจาก NAND/NOR · dual symbol	NAND: 1-2-3-4-4-5 · NOR: 1-2-3-4-5-6 · bubble = ขีดบน



ชีทชุดนี้ครอบ Week 1–4 ครบแล้วครับ · Week 5–7 ต้องรอสไลด์จากอาจารย์

ภาพรวมทั้งวิชาถึงตอนนี้ — กล่องเส้นประคือส่วนที่**ยังทำไม่ได้เพราะยังไม่มีสไลด์**ครับ

★ เช็คลิสต์เตรียม Midterm — ติ๊กทีละข้อ

ข้อมูลการสอบ

วันสอบ: 23 กันยายน 2026 · เวลา 12:00–14:00 (2 ชั่วโมง) · ครอบเนื้อหา Week 1–7 · คิดเป็น 35% ของคะแนนทั้งวิชา

2 ชั่วโมงสำหรับ 7 สัปดาห์แปลว่าเวลาน้อยมากครับ — ข้อที่ต้องกดตารางค่าความจริง 16 แถวใช้เวลาราว 4–5 นาที ดังนั้น**ต้องซ้อมให้เร็วจนไม่ต้องคิด** ไม่ใช่แค่ “ทำได้”

ส่วนที่ 1 — ทักษะที่ต้องทำได้โดยไม่ต้องเปิดสมุด (Week 1–4)

ติ๊ก	EP	ทำสิ่งนี้ได้ไหม
☐	01	แปลงเลขฐานสิบเป็นฐานสอง/แปด/สิบหก และย้อนกลับ **ภายใน 1 นาทีต่อข้อ**

ติ๊ก	EP	ทำสิ่งนี้ได้ไหม
☐	01	แปลง binary ↔ octal ↔ hex ด้วยการจับกลุ่ม 3 บิต / 4 บิต โดยไม่ผ่านฐานสิบ
☐	02	บวก-ลบ-คูณ-หารฐานสอง โดยเขียนตัวทด/ตัวยืมครบทุกหลัก
☐	02	หา 2's complement และบอกค่าฐานสิบของเลขมีเครื่องหมาย n บิต
☐	02	บอกพิสัยของเลข n บิตในทั้งสามระบบ (sign-magnitude / 1's / 2's)
☐	03	แปลงเลขทศนิยมเป็น IEEE 32 บิต และย้อนกลับ (จำ bias 127 ได้)
☐	03	บอกได้ว่าเมื่อไหร่เกิด overflow ในการบวกเลข 2's complement
☐	04	แปลง BCD และบอกว่าทำไม 1010–1111 ใช้ไม่ได้
☐	04	แปลง binary ↔ Gray code ไป-กลับ
☐	04	คำนวณ CRC (หาร XOR ทุกรอบ) และตรวจว่าข้อมูลที่รับมาผิดไหม
☐	04	เข้ารหัส Hamming และหาตำแหน่งบิตที่ผิดได้
☐	05	วาดสัญลักษณ์ + ตารางค่าความจริงของเกตทั้ง 7 ตัวจากความจำ
☐	05	อ่าน timing diagram และเติมรูปคลื่นเอาต์พุตได้
☐	06	ย่อนิพจน์บูลีนพร้อม**เขียนชื่อกฎกำกับทุกบรรทัด**
☐	06	แตกขีดซ้อนขีดทีละชั้นด้วย De Morgan โดยไม่ข้ามขั้น
☐	07	สร้างตารางค่าความจริงจากนิพจน์ n ตัวแปรโดยไม่ตกแถว
☐	07	อ่าน SOP จากแถว 1 และ POS จากแถว 0 (**กลับค่าตัวแปรใน POS**)
☐	07	เขียนสัญกรณ์ $\sum m(\dots)$ และ $\prod M(\dots)$ ได้ถูก
☐	08	ไล่นิพจน์จากรูปวงจรทีละเกต พร้อมเขียนกำกับบนสายทุกเส้น
☐	08	บอกจำนวน NAND / NOR ที่ต้องใช้สร้างเกตแต่ละชนิดจากความจำ
☐	08	แปลง SOP เป็น NAND ล้วน และ POS เป็น NOR ล้วนด้วยเทคนิค bubble คู่
☐	08	วาดวงจรใหม่ด้วย dual symbol แล้วอ่านนิพจน์จากภาพตรง ๆ

ส่วนที่ 2 — Week 5–7 (ยังไม่มีสไลด์ ต้องเติมเองหลังอาจารย์สอน)

ติ๊ก	สัปดาห์	หัวข้อที่ต้องเตรียม
☐	**Week 5**	**Karnaugh Maps (K-map)** — วิธีย่อนิพจน์แบบมองภาพ · การจัดกลุ่ม 2, 4, 8 ช่อง · don't care · การย่อทั้ง SOP และ POS จาก K-map
☐	**Week 6**	**Applications ของ combinational logic** — adder / subtractor · comparator · encoder / decoder · multiplexer / demultiplexer
☐	**Week 7**	**Digital ICs** — ตระกูล TTL / CMOS · fan-out · propagation delay · noise margin

ส่วนที่ 3 — กลยุทธ์ในห้องสอบ 2 ชั่วโมง

ก่อนลงมือ 3 นาทีแรก

อ่านโจทย์ทุกข้อก่อน แล้ว**ทำข้อที่มั่นใจที่สุดก่อน**ครับ — ข้อที่ต้องกดตาราง 16 แถวให้เก็บไว้ทีหลัง เพราะกินเวลามากและพลาดง่ายเมื่อรีบ

ทุกข้อที่มีวงจร

เขียนนิพจน์กำกับบนสายทุกเส้นในกระดาษข้อสอบ อาจารย์ให้คะแนนกระบวนการด้วย และมันช่วยกันพลาดเองด้วยครับ

ทุกข้อที่ย่อนิพจน์

เขียนชื่อกฎกำกับทุกบรรทัด และถ้าจำชื่อไม่ได้ ให้เขียนสูตรกำกับแทน เช่น *"by* $A + \overline{A} = 1$*"* — ปลอดภัยกว่าเขียนชื่อผิดครับ

5 นาทีสุดท้าย

ไล่ตรวจ**เฉพาะเรื่องขีดบน**ครับ — ขีดคลุมถูกก้อนไหม · ขีดซ้อนกันหักล้างครบไหม · POS กลับค่าครบทุกตัวไหม — **ข้อสอบวิชานี้เสียคะแนนที่ขีดบนมากที่สุด**

EP ถัดไปเจออะไร · Week 5 คือ Karnaugh Maps

ชีทชุดนี้จบที่ EP08 ครับ เพราะครอบ Week 1–4 ครบแล้ว — เนื้อหาถัดไปคือ **Week 5: Karnaugh Maps** ซึ่ง**ยังทำไม่ได้ตอนนี้เพราะยังไม่มีสไลด์จากอาจารย์** พี่จะไม่ทำล้ำหน้าสไลด์เด็ดขาด เพราะเสี่ยงสอนไม่ตรงกับที่อาจารย์จะออกสอบครับ

แต่พี่บอกได้ว่า K-map จะตอบคำถามที่ EP08 ทิ้งค้างไว้: ใน EP08 ข้อ 5 เราได้ SOP ยาว 10 พจน์ แล้วพี่บอกว่า "ไม่ต้องย่อ" — **คำถามคือแล้วถ้าอยากย่อจริง ๆ ล่ะครับ จะย่อ 4 ตัวแปรด้วยมือยังไงไม่ให้พลาด** คำตอบคือ K-map ซึ่งเปลี่ยนการย่อนิพจน์ให้กลายเป็น**การมองภาพและวงกลุ่ม** แทนที่จะไล่กฎบูลีนทีละบรรทัด

Week 6–7 จะเป็น **Applications** (adder, decoder, multiplexer) และ **Digital ICs** ครับ — ทั้งสองอันต่อยอดจาก EP08 โดยตรง เพราะวงจรพวกนั้น **สร้างจากทรง AND-OR และ NAND ล้วนที่น้องเพิ่งเรียนไปทั้งหมด**

ตารางแหล่งอ้างอิง

แหล่ง	หน้า/ข้อ	ใช้ในหัวข้อไหนของ EP นี้
สไลด์ EES 271-04 Combinational Logic Circuits-2026	หน้า 31, 33	หัวข้อ 1–2 (นิยาม combinational · โจทย์วิเคราะห์วงจร 3 อัน)
สไลด์เดียวกัน	หน้า 37, 39, 43, 44	หัวข้อ 3 (AND-OR · AOI · XOR · XNOR)
สไลด์เดียวกัน	หน้า 45–46	หัวข้อ 4 (gate conversion ด้วย NOT ทั้ง 4 คู่)
สไลด์เดียวกัน	หน้า 50–52	หัวข้อ 5 (NAND universal ครบ 6 แบบ)
สไลด์เดียวกัน	หน้า 53–55	หัวข้อ 6 (NOR universal ครบ 6 แบบ)
สไลด์เดียวกัน	หน้า 59–61, 65–66	หัวข้อ 7–8 (dual symbol · กฎ bubble)
สไลด์เดียวกัน	หน้า 62–64, 67–69	หัวข้อ 8 (ตัวอย่างวงจร 5 เกต ทั้ง NAND และ NOR)
EES271-4 Assignment #4 (2026)	ข้อ 4	หัวข้อ 9 (majority ของสามอินพุต)
EES271-4 Assignment #4 (2026)	ข้อ 5	หัวข้อ 10 (จำนวน HIGH ไม่เท่ากับ LOW)
EES271-4 Assignment #4 (2026)	ข้อ 6 (a), (b)	หัวข้อ 11 (redraw ด้วย dual symbol)
Floyd, Digital Fundamentals 10th ed.	Ch. 5 (Combinational Logic Analysis)	สัญลักษณ์เกตแบบ distinctive shape · กฎ bubble-to-bubble · universal gates

แหล่ง	หน้า/ข้อ	ใช้ในหัวข้อไหนของ EP นี้
ชีท EP05–EP07 ของชุดนี้	—	callback: dual symbol (EP05) · De Morgan และ involution (EP06) · SOP/POS (EP07)
`build/tools/dig.py`	`truth_table` · `same_logic` · `minterms` · `maxterms`	ตรวจคำตอบทุกข้อในชั้น ⑤ ก่อนเขียนลงชีท

คำส่งท้าย

จบ EP08 แล้วครับ — น้องเดินทางจาก “ดิจิทัลคืออะไร” ใน EP01 มาจนถึง “สร้างวงจรทั้งวงจากเกตชนิดเดียว” ใน EP08 แล้ว **ทั้ง 8 เล่มนี้คือเนื้อหา Week 1–4 ครบถ้วน**

สิ่งที่พี่อยากให้น้องจำที่สุดจาก EP นี้มีประโยคเดียวครับ: “bubble คือขีดบน และขีดบนสองชั้นหักล้างกัน” — ทุกอย่างใน EP นี้ ตั้งแต่ gate conversion ยัน universal gate ยัน dual symbol ล้วนเป็นผลจากประโยคเดียวนี้ทั้งนั้น

เจอกันตอน Week 5 เรื่อง Karnaugh Maps ครับ — **ขอให้สอบผ่านนะครับ**