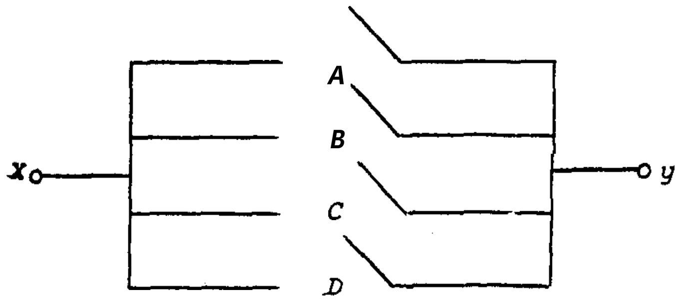
บทที่ 5
วงจรสวิทและวงจรลอกจิค
(Switching Circuit and Logic Circuit)

5.1 วงจรสวิท *(Switching Circuit)*

สวิทคือ เครื่องมือที่ติดแนบเข้าไปที่จุดในวงจรไฟฟ้า ซึ่งเครื่องมืออันนี้เราสมมติว่ามันมีสภาพอยู่ 2 อย่างคือ ปิด หรือเปิด ถ้ามันมีสภาพปิด สวิทจะยอมให้กระแสไฟไหลผ่านจุดได้ในขณะที่ถ้ามันมีสภาพเปิดกระแสไฟจะไม่สามารถไหลผ่านจุดได้เลย เราสามารถจะเขียนแสดงสวิทโดยสัญลักษณ์ —— A\\—— โดยที่ A แทนประโยคซึ่งสวิทปิดเมื่อ A มี *truth value* เป็นจริงและมีสวิทเปิดเมื่อประโยค A มี *truth value* เป็นเท็จ

เราจะกล่าวว่าจุด 2 จุด ติดต่อกันโดยวงจรสวิทก็ต่อเมื่อ จุด 2 จุดนี้เชื่อมกันด้วยขวดลวดซึ่งมีสวิทติดแนบอยู่เป็นจำนวนจำกัด *(finite)*

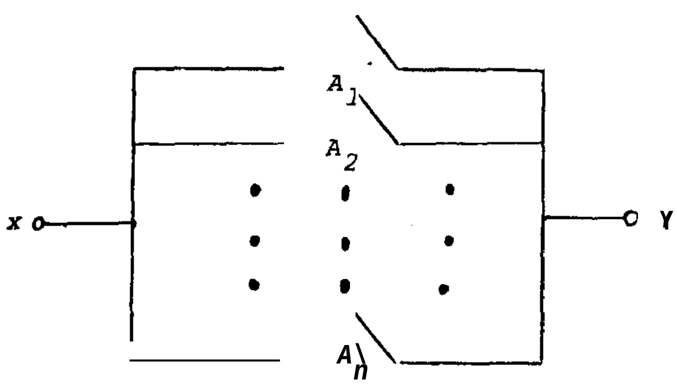
ตัวอย่าง 5.1.1



รูป 5.1.1

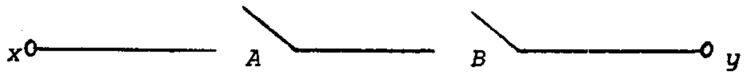
รูป 5.1.1 นี้จุด x และ y เชื่อมติดกันโดยวงจรสวิทในกรณีอย่างนี้เรากล่าวว่าสวิททั้งสี่นี้อยู่ในตำแหน่งขนานกัน *(in parallel)* และแน่นอนตามรูปนี้กระแสไฟจะไหลผ่านระหว่าง x กับ y ก็ต่อเมื่อ $A \vee B \vee C \vee D$ เป็นจริง

จากตัวอย่างนี้เราอาจจะขยายออกไปถึงกรณีที่สวิทมีจำนวน n (โดยที่ n เป็น *finite)* และเชื่อมกันในตำแหน่งขนานกัน ซึ่งกระแสไฟจะไหลผ่านวงจรก็ต่อเมื่อประโยค $A_1 \vee A_2 \vee \ldots A_n$ เป็นจริงดังรูป 5.1.2



รูป 5.1.2

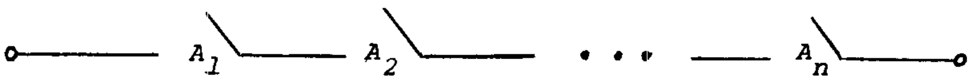
<u>ตัวอย่าง</u> 5.1.2



รูป 5.1.3

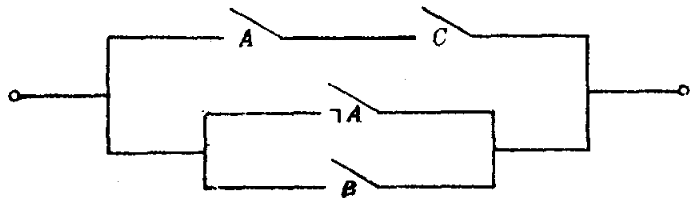
วงจรสวิทตามรูป 5.1.3 นี้กระแสไฟสามารถจะไหลระหว่างจุด x กับจุด y ก็ต่อเมื่อ A & B เป็นจริง ในกรณีอย่างนี้เรากล่าวว่าสวิทอยู่ในอนุกรม *(in series)*

จากตัวอย่างนี้เราอาจขยายออกไปถึงกรณีที่สวิทมีจำนวน n (โดยที่ n เป็น *finite)* เชื่อมต่อกันอยู่ในอนุกรมและเงื่อนไขที่กระแสไฟจะไหลผ่านวงจร ก็คือ A_1& A_2& A_3& ...& A_n เป็นจริงดังรูป 5.1.4



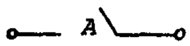
รูป 5.1.4

ตัวอย่าง ***5.1.3***

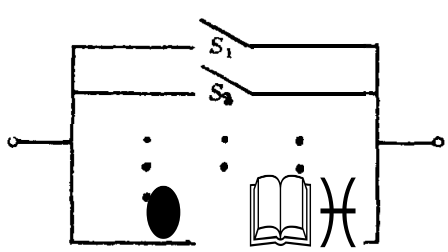


รูป 5.1.5

วงจรสวิทตามรูป 5.1.5 นี้กระแสไฟจะสามารถไหลผ่านได้ก็ต่อเมื่อ $(A \& C) \vee (\neg A \vee B)$ เป็นจริง

จากตัวอย่าง 5.1.3 นี้จะเห็นว่าเราสามารถจะรวมสวิทในตำแหน่งขนานกันกับสวิทในอนุกรมเข้าด้วยกันเป็นวงจรเดียวกัน ซึ่งวงจรประเภทนี้เราเรียกว่า วงจรสวิทอนุกรมแบบขนาน *(Series-parallel witching circuit)* ยิ่งกว่านั้น ถ้า A เป็นประโยคใด ๆ แล้ว  เป็นวงจรสวิทอนุกรมแบบขนานและถ้า $S, S_1, S_2, \ldots, S_n$ เป็นวงจรสวิทอนุกรมแบบขนาน เราสามารถจะสร้างวงจรสวิทอนุกรมแบบขนานวงจรใหม่ได้โดยแทนที่สวิทใด ๆ ใน S โดย

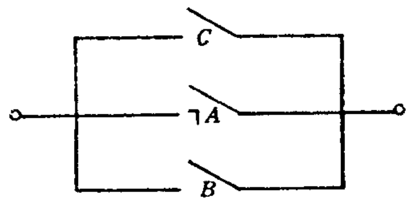
หรือมิฉะนั้นก็



ซึ่งแน่นอนเงื่อนไขที่กระแสจะไหลผ่านวงจรก็ต่อเมื่อ สามารถเขียนออกมาในรูปประโยคในลักษณะที่เป็นคอนจังชัน และดิสจังชัน *(Conjunction and disjunction)* เริ่มต้นจากประโยคแต่ละประโยคที่แทนสวิทปิด

5.2 การทำวงจรให้ง่ายลง

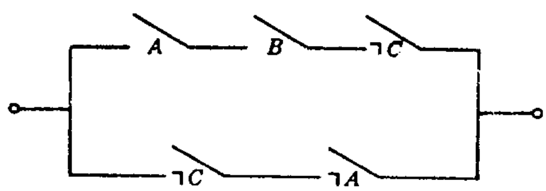
ในหัวข้อ 5.1 เงื่อนไขที่กระแสไฟจะสามารถไหลผ่านวงจรในตัวอย่าง 5.1.3 (รูป 5.1.5) คือ $(A \,\&\, C) \vee (\neg A \vee B)$ เป็นจริง ซึ่งประโยคนี้ *logically equivalent* กับ $((A \,\&\, C) \vee \neg A) \vee B$. และประโยคนี้ก็ *l gically equivalent* กับ $C \vee \neg A \vee B$ ด้วยเหตุนี้วงจรในรูป 5.1.5 ซึ่งสามารถแทนได้ด้วยวงจรในรูป 5.2.1



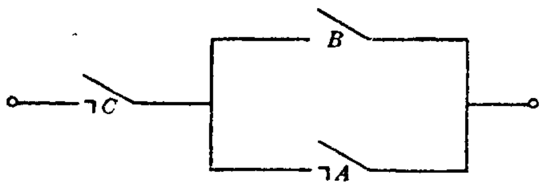
รูป 5.2.1

จะเห็นว่าวงจรในรูป 5.2.1 นี้ง่ายกว่าวงจรในรูป 5.1.5 เพราะเกี่ยวข้องกับสวิทน้อยอันกว่า

<u>ตัวอย่าง</u> 5.2.1



เงื่อนไขที่กระแสจะไหลผ่าน วงจรในรูป 5.2.2 คือ $(A \& B \& C) \vee (\neg C \& \neg A)$ เป็นจริง แต่อย่างไรก็ตามประโยคนี้ *logically equivalent* กับประโยคที่ว่า $\neg C \& [(A \& B) \vee \neg A]$ ซึ่งในทางกลับกันก็ *logically equivalent* กับ $\neg C \& (B \vee \neg A)$ ด้วยเหตุนี้เราจึงสามารถจะแทนวงจรในรูป 5.2.2 ด้วยวงจรในรูป 5.2.3 ซึ่งง่ายกว่า



รูป 5.2.3

<u>ตัวอย่าง</u> 5.2.2

คณะกรรมการชุดหนึ่งซึ่งประกอบด้วยกรรมการ 3 คน ทำการตัดสินปัญหาโดยใช้เสียงส่วนมากของกรรมการ ในการออกเสียงกำหนดให้กรรมการแต่ละคนกดปุ่มไฟฟ้าถ้าต้องการออกเสียง "เห็นด้วย" และให้อยู่เฉย ๆ เมื่อไม่เห็นด้วย เราจะสร้างวงจรสวิทเพื่อให้กระแสไฟไหลผ่านได้เฉพาะเมื่อเสียงส่วนมากของกรรมการต้องการออกเสียง "เห็นด้วย"

ให้ A แทนกรรมการคนที่หนึ่งออกเสียงเห็นด้วย

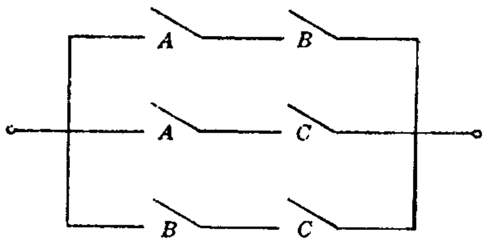
B แทนกรรมการคนที่สองออกเสียงเห็นด้วย

C แทนกรรมการคนที่สามออกเสียงเห็นด้วย

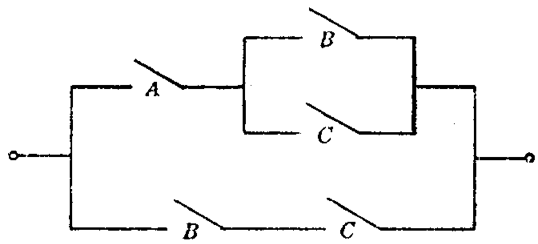
ดังนั้นเงื่อนไขที่จำเป็นและเพียงพอสำหรับเสียงส่วนมากเห็นด้วยคือ

$$(A \& B) \vee (A \& C) \vee (B \& C)$$

ซึ่งจะเขียนรูปได้ดังรูป 5.2.4 แต่อย่างไรก็ตามประโยคดังกล่าวนี้ *logically equivalent* กับประโยค $(A \;\&\; (B \vee C)) \vee (B \;\&\; C)$ ซึ่งจะสร้างวงจรได้ง่ายกว่า เราจึงเปลี่ยนมาใช้วงจรตามรูป 5.2.5 แทนได้



รูป 5.2.4



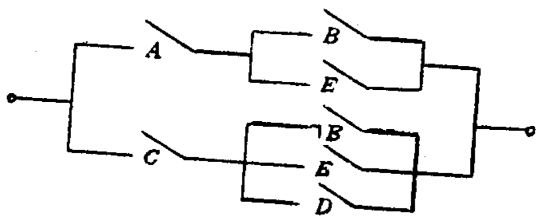
รูป 5.2.5

5.3 *Bridge Circuits*

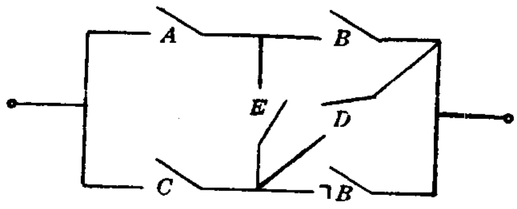
บางครั้งเราสามารถจะแทนที่วงจรอนุกรมแบบขนานได้ด้วยวงจรอื่นที่สมนัย *(equivalent circuit)* ซึ่งเป็นวงจรที่ไม่ใช่วงจรอนุกรมแบบขนาน

<u>ตัวอย่าง</u> 5.3.1

วงจรอนุกรมแบบขนานซึ่งสอดคล้องกับเงื่อนไขที่ว่า $[A \,\&\, (B \vee E)]$ $[C \,\&\, (\neg B \vee E \vee D)]$ ซึ่ง เขียนรูปได้ดังรูป 5.3.1 ซึ่งวงจรนี้สมนัย *(equivalent)* ดังวงจรในรูป 5.3.2 แน่นอน ทางที่กระแสจะไหลผ่านวงจรนี้คือ $A \,\&\, B$, $A \,\&\, E \,\&\, D$, $A \,\&\, E \,\&\, \neg B$, $C \,\&\, E \,\&\, B$, $C \,\&\, D$, $C \,\&\, \neg B$ ด้วยเหตุนี้เงื่อนไขที่กระแสไฟจะไหลผ่านวงจรนี้คือ $(A \,\&\, B) \vee (A \,\&\, E \,\&\, D) \vee (A \,\&\, E \,\&\, \neg B) \vee (C \,\&\, E \,\&\, B) \vee (C \,\&\, D) \vee (C \,\&\, \neg B)$ ซึ่ง *logically equivalent* กับ $[A \,\&\, (B \vee E)] \vee [C \,\&\, (\neg B \vee E \vee D)]$



รูป 5.3.1



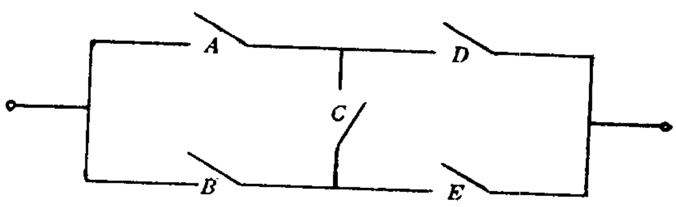
รูป 5.3.2

วงจรตามรูป 5.3.2 นี้เป็นตัวอย่างของวงจรที่ไม่ใช่วงจรอนุกรมแบบขนาน วงจรแบบนี้เราเรียกว่า *bridge circuits*

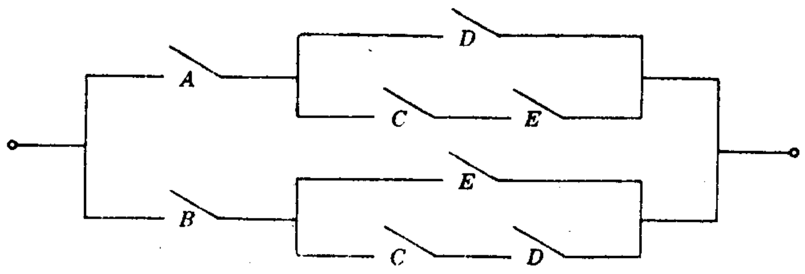
ในตัวอย่าง 5.3.1 วงจร *bridge circuits* มีสวิทจำนวนน้อยกว่าสวิทของวงจรอนุกรมแบบขนาน

<u>ตัวอย่าง</u> 5.3.2

ประโยค $(A \,\&\, \left[D \vee (C \,\&\, E)\right]) \vee \left[B \,\&\, (E \vee (C \,\&\, D))\right]$ เป็น *bridge circuit* ซึ่งเขียนรูปได้ดังรูป 5.3.3 และเป็นวงจรที่มีวงจรอนุกรมแบบขนานเป็นดังรูป 5.3.4



รูป 5.3.3

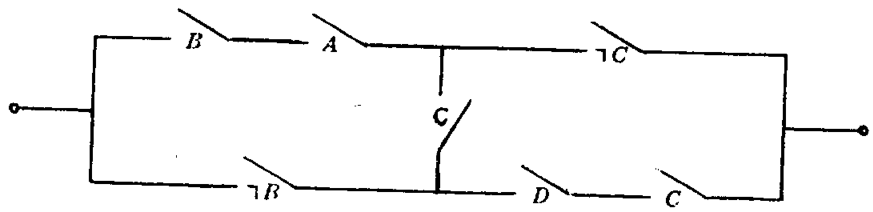


รูป 5.3.4

ขอให้สังเกตว่า *bridge circuit* ใด ๆ จะกำหนด *truth function* โดยที่รูปแบบของประโยค *truth function* จะได้จากการหาทางเดินทั้งหมดที่อาจเป็นไปได้ของกระแสผ่านวงจร

ตัวอย่างเช่น *bridge circuit* ที่แสดงดังรูป 5.3.5 จะสอดคล้องกับรูปแบบประโยคที่ว่า

$$(A \,\&\, B \,\&\, C \,\&\, D) \vee (A \,\&\, B \,\&\, \neg C) \vee (\neg B \,\&\, C \,\&\, D)$$



รูป 5.3.5

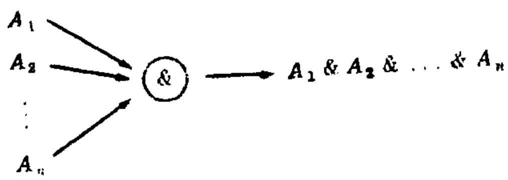
5.4 วงจรลอกจิค *(Logic Circuits)*

ขบวนการของการให้ข้อมูลเป็นบทบาทที่สำคัญที่สุดอันหนึ่งของ *digital computer* สมัยใหม่ เพื่อให้บรรลุจุดประสงค์นี้เครื่องมือแบบพิเศษจึงได้รับการประดิษฐ์ขึ้นใช้ วงจรลอกจิคเป็นเครื่องมือชนิดหนึ่งที่ประดิษฐ์ขึ้นเพื่อวัตถุประสงค์นี้

5.4.1 *and gate* &

นิยาม

> ให้ $A_1, A_2, \ldots, A_n$ โดยที่ $n \geq 2$ เป็น *input* แล้ว *and gate operation* คือ การกระทำบน $A_1, A_2, \ldots, A_n$ โดยที่ $n \geq 2$ และสร้างคอนจังชัน $A_1 \& A_2 \& A_3 \&, \ldots \& A_n$ ใช้สัญลักษณ์ (&)



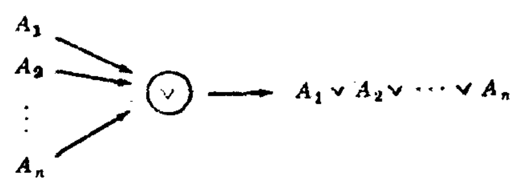
รูปที่ 5.4.1

แต่ละ *input* A_i จะมีปริมาณทางฟิสิคเรียก โวลท์ ซึ่งเราจะจำแนกความแตกต่างออกมา 2 สภาพ เขียนแทนด้วย 0 และ 1 สภาพ 1 จะปรากฏเมื่อ A_i เป็นจริงและสภาพ 0 จะปรากฏเมื่อ A_i เป็นเท็จทำนองเดียวกัน *output* ของ *and gate* จะมีสภาพที่เป็นไปได้ 2 สภาพ คือ 0 และ 1 เช่นกัน จะเป็น 1 เมื่อ $A_1 \& A_2 \& A_3 \& \ldots \& A_n$ เป็นจริงและจะเป็น 0 เมื่อ $A_1 \& A_2 \& A_3 \& \ldots \& A_n$ เป็นเท็จ

5.4.2 *or gate* (V)

นิยาม

> ถ้า $A_1, A_2, \ldots, A_n$ $(n \geq 2)$ เป็น *input*
> แล้ว *output* คือ $A_1 \vee A_2 \vee \ldots \vee A_n$
> แล้วการกระทำนี้เรียกว่า *or gate operations*
> ใช้สัญลักษณ์ (V)



รูป 5.4.2

ดังนั้นตามนิยามนี้ *output* จะมีสภาพ 1 เมื่อ A_i ใน *output* อย่างน้อยที่สุดหนึ่งประพจน์เป็น 1 (เป็นจริง) ⓥ

5.4.3 *inverter*

นิยาม

inverter (¬) คือ เครื่องมือซึ่งมี *input A* (หนึ่ง *input*) และสร้าง *output* ¬A

ดังนั้น *output* ¬A จะมีสภาพ 1 เมื่อ *input A* มีสภาพ 0

และ *output* ¬A จะมีสภาพ 0 เมื่อ *input A* มีสภาพ 1

$$A \longrightarrow (\neg) \longrightarrow \neg A$$

รูป 5.4.3

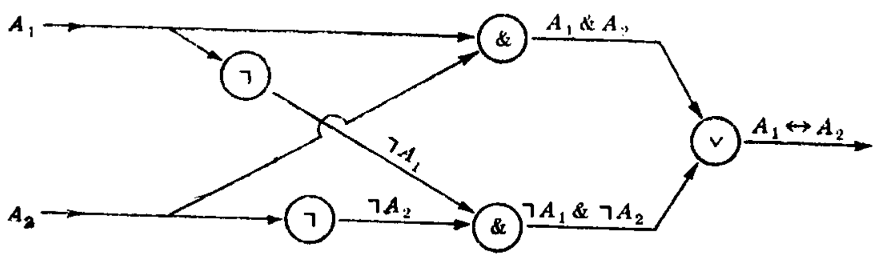
นิยาม

วงจรลอกจิค *(logic Cicuit)* คือวงจรที่สร้างจาก *inputs* หลาย ๆ อันโดยใช้ *and gate or gate, inverter* และเครื่องมือที่อาจเป็นไปได้อื่น ๆ เพื่อให้ได้ทรูยฟังก์ชันนัลโอเปอเรชัน

เครื่องมืออีเล็คโทรนิคจริง ๆ ที่ใช้สร้าง *and gate, or gate* และ *inverters* จะแปรไปตามสภาพของเทคโนโลยี ด้วยเหตุผลนี้จึงเป็นความสะดวกที่สุดที่จะไม่เอาใจใส่ *(ignore)* คำถามทาง *hardware* (เช่น ไดโอด *(diodes)*, ทรานซิสเตอร์ *(transistors)* เป็นต้น) ผู้อ่านที่สนใจจะหาอ่านได้โดยตรงจากหนังสือของภาควิชาฟิสิกค์

<u>ตัวอย่าง</u> 5.4.1

ในการสร้างวงจรลอกจิคซึ่งจะได้ *output* $A_1 \longleftrightarrow A_2$ จะทำได้ดังรูป 5.4.4

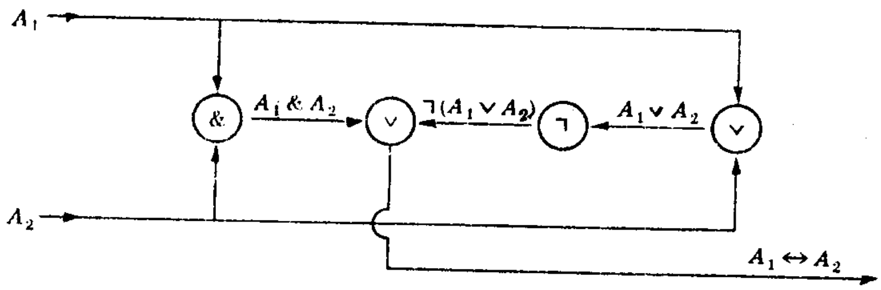


รูป 5.4.4

ขอให้สังเกตว่า $A_1 \leftrightarrow A_2$ สมนัยกับประพจน์

$(A_1 \& A_2) \vee (\neg A_1 \& \neg A_2)$ และทำนองเดียวกันก็สมนัยกับประพจน์

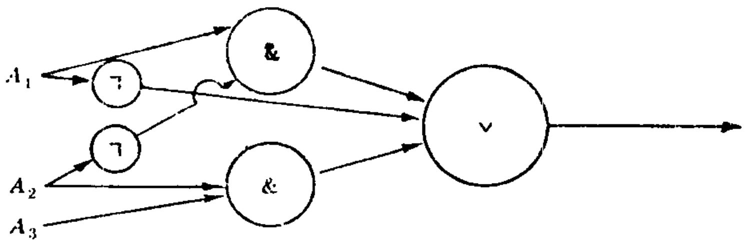
$(A_1 \& A_2) \vee \neg (A_1 \vee A_2)$ ซึ่งประพจน์หลังจะสร้างได้ง่ายกว่า ดังรูป 5.4.5



รูป 5.4.5

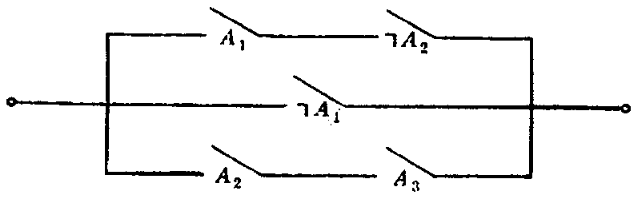
<u>ตัวอย่าง</u> 5.4.2

สร้างวงจรลอกจิคซึ่งให้ *output* $(A_1 \& \neg A_2) \vee \neg A_1 \vee (A_2 \& A_3)$



รูป 5.4.6

แทนที่จะสร้างวงจรลอจิคดังรูป 5.4.6 เราอาจจะสร้างโดยใช้วงจรอนุกรมแบบขนาน *(series - parallel circuit)* ซึ่งกระแสจะไหลผ่านเมื่อ 1 เป็นจริงดังรูป 5.4.7



รูป 5.4.7

5.5 ระบบเลขฐานสอง *(The Binary Number System)*

ปกติเราเคยชินกับการใช้เลขฐานสิบ เช่น 34,062 ซึ่งหมายถึง

$2 + 6.10 + 0.10^2 + 4.10^3 + 3.10^4$

โดยทั่ว ๆ ไปจำนวนเต็มบวกใด ๆ สามารถจะเขียนได้ในรูป

$a_0 + a_1.10 + a_2.10^2 + a_3.10^3 + \dots + a_k.10^k$

โดยที่ $0 \leq A_i \leq 9$ สำหรับ $0 \leq i \leq k$ และ $a_k > 0$

ดังนั้นจำนวนนี้จะเขียนแทนจำนวนเลขฐานสิบในรูป $a_k a_{k-1} \cdots a_2 a_1 a_0$

อย่างไรก็ตามสำหรับจำนวนเต็ม $r > 1$, ทุก ๆ จำนวนเต็มบวก n จะสามารถเขียนแทนได้รูปเดียว *(unique)* ในรูป

$a_0 + a_1.r + a_2.r^2 + \dots + a_m.r^m$

โดยที่ $0 \leq a_i \leq r - 1$ สำหรับ $0 \leq i \leq m$ และ $a_m > 0$

ในกรณีเฉพาะทุก ๆ จำนวนเต็มบวกเราสามารถจะเขียนได้ในรูปเลขฐานสอง

$a_0 + a_1.2 + a_2.2^2 + a_3.2^3 + \ldots + a_m.2^m$

โดยที่ $0 \leq a_i \leq 1$ สำหรับ $0 \leq i \leq m$ และ $a_m = 1$

<u>ตัวอย่าง</u> 5.5.1

จำนวน 23 (ในเลขฐานสิบ) จะเขียนในรูปเลขฐาน 2 ได้เป็น

10111, ie $2^4 + 2^2 + 2 + 1$

จำนวน 101 (ในเลขฐานสิบ) จะเขียนในรูปเลขฐานสอง ได้เป็น

1100101 ie $2^6 + 2^5 + 2^2 + 1$

ขบวนการในการหาเลขฐาน 2 ที่จะเขียนแทนจำนวน n คือการหากำลังสูงสุด 2^m ซึ่ง $\leq n$ ลบ 2^m ออกจาก n แล้วหากำลังสูงสุด 2^j ซึ่ง $\leq n - 2^m$

<u>ตัวอย่าง</u> 5.5.2

เลขฐานสิบ	เลขฐานสอง
1	1
2	10
3	11
4	100
5	101
6	110
7	111

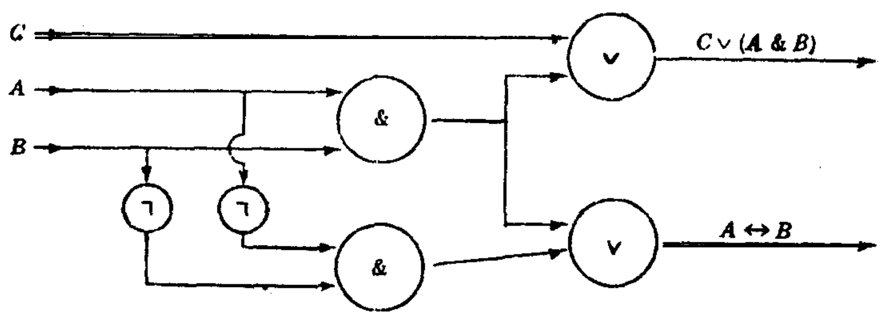
เลขฐานสิบ	เลขฐานสอง
8	1000
11	1011
16	10000
35	100011
52	110100
117	1110101

5.6 วงจรลอกจิคแบบทวิคูณเอาท์พุท

(Multiple Output Logic Circuit)

ปรากฏว่ามีวงจรลอกจิคแบบเดียวกันไปเป็นส่วนหนึ่งของวงจรลอกจิคอันอื่น นำมาซึ่งการใช้วงจรลอกจิคกับ *output* ที่มากกว่าหนึ่ง *output* ดังตัวอย่างข้างล่างต่อไปนี้

ตัวอย่าง 5.6.1



รูป 5.6.1

ตัวอย่าง 5.6.2

นำจำนวนในเลขฐานสองทั้งหมด 2 จำนวนมารวมกันโดยวิธีเดียวกับเลขฐานสิบ

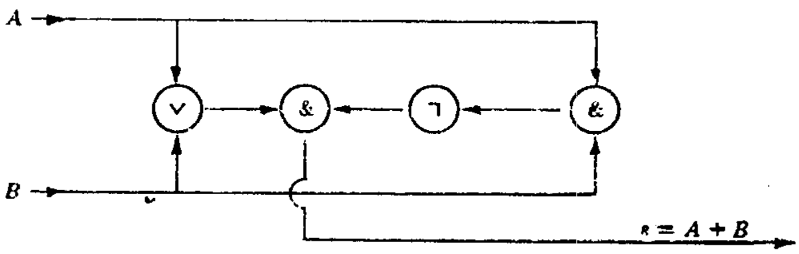
เลขฐานสอง :	100101	เลขฐานสิบ :	37
	10111		23
	111100		60

ถ้าเราจะพิจารณาเฉพาะผลบวกของเลขตัวเดียว ๆ 0 และ 1 เราจะได้ค่าของผลรวมของจำนวนเลข (S) และจำนวนทด (C) ดังต่อไปนี้

A	B	s	c
1	1	0	1
0	1	1	0
1	0	1	0
0	0	0	0

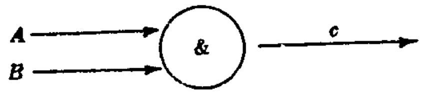
ดังนั้น s สมนัยกับ "หรือ" (or) ซึ่งเราอาจจะเขียนแทนด้วยสัญญลักษณ์ $A + B$ ในขณะที่ c สมนัยกับคอนจังชัน

ถ้าเราต้องการจะสร้างวงจรลอกจิคสำหรับ s อย่างเดียวเราจะได้ดังนี้



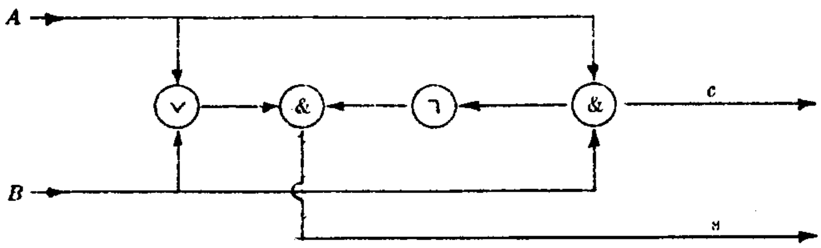
รูป 5.6.2

ในทำนองเดียวกันเราสามารถสร้างวงจรลอจิคสำหรับ C ได้ดังนี้



รูป 5.6.3

อย่างไรก็ตามเราสามารถจะรวม 2 วงจรนี้เข้าด้วยกันเป็นวงจรลอจิค แบบทวิคูณเอาท์พุท ดังรูป 5.6.4



รูป 5.6.4

วงจรตามรูป 5.6.4 มีชื่อเรียกว่า <u>*half adder*</u>

<u>ตัวอย่าง</u> 5.6.3

ถ้าเราต้องการจะบวก ตัวเลขเดียว ๆ A และ B แล้วนำผลไปรวมกับ ตัวเลขทด C จากการบวกครั้งก่อน ๆ หน้านี้ จะได้ผลดังตารางข้างล่างนี้

A	B	C	S	C
1	1	1	1	1
0	1	1	0	1
1	0	1	0	1
0	0	1	1	0
1	1	0	0	1
0	1	0	1	0
1	0	0	1	0
0	0	0	0	0

ดังนั้น S จะสมนัยกับประพจน์

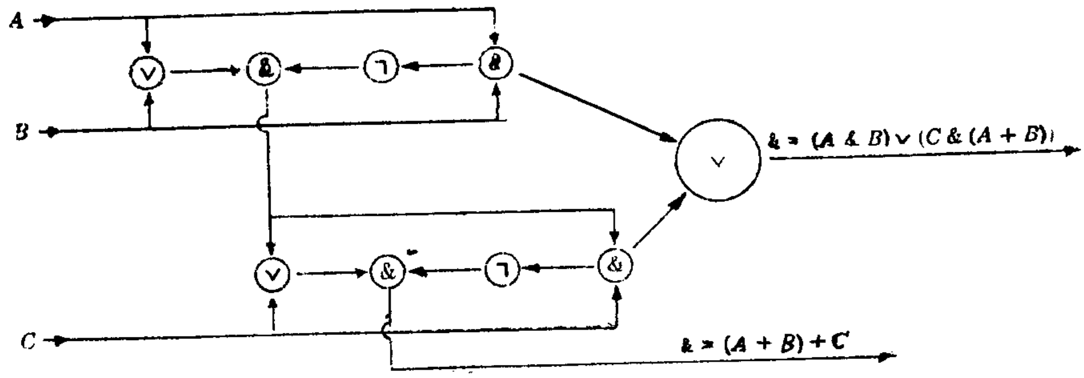
$$(C \,\&\, \neg(A + B)) \vee ((A + B) \,\&\, \neg C)$$

ซึ่ง *logically equivalent* กับ $(A + B) + C$

และเลขทด C สมนัยกับประพจน์

$$(A \,\&\, B) \vee (C \,\&\, (A + B)$$

เราสามารถจะใช้วงจรสำหรับ $A + B$ จากรูป 5.6.2 ในตัวอย่างที่ 5.6.2 และสามารถสร้างวงจรสำหรับ S และ C ในตัวอย่าง 5.6.3 ได้ดังนี้

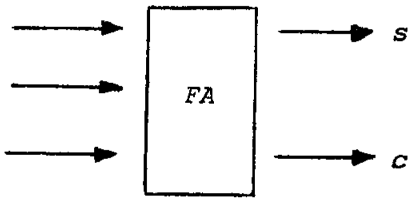


รูป 5.6.5

วงจรตามรูป 5.6.5 มีชื่อเรียกว่า <u>*full adder*</u>

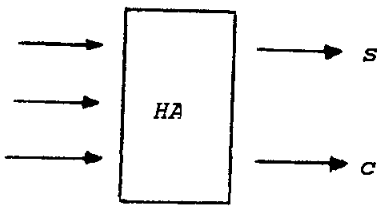
เราสามารถจะสร้างวงจรสำหรับการบวกจำนวนในเลขฐานสอง 2 จำนวนแต่ละจำนวนประกอบด้วย 3 หลัก $A_2A_1A_0$ และ $B_2B_1B_0$

ให้



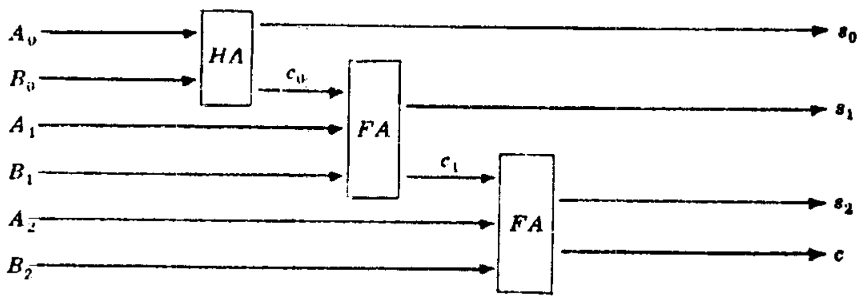
แทน *full adder*

และให้



แทน *half adder*

แล้วผลบวกของ 2 จำนวนดังกล่าวจะแทนได้ด้วยรูป 5.6.6 โดย $CS_2S_1S_0$



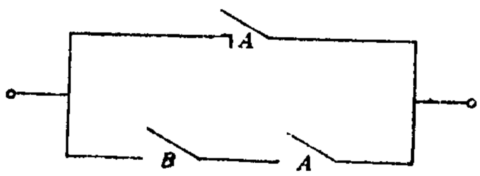
รูป 5.6.6

5.7 การหาค่าต่ำสุด *(Minimization)*

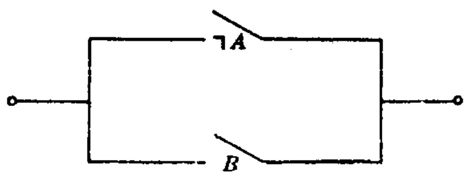
ราคาในการสร้างและใช้วงจรสวิท และวงจรลอจิคขึ้นอยู่กับสภาพของเทคโนโลยี่และเวลา แต่อย่างไรก็ตาม ณ เวลาที่กำหนดให้ วงจรบางวงจรอาจจะเสียค่าใช้จ่ายน้อยกว่าวงจรอื่น ๆ ที่ใช้แทนกันได้

<u>ตัวอย่าง</u> 5.7.1

วงจรในรูป 5.7.1 ซึ่งสมนัยกับประพจน์ $\neg A \vee (B \,\&\, A)$ จะเสียค่าใช้จ่ายแพงกว่าวงจรในรูป 5.7.2 ซึ่งสมนัยกับประพจน์ $\neg A \vee B$ ซึ่งวงจรหลังนี้ สมนัยกับวงจรในรูป 5.7.1



รูป 5.7.1



รูป 5.7.2

ปัญหาการหาค่า (ใช้จ่าย) ต่ำสุด *(minimization problem)* ประกอบด้วยการกำหนดวิธีการสำหรับหาวงจรที่ง่ายที่สุด (*ie* ที่ราคาถูกที่สุด) ซึ่งเป็นวงจรที่สมมัย *(equivalent)* กับวงจรที่กำหนดมาให้ ในตัวอย่าง 5.7.1 จะเห็นได้ชัดเจนว่า วงจรในรูป 5.7.2 เป็นวงจรที่ง่าย (ซึ่งย่อมหมายถึงราคาถูก) ที่สุดซึ่งแทน ทรูธฟังก์ชัน $\neg A \vee B$ ที่สมมัยกับวงจรในรูป 5.7.1

ดังนั้นเป็นการแน่นอนว่า สำหรับทรูธฟังก์ชันใด ๆ ที่ให้มาเราสามารถจะหาวงจรที่แทนทรูธฟังก์ชันที่ให้มานั้นและสามารถจะตรวจสอบค่าใช้จ่ายของวงจรนั้นเพื่อหาวงจรที่ให้ค่าใช้จ่ายถูกกว่าได้เสมอ

แต่วิธีการดังกล่าวนี้จะใช้ได้ดีเฉพาะกับวงจรแบบง่าย ๆ เท่านั้น แต่ถ้าเป็นวงจรที่มีตัวแปรตั้งแต่ 3 ตัวขึ้นไปวิธีการตรวจสอบแบบนี้ใช้ไม่ได้เพราะค่าใช้จ่ายจะเกี่ยวข้องกับปัจจัยอื่นด้วย ดังนั้นจึงจำเป็นต้องพิจารณาประโยชน์, ความสะดวก และรวดเร็ว

ดังนั้นการหา *minimization problem* จึงมีด้วยกันหลายแบบ

สมมติว่าเรามีทรูธฟังก์ชันใด ๆ ประพจน์ เราจะมีหลายแบบ *(form)* ในการทำ *minimization problem*

1. หา *disjunctive normal form (dnf)* ที่ "ง่ายที่สุด" ซึ่งแทนหรือสมนัยกับ ทรูธฟังก์ชันที่กำหนดมาให้

2. หาวงจร *series - parallel switching* (หรือวงจรลอกจิค) ที่ "ง่ายที่สุด" ซึ่งแทนทรูธฟังก์ชันที่กำหนดมาให้

3. หาวงจรสวิท *(series - parallel* หรือมิฉะนั้นก็ *bridge* อันใดอันหนึ่ง) ที่ "ง่ายที่สุด" ซึ่งแทนทรูธฟังก์ชันที่กำหนดมาให้

<u>ตัวอย่าง</u> 5.7.2

พิจารณาทรูธฟังก์ชันซึ่งกำหนดมาให้ดังตารางข้างล่างนี้

A	B	C	$g(A,B,C)$
T	T	T	T
F	T	T	F
T	F	T	T
F	F	T	F
T	T	F	T
F	T	F	F
T	F	F	F
F	F	F	F

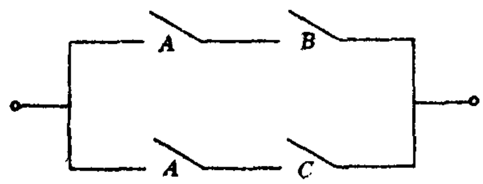
dijunction normal form ของฟังก์ชันนี้ คือ

$(A \,\&\, B \,\&\, C) \vee (A \,\&\, \neg B \,\&\, C) \vee (A \,\&\, B \,\&\, \neg C)$

ซึ่ง *logically equivalent* กับ $(A \,\&\, B) \vee (A \,\&\, C)$

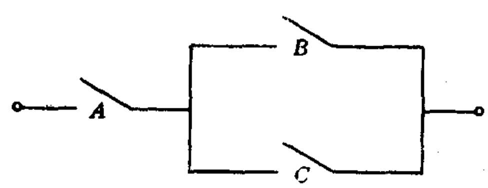
และเป็น *dnf* ที่ง่ายที่สุดของ g ดังนั้นการแก้ปัญหาแบบ 1 วงจร

ที่ใช้แทนหรือสมนัย วงจร *series - parallel* จะเป็นวงจรดังรูป 5.7.3



รูป 5.7.3

แต่อย่างไรก็ตาม รูปแบบประพจน์ที่ *logically equivalent* กับ $(A \& B) \vee (A \& C)$ คือ $A \& (B \vee C)$ ซึ่งจะได้วงจรอนุกรมแบบขนาน *(series - parallel circuit)* ที่ง่ายกว่าดังรูป 5.7.4



รูป 5.7.4

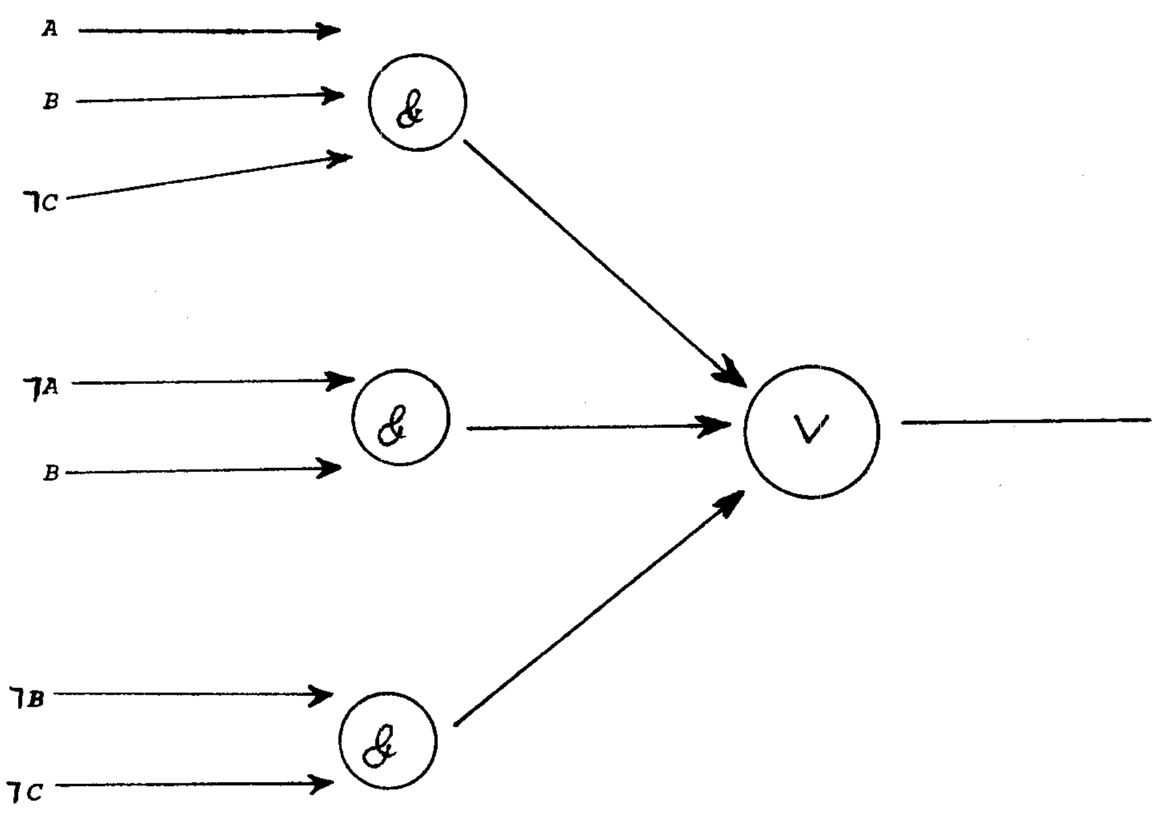
เราจะเห็นอยู่แล้วว่าวงจรในรูป 5.7.4 นี้ไม่สามารถจะหาวงจรอื่นที่ง่ายกว่ามาแทนได้ ดังนั้น *minimal dnf* ซึ่งแก้ปัญหาแบบ (1) จะไม่จำเป็นต้องเป็นคำตอบของปัญหาแบบที่ (2) นั่นคือ คำตอบของปัญหาแบบที่ (2) ไม่จำเป็นต้องเป็น *dnf*

ในตัวอย่างที่แสดงในรูป 5.3.3 เป็นตัวอย่างที่แสดงให้เห็นว่ามี *bridge circuit* ที่ง่ายกว่าวงจรอนุกรมแบบขนาน *(series -parallel circuit)* ใด ๆ ที่สมนัยกันด้วยเหตุนี้คำตอบของปัญหาแบบที่ (3) ไม่จำเป็นต้องเป็นคำตอบของปัญหาแบบที่ (2)

<u>ข้อสังเกต</u>

1. การแก้ปัญหา *minimization problem* แบบที่ (1) ต้องการพิจารณาเฉพาะ *dnf* เท่านั้น คำตอบของปัญหาจะจัดหาวงจร 2 สภาพที่เรียกว่า *and - or logic circuit* ตัวอย่างเช่น

$dnf(A \,\&\, B \,\&\, \neg C) \vee (\neg A \,\&\, B) \vee (\neg B \,\&\, \neg C)$ ซึ่งจะแทนได้ด้วยวงจรในรูป 5.7.5 ดังนี้



รูป 5.7.5

2. ในการแก้ปัญหา *minimization problem* แบบ *(II)* ในกรณีของวงจรลอกจิค เราจะต้องพิจารณารูปแบบของประพจน์ *(arbitrary statement forms)* ไม่ใช่แต่ *dnf* ในกรณีนี้ *inverter* จะถูกนับในการคำนวณ ค่าใช้จ่ายด้วย เพราะนิเกชันอาจจะใช้ไม่แต่กับตัวอักษร (คำ) แต่รวมใช้ทั้งประพจน์ด้วย แต่อย่างไรก็ตาม ถ้าเราสนใจแต่เพียงวงจรสวิทเท่านั้น การพิจารณาก็จะจำกัดอยู่เฉพาะที่รูปแบบประพจน์ซึ่งนิเกชันถูกใช้เฉพาะกับ *statement letter* เท่านั้น

3. ในการเขียนรูปแบบประพจน์ บ่อยครั้งเพื่อความสะดวกในการเขียนจะละเครื่องหมาย & เสียและเขียน $\bar{A}$ แทน $\neg A$

__ตัวอย่าง__ 5.7.3

เขียน $A\bar{B}\bar{C} \vee AB\bar{C}$ แทน $(A \mathbin{\&} \neg B \mathbin{\&} C) \vee (\neg A \mathbin{\&} B \mathbin{\&} \neg C)$

เขียน $\bar{A}\bar{B} \vee \bar{A}B\bar{C}D \vee \bar{A}C$ แทน $(\neg A \mathbin{\&} \neg B)$

$\vee (\neg A \mathbin{\&} B \mathbin{\&} \neg C \mathbin{\&} D) \vee (\neg A \mathbin{\&} C)$

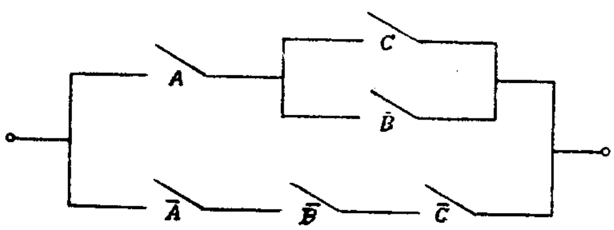
เขียน $(\bar{A} \vee B)(A \vee \bar{B} \vee \bar{C})$ แทน $(\neg A \vee B) \mathbin{\&} (A \vee \neg B \vee \neg C)$

5.8 *Don't Care Condition*

ในปัญหาหลาย ๆ ปัญหาที่เกี่ยวข้องกับการกำหนดวงจร จะมีเงื่อนไขที่แน่นอนซึ่งเป็นไม่ได้หรือไม่เป็นที่ต้องการในการทำงานของวงจร เงื่อนไขประเภทนี้เราเรียกว่า *Don't Care Conditions*

__ตัวอย่าง__ 5.8.1

ประพจน์ $A(\bar{B} \vee C) \vee \bar{A}\bar{B}\bar{C}$ มีวงจรสวิทดังรูป 5.8.1



รูป 5.8.1

ตามรูปนี้ในกรณีที่ A คือประพจน์ที่ว่า "x เป็นจำนวนเต็มคู่"

B คือประพจน์ที่ว่า "x เป็นกำลังสองสมบูรณ์"

C คือประพจน์ที่ว่า "x เป็นจำนวนเต็มที่หารลงตัวด้วย 4"

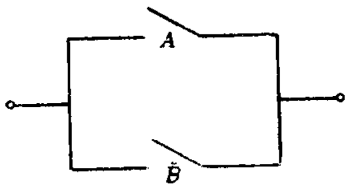
จะเห็นว่าเงื่อนไข $AB\bar{C}$, $\bar{A}\bar{B}C$, $\bar{A}BC$ ทั้งสามเป็นไปไม่ได้

ด้วยเหตุนี้จึงไม่เป็นไร ถ้าเราจะสร้างวงจรที่จะยอมให้กระแสไหลผ่านถ้ามีบางเงื่อนไขใน 3 เงื่อนไขนี้เป็นไปได้

ในกรณีเฉพาะวงจรที่แทนได้ด้วยประพจน์ในรูป

$A(\bar{B} \vee C) \vee \bar{A}\bar{B}\bar{C} \vee AB\bar{C} \vee \bar{A}BC$

จะให้ผลสำเร็จตามต้องการเช่นเดียวกับวงจรในรูป 5.8.1 แต่ประพจน์นี้ยัง *logically equivalent* กับ $A \vee \bar{B}$ ซึ่งจะให้วงจรที่ง่ายกว่าขึ้นไปอีกดังรูป 5.8.2



รูป 5.8.2

จากตัวอย่าง 5.8.1 จะเห็นได้ว่าเงื่อนไขของ *Don't Care Condition* บางทีก็ให้วงจรที่ง่ายลงไปอีก

<u>ตัวอย่าง</u> 5.8.2

ตัวเลข 0 ถึง 9 ในเลขฐานสิบสามารถเขียนแทนได้ด้วยตัวเลขในเลขฐานสองได้ดังต่อไปนี้

เลขฐาน 10	0	1	2	3	4	5	6	7	8	9
เลขฐาน 2	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

พิจารณาประโยค

A เลขตัวแรก (ทางขวามือสุด) ของเลขฐานสองเป็น 1

B เลขตัวที่สองของเลขฐานสองเป็น 1

C เลขตัวที่สามของเลขฐานสองเป็น 1

D เลขตัวที่สี่ของเลขฐานสองเป็น 1

ดังนี้

$\bar{A}\bar{B}\bar{C}\bar{D}$ แทน 0

$A\bar{B}\bar{C}\bar{D}$ แทน 1

$A\bar{B}\bar{C}D$ แทน 9

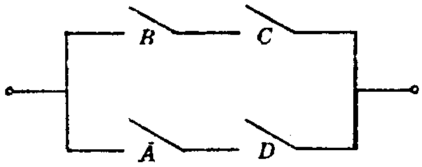
จาก *input* A,B,C,D นี้เราจะสร้างวงจรสวิทซึ่งกระแสจะไหลผ่านก็ต่อเมื่อ *input* เหล่านี้แทนจำนวน 6, 7 หรือ 8 นั่นคือภายใต้เงื่อนไขที่ว่า

$\bar{A}BC\bar{D} \vee ABC\bar{D} \vee \bar{A}\bar{B}\bar{C}D$

ถ้า *input* A,B,C,D แทนจำนวนระหว่าง 0 ถึง 9 เสมอ แล้วเราสามารถจะเพิกเฉยต่อจำนวน 6 จำนวนที่อาจเป็นไปได้คือ 1010, 1011,...,1111 (นั่นคือเลขฐานสองที่แทนจำนวน 10 ถึง 15) ด้วยเหตุนี้ *Don't Care Condition* $\bar{A}B\bar{C}D$, $AB\bar{C}D$, $\bar{A}\bar{B}CD$, $A\bar{B}CD$, $\bar{A}BCD$, $ABCD$ ในกรณีเฉพาะเราสามารถใช้

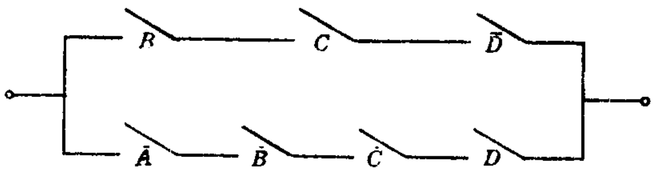
$$\bar{A}BC\bar{D} \vee ABC\bar{D} \vee \bar{A}\bar{B}\bar{C}D \vee ABCD \vee \bar{A}BCD \vee \bar{A}\bar{B}CD \vee \bar{A}B\bar{C}D \quad \ldots\ldots (*)$$

(ดังนั้นเราใช้เงื่อนไข 4 เงื่อนไขจาก 6 *don't care conditions*) ซึ่งประพจน์นี้ *logically equivalent* กับ $BC \vee \bar{A}D$ วงจรของ $BC \vee \bar{A}D$ แสดงดังรูป 5.8.3



รูป 5.8.3

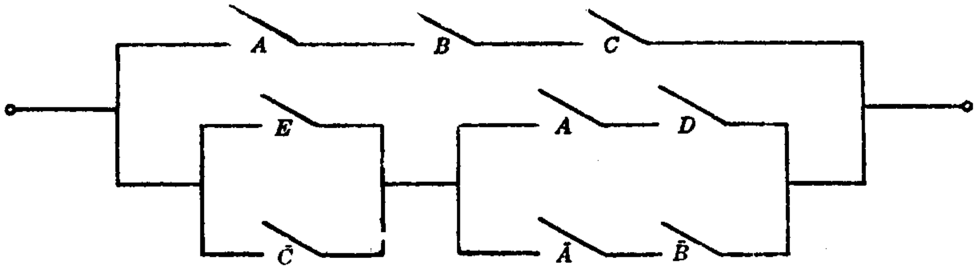
ถ้าเราไม่ใช้ *don't care condition* ในการสร้างวงจรประพจน์ดั้งเดิม $\bar{A}BC\bar{D} \vee ABC\bar{D} \vee \bar{A}\bar{B}\bar{C}D$ จะลดรูปลงมาได้เป็นแค่ $BC\bar{D} \vee \bar{A}\bar{B}\bar{C}D$ ซึ่งแสดงได้ด้วยวงจรในรูป 5.8.4



รูป 5.8.4

แบบฝึกหัดที่ 5

1. จงเขียนวงจร *bridge circuit* แทนวงจรอนุกรมแบบขนานตามรูปข้างล่างนี้



2. คณะกรรมการคณะหนึ่งประกอบด้วยประธานในที่ประชุม ประธานกรรมการเลขานุการ และเหรัญญิก ญัตติในการประชุมครั้งนี้จะผ่าน (คือได้รับการยอมรับในที่ประชุม) ก็ต่อเมื่อ ญัตตินี้ได้รับการยอมรับด้วยเสียงส่วนใหญ่ของที่ประชุม หรือได้เสียงสนับสนุนจากประธานในที่ประชุมรวมกับสมาชิกคนอื่นอีก 1 เสียง สมาชิกแต่ละคนจะกดปุ่มสวิทไฟ ถ้าเห็นด้วยกับญัตติจงออกแบบวงจรสวิทซึ่งควบคุมการไหลของกระแสไฟ โดยจะยอมให้กระแสไฟไหลผ่านวงจรก็ต่อเมื่อ ญัตติได้รับการยอมรับ

3. ให้จำนวนเต็มที่มากกว่าหรือเท่ากับ 0 และน้อยกว่าสิบจำนวนหนึ่งเขียนแทนด้วยตัวเลขในเลขฐานสองในรูป $a_3a_2a_1a_0$ (ตัวอย่างเช่น ถ้าจำนวนเต็มจำนวนนั้นคือ 3 แล้ว $a_3= a_2= 0$ และ $a_1= a_0= 1$ ในขณะที่ถ้าจำนวนเต็มจำนวนนั้นคือ 9 แล้ว $a_3= a_0= 1$ และ $a_2= a_1= 0$) ถ้า A_i เป็นประพจน์ที่ $a_i= 1$ จงสร้างวงจรลอจิคที่แทนเงื่อนไขที่ว่าจำนวนเต็มที่กำหนดให้เป็นเลขปฐม *(prime)*

4. จงสร้างวงจรลอจิกสำหรับบวก 1 เข้ากับเลขฐานสองซึ่งประกอบด้วยเลข 4 หลัก $a_3a_2a_1a_0$

5. จงบรรยายวิธีลดรูปการลบของเลขฐานสองไปยังการใช้การบวกเท่านั้น

6. สมมติว่าจำนวนระหว่าง 0 และ 9 ให้มาในรูปเลขฐานสองซึ่งประกอบด้วย 4 หลัก โดยการใช้สัญลักษณ์แบบตัวอย่าง 5.8.2 จงใช้ *don't care condition* เพื่อสร้างวงจรสวิท (หรือวงจรลอจิก) อย่างง่ายสำหรับเงื่อนไขที่ว่าจำนวนที่ให้มาเป็น จำนวนปฐม *(prime)*

7. จงเปลี่ยนเลขฐานสิบต่อไปนี้ให้อยู่ในรูปเลขฐานสอง

   35, 74, 90, 150, 320

8. จงเปลี่ยนเลขฐานสองต่อไปนี้ให้อยู่ในรูปเลขฐานสิบ

   10110, 111011, 10001101

9. จงบวกเลขข้างล่างต่อไปนี้โดยใช้ระบบเลขฐานสอง

   ก) $\begin{array}{r} 11000 \\ + \\ \underline{101110} \end{array}$ ข) $\begin{array}{r} 11101 \\ + \\ \underline{1011} \end{array}$