เอกสารประกอบการสอน

การออกแบบวงจรดิจิตอลและวงจรตรรกะ

Digital Circuit and Logic Design

ศิริพร เดชะศิลารักษ์

ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

อำเภอเมือง จังหวัดพิษณุโลก

การวัดประเมินผล

ปฏิบัติการ	15% (เข้าทำแลบ 5%, รายงานแลบ10%)
การบ้าน+รายงาน	5%
สอบย่อย	20%
สอบกลางภาค	30%
สอบปลายภาค	30%

การตัดเกรด : อิงกลุ่ม

ลงชื่อ.....

(อาจารย์ศิริพร เดชะศิลารักษ์)

...../...../.....

มหาวิทยาลัยนเรศวร

แผนการสอน (Course Syllabus)

ภาคเรียนปลาย ปีการศึกษา 2550

รหัสวิชา : 305122 **จำนวนหน่วยกิต :** 3(2-3)

ชื่อวิชา : การออกแบบวงจรดิจิตอลและวงจรตรรกะ 1

Digital Circuit and Logic Design I

วิชาบังคับก่อน: -

ผู้สอน : อาจารย์ศิริพร เดชะศิลารักษ์ ห้อง EE 413

http://www.ecpe.nu.ac.th/professors_siriporn.htm

เนื้อหารายวิชา

ทฤษฎีเบื้องต้นของวงจรสวิทชิ่ง พีชคณิตแบบบูลีน รหัสคอมพิวเตอร์ การตรวจสอบความผิดพลาด ตารางความจริง วิธีการลดรูปสมการแบบบูลีน วงจรตรรกชนิดต่างๆ มัลติเพลกเซอร์ ดีมัลติเพลกเซอร์ วงจรเข้ารหัสและถอดรหัส วงจรบวกลบ วงจรเชิงลำดับ ฟลิปฟลอป วงจรนับ วงจรชีฟรีจิสเตอร์

จุดประสงค์วิชา

ให้นิสิตเข้าใจการออกแบบวงจรดิจิตอลเพื่อเป็นพื้นฐานของการเรียนรู้ในรายวิชาการออกแบบวงจรดิจิตอลและวงจรตรรกะ 2 รายวิชาไมโครโปรเซสเซอร์และภาษาแอสเซมบลี และรายวิชาอื่นๆ ต่อไป

เนื้อหารายวิชา (บรรยาย)

สัปดาห์ที่	เนื้อหาการสอน	หมายเหตุ
1	บทนำ ระบบดิจิตอล ระบบตัวเลข (1)	
2	ระบบตัวเลข (2)	
3	รหัสคอมพิวเตอร์และการตรวจสอบความผิดพลาด	
4	ลอจิกเกต	สอบย่อย 1
5	การลดรูปโดยใช้พีชคณิตบูลีน	
6	การลดรูปโดยใช้ แผนผังคาร์โนห์	
7	ทบทวนเนื้อหา	
8	**สอบกลางภาค**	
9	วงจรคอมบิเนชัน 1 : วงจรเปรียบเทียบและวงจรคำนวณทางคณิตศาสตร์	
10	วงจรคอมบิเนชัน 2 : วงจรเข้ารหัส ถอดรหัส มัลติเพล็กซ์ ดีมัลติเพล็กซ์	
11	ฟลิปฟลอป	
12	วงจรชีเควนเชียล	สอบย่อย 2
13	วงจรนับอะซิงโครนัสและซิงโครนัส	
14	ชิฟรีจิสเตอร์	
15	ทบทวนเนื้อหา	
16	**สอบปลายภาค**	

ตารางปฏิบัติการ

สัปดาห์ที่	ปฏิบัติการ	รายละเอียด	หมายเหตุ
1	-	ชี้แจงแผนการเรียน ระเบียบปฏิบัติของรายวิชาและ แบ่งกลุ่ม	
		แนะนำอุปกรณ์ต่าง ๆ ที่ใช้ในการทดลอง	
2	1	การทดลอง 1 ลอจิกเกตชนิดทีทีแอล	
		การทดลอง 2 ลอจิกเกตชนิดซีมอส	
3	2	การทดลอง 3 ลอจิกเกตทีทีแอลที่มีเอาต์พุตคอลเลคเตอร์เปิด	
		การทดลอง 4 เกตเอนกประสงค์ชนิดแนนด์และนอร์	
4	3	การทดลอง 5 การลดรูปเกตโดยใช้แผนผังคาร์โนห์	
		การทดลอง 7 ลอจิกเกตที่มีเอาต์พุต 3 สถานะ	
5	4	การทดลอง 16 วงจรบวกและวงจรลบเลขฐานสอง	
		การทดลอง 17 วงจรทางคณิตศาสตร์	
6	5	การทดลอง 13 วงจรเข้ารหัส วงจรถอดรหัส และตัวแสดงผล	
7	-	ทบทวน	
8		สอบกลางภาค	
9	6	การทดลอง 18 มัลติเพลกซ์และดีมัลติเพลกซ์	
		การทดลอง 9 ฟลิปฟลอป	
10	7	การทดลอง 10 วงจรนับแบบอะซิงโครนัส	
		การทดลอง 11 วงจรนับแบบซิงโครนัส	
11	8	การทดลอง 14 วงจรนับสิบ	
		การทดลอง 15 วงจรนับเลขฐานสอง	
12	9	การทดลอง 12 วงจรเลื่อนข้อมูล	
13	10	การทดลอง 19 ตัวแปลงดิจิตอลเป็นแอนะลอก	
		การทดลอง 20 ตัวแปลงแอนะลอกเป็นดิจิตอล	
14	-	ทบทวน	
15	-	ทบทวน	
16		สอบปลายภาค	

หนังสืออ้างอิง/หนังสืออ่านประกอบ

1. เอกสารประกอบการสอนรายวิชาการออกแบบวงจรดิจิตอลและตรรกะ, ศิริพร เดชะศิลารักษ์, ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์, คณะวิศวกรรมศาสตร์, ม.นเรศวร, 2550
2. ดิจิตอลและการออกแบบวงจรลอจิก, สุวัฒน์ รอดผล, สำนักพิมพ์ ส.ส.ท, 2544
3. Digital Fundamentals, 7th Edition, Thomas L. Floyd, Prentice-Hall, 2000.
4. Digital Principle and Design, Donald D.Givone, McGraw Hill, 2003.
5. Digital Design, M.Mano, 3th Edition, Prentice-Hall, 2002.

คำนำ

เอกสารเล่มนี้ได้เขียนและเรียบเรียงขึ้นมาเพื่อใช้ในเป็นเอกสารประกอบการเรียนการสอนรายวิชา 305122 การออกแบบวงจรดิจิตอลและวงจรตรรกะ 1 (Digital Circuit and Logic Design I)และรายวิชา 303262 การออกแบบวงจรดิจิตอลและวงจรตรรกะ (Digital Circuit and Logic Design) ซึ่งอยู่ในกลุ่มวิชาบังคับเฉพาะสาขาตามหลักสูตรวิศวกรรมคอมพิวเตอร์และวิศวกรรมไฟฟ้า ของภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร สำหรับเนื้อหาในเอกสารเล่มนี้จะกล่าวถึงความเป็นมาของระบบดิจิตอล ทฤษฎีเบื้องต้นของวงจรสวิทชิ่ง รหัสคอมพิวเตอร์ การตรวจสอบความผิดพลาด ตารางความจริง วิธีการลดรูปสมการแบบบูลีน พีชคณิตแบบบูลีน แผนผังคาร์โนห์ วงจรลอจิกเชิงจัดหมู่ต่างๆ เช่น มัลติเพลกเซอร์ ดีมัลติเพลกเซอร์ วงจรเข้ารหัสและถอดรหัส วงจรบวกลบ เป็นต้น วงจรลอจิกเชิงลำดับ ฟลิปฟลอป วงจรนับ วงจรชีฟรีจิสเตอร์ รวมทั้งการประยุกต์ใช้งานเบื้องต้น

ผู้เขียนหวังว่า นิสิตจะได้รับความรู้จากเอกสารเหล่านี้เพื่อใช้ในการเรียน ทบทวนหรือทำความเข้าใจในเนื้อหาได้มากขึ้น นอกเหนือจากการฟังคำบรรยายในชั้นเรียน อย่างไรก็ตามนิสิตต้องฝึกฝนแบบฝึกหัดท้ายบทหรือจากเอกสารอ้างอิงด้วยตนเอง เพื่อให้เกิดทักษะในการออกแบบวงจรดิจิตอลมากขึ้น ซึ่งจะเป็นประโยชน์สำหรับนิสิตในการศึกษาขั้นสูงต่อไป

สุดท้ายนี้ ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร. สุชาติ แย้มเม่น ดร.อัครพันธ์ วงค์กังแห ผู้ช่วยศาสตราจารย์ ดร. สุรเชษฐ์ กานต์ประชา ที่ให้คำแนะนำอันเป็นประโยชน์อย่างยิ่งในการทำเอกสารเล่มนี้ รวมทั้งนิสิตทุกท่าน ที่ช่วยเหลือในการตรวจเอกสาร ทำให้เอกสารเล่มนี้สำเร็จลุล่วงไปด้วยดี

ศิริพร เดชะศิลารักษ์

27 พฤศจิกายน 2550

สารบัญ

		หน้า
คำนำ		ก
สารบัญ		ข-ฉ
บทที่ 1 บทนำ		
1.1	ระบบดิจิตอลและอนาลอก	1
1.2	วิวัฒนาการของระบบดิจิตอล	4
1.3	โครงสร้างและการทำงานของวงจรดิจิตอล	5
บทที่ 2 ระบบจำนวน		
2.1	ระบบเลขฐาน	10
2.2	การแปลงเลขฐาน	12
2.3	การดำเนินการทางคณิตศาสตร์ของเลขฐาน	19
2.4	ตัวเลขที่มีเครื่องหมาย	31
2.5	การบวกลบเลข Signed 2's complement	35
บทที่ 3 รหัส		
3.1	รหัสบีซีดี	42
3.2	รหัสอักขระ	51
3.3	รหัสแสดงผล	55
3.4	การตรวจสอบความผิดพลาดของข้อมูล	56
บทที่ 4 ลอจิกเกต		
4.1	บทนำ	65
4.2	สัญลักษณ์ลอจิกเกต	67
4.3	การทำงานของลอจิกเกต	68
4.4	ชนิดของลอจิกเกต	70
4.5	แผนภาพเวนน์	81
4.6	ตัวอย่างการประยุกต์ใช้งานลอจิกเกต	81

สารบัญ (ต่อ)

		หน้า
4.7	เกตสมมูลย์	83
4.8	การต่อขยายเกต	86
4.9	เกตเอนกประสงค์	87
4.10	เทคโนโลยีของวงจรรวมหรือไอซี	90
บทที่ 5 พีชคณิตบูลีน		
5.1	บทนำ	97
5.2	ทฤษฎีของพีชคณิตบูลีน	98
5.3	การวิเคราะห์วงจรลอจิกด้วยสมการบูลีน	102
5.4	รูปแบบมาตรฐานของสมการบูลีน	104
5.5	สมการบูลีนและตารางความจริง	110
5.6	การลดรูปสมการโดยใช้พีชคณิตบูลีน	112
5.7	การออกแบบวงจรโดยใช้พีชคณิตบูลีน	115
5.8	การออกแบบวงจรโดยใช้แนนด์หรือนอร์เกต	119
บทที่ 6 แผนผังคาร์โนห์		
6.1	บทนำ	123
6.2	รูปแบบของ KM	123
6.3	การเขียนตารางความจริงและสมการบูลีนใน KM	129
6.4	การรวมกลุ่มใน KM เพื่อหาคำตอบ	133
6.5	Don't care term	138
6.6	การออกแบบวงจรลอจิกโดยใช้ KM	138
บทที่ 7 วงจรคำนวณคณิตศาสตร์และลอจิก		
7.1	บทนำ	149
7.2	วงจรบวกเลขฐานสอง	150
7.3	วงจรบวกเลขบีซีดี	158
7.4	วงจรลบเลขฐานสอง	160

สารบัญ (ต่อ)

		หน้า
7.5	วงจรเปรียบเทียบ	164
7.6	วงจรคูณ	169
บทที่ 8 วงจรลอจิกเชิงจัดหมู่		
8.1	บทนำ	171
8.2	วงจรเข้ารหัส	171
8.3	วงจรถอดรหัส	176
8.4	วงจรพาริตี้	193
8.5	วงจรมัลติเพล็กซ์และดีมัลติเพล็กซ์	198
บทที่ 9 ฟลิปฟลอป		
9.1	บทนำ	205
9.2	สัญญาณนาฬิกาและสัญญาณพัลส์	206
9.3	ชนิดของฟลิปฟลอป	210
9.4	ประเภทอินพุตของฟลิปฟลอป	224
9.5	Master-Slave ฟลิปฟลอป	227
9.6	การประยุกต์ใช้งานฟลิปฟลอป	228
9.7	เวลาที่ใช้ในการทำงานของฟลิปฟลอป	230
บทที่ 10 วงจรซีเควนเชียลแบบซิงโครนัส		
10.1	บทนำ	232
10.2	หลักการเบื้องต้น	233
10.3	การวิเคราะห์วงจรซีเควนเชียล	237
10.4	การออกแบบวงจรซีเควนเชียล	242
บทที่ 11 วงจรนับ		
11.1	ประเภทของวงจรนับ	246
11.2	วงจรนับอะซิงโครนัส	247
11.3	วงจรนับซิงโครนัส	252

สารบัญ (ต่อ)

		หน้า
11.4	การหาลำดับการนับของวงจร	258
11.5	การนำวงจรนับมาต่อแบบเรียงลำดับ	259
11.6	การถอดรหัสวงจรนับ	260
11.7	การประยุกต์ใช้งานวงจรนับ	261
บทที่ 12 วงจรเลื่อนข้อมูล		
12.1	ชิฟรีจิสเตอร์แบบเข้าอนุกรม-ออกอนุกรม	264
12.2	ชิฟรีจิสเตอร์แบบเข้าอนุกรม-ออกขนาน	268
12.3	ชิฟรีจิสเตอร์แบบเข้าขนาน-ออกอนุกรม	269
12.4	ชิฟรีจิสเตอร์แบบเข้าขนาน-ออกขนาน	271
12.5	ชิฟรีจิสเตอร์แบบสองทิศทาง	272
12.6	ไอซีชิฟรีจิสเตอร์	273
12.7	วงจรนับด้วยชิฟรีจิสเตอร์	278
12.8	การประยุกต์ใช้งานชิฟรีจิสเตอร์	281
บรรณานุกรม		287

มหาวิทยาลัยนเรศวร

บทที่ 1
บทนำ (Introduction)

วัตถุประสงค์

1. สามารถอธิบายและเข้าใจความแตกต่างระหว่างระบบดิจิตอลและระบบอนาลอกได้
2. สามารถอธิบายวิวัฒนาการ ความสำคัญของระบบดิจิตอลได้

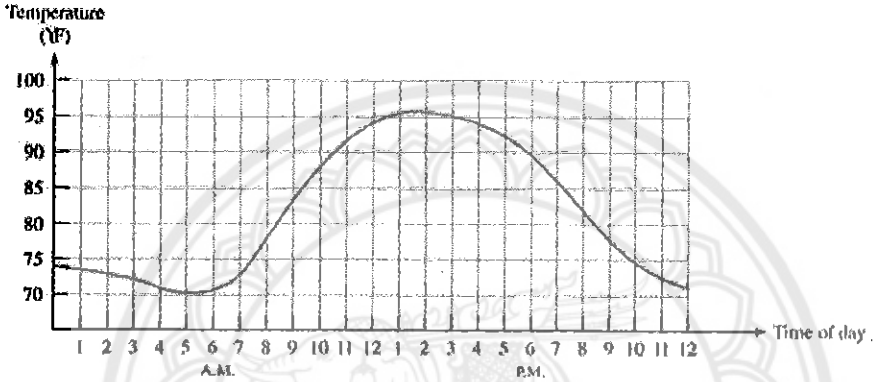
ในปัจจุบัน คำว่า "ดิจิตอล (Digital)" ได้เป็นคำที่นิยมใช้กันทั่วไป เนื่องจากระบบดิจิตอล (Digital system) ได้เข้ามามีบทบาทในชีวิตประจำวันของคนเราอย่างกว้างขวางและครอบคลุมในทุกด้าน เช่น คอมพิวเตอร์ ระบบอัตโนมัติ หุ่นยนต์ ด้านการแพทย์ การขนส่ง การสื่อสาร ความบันเทิง การสำรวจอวกาศ เป็นต้น จึงสามารถกล่าวได้ว่าเป็นยุคของดิจิตอลอย่างแท้จริง ดังนั้นการเรียนรู้หลักการ แนวคิดและการดำเนินการเบื้องต้นเกี่ยวกับระบบดิจิตอลจึงเป็นสิ่งที่จำเป็นอย่างยิ่ง เนื่องจากระบบดิจิตอลประกอบด้วยวงจรดิจิตอลจำนวนหลายวงจรที่เชื่อมต่อเข้าด้วยกัน เพื่อให้เข้าใจหลักการทำงานของระบบดิจิตอล จำเป็นต้องมีความรู้ความเข้าใจพื้นฐานเกี่ยววงจรดิจิตอลและการทำงานของวงจรในแต่ละส่วน รวมทั้งการออกแบบวงจรเพื่อนำไปประยุกต์ใช้งานต่างๆ ได้

1.1 ระบบดิจิตอลและอนาลอก (Digital and Analog Systems)

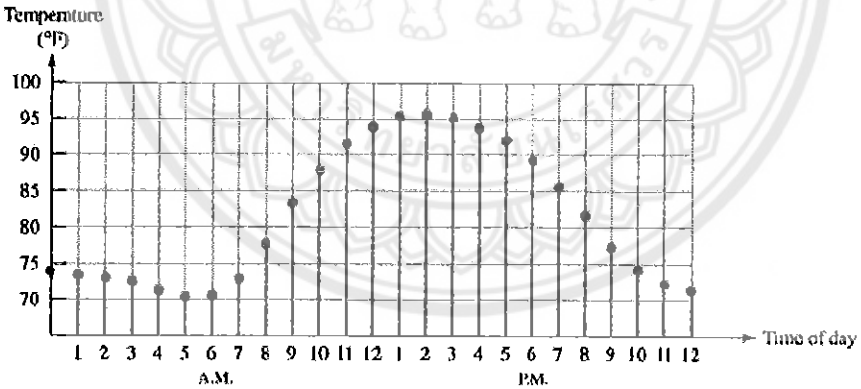
ระบบทางไฟฟ้าสามารถแบ่งระดับสัญญาณได้เป็น 2 ระบบคือ อนาลอกและดิจิตอล **ระบบอนาลอก** โดยทั่วไปปริมาณต่างๆ ในธรรมชาติจะอยู่ในรูปของสัญญาณอนาลอก ที่มีค่าของสัญญาณแบบต่อเนื่อง เช่น แรงดัน กระแสไฟฟ้า อุณหภูมิ ความดัน เสียง ระยะทาง เป็นต้น ดังนั้นระบบอนาลอกจึงเป็นระบบที่มีความต่อเนื่อง (Continuous system) ตัวอย่างของอุปกรณ์ที่มีการวัดปริมาณในรูปของสัญญาณอนาลอก ค่าตัวเลขที่วัดได้เป็นค่าโดยประมาณเท่านั้น ไม่สามารถบอกเป็นค่าตัวเลขอย่างละเอียดได้ เช่น มิเตอร์วัดความเร็วรถแบบที่ใช้เข็ม โดยเข็มของมิเตอร์มีมุมเปลี่ยนแปลงตามความเร็วของรถที่มีความเร็วเพิ่มขึ้นหรือลดลง ความเร็วของรถอาจมีค่าตั้งแต่ 0 จนถึง 100 ไมล์ต่อชั่วโมง (mph) หรือปรอทวัดอุณหภูมิ เมื่อปรอทอยู่ระหว่างเส้น 2 เส้นที่บอกระดับของอุณหภูมิ การอ่านค่าอุณหภูมิจะพิจารณาค่าของปรอทที่อยู่ใกล้เส้นนั้นมากที่สุดและบอกค่าอุณหภูมิที่วัดได้เป็นค่าโดยประมาณ **ระบบดิจิตอล** มีค่าสัญญาณที่ไม่ต่อเนื่องหรือเป็นค่าตัวเลข (Discrete) ดังนั้นระบบดิจิตอลจึงเป็นระบบที่ไม่มีความต่อเนื่อง (Discrete system) อุปกรณ์อิเล็กทรอนิกส์ส่วนใหญ่จะเป็นระบบดิจิตอล รวมทั้งคอมพิวเตอร์ เครื่องคำนวณ เครื่องเล่นวิดีโอ ระบบโทรศัพท์ต่างๆ เป็นต้น จะเห็นได้ว่าระบบดิจิตอลเป็นระบบที่ใหญ่ที่สุดในโลกปัจจุบัน ตัวอย่างของอุปกรณ์ที่มีการวัดปริมาณในรูปของสัญญาณดิจิตอล เช่น นาฬิกาดิจิตอลที่บอกเวลา

ในรูปของตัวเลขที่แสดงชั่วโมง นาทีหรือวินาที แม้ว่าปริมาณเวลามีการเปลี่ยนแปลงแบบต่อเนื่อง แต่นาฬิกาดิจิตอลสามารถบอกเวลาได้ทุกๆ วินาทีหรือนาที เมื่อเปรียบเทียบกับนาฬิกาไขลานที่บอกเวลาอย่างต่อเนื่องขึ้นอยู่กับเข็มของนาฬิกาที่หมุนไป

ตัวอย่างการเปรียบเทียบข้อมูลที่เป็นปริมาณอนาลอกและดิจิตอล ได้แก่ การวัดอุณหภูมิในแต่ละวัน จากรูปที่ 1.1 แสดงค่าอุณหภูมิที่มีลักษณะต่อเนื่องกันเรียกว่าเป็นปริมาณอนาลอก แต่จากรูปที่ 1.2 แสดงค่าอุณหภูมิที่มีลักษณะไม่ต่อเนื่องกันเรียกว่าเป็นปริมาณดิจิตอล เป็นการวัดอุณหภูมิในทุกๆ 1 ชั่วโมงและแสดงค่าตัวเลขของอุณหภูมิในแต่ละชั่วโมง



รูปที่ 1.1 ปริมาณอนาลอกของการวัดค่าอุณหภูมิใน 1 วัน



รูปที่ 1.2 ปริมาณดิจิตอลของการวัดค่าอุณหภูมิใน 1 วัน

ข้อดีของระบบดิจิตอล

ในปัจจุบันอุปกรณ์อิเล็กทรอนิกส์ต่างๆ ได้ใช้เทคนิคของดิจิตอลแทนที่อนาลอกเพิ่มขึ้น เนื่องจากเหตุผลดังนี้

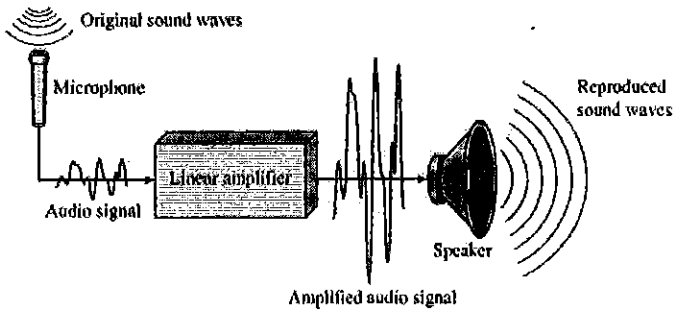
1. การออกแบบระบบดิจิตอลสามารถทำได้ง่าย เนื่องจากระบบดิจิตอลมีการทำงานแบบวงจรสวิตช์ที่ใช้ค่าระดับแรงดันเป็น HIGH หรือ LOW เท่านั้น โดยไม่ได้ใช้ค่าแรงดันหรือกระแสไฟฟ้าที่ใช้งานจริง

2. การเก็บข้อมูลดิจิตอลสามารถทำได้ง่าย เนื่องจากระบบดิจิตอลสามารถเก็บข้อมูลได้จำนวนมากและประหยัดพื้นที่ของวงจร ซึ่งตรงกันข้ามกับระบบอนาลอก
3. ระบบดิจิตอลให้ข้อมูลที่มีความถูกต้องและแม่นยำมากกว่าระบบอนาลอก เนื่องจากข้อมูลของระบบดิจิตอลจะไม่มีการสูญเสียไปในขณะที่มีการประมวลผล แต่ระบบอนาลอก สัญญาณต่างๆ เช่น แรงดันไฟฟ้าหรือกระแสไฟฟ้าจะมีการสูญเสียไปเนื่องจากอุณหภูมิ ความชื้นและความคลาดเคลื่อนที่เกิดขึ้นในวงจรเอง
4. ระบบดิจิตอลสามารถโปรแกรมการทำงานได้ การออกแบบระบบดิจิตอลสามารถทำได้ง่ายด้วยการควบคุมชุดคำสั่งหรือที่เรียกว่าโปรแกรม แม้ว่าระบบอนาลอกสามารถโปรแกรมได้เช่นเดียวกันแต่มีข้อจำกัดในเรื่องการใช้งานที่มีความหลากหลายและการทำงานที่มีความซับซ้อน
5. ระบบดิจิตอลจะเกิดสัญญาณรบกวนได้น้อยกว่าระบบอนาลอก สัญญาณรบกวน (Noise) แทบไม่มีผลต่อระบบดิจิตอล เนื่องจากระบบดิจิตอลไม่ได้ใช้ค่าแรงดันไฟฟ้าจริงในการทำงานเพียงแต่ใช้ค่าของแรงดันไฟฟ้าเพื่อบอกระดับของสัญญาณว่าเป็น HIGH หรือ LOW ในการทำงานเท่านั้น
6. วงจรดิจิตอลสามารถผลิตเป็นไอซีได้ แม้ว่าระบบอนาลอกมีประโยชน์ต่อการพัฒนาเทคโนโลยีของไอซี แต่วงจรอนาลอกเองไม่สามารถทำเป็นวงจรรวมหรือไอซีได้เนื่องจากข้อจำกัดของอุปกรณ์ที่ใช้ในวงจรอนาลอกเอง เช่น ตัวต้านทาน คาปาซิเตอร์ ตัวเหนี่ยวนำ หรือหม้อแปลงเป็นต้น ซึ่งเป็นอุปกรณ์ขนาดใหญ่ไม่สามารถบรรจุลงในไอซีได้

ข้อจำกัดของระบบดิจิตอล

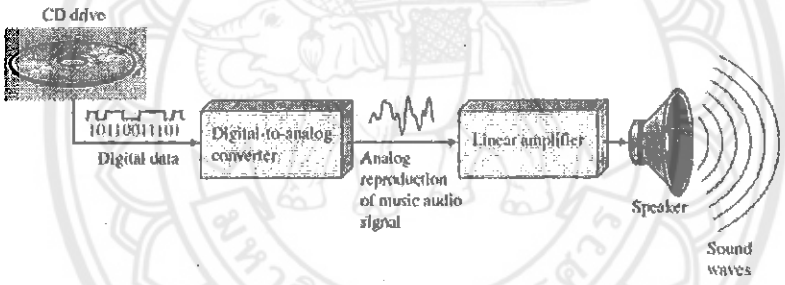
ข้อจำกัดของการใช้งานระบบดิจิตอลมีน้อยมาก แต่ปัญหาที่สำคัญอย่างหนึ่งของระบบดิจิตอลคือเนื่องจากสัญญาณต่างๆ ในธรรมชาติเป็นแบบอนาลอก จึงมีความจำเป็นที่จะต้องทำการแปลงจากสัญญาณอนาลอกเป็นดิจิตอล เพื่อสามารถทำการประมวลผลข้อมูลในระบบดิจิตอลได้ อุปกรณ์ที่ใช้ในการแปลงสัญญาณจากอนาลอกเป็นดิจิตอลเรียกว่า ADC (Analog to Digital Converter) และอุปกรณ์ที่ใช้ในการแปลงสัญญาณจากดิจิตอลเป็นอนาลอกเรียกว่า DAC (Digital to Analog Converter)

จากรูปที่ 1.3 แสดงการทำงานของเครื่องขยายเสียงในระบบอนาลอก เนื่องจากเสียงที่คนเราพูดนั้น จัดเป็นปริมาณอนาลอก เสียงพูดจะผ่านไมโครโฟนและทำการแปลงเป็นค่าระดับแรงดันที่ต่อเนื่องเรียกว่าสัญญาณเสียง (Audio signal) สัญญาณเสียงนี้จะถูกส่งต่อไปยังเครื่องขยายเสียง (Linear amplifier) เพื่อทำการขยายเสียงและส่งออกยังลำโพง (Speaker)



รูปที่ 1.3 ระบบเครื่องเสียงแบบอนาลอก

จากรูปที่ 1.4 เป็นตัวอย่างของเครื่องเล่นซีดีที่ใช้ทั้งระบบดิจิตอลและอนาลอก โดยเพลงต่างๆ จะถูกเก็บไว้ในแผ่นซีดี (Compact disk) ในลักษณะข้อมูลดิจิตอล เมื่อเปิดแผ่นซีดีเพื่อฟังเพลงต่างๆ หัวอ่านของเครื่องเล่นซีดีจะทำการอ่านข้อมูลจากแผ่นซีดีและส่งไปยังวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอก ทำให้ได้สัญญาณเสียงออกมา จากนั้นสัญญาณเสียงจะส่งต่อไปยังเครื่องขยายเสียง (Linear amplifier) เพื่อทำการขยายเสียงและส่งออกยังลำโพง (Speaker) ต่อไป



รูปที่ 1.4 ระบบเครื่องเล่นซีดี

1.2 วิวัฒนาการของระบบดิจิตอล

เทคโนโลยีของระบบดิจิตอลได้พัฒนาขึ้นมาจากความพยายามที่จะให้การคำนวณมีความรวดเร็วแม่นยำ เครื่องคำนวณเครื่องแรกที่ถือเป็นต้นแบบของคอมพิวเตอร์ในปัจจุบัน ออกแบบโดยนักคณิตศาสตร์ ชาวอังกฤษชื่อ Charles Babbage ในปี ค.ศ. 1800 เรียกว่าเครื่องจักรกลวิเคราะห์ (Analytical Engine) แต่ยังไม่สามารถใช้งานจริงได้ แต่ถือว่าเป็นต้นแบบของเครื่องคอมพิวเตอร์ในปัจจุบัน ต่อมาในปี ค.ศ.1946 Eckert กับ Mauchly ได้สร้างเครื่องคอมพิวเตอร์ชื่อ ENIAC (Electronic Numerical Integrator And Computer) ที่ประกอบด้วยอุปกรณ์อิเล็กทรอนิกส์คือหลอดสุญญากาศจำนวน 18,000 หลอด ความสามารถการทำงานของเครื่องคอมพิวเตอร์นี้เทียบได้กับเครื่องคิดเลขแบบมือถือที่ใช้ในปัจจุบันเท่านั้น เครื่องคอมพิวเตอร์หลอดสุญญากาศใช้กำลังไฟมากและเสียหายบ่อย ทำให้ใช้งานได้ไม่นาน ต่อมาในปี ค.ศ. 1951 มีการเสนอแนวคิดเรื่องการ

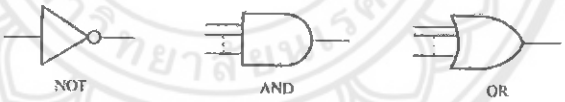
เก็บโปรแกรมไว้ในหน่วยความจำของเครื่องคอมพิวเตอร์สร้างเป็นเครื่อง UNIVAC I ของบริษัท Remington Rand ถือว่าเป็นเครื่องคอมพิวเตอร์เครื่องแรกที่ผลิตออกมาเพื่อการค้า

ความก้าวหน้าของเครื่องคอมพิวเตอร์ได้พัฒนาควบคู่ไปกับการพัฒนาอุปกรณ์อิเล็กทรอนิกส์จากทรานซิสเตอร์ไปจนถึงสารกึ่งตัวนำ ทำให้เครื่องคอมพิวเตอร์มีขนาดเล็กลงแต่ประสิทธิภาพการใช้งานสูงขึ้น

1.3 โครงสร้างและการทำงานของวงจรดิจิตอล

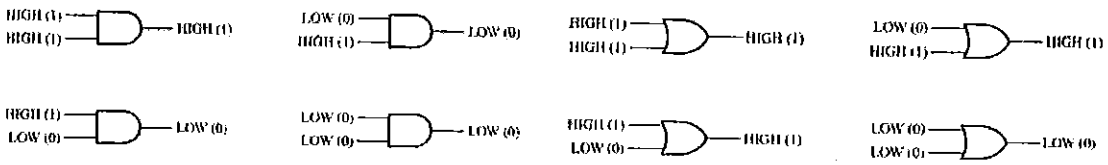
ระบบดิจิตอลประกอบด้วยวงจรดิจิตอลจำนวนหลายวงจร ในแต่ละวงจรดิจิตอลประกอบด้วยเกต (Gate) ต่างๆ มาต่อกันเป็นวงจร ภายในโครงสร้างของเกตเป็นวงจรอิเล็กทรอนิกส์ ที่ประกอบด้วยทรานซิสเตอร์ ไดโอด มาต่อกัน ในที่นี้การออกแบบระบบดิจิตอลจะอ้างถึงการทำงานของเกตแต่ละชนิดและการนำเกตมาสร้างเป็นวงจรดิจิตอลเท่านั้น

เกตพื้นฐานที่นำมาใช้ในการออกแบบมี 3 ชนิดคือ นอตเกตหรืออินเวอร์เตอร์(NOT gate) แอนด์เกต (AND gate) และออร์เกต (OR gate) สัญลักษณ์ของเกตพื้นฐานแสดงดังรูปที่ 1.5 จากสัญลักษณ์ของเกตทางด้านซ้ายเป็นอินพุตและด้านขวาเป็นเอาต์พุต จะเห็นว่านอตเกตมีเพียงหนึ่งอินพุต แต่แอนด์เกตและออร์เกตจะอินพุตตั้งแต่หนึ่งอินพุตขึ้นไป จากรูปที่ 1.6 แสดงการทำงานของเกตแต่ละชนิด ที่มีลักษณะเหมือนกับตรรกศาสตร์โดยใช้เงื่อนไขจริงหรือเท็จ ซึ่งจริงหรือเท็จในทางดิจิตอลหมายถึงค่าของระดับแรงดัน HIGH หรือ LOW นั่นเอง



รูปที่ 1.5 สัญลักษณ์ของเกตพื้นฐาน

(ก) การทำงานของนอตเกต

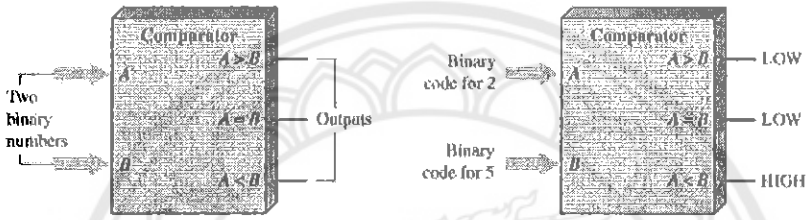


(ข) การทำงานของแอนด์เกต (ค) การทำงานของออร์เกต

รูปที่ 1.6 การทำงานของเกตพื้นฐาน

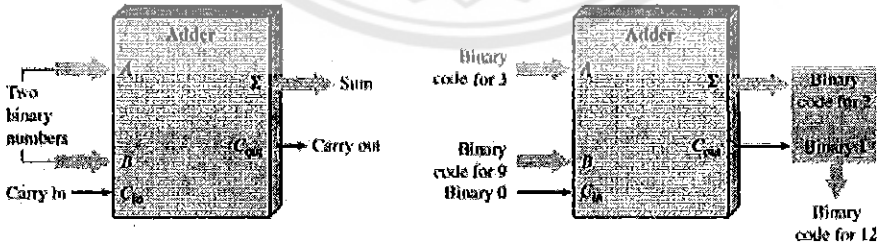
เมื่อนำเกตพื้นฐานมาประกอบเป็นวงจรดิจิตอลอย่างง่ายจนถึงวงจรที่มีความซับซ้อน เพื่อให้วงจรทำงานต่างๆ ได้ ตัวอย่างวงจรดิจิตอลได้แก่ วงจรเปรียบเทียบ (Comparator) วงจรบวกเลข วงจรเข้ารหัสและถอดรหัส (Encoder and Decoder) วงจรนับ (Counter) วงจรเลื่อนข้อมูล (Shift register) เป็นต้น

วงจรเปรียบเทียบ เป็นวงจรที่นำเลขฐานสอง 2 จำนวน มาทำการเปรียบเทียบกัน เพื่อต้องการทราบว่าข้อมูลใดมีค่ามากกว่า น้อยกว่า หรือเท่ากัน จากตัวอย่างดังรูปที่ 1.7 เป็นการเปรียบเทียบเลข 2 และ 5 โดยป้อนข้อมูลทั้งสองในรูปของเลขฐานสองที่อินพุต A และ B ตามลำดับ จะได้เอาต์พุตขา A<B เป็น HIGH ส่วนขาอื่นเป็น LOW



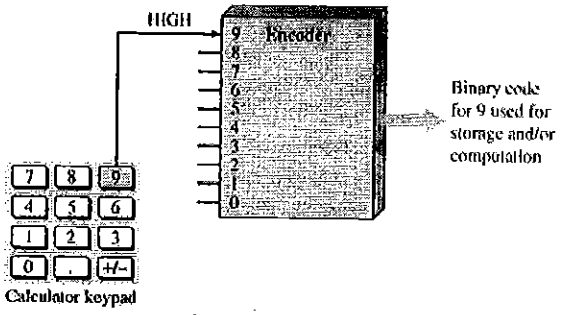
รูปที่ 1.7 วงจรเปรียบเทียบ

วงจรบวกเลข เป็นวงจรที่นำเลขฐานสอง 2 จำนวน มาทำการบวกกัน จะได้เอาต์พุตเป็นผลบวก (Sum) และตัวทดออก (Carry out) จากตัวอย่างดังรูปที่ 1.8 เป็นการบวกเลข 3 และ 9 โดยป้อนข้อมูลทั้งสองในรูปของเลขฐานสองที่อินพุต A และ B ตามลำดับ ส่วนตัวทดเข้า (Carry in) มีค่าเป็น 0 จะได้เอาต์พุตของผลบวกเป็น 2 และตัวทดออกเป็น 1 ในรูปของเลขฐานสอง



รูปที่ 1.8 วงจรบวกเลข

วงจรเข้ารหัส เป็นวงจรที่นำข้อมูลใดๆ เช่น เลขฐานสิบหรือตัวอักษรเป็นต้น มาทำการแปลงเป็นเลขฐานสอง จากตัวอย่างดังรูปที่ 1.9 เป็นการเข้ารหัสเลข 9 โดยกดคีย์เลข 9 สัญญาณจากแป้นคีย์จะส่งไปยังอินพุตขาที่ 9 ของวงจร และให้เอาต์พุตเป็นเลข 9 ที่อยู่รูปของเลขฐานสอง

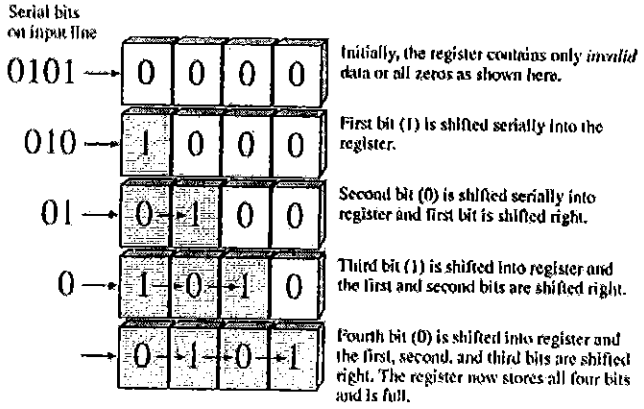


รูปที่ 1.9 **วงจรเข้ารหัส**

วงจรถอดรหัส เป็นวงจรที่นำข้อมูลเลขฐานสองมาแปลงเป็นข้อมูลที่คนเราสามารถเข้าใจได้ ตัวอย่างดังรูปที่ 1.10 **เป็นการถอดรหัสเลข** 8 **โดยทำการป้อนเลข** 8 **ในรูปของเลขฐานสองที่อินพุตของวงจรและเอาต์พุตแสดงผลเป็นเลขฐานสิบออกมา**

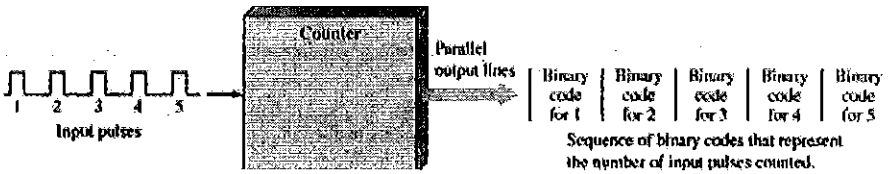
รูปที่ 1.10 **วงจรถอดรหัส**

วงจรเลื่อนข้อมูล เป็นวงจรทำหน้าที่เลื่อนข้อมูลเข้ามาเก็บในวงจรหรือส่งข้อมูลออกจากวงจร เรียกอุปกรณ์ที่ทำหน้าเก็บข้อมูลนี้ว่ารีจิสเตอร์ จากตัวอย่างดังรูปที่ 1.11 **เป็นการเลื่อนข้อมูล** 0101 **มาเก็บไว้ที่รีจิสเตอร์**



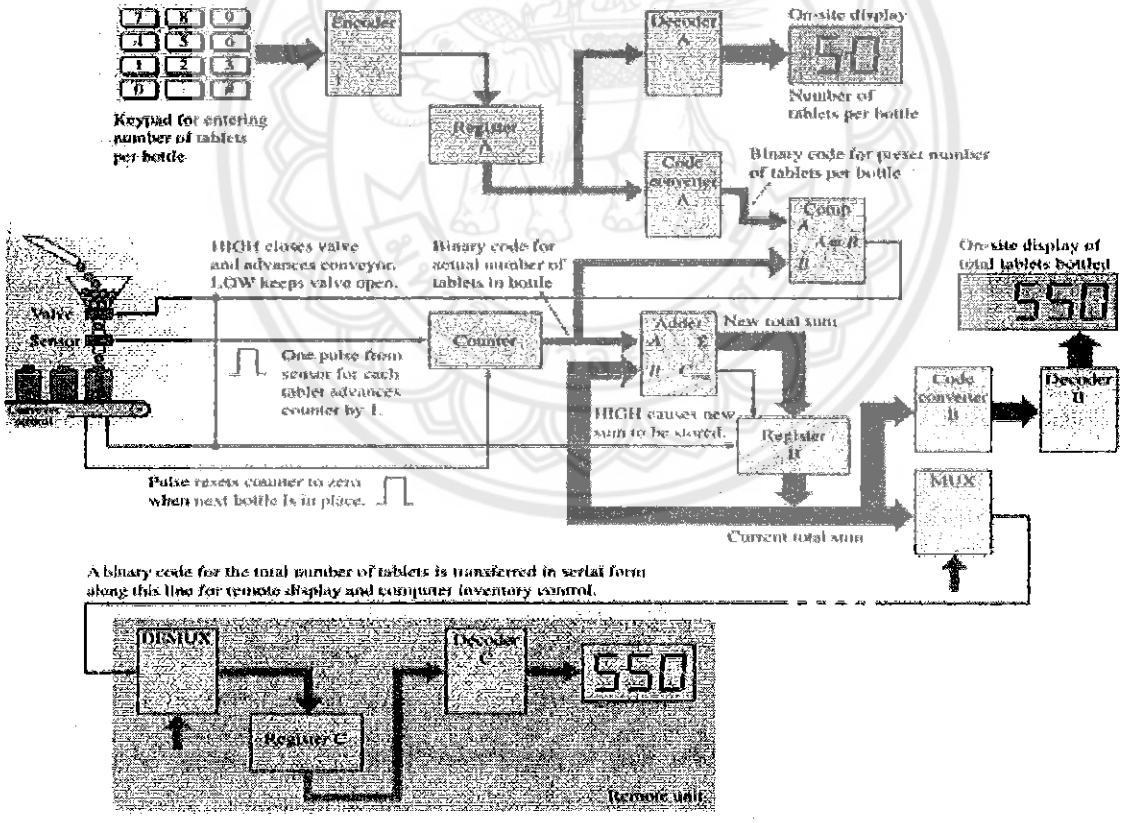
รูปที่ 1.11 **วงจรเลื่อนข้อมูล**

วงจรนับ เป็นวงจรที่ใช้นับเลขตามค่าที่ต้องการ เป็นวงจรหนึ่งที่สำคัญมากในระบบดิจิตอล ดังรูปที่ 1.12



รูปที่ 1.12 วงจรนับเลข

ระบบควบคุมการบรรจุเม็ดยา จากรูปที่ 1.13 เป็นการนำวงจรแต่ละส่วนมาประกอบเข้าด้วยกันเพื่อควบคุมการบรรจุเม็ดยาใส่ขวด



รูปที่ 1.13 บล็อกไดอะแกรมของระบบควบคุมการบรรจุเม็ดยา

การทำงานของระบบเริ่มจากการป้อนตัวเลขจำนวนของเม็ดยาทางคีย์บอร์ด ในที่นี้สมมุติต้องการบรรจุยาจำนวน 50 เม็ดต่อขวด ข้อมูลจากแป้นคีย์จะส่งไปยังวงจรเข้ารหัสและเก็บข้อมูลไว้ที่รีจิสเตอร์ A เพื่อเป็นการกำหนดจำนวนเม็ดยาต่อขวด จากนั้นที่ขวดจะทำการนับจำนวนเม็ดยาโดยมีเซนเซอร์ทำหน้าที่ตรวจจับเม็ดยาที่ไหลลงขวดแต่ละเม็ด สัญญาณจากเซนเซอร์จะส่งต่อไปยังวงจรนับเพื่อทำการนับจำนวนเม็ดยาและส่งค่าการนับไปยังวงจรบวกและเก็บค่าผลบวกไว้ที่รีจิสเตอร์ B เมื่อบรรจุเม็ดยาครบ 50 เม็ดแล้ว จะส่งค่าตัวเลขของการนับที่ได้ไปยังวงจรเปรียบเทียบเพื่อทำการเปรียบเทียบว่าตัวเลขที่ได้มีค่าเท่ากับที่กำหนดไว้ในตอนแรกหรือไม่ ถ้าตัวเลขทั้งสองมีค่าเท่ากันแล้ว วงจรเปรียบเทียบจะส่งสัญญาณเพื่อทำการปิดวาล์วที่ขวดและเลื่อนขวดใหม่มาแทนที่ จากนั้นจะเริ่มบรรจุเม็ดยาใส่ขวดอีกครั้ง

แบบฝึกหัดท้ายบท

1. จงอธิบายความแตกต่างของระบบดิจิตอลและระบบอนาลอก
2. จงบอกข้อดีและข้อเสียของระบบดิจิตอล
3. จงยกตัวอย่างของระบบดิจิตอลที่ใช้งานในปัจจุบัน

บทที่ 2
ระบบจำนวน (Number System)

วัตถุประสงค์

1. เพื่อให้นิสิตเข้าใจเกี่ยวกับระบบเลขฐานและการแปลงเลขฐานต่างๆ
2. เพื่อให้นิสิตเข้าใจเกี่ยวกับการดำเนินการทางคณิตศาสตร์ของเลขฐาน
3. เพื่อให้นิสิตเข้าใจการเก็บข้อมูลของระบบคอมพิวเตอร์

ระบบเลขฐานสอง (Binary number system) เป็นระบบเลขพื้นฐานที่สำคัญมากในระบบดิจิตอล โดยเฉพาะในคอมพิวเตอร์และวงจรอิเล็กทรอนิกส์ต่างๆ สำหรับเนื้อหาบทนี้จะกล่าวถึงระบบเลขฐานสองและความสัมพันธ์ของระบบเลขฐานสองกับระบบเลขฐานอื่นๆ เช่น เลขฐานสิบ (Decimal) เลขฐานแปด (Octal) และเลขฐานสิบหก (Hexadecimal) รวมทั้งการดำเนินการทางคณิตศาสตร์ (Arithmetic operation) ของเลขฐาน เพื่อสามารถเข้าใจการทำงานพื้นฐานของระบบดิจิตอลได้

2.1 ระบบเลขฐาน (Base)

โดยทั่วไประบบเลขฐานจะเริ่มจากเลขฐานหนึ่ง แต่เนื่องจากเลขฐานหนึ่งมีเพียงเลขศูนย์ตัวเดียวเท่านั้น จึงเป็นระบบเลขที่ไม่มีความสำคัญและไม่น่าเรียนรู้ในวงจรดิจิตอล ระบบเลขฐานที่นำมาใช้ในวงจรดิจิตอลคือระบบเลขฐานสองที่ประกอบด้วยเลข 0 และ 1 อย่างไรก็ตามการศึกษาเรื่องระบบเลขฐาน จำเป็นต้องเข้าใจระบบเลขฐานอื่นด้วย เช่น เลขฐานแปด เลขฐานสิบหก เป็นต้น ซึ่งมีบทบาทสำคัญในเทคโนโลยีด้านคอมพิวเตอร์เนื่องจากข้อมูลในระบบดิจิตอลประกอบด้วยเลขฐานสองจำนวนมากมาย ทำให้สามารถเข้าใจข้อมูลเหล่านี้ได้ยากและโอกาสเกิดความผิดพลาดของข้อมูลก็มีมากเช่นกัน ดังนั้นจึงนำเลขฐานแปดและเลขฐานสิบหกมาใช้แทนเลขฐานสองเพื่อสามารถเข้าใจข้อมูลได้ง่ายขึ้น นอกจากนี้ในชีวิตประจำวันของคนเราจะใช้เลขฐานสิบ ดังนั้นจึงต้องเข้าใจความสัมพันธ์ของระบบเลขฐานต่างๆ สามารถแสดงตัวเลขและตัวอักษรของเลขฐานต่างๆ ดังตารางที่ 2.1

เลขฐานสองมีตัวเลข 2 ตัวเท่านั้นคือ 0 และ 1 ค่าตัวเลขทั้งสองนี้เรียกว่าบิต (Bit) ที่มาจากคำว่า <u>Bi</u>nary Digi<u>t</u> ตัวอย่างเช่น เลขฐานสองของ 1101.101 ประกอบด้วยเลขฐานสองจำนวน 7 บิต เป็นต้น สำหรับคำว่า "Digit" มาจากรากศัพท์ของคำว่า "Digitus" ซึ่งเป็นภาษาละตินมีความหมายว่านิ้ว เนื่องจากสมัยก่อนการนับค่าตัวเลขจะใช้นิ้วมือมาช่วยในการนับ ดังนั้น "Digit" ในที่นี้จึงหมายถึงตัวเลข ตำแหน่งของเลขฐานสองในแต่ละบิต มีความหมายดังนี้ บิตที่อยู่ทางด้าน

ขวาสุดจะมีค่าน้อยที่สุดเรียกว่าบิตนัยสำคัญที่น้อยที่สุดหรือบิตต่ำ (Least Significant Bit, LSB) บิตที่อยู่ทางด้านซ้ายสุดจะมีค่ามากที่สุดเรียกว่าบิตนัยสำคัญที่มากที่สุดหรือบิตสูง (Most Significant Bit, MSB) การรวมกลุ่มบิตของเลขฐานสองจะมีชื่อเรียกแตกต่างกันดังตารางที่ 2.2

$$Most\ \ Significant\ \ Bit(MSB)\ \Leftarrow\ \underline{1}010\underline{1}\ \Rightarrow\ Least\ \ Significant\ \ Bit(LSB)$$

ตารางที่ 2.1 ระบบเลขฐานต่างๆ

Base	Number system	Digit symbols
2	Binary	0,1
3	Ternary	0,1,2
4	Quaternary	0,1,2,3
5	Quinary	0,1,2,3,4
8	Octal	0,1,2,3,4,5,6,7
10	Decimal	0,1,2,3,4,5,6,7,8,9
12	Duodecimal	0,1,2,3,4,5,6,7,8,9,A,B
16	Hexadecimal	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

ตารางที่ 2.2 การเรียกชื่อจำนวนบิตข้อมูล

จำนวนบิตข้อมูล	ชื่อเรียก
4	Nibble
8	Byte
16	Word
32	Double Word

ความสัมพันธ์ระหว่างเลขฐานสองและเลขฐานสิบจะอยู่ในรูปของ $N=2^n$ เมื่อ n คือจำนวนบิตของเลขฐานสองและ N คือจำนวนค่าของเลขฐานสิบที่นับได้ทั้งหมด ค่าที่นับได้มีค่าตั้งแต่ 0 ถึง 2^n-1 เช่น เลขฐานสองจำนวน 3 บิตแสดงการนับตั้งแต่ 0-7 (จำนวน 8 ค่า) เลขฐานสองจำนวน 4 บิตแสดงการนับตั้งแต่ 0-15 (จำนวน 16 ค่า) เป็นต้น เลขฐานสิบมีตัวเลข 10 ตัว คือ 0 ถึง 9 ถ้าต้องการแสดงค่าที่มากกว่า 9 ต้องใช้ตัวเลขจำนวน 2 ตัวหรือมากกว่า **เลขฐานแปด**มีตัวเลข 8 ตัว คือ 0 ถึง 7 **เลขฐานสิบหก** มีตัวเลข 16 ตัว คือ 0 ถึง 15 และใช้ตัวอักษร A, B, C, D, E, F แทนตัวเลข 10 ถึง 15 ตามลำดับ จากตารางที่ 2.1 จะเห็นว่าระบบเลขฐานที่ต่างกัน มีตัวเลขบางตัวซ้ำกัน เช่น เลข 0 และ 1 จะมีอยู่ในทุกระบบเลขฐาน เป็นต้น ดังนั้นเพื่อไม่ให้เกิดความสับสนจึงเขียนเลขฐานลงท้ายกำกับไว้ เช่น 101_2, 101_8, 101_{10}, 101_{16} เป็นต้น ยกเว้นเลขฐานสิบจะไม่เขียนเลขฐานกำกับไว้ก็ได้ เนื่องจากเป็นเลขฐานที่ใช้กันในชีวิตประจำวันอยู่แล้ว

การนับค่าของระบบเลขฐานต่างๆ มีวิธีการนับเช่นเดียวกับเลขฐานสิบ โดยการนับจะเริ่มนับในคอลัมน์ที่หนึ่งตั้งแต่ 0 จนถึง 9 เมื่อคอลัมน์ที่หนึ่งมีค่าถึง 9 หรือครบรอบ จะกลับมานับที่ 0

ใหม่และคอลัมน์ถัดไปทางด้านซ้ายจะบวก 1 เข้าไป จากนั้นการนับค่าในคอลัมน์ที่หนึ่งจะเพิ่มขึ้นเป็น 10, 11, 12, ..., 19 สามารถแสดงลำดับการนับของเลขฐานสิบ เลขฐานสอง เลขฐานสาม เลขฐานแปด เลขฐานสิบหก ตั้งแต่ 0 ถึง 31 ดังตารางที่ 2.3

ตารางที่ 2.3 ลำดับการนับของระบบเลขฐาน

Decimal	Binary	Ternary	Octal	Hexadecimal
0	0	0	0	0
1	1	1	1	1
2	10	2	2	2
3	11	10	3	3
4	100	11	4	4
5	101	12	5	5
6	110	20	6	6
7	111	21	7	7
8	1000	22	10	8
9	1001	100	11	9
10	1010	101	12	A
11	1011	102	13	B
12	1100	110	14	C
13	1101	111	15	D
14	1110	112	16	E
15	1111	120	17	F
16	10000	121	20	10
17	10001	122	21	11
18	10010	200	22	12
19	10011	201	23	13
20	10100	202	24	14
21	10101	210	25	15
22	10110	211	26	16
23	10111	212	27	17
24	11000	220	30	18
25	11001	221	31	19
26	11010	222	32	1A
27	11011	1000	33	1B
28	11100	1001	34	1C
29	11101	1002	35	1D
30	11110	1010	36	1E
31	11111	1011	37	1F

2.2 การแปลงเลขฐาน

1) การแปลงจากเลขฐานสอง เลขฐานแปด เลขฐานสิบหกเป็นเลขฐานสิบ

ตัวเลขในแต่ละตำแหน่งจะบอกให้ทราบถึงปริมาณของตัวเลขนั้นๆ สามารถเขียนอยู่ในรูปสมการของโพลิโนเมียลได้ ตัวอย่างเช่น เลข 872.64 เขียนในรูปของโพลิโนเมียลได้ดังนี้

$$
\begin{aligned}
872.64 &= 800 + 70 + 2 + 0.6 + 0.04 \\
&= 8\times100 + 7\times10 + 2\times1 + 6\times0.1 + 4\times0.01 \\
&= (8\times10^{2}) + (7\times10^{1}) + (2\times10^{0}) + (6\times10^{-1}) + (4\times10^{-2})
\end{aligned}
$$

ตำแหน่งของตัวเลขแต่ละตัวแสดงค่าน้ำหนัก (Weight) ของตัวเลขนั้น กรณีที่เป็นเลขจำนวนเต็มแสดงค่าน้ำหนักอยู่ในรูปของเลข 10 ยกกำลังด้วยเลขจำนวนเต็มบวก โดยที่ค่าเลขยกกำลังจำนวนเต็มบวกจะเพิ่มขึ้นจากขวาไปซ้าย โดยเริ่มที่ 10^0 กรณีที่เป็นเลขเศษส่วน (Fractional number) แสดงค่าน้ำหนักในรูปของเลข 10 ยกกำลังด้วยเลขจำนวนเต็มลบ ค่าของเลขยกกำลังจำนวนเต็มลบจะเพิ่มขึ้นจากซ้ายไปขวา โดยเริ่มที่ 10^{-1} ดังนั้นค่าเลขฐานสิบใดๆ หาได้จากผลรวมของตัวเลขในแต่ละตำแหน่งคูณกับค่าน้ำหนักของตำแหน่งนั้นๆ

$$\ldots\ 10^5 10^4 10^3 10^2 10^1 10^0 . 10^{-1} 10^{-2} 10^{-3} 10^{-4} 10^{-5} 10^{-6} \ldots$$

ตัวอย่างเช่นเลข 872.64 จะเห็นว่าเลข 8 มีค่าน้ำหนักเท่ากับ 100(10^2) เลข 7 มีค่าน้ำหนักเท่ากับ 10(10^1) ถ้าเลข 8 และ 7 สลับตำแหน่งกันค่าน้ำหนักของตัวเลขจะเปลี่ยนไปเช่นกัน ดังนั้นค่าน้ำหนักขึ้นอยู่กับตำแหน่งของตัวเลขนั้น ลักษณะดังกล่าวข้างต้นสามารถเขียนสมการโพลิโนเมียลของเลขฐานต่างๆ ได้ดังนี้

$$\begin{aligned} N &= a_n a_{n-1} a_{n-2} .. a_1 a_0 . a_{-1} a_{-2} .. a_{-m} \\ &= a_n r^n + a_{n-1} r^{n-1} + a_{n-2} r^{n-2} + \ldots + a_1 r^1 + a_0 r^0 + a_{-1} r^{-1} + \ldots + a_{-m} r^{-m} \\ &= \sum_{i=-m}^{n} a_i \cdot r^i \end{aligned}$$

เมื่อ N คือค่าของเลขฐานสิบ i คือค่าประจำหลักหรือในตำแหน่งนั้น a_i คือค่าของตัวเลขในหลักหรือตำแหน่งที่ i ของเลขฐานที่ต้องการหา r คือค่าเลขฐาน

ตัวอย่างที่ 2.1 จงแปลงเลขต่อไปนี้เป็นเลขฐานสิบ

(1) 101.01_2 (2) 136.2_8 (3) $1F.8_{16}$

<u>วิธีทำ</u>

(1) $$\begin{aligned} 101.01_2 &= (1\times 2^2)+(0\times 2^1)+(1\times 2^0)+(0\times 2^{-1})+(1\times 2^{-2}) \\ &= 4+1+0.25 \\ &= 5.25 \end{aligned}$$

(2) $$\begin{aligned} 136.2_8 &= (1\times 8^2)+(3\times 8^1)+(6\times 8^0)+(2\times 8^{-1}) \\ &= 64+24+6+0.25 \\ &= 94.25 \end{aligned}$$

(3) $1F.8_{16} = (1\times16^{1}) + (15\times16^{0}) + (8\times16^{-1})$

$= 16 + 15 + 0.5$

$= 31.5$

2) **การแปลงจากเลขฐานสิบเป็นเลขฐานสอง เลขฐานแปด เลขฐานสิบหก**

การแปลงเลขฐานสามารถพิจารณาเป็น 2 กรณี คือเลขจำนวนเต็ม(Integer) และเลขทศนิยมหรือเลขเศษส่วน (Fraction)

กรณีเลขจำนวนเต็ม มีขั้นตอนดังนี้

1. **นำเลขฐานที่ต้องการหาไปหารเลขฐานสิบ ทำการหารจนกระทั่งผลหารมีค่าเท่ากับศูนย์**
2. นำค่าเศษที่ได้จากการหาร มาเขียนเรียงลำดับจากเศษตัวสุดท้ายไปจนถึงค่าเศษตัวแรกหรือเขียนค่าเศษจากล่างขึ้นบน โดยค่า**เศษตัวแรกจะมีค่านัยสำคัญที่น้อยที่สุดและเศษตัวสุดท้ายจะมีค่านัยสำคัญมากที่สุด**

ตัวอย่างที่ 2.2 จงแปลงเลข 13_{10} เป็นเลขฐานสอง ฐานแปด ฐานสิบหก

<u>วิธีทำ</u>

(1) การแปลงเลข 13_{10} เป็นเลขฐานสอง

$13 \div 2 = 6$ เศษ 1 (LSB)

$6 \div 2 = 3$ เศษ 0

$3 \div 2 = 1$ เศษ 1

$1 \div 2 = 0$ เศษ 1 (MSB)

ดังนั้นจะได้ว่า $13_{10} = 1101_2$

(2) การแปลงเลข 13_{10} เป็นเลขฐานแปด

$13 \div 8 = 1$ เศษ 5

$1 \div 8 = 0$ เศษ 1

ดังนั้นจะได้ว่า $13_{10} = 15_8$

(3) การแปลงเลข 13_{10} เป็นเลขฐานสิบหก

$13 \div 16 = 0$ เศษ 13 (13 ในเลขฐานสิบหกมีค่าเท่ากับ D)

ดังนั้นจะได้ว่า $13_{10} = D_{16}$

ตัวอย่างที่ 2.3 จงแปลงเลข 58_{10} เป็นเลขฐานสอง ฐานแปด ฐานสิบหก

<u>วิธีทำ</u>

(1) การแปลงเลข 58_{10} เป็นเลขฐานสอง

$58 \div 2 = 29$ เศษ 0

$29 \div 2 = 14$ เศษ 1

$14 \div 2 = 7$ เศษ 0

$7 \div 2 = 3$ เศษ 1

$3 \div 2 = 1$ เศษ 1

$1 \div 2 = 0$ เศษ 1

ดังนั้นจะได้ว่า $58_{10} = 111010_2$

(2) การแปลงเลข 58_{10} เป็นเลขฐานแปด

$58 \div 8 = 7$ เศษ 2

$7 \div 8 = 0$ เศษ 7

ดังนั้นจะได้ว่า $58_{10} = 72_8$

(3) การแปลงเลข 58_{10} เป็นเลขฐานสิบหก

$58 \div 16 = 3$ เศษ 10 (10 ในเลขฐานสิบหกมีค่าเท่ากับ A)

$3 \div 16 = 0$ เศษ 3

ดังนั้นจะได้ว่า $58_{10} = 3A_{16}$

กรณีเลขทศนิยมหรือเศษส่วน มีขั้นตอนดังนี้

1. นำเลขฐานที่ต้องการหาไปคูณกับเลขทศนิยมที่กำหนด ทำการคูณเฉพาะส่วนที่เป็นเลขทศนิยมเท่านั้น จนกระทั่งผลคูณของเลขทศนิยมที่ได้มีค่าเท่ากับศูนย์
2. นำเลขจำนวนเต็มของผลคูณที่ได้ มาเขียนเรียงลำดับจากเลขจำนวนเต็มตัวแรกไปจนถึงจำนวนเต็มตัวสุดท้าย หรือเขียนเลขจำนวนเต็มจากบนลงมาล่าง โดยที่เลขจำนวนเต็มตัวแรกมีค่านัยสำคัญที่มากที่สุดและเลขจำนวนเต็มตัวสุดท้ายมีค่านัยสำคัญที่น้อยที่สุด

ตัวอย่างที่ 2.4 จงแปลงเลข 0.65625_{10} เป็นเลขฐานสอง ฐานแปด ฐานสิบหก

<u>วิธีทำ</u>

เลขฐานสอง	เลขฐานแปด	เลขฐานสิบหก
0.65625×	0.65625×	0.65625×
2	8	16
1.31250×	5.25000×	10.50000×
2	8	16
0.62500×	2.00000	8.00000
2		
1.25000×		
2		
0.50000×		
2		
1.00000		

ดังนั้นจะได้ว่า $0.65625_{10} = 0.10101_2 = 0.52_8 = 0.A8_{16}$

ตัวอย่างที่ 2.5 จงแปลงเลข 0.1875_{10} เป็นเลขฐานสอง ฐานแปด ฐานสิบหก

<u>วิธีทำ</u>

เลขฐานสอง	เลขฐานแปด	เลขฐานสิบหก
0.1875×	0.1875×	0.1875×
2	8	16
0.3750×	1.5000×	3.0000
2	8	
0.7500×	4.0000	
2		
0.5000×		
2		
1.0000		

ดังนั้นจะได้ว่า $0.1875_{10} = 0.0011_2 = 0.14_8 = 0.3_{16}$

ตัวอย่างที่ 2.6 จงแปลงเลข 0.3_{10} เป็นเลขฐานสอง ฐานแปด

<u>วิธีทำ</u>

เลขฐานสอง	เลขฐานแปด
$0.3\times$	$0.3\times$
$\underline{\quad 2}$	$\underline{\quad 8}$
$0.6\times$	$2.4\times$
$\underline{\quad 2}$	$\underline{\quad 8}$
$1.2\times$	$3.2\times$
$\underline{\quad 2}$	$\underline{\quad 8}$
0.4	1.6
$\vdots$	$\vdots$

ดังนั้นจะได้ว่า $0.3 \approx (0.010...)_2 \approx (0.231...)_8$

ตัวอย่างที่ 2.7 จงแปลงเลข 0.57251_{10} เป็นเลขฐานสอง

<u>วิธีทำ</u>

$$
\begin{array}{r}
0.57251\times \\
\underline{\qquad 2} \\
1.14502\times \\
\underline{\qquad 2} \\
0.29004\times \\
\underline{\qquad 2} \\
0.58008\times \\
\underline{\qquad 2} \\
1.16016\times \\
\underline{\qquad 2} \\
0.32032 \\
\vdots
\end{array}
$$

ดังนั้น $0.57251 \approx 0.10010_2 ...$

3) **การแปลงระหว่างเลขฐานสอง ฐานแปด ฐานสิบหก**

จากความสัมพันธ์ระหว่างเลขฐานสอง เลขฐานแปดและเลขฐานสิบหก แสดงดังตารางที่ 2.4 จะเห็นว่าเลขฐานแปด 1 หลักสามารถแทนด้วยเลขฐานสองจำนวน 3 บิต และเลขฐานสิบหก 1 หลักสามารถแทนด้วยเลขฐานสองจำนวน 4 บิต ทำให้การแปลงระหว่างเลขฐานเหล่านี้ง่ายขึ้น

ตารางที่ 2.4 ความสัมพันธ์ระหว่างเลขฐานสอง เลขฐานแปดและเลขฐานสิบหก

เลขฐานแปด	เลขฐานสอง	เลขฐานสิบหก	เลขฐานสอง
0	000	0	0000
1	001	1	0001
2	010	2	0010
3	011	3	0011
4	100	4	0100
5	101	5	0101
6	110	6	0110
7	111	7	0111
		8	1000
		9	1001
		A	1010
		B	1011
		C	1100
		D	1101
		E	1110
		F	1111

การแปลงระหว่างเลขฐานสองกับเลขฐานแปด มีขั้นตอนดังนี้

1. การแปลงจากเลขฐานสองเป็นเลขฐานแปด เริ่มจากแบ่งเลขฐานสองออกเป็นกลุ่มๆ ละ 3 บิต ถ้าแบ่งกลุ่มไปทางขวาของจุดทศนิยมแล้วมีจำนวนไม่ครบ 3 บิต ให้เติมบิตศูนย์ไปทางขวาจนกระทั่งกลุ่มนั้นมีจำนวนครบ 3 บิต ถ้าแบ่งกลุ่มไปทางซ้ายของจุดทศนิยมแล้วมีจำนวนไม่ครบ 3 บิต ให้เติมบิตศูนย์ไปทางซ้ายจนกระทั่งกลุ่มนั้นมีจำนวนครบ 3 บิต เขียนเลขฐานแปดแทนเลขฐานสองในแต่ละกลุ่ม

2. การแปลงจากเลขฐานแปดเป็นเลขฐานสอง เลขฐานแปดแต่ละหลักสามารถแทนด้วยเลขฐานสองจำนวน 3 บิต นำเลขฐานสองมาเขียนเรียงต่อกัน เลขฐานสองที่เป็นศูนย์ที่อยู่ทางด้านขวาสุดของเลขทศนิยมหรือซ้ายสุดของเลขจำนวนเต็ม สามารถตัดทิ้งได้

การแปลงระหว่างเลขฐานสองกับเลขฐานสิบหก มีขั้นตอนดังนี้

1. การแปลงจากเลขฐานสองเป็นเลขฐานสิบหก เริ่มจากแบ่งเลขฐานสองออกเป็นกลุ่มๆ ละ 4 บิต ถ้าแบ่งกลุ่มไปทางขวาของจุดทศนิยมแล้วมีจำนวนไม่ครบ 4 บิต ให้เติมบิตศูนย์ไปทางขวาจนกระทั่งกลุ่มนั้นมีจำนวนครบ 4 บิต ถ้าแบ่งกลุ่มไปทางซ้ายของจุดทศนิยมแล้วมีจำนวนไม่ครบ 4 บิต ให้เติมบิตศูนย์ไปทางซ้ายจนกระทั่งกลุ่มนั้นมีจำนวนครบ 4 บิต เขียนเลขฐานสิบหกแทนเลขฐานสองในแต่ละกลุ่ม

2. การแปลงจากเลขฐานสิบหกเป็นเลขฐานสอง เลขฐานสิบหกแต่ละหลักสามารถแทนด้วยเลขฐานสองจำนวน 4 บิต นำเลขฐานสองมาเขียนเรียงต่อกัน เลขฐานสองที่เป็นศูนย์ที่อยู่ทางด้านขวาสุดของเลขทศนิยมหรือซ้ายสุดของเลขจำนวนเต็ม สามารถตัดทิ้งได้

ตัวอย่างที่ 2.8 จงแปลงเลขฐานต่อไปนี้ เป็นเลขฐานสอง

(1) 275_8 (2) 13.75_8 (3) $1A6_{16}$ (4) 13.5_{16}

<u>วิธีทำ</u>

(1) 275_8 = 010 111 101 = 10111101_2

(2) 13.75_8 = 001 011.111 101 = 1011.111101_2

(3) $1A6_{16}$ = 0001 1010 0110 = 110100110_2

(4) 13.5_{16} = 0001 0011.0101 = 10011.0101_2

ตัวอย่างที่ 2.9 จงแปลงเลขฐานสองต่อไปนี้ เป็นเลขฐานแปดและสิบหก

(1) 10101_2 (2) 10111.1101_2

<u>วิธีทำ</u>

(1) 10101_2 = 010 101 = 25_8

10101_2 = 0001 0101 = 15_{16}

(2) 10111.1101_2 = 010 111. 110 100 = 27.64_8

10111.1101_2 = 0001 0111.1101 = $17.D_{16}$

2.3 การดำเนินการทางคณิตศาสตร์ของระบบเลขฐาน

เลขฐานต่างๆ สามารถทำการบวก ลบ คูณ หาร โดยตัวตั้งและตัวกระทำต้องเป็นเลขฐานเดียวกัน ในกรณีที่มีเลขฐานไม่เหมือนกันให้ทำการแปลงเป็นเลขฐานเดียวกันก่อน โดยใช้วิธีการแปลงเลขฐานที่กล่าวมาแล้วก่อนหน้านี้

การบวกเลขฐาน (Addition) มีขั้นตอนดังนี้

1. จัดวางตำแหน่งตัวตั้งกับตัวบวกให้หลักที่มีค่าน้ำหนักเหมือนกัน อยู่ตรงกัน
2. ทำการบวกแต่ละหลักเข้าด้วยกัน โดยเริ่มจากหลักที่อยู่ทางขวาสุดก่อน การบวกเลขฐานต่างๆ จะต่างกันที่ตัวทด โดยเลขฐานสองจะทดที่ 2 เลขฐานแปดจะทดที่ 8 เลขฐานสิบจะทดที่ 10 เลขฐานสิบหกจะทดที่ 16 ตามลำดับ

ตัวอย่างที่ 2.10 จงบวกเลขฐานต่อไปนี้

(1) $1101_2+100101_2$ (2) 235_8+126_8 (3) $207A_{16}+8194_{16}$

<u>วิธีทำ</u>

```
     1 1 0 1+      2 3 5+      2 0 7 A+
1 0 0 1 0 1        1 2 6       8 1 9 4
1 1 0 0 1 0        3 6 3       A 2 0 E
```

ดังนั้นจะได้ว่า $1101_2+100101_2 = 110010_2$

$235_8+126_8 = 363_8$

$207A_{16}+8194_{16} = A20E_{16}$

การลบเลขฐาน (Subtraction)

การลบเลขฐานมี 2 วิธี คือ การลบโดยตรงเป็นการลบเช่นเดียวกับเลขฐานสิบ และการลบแบบคอมพลีเมนต์เป็นการลบที่ใช้ในวงจรคอมพิวเตอร์

การลบโดยตรง มีขั้นตอนดังนี้

1. จัดวางตำแหน่งของตัวตั้งและตัวลบที่มีค่าน้ำหนักเหมือนกัน ให้อยู่ตรงกัน
2. ทำการลบจากหลักที่อยู่ทางขวาสุดก่อน
3. ถ้าตัวตั้งมีค่ามากกว่าตัวลบสามารถลบได้เลย ผลลัพธ์ที่ได้มีค่าเป็นบวก ถ้าตัวตั้งน้อยกว่าตัวลบ ให้ทำการสลับตำแหน่งของตัวตั้งกับตัวลบ ผลลัพธ์ที่ได้มีค่าเป็นลบ (ให้ใส่เครื่องหมายลบไว้ข้างหน้า)

ตัวอย่างที่ 2.11 จงลบเลขฐานต่อไปนี้ ด้วยวิธีการลบโดยตรง

(1) $110010_2 - 101101_2$ (2) 435_8-176_8 (3) $C51_{16}-AB3_{16}$

<u>วิธีทำ</u>

```
1 1 0 0 1 0-      4 3 5₈-      C 5 1₁₆-
1 0 1 1 0 1       1 7 6₈       A B 3₁₆
0 0 0 1 0 1       2 3 7₈       1 9 E₁₆
```

ดังนั้นจะได้ว่า $110010_2 - 101101_2 = 101_2$

$435_8-176_8 = 237_8$

$C51_{16}-AB3_{16} = 19E_{16}$

การลบแบบคอมพลีเมนต์

เป็นเทคนิคการลบเลขอย่างหนึ่งที่ช่วยให้การลบเลขฐานต่างๆ เร็วขึ้น เนื่องจากระบบเลขฐานที่คนเราไม่คุ้นเคย เช่นเลขฐานแปด ฐานสิบหก เมื่อทำการลบโดยตรง จำเป็นต้องมีการยืมจากตัวเลขที่อยู่ข้างหน้ามา ทำให้การลบยุ่งยากและสับสนได้ การลบแบบคอมพลีเมนต์ทำได้โดยการหาค่าคอมพลีเมนต์ของตัวลบและนำไปบวกกับตัวตั้ง วิธีการลบแบบนี้นิยมใช้ในวงจรของคอมพิวเตอร์ ทำให้สามารถใช้วงจรบวกเพียงวงจรเดียว เพื่อใช้ในการลบและบวกเลขฐานต่างๆ ได้

คอมพลีเมนต์ของเลขฐานต่างๆ มี 2 แบบ คือ r-1' s complement และ r' s complement เมื่อ r คือเลขฐาน ดังนั้นเลขฐานสองมี 1's complement ($\overline{1S}$) และ 2's complement ($\overline{2S}$) เลขฐานแปดมี 7's complement ($\overline{7S}$) และ 8's complement ($\overline{8S}$) เลขฐานสิบมี 9's complement ($\overline{9S}$) และ 10's complement ($\overline{10S}$) เลขฐานสิบหกมี 15's complement ($\overline{15S}$) และ 16's complement ($\overline{16S}$) ค่าคอมพลีเมนต์ของเลขฐานแสดงดังตารางที่ 2.5

การลบแบบ r-1's complement มีขั้นตอนดังนี้

1. ทำตัวตั้งและตัวลบให้มีจำนวนหลักเท่ากัน โดยการเพิ่มบิต 0 เข้าไป
2. หาค่า r-1's complement ของตัวลบ จากนั้นนำค่า r-1's complement ของตัวลบมาบวกกับตัวตั้งทีละหลัก โดยเริ่มจากหลักที่อยู่ทางขวาสุดก่อนเสมอ
3. ผลลัพธ์หรือคำตอบที่ได้จะพิจารณาเป็น 2 กรณี คือ กรณีที่มีตัวทดสุดท้าย (End Around Carry , EAC) ให้นำตัวทดไปบวกกับหลักที่อยู่ทางขวาสุด จะได้ผลลัพธ์ที่ต้องการและมีค่าเป็นบวก กรณีที่ไม่มีตัวทดสุดท้าย ให้หาค่า r-1's complement ของผลบวก จะได้ผลลัพธ์ที่ต้องการและมีค่าเป็นลบ

การลบแบบ r's complement มีขั้นตอนดังนี้

1. ทำตัวตั้งและตัวลบให้มีจำนวนหลักเท่ากัน โดยการเพิ่มบิต 0 เข้าไป
2. หาค่า r 's complement ของตัวลบ จากนั้นนำค่า r 's complement ของตัวลบมาบวกกับตัวตั้งทีละหลัก โดยเริ่มจากหลักที่อยู่ทางขวาสุดก่อนเสมอ
3. ผลลัพธ์หรือคำตอบที่ได้จะพิจารณาเป็น 2 กรณี คือ กรณีที่มีตัวทดสุดท้าย (End Around Carry , EAC) ให้ตัดตัวทดทิ้งไป จะได้ผลลัพธ์ที่ต้องการและมีค่าเป็นบวก กรณีที่ไม่มีตัวทดสุดท้ายให้หาค่า r 's complement ของผลบวก จะได้ผลลัพธ์ที่ต้องการและมีค่าเป็นลบ

ตารางที่ 2.5 ค่าคอมพลีเมนต์ของเลขฐานสอง ฐานแปด ฐานสิบ ฐานสิบหก

Binary	1's complement	2's complement
0	1	10
1	0	1

Octal	7's complement	8's complement
0	7	10
1	6	7
2	5	6
3	4	5
4	3	4
5	2	3
6	1	2
7	0	1

Decimal	9's complement	10's complement
0	9	10
1	8	9
2	7	8
3	6	7
4	5	6
5	4	5
6	3	4
7	2	3
8	1	2
9	0	1

Hexadecimal	15's complement	16's complement
0	F	10
1	E	F
2	D	E
3	C	D
4	B	C
5	A	B
6	9	A
7	8	9
8	7	8
9	6	7
A	5	6
B	4	5
C	3	4
D	2	3
E	1	2
F	0	1

ตารางที่ 2.6 สรุปการลบแบบคอมพลีเมนต์

การลบ	มีตัวทดสุดท้าย (EAC) (ผลลัพธ์เป็นบวก)	ไม่มีตัวทดสุดท้าย (EAC) (ผลลัพธ์เป็นลบ)
r-1's complement	นำ EAC + ผลบวกที่ได้หลักขวาสุด	ทำคอมพลีเมนต์ของผลบวกที่ได้
r' s complement	ตัด EAC ทิ้งไป	

ตัวอย่างที่ 2.12 จงหาค่า $\overline{1S}$ และ $\overline{2S}$ ของเลขฐานสองต่อไปนี้

(1) 101101_2 (2) 1011_2

<u>วิธีทำ</u>

(1) 1' s complement ของ 101101_2 = 010010

2' s complement ของ 101101_2 = 1's complement +1

= 010010+1

= 010011

(2) 1' s complement ของ 1011_2 = 0100

2' s complement ของ 1011_2 = 1's complement +1

= 0100+1

= 0101

ตัวอย่างที่ 2.13 จงหาค่า $\overline{7S}$ และ $\overline{8S}$ ของเลขฐานแปดต่อไปนี้

(1) 325_8 (2) 6072_8

<u>วิธีทำ</u>

(1) 7' s complement ของ 325_8 = 452

8' s complement ของ 325_8 = 7's complement +1

= 452+1

= 453

(2) 7' s complement ของ 6072_8 = 1705

8' s complement ของ 6072_8 = 7's complement +1

= 1705+1

= 1706

ตัวอย่างที่ 2.14 จงหาค่า $\overline{9S}$ และ $\overline{10S}$ ของเลขฐานสิบต่อไปนี้

(1) 325_{10} (2) 6072_{10}

<u>วิธีทำ</u>

(1) 9' s complement ของ 325_{10} = 674

10' s complement ของ 325_{10} = 9's complement +1

= 674+1

= 675

(2) 9' s complement ของ 6072_{10} = 3927

10' s complement ของ 6072_{10} = 9's complement +1

= 3927+1

= 3928

ตัวอย่างที่ 2.15 จงหาค่า $\overline{15S}$ และ $\overline{16S}$ ของเลขฐานสิบหกต่อไปนี้

(1) 325_{16} (2) 6072_{16}

<u>วิธีทำ</u>

(1) 15' s complement ของ 325_{16} = CDA

16' s complement ของ 325_{16} = 15's complement +1

= CDA+1

= CDB

(2) 15' s complement ของ 6072_{16} = 9F8D

16' s complement ของ 6072_{16} = 15's complement +1

= 9F8D+1

= 9F8E

ตัวอย่างที่ 2.16 จงลบเลขฐานต่อไปนี้ด้วยวิธีการคอมพลีเมนต์

(1) 110010_2-101101_2 (2) 435_8-176_8

(3) 653_{10}-475_{10} (4) $C51_{16}$-$AB3_{16}$

<u>วิธีทำ</u>

(1) 110010_2-101101_2

การลบโดยใช้ 1's complement

1's complement ของตัวลบ 101101 = 010010

```
  1 1 0 0 1 0+
  0 1 0 0 1 0
 ------------
1 0 0 0 1 0 0+
 └──────────► 1
 ------------
  0 0 0 1 0 1
```

การลบโดยใช้ 2's complement

2's complement ของตัวลบ 101101 = 010010+1

= 010011

```
  1 1 0 0 1 0+
  0 1 0 0 1 1
 ------------
1 0 0 0 1 0 1
```

นั่นคือ $110010_2 - 101101_2 = 101_2$

(2) $435_8 - 176_8$

การลบโดยใช้ 7's complement

7's complement ของตัวลบ 176 = 601

```
  4 3 5+
  6 0 1
 --------
1 2 3 6+
 └──► 1
 --------
  2 3 7
```

การลบโดยใช้ 8's complement

8's complement ของตัวลบ 176 = 601+1

= 602

```
  4 3 5+
  6 0 2
 --------
1 2 3 7
```

นั่นคือ $435_8 - 176_8 = 237_8$

(3) $653_{10} - 475_{10}$

การลบโดยใช้ 9 's complement

9's complement ของตัวลบ 475_{10} = 524

```
  6 5 3+
  5 2 4
 --------
1 1 7 7+
 └──► 1
 --------
  1 7 8
```

การลบโดยใช้ 10 's complement

10's complement ของตัวลบ 475_{10} = 524+1

= 525

```
  6 5 3+
  5 2 5
 --------
1 1 7 8
```

นั่นคือ $653_{10} - 475_{10} = 178_{10}$

(4) $C51_{16} - AB3_{16}$

การลบโดยใช้ 15 's complement

15's complement ของตัวลบ AB3 = 54C

```
  C 5 1+
  5 4 C
 -------
1 1 9 D+
 |
 ------> 1
 -------
  1 9 E
 =======
```

การลบโดยใช้ 16's complement

16's complement ของตัวลบ AB3 = 54C +1

= 54D

```
  C 5 1+
  5 4 D
 -------
1 1 9 E
 =======
```

นั่นคือ $C51_{16}-AB3_{16} = 19E_{16}$

ตัวอย่างที่ 2.17 จงลบเลขฐานต่อไปนี้ด้วยวิธีการคอมพลีเมนต์

(1) $101101_2-110010_2$ (2) 176_8-435_8

(3) $475_{10}-653_{10}$ (4) $AB3_{16}- C51_{16}$

<u>วิธีทำ</u>

(1) $101101_2-110010_2$

การลบโดยใช้ 1's complement

1's complement ของตัวลบ 110010 = 001101

```
  1 0 1 1 0 1+
  0 0 1 1 0 1
 -------------
  1 1 1 0 1 0
      ⇓ 1s
  0 0 0 1 0 1
 =============
```

การลบโดยใช้ 2's complement

2's complement ของตัวลบ 110010 = 001101+1

= 001110

```
  1 0 1 1 0 1+
  0 0 1 1 1 0
 -------------
  1 1 1 0 1 1
      ⇓ 2s
  0 0 0 1 0 1
 =============
```

นั่นคือ $101101_2-110010_2= -101_2$

(2) 176_8-435_8

การลบโดยใช้ 7's complement

7's complement ของตัวลบ 435 = 342

```
  1 7 6+
  3 4 2
 -------
  5 4 0+
      ⇓ 7s
  2 3 7
 =======
```

การลบโดยใช้ 8's complement

8's complement ของตัวลบ 435 = 342+1

= 343

```
  1 7 6+
  3 4 3
 -------
  5 4 1
      ⇓ 8s
  2 3 7
 =======
```

นั่นคือ $176_8 - 435_8 = -237_8$

(3) $475_{10}-653_{10}$

การลบโดยใช้ 9 's complement

9's complement ของตัวลบ 653 = 346

```
4 7 5+
3 4 6
------
8 2 1
  ⇓ 9s
1 7 8
```

การลบโดยใช้ 10 's complement

10's complement ของตัวลบ 635 = 346+1

= 347

```
4 7 5+
3 4 7
------
8 2 2
  ⇓ 10s
1 7 8
```

นั่นคือ $475_{10}-653_{10} = -178_{10}$

(4) $AB3_{16}- C51_{16}$

การลบโดยใช้ 15 's complement

15's complement ของตัวลบ C51 = 3AE

```
A B 3+
3 A E
------
E 6 1
  ⇓ 15s
1 9 E
```

การลบโดยใช้ 16's complement

16's complement ของตัวลบ C51 = 3AE +1

= 3AF

```
A B 3+
3 A F
------
E 6 2
  ⇓ 16s
1 9 E
```

นั่นคือ $AB3_{16} - C51_{16} = -19E_{16}$

การคูณเลขฐาน (Multiplication)

การคูณเลขฐานสอง เลขฐานแปด เลขฐานสิบหก ใช้หลักการเดียวกับเลขฐานสิบ โดยนำตัวคูณไปคูณกับตัวตั้งทีละหลัก แล้วนำผลคูณที่ได้ในแต่ละหลักมาบวกกัน สามารถใช้ตารางการคูณเลขฐานช่วยในการคูณได้

ตัวอย่างที่ 2.18 จงหาผลคูณของเลขฐานต่อไปนี้

(1) $1010_2 \times 101_2$ (2) $247_8 \times 36_8$ (3) $1B2_{16} \times A8_{16}$

<u>วิธีทำ</u>

(1) $1010_2 \times 101_2$

```
        1 0 1 0×
          1 0 1
        -------
        1 0 1 0
      0 0 0 0   +
    1 0 1 0
    ---------------
    1 1 0 0 1 0
```

ดังนั้นจะได้ว่า $1010_2 \times 101_2 = 110010_2$

(2) $247_8 \times 36_8$

แบบที่ 1

```
        2 4 7×
          3 6
        -------
        5 2
      3 0
    1 4       +
    -----------
      2 5
    1 4
    6
    -----------
  1 1 6 2 2
```

แบบที่ 2

```
          2   4   7×
              3   6
        -------------
        1  4+3 0+5  2
      6+1  4+2   5
    -------------------
  1   1   6   2   2
```

ดังนั้นจะได้ว่า $247_8 \times 36_8 = 11622_8$

(3) $1B2_{16} \times A8_{16}$

แบบที่ 1

แบบที่ 2

```
          1   B   2×
              A   8
        -------------
        8+5 8+1   0
    A+6 E+1   4
    -------------------
  1   1   C   D   0
```

```
        1 B 2×
          A 8
-----------------
          1 0
        5 8
        8       +
-----------------
        1 4
    6 E
    A
-----------------
  1 1 C D 0
```

ดังนั้นจะได้ว่า $1B2_{16} x A8_{16} = 11CD0_{16}$

ตารางที่ 2.7 ตารางการคูณของเลขฐานสอง

×		ตัวคูณ	
		0	1
ตัวตั้ง	0	0	0
	1	0	1

ตารางที่ 2.8 ตารางการคูณของเลขฐานแปด

	0							
	0	0	0	0	0	0	0	0
	0	1	2	3	4	5	6	7
	0	2	4	6	10	12	14	16
	0	3	6	11	14	17	22	25
	0	4	10	14	20	24	30	34
	0	5	12	17	24	31	36	43
	0	6	14	22	30	36	44	52
	0	7	16	25	34	43	52	61

ตารางที่ 2.9 ตารางการคูณของเลขฐานสิบหก

×	ตัวคูณ															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	0	2	4	6	8	A	C	E	10	12	14	16	18	1A	1C	1E
3	0	3	6	9	C	F	12	15	18	1B	1E	21	24	27	2A	2D
4	0	4	8	C	10	14	18	1C	20	24	28	2C	30	34	38	3C
5	0	5	A	F	14	19	1E	23	28	2D	32	37	3C	41	46	4B
6	0	6	C	12	18	1E	24	2A	30	36	3C	42	48	4E	54	5A
7	0	7	E	15	1C	23	2A	31	38	3E	46	4D	54	5B	62	69
8	0	8	10	18	20	28	30	38	40	48	50	58	60	68	70	78
9	0	9	12	1B	24	2D	36	3E	48	51	5A	63	6C	75	7E	87
A	0	A	14	1E	28	32	3C	46	50	5A	64	6E	78	82	8C	96
B	0	B	16	21	2C	37	42	4D	58	63	6E	79	84	8F	9A	A5
C	0	C	18	24	30	3C	48	54	60	6C	78	84	90	9C	A8	B4
D	0	D	1A	27	34	41	4E	5B	68	75	82	8F	9C	A9	B6	C3
E	0	E	1C	2A	38	46	54	62	70	7E	8C	9A	A8	B6	C4	D2
F	0	F	1E	2D	3C	4B	5A	69	78	87	96	A5	B4	C3	D2	E1

การหารเลขฐาน (Division)

การหารเลขฐานสอง เลขฐานแปด เลขฐานสิบหก ใช้หลักการหารเช่นเดียวกับเลขฐานสิบ

ตัวอย่างที่ 2.19 จงหาผลหารของเลขฐานต่อไปนี้

(1) $10100_2 \div 100_2$ (2) $11622_8 \div 247_8$ (3) $ED.54_{16} \div C_{16}$

<u>วิธีทำ</u>

(1) $10100_2 \div 100_2$

$$
\begin{array}{r}
1\ 0\ 1 \\
100\overline{)1\ 0\ 1\ 0\ 0} \\
1\ 0\ 0 \\
\hline
1\ 0\ 0 \\
1\ 0\ 0 \\
\hline\hline
\end{array}
$$

ดังนั้นจะได้ว่า $10100_2 \div 100_2 = 101_2$

(2) $11622_8 \div 247_8$

```
            3 6
247)1 1 6 2 2
    7 6 5
    -------
    1 7 5 2
    1 7 5 2
    =======
```

ดังนั้นจะได้ว่า $11622_8 \div 247_8 = 36_8$

(3) $ED.54_{16} \div C_{16}$

```
    1 3 . C 7
C)E D . 5 4
  C
  -----------
  2 D
  2 4
  -----------
    9 . 5
    9 . 0
  -----------
        5 4
        5 4
  -----------
```

ดังนั้นจะได้ว่า $ED.54_{16} \div C_{16} = 13.C7_{16}$

2.4 ตัวเลขแบบคิดเครื่องหมาย (Signed Number)

การดำเนินการทางคณิตศาสตร์ของเลขฐานสอง ผลลัพธ์ที่ได้มีทั้งค่าบวกและค่าลบ แต่ในวงจรดิจิตอลหรือการประมวลผลด้วยคอมพิวเตอร์ จะใช้เลขฐานสองแทนเครื่องหมายบวกและลบ โดยกำหนดให้บิตแรกซ้ายสุดหรือบิตสูงแทนบิตเครื่องหมาย (Signed bit) ถ้าบิตแรกมีค่าเป็น 0 ใช้แทนเครื่องหมายบวกและถ้าบิตแรกมีค่าเป็น 1 ใช้แทนเครื่องหมายลบ ส่วนบิตที่เหลือแทนขนาดของตัวเลข (Magnitude) กรณีที่เป็นเลขที่มีเครื่องหมายบวกต้องเขียน 0 นำหน้าไว้ด้วยเสมอ ไม่สามารถตัดทิ้งได้เนื่องจากเป็นบิตเครื่องหมาย ซึ่งแตกต่างจากตัวเลขที่ไม่คิดเครื่องหมายที่สามารถตัดศูนย์ที่อยู่ข้างหน้าทิ้งได้ ค่าของตัวเลขที่คิดเครื่องหมายสามารถมีค่าได้ตั้งแต่ (-2^{n-1}) ถึง $(2^{n-1}-1)$ เมื่อ n คือจำนวนบิต ตัวอย่างเช่น เลขที่คิดเครื่องหมายขนาด 4 บิต มีค่าได้ตั้งแต่ (-2^{4-1}) ถึง $(2^{4-1}-1)$ หรือ (-8) – (+7) เป็นต้น รูปแบบตัวเลขที่คิดเครื่องหมายมี 3 แบบดังนี้

1. แบบ Signed-Magnitude (SM)

กำหนดให้บิตแรกซ้ายสุดแทนบิตเครื่องหมาย ส่วนบิตที่เหลือแทนขนาดของตัวเลข เช่น การเขียนเลข +6 และ –6 แบบ SM เริ่มจากการแปลงเลข 6 ที่อยู่ในรูปของเลขฐานสิบไปเป็นเลขฐานสอง จะได้ $6 = 110_2$ ซึ่งค่า 110_2 เป็นบิตที่แสดงค่าขนาดของตัวเลข ดังนั้นการเขียน +6 แบบ SM ให้ใส่บิต 0 (+) เข้าไปข้างหน้า จะได้เลข SM ของ $+6 = 0110_2$ และการเขียน -6 แบบ SM ให้ใส่บิต 1 (-) เข้าไปข้างหน้า จะได้เลข SM ของ $-6 = 1110_2$

$+6 = 0\underline{110}_2$

บิตเครื่องหมาย ← → บิตขนาดของเลข 6

$-6 = 1\underline{110}_2$

บิตเครื่องหมาย ← → บิตขนาดของเลข 6

2. แบบ Signed 1's complement

กำหนดให้บิตแรกซ้ายสุดแทนบิตเครื่องหมาย เช่นเดียวกับแบบ SM ส่วนบิตที่เหลือแทนขนาดของตัวเลข ซึ่งมีรูปแบบการเก็บข้อมูลดังนี้ ถ้าบิตเครื่องหมายมีค่าเป็น 0 (+) บิตที่เหลือจะเหมือนกับแบบ SM ถ้าบิตเครื่องหมายมีค่าเป็น 1 (-) บิตที่เหลือจะเก็บข้อมูลในรูปของ 1's complement ตัวอย่างเช่น การเขียนเลข +6 และ –6 แบบ Signed 1's complement เริ่มจากการแปลงเลข 6 ที่อยู่ในรูปของเลขฐานสิบเป็นเลขฐานสองจะได้ $6 = 110_2$ ซึ่งค่า 110_2 เป็นบิตที่แสดงค่าขนาดของตัวเลข ดังนั้นการเขียน +6 แบบ Signed 1's complement ให้ใส่บิต 0 (+) เข้าไปข้างหน้า จะได้เลข Signed 1's complement ของ $+6 = 0110_2$ และการเขียน -6 แบบ Signed 1's complement ต้องหาค่า 1's complement ของ 110_2 ซึ่งมีค่าเท่ากับ 001 จากนั้นให้ใส่บิต 1 (-) เข้าไปข้างหน้า จะได้เลข Signed 1's complement ของ $-6 = 1001_2$

$+6 = 0\underline{110}_2$

บิตเครื่องหมาย ← → บิตขนาดของเลข 6

$-6 = 1\underline{001}_2$

บิตเครื่องหมาย ← → บิตขนาดของเลข 6 ในรูป 1's complement

3. แบบ Signed 2's complement

บิตแรกซ้ายสุดแทนบิตเครื่องหมาย ส่วนบิตที่เหลือแทนขนาดของตัวเลข ซึ่งมีรูปแบบการเก็บข้อมูลดังนี้ ถ้าบิตเครื่องหมายมีค่าเป็น 0 (+) บิตที่เหลือจะเหมือนกับแบบ SM ถ้าบิตเครื่องหมายมีค่าเป็น 1 (-) บิตที่เหลือจะเก็บในรูปของ 2's complement ตัวอย่างเช่น การเขียน +6 และ –6 แบบ Signed 2's complement จะเริ่มจากการแปลงเลข 6 ที่อยู่ในรูปของเลขฐานสิบเป็นเลขฐานสองจะได้ $6 = 110_2$ ซึ่งค่า 110_2 เป็นบิตที่แสดงขนาดของตัวเลข ดังนั้นการเขียน +6 แบบ

Signed 2's complement ให้ใส่บิต 0 (+) เข้าไปข้างหน้า จะได้เลข Signed 2's complement ของ +6 = 0110_2 และการเขียน -6 แบบ Signed 2's complement จะต้องหาค่า 2's complement ของ 110_2 ซึ่งมีค่าเท่ากับ 010 จากนั้นให้ใส่บิต 1 (-) เข้าไปข้างหน้า จะได้เลข Signed 2's complement ของ -6 = 1010_2

$+6 = 0\underline{110}_2$

บิตเครื่องหมาย ← → บิตขนาดของเลข 6

$-6 = 1\underline{010}_2$

บิตเครื่องหมาย ← → บิตขนาดของเลข 6 ในรูป 2's complement

ตัวอย่างที่ 2.20 จงแปลงเลขฐานสิบต่อไปนี้เป็นตัวเลขที่คิดเครื่องหมายในรูปแบบ SM, Signed 1's complement และ Signed 2's complement

(1) -28 (2) 85 (3) -1.25

<u>วิธีทำ</u>

(1) -28

$28_{10} = 11100_2$

1's complement ของ $11100_2 = 00011_2$

2's complement ของ $11100_2 = 00100_2$

ดังนั้นจะได้ว่า

เลข SM ของ –28 = 111100_2

เลข Signed 1's complement ของ –28 = 100011_2

เลข Signed 2's complement ของ –28 = 100100_2

(2) 85

$85_{10} = 1010101_2$

ดังนั้นจะได้ว่า

เลข SM ของ 85 = 01010101_2

เลข Signed 1's complement ของ 85 = 01010101_2

เลข Signed 2's complement ของ 85 = 01010101_2

(3) -1.25

$1.25_{10} = 1.01_2$

1's complement ของ $1.01_2 = 0.10_2$

2's complement ของ $1.01_2 = 0.11_2$

ดังนั้นจะได้ว่า

เลข SM ของ -1.25 = 11.01_2

เลข Signed 1's complement ของ -1.25 = 10.10_2

เลข Signed 2's complement ของ -1.25 = 10.11_2

ตัวอย่างที่ 2.21 จงแปลงเลข SM ต่อไปนี้ให้อยู่ในรูปของเลขฐานสิบ

(1) 011001_2 (2) 111001_2

<u>วิธีทำ</u>

(1) 011001_2

ขนาดของเลข 011001_2 คือ 11001_2 ดังนั้น $11001_2 = 25_{10}$

บิตเครื่องหมายเป็น 0 แสดงเครื่องหมายบวก ดังนั้น $011001_2 = +25_{10}$

(2) 111001_2

ขนาดของเลข 111001_2 คือ 11001_2 ดังนั้น $11001_2 = 25_{10}$

บิตเครื่องหมายเป็น 1 แสดงเครื่องหมายลบ ดังนั้น $111001_2 = -25_{10}$

ตัวอย่างที่ 2.22 จงแปลงเลข Signed 1's complement ให้อยู่ในรูปของเลขฐานสิบ

(1) 10010_2 (2) 011010_2

<u>วิธีทำ</u>

(1) 10010_2

ค่าขนาดของเลข 10010_2 คือ 0010_2 ดังนั้น 1's complement ของ 0010_2 = $1101_2 = 13_{10}$

บิตเครื่องหมายเป็น 1 แสดงเครื่องหมายลบ ดังนั้น $10010_2 = -13_{10}$

(2) 011010_2

ค่าขนาดของเลข 011010_2 คือ 11010_2 ดังนั้น $11010_2 = 26_{10}$

บิตเครื่องหมายเป็น 0 แสดงเครื่องหมายบวก ดังนั้น $011010_2 = +26_{10}$

ตัวอย่างที่ 2.23 จงเขียนเลข Signed 2's complement ให้อยู่ในรูปของเลขฐานสิบ

(1) 00101101_2 (2) 10010011_2

วิธีทำ

(1) 00101101_2

ขนาดของเลข 00101101_2 คือ 0101101_2 ดังนั้น $0101101_2 = 45_{10}$

บิตเครื่องหมายเป็น 0 แสดงเครื่องหมายบวก ดังนั้น $00101101_2 = +45_{10}$

(2) 10010011_2

ขนาดของเลข 10010011_2 คือ 0010011_2 ดังนั้น 2's complement ของ

$0010011_2 = 1101101_2 = 109_{10}$

บิตเครื่องหมายเป็น 1 แสดงเครื่องหมายลบ ดังนั้น $10010011_2 = -109_{10}$

2.5 การบวกและลบเลข Signed 2's complement

จากตารางที่ 2.10 จะเห็นว่าตัวเลขที่คิดเครื่องหมายขนาด 4 บิต จะมีบิตเครื่องหมายจำนวน 1 บิตและบิตขนาดจำนวน 3 บิต มีค่าตั้งแต่ –8 ถึง +7 จากตารางที่ 2.11 จะเห็นว่าตัวเลขที่คิดเครื่องหมายขนาด 8 บิต จะมีบิตเครื่องหมายจำนวน 1 บิตและบิตขนาดจำนวน 7 บิต มีค่าตั้งแต่ –128 ถึง +127 จากตารางทั้งสองจะเห็นได้ว่าเลขที่คิดเครื่องหมายแบบ SM และ Signed 1's complement จะไม่มีค่า –8 และ –128 นอกจากนี้การแสดงค่าเลข 0 แบบ SM ขนาด 4 บิตมี 2 ตัวคือ 0000 และ 1000 ค่าเลข 0 แบบ Signed 1's complement ขนาด 4 บิต มี 2 ตัวคือ 0000 และ 1111 การที่เลข 0 สามารถแทนด้วยตัวเลข 2 ตัวดังกล่าว ทำให้เกิดความสับสนในการใช้งานได้ ดังนั้นตัวเลขแบบ Signed 2's complement จึงเป็นที่นิยมในการใช้งานมากกว่า

ตารางที่ 2.10 ตัวอย่างของเลขที่คิดเครื่องหมายขนาด 4 บิต

Decimal	Signed Magnitude	Signed 1's complement	Signed 2's complement
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
-0	1000	1111	-
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	-	-	1000

ตารางที่ 2.11 ตัวอย่างของเลขที่คิดเครื่องหมายขนาด 8 บิต

Decimal	Signed Magnitude	Signed 1's complement	Signed 2's complement
+127	01111111	01111111	01111111
+126	01111110	01111110	01111110
⋮	⋮	⋮	⋮
+6	00000110	00000110	00000110
+5	00000101	00000101	00000101
+4	00000100	00000100	00000100
+3	00000011	00000011	00000011
+2	00000010	00000010	00000010
+1	00000001	00000001	00000001
+0	00000000	00000000	00000000
-0	10000000	11111111	-
-1	10000001	11111110	11111111
-2	10000010	11111101	11111110
-3	10000011	11111100	11111101
-4	10000100	11111011	11111100
-5	10000101	11111010	11111011
-6	10000110	11111001	11111010
⋮	⋮	⋮	⋮
-126	11111110	10000001	10000010
-127	11111111	10000000	10000001
-128	-	-	10000000

ในหัวข้อนี้จะเน้นที่การบวกและลบเลข Signed 2's complement ขนาด 8 บิต ในการบวกและลบเลขจะต้องมีค่าผลลัพธ์ที่ได้อยู่ในช่วง –128 ถึง +127 ถ้ามีค่าน้อยกว่า –128 หรือมากกว่า +127 จะเกิดโอเวอร์โฟลว์ (Overflow) ซึ่งเลข Signed 2's complement ขนาด 8 บิตไม่สามารถเก็บข้อมูลได้ ทำให้ผลลัพธ์ที่ได้ไม่ถูกต้อง ดังนั้นเพื่อไม่ให้เกิดโอเวอร์โฟลว์ในการบวกหรือลบตัวเลขขนาด 8 บิตดังกล่าว จะต้องใช้จำนวนบิตของตัวเลขเพิ่มขึ้น เช่น 16 บิต เป็นต้น สำหรับวิธีการบวกหรือลบเลข Signed 2's complement ใช้วิธีการเช่นเดียวกับเลขฐานสอง

ตัวอย่างที่ 2.24 จงบวกและลบเลข Signed 2's complement ขนาด 8 บิตต่อไปนี้ พร้อมตรวจสอบผลลัพธ์ที่ได้ว่าถูกต้องหรือไม่

(1) $01111110_2 + 00111101_2$ (2) $11011001_2 + 10101101_2$

(3) $10101010_2 - 10011010_2$ (4) $10001011_2 - 01110101_2$

<u>วิธีทำ</u>

(1) $01111110_2 + 00111101_2$

การบวก	ตรวจสอบในรูปของเลขฐานสิบ
0111 1110+	1 2 6+
0011 1101	6 1
1011 1011	1 8 7

จะเห็นว่าการบวกเลข Signed 2's complement เมื่อทำการตรวจสอบในรูปของเลขฐานสิบแล้ว ผลลัพธ์ที่ได้มีค่าเท่ากับ 187 ซึ่งเป็นค่าตัวเลขที่ไม่สามารถเก็บได้ในข้อมูลขนาด 8 บิต เมื่อพิจารณาจากการบวกเลข Signed 2's complement ผลลัพธ์ที่ได้มีค่าเท่ากับ 10111011_2 เมื่อแปลงเป็นเลขฐานสิบแล้วมีค่าเท่ากับ –69 ซึ่งเป็นคำตอบที่ไม่ถูกต้อง ถ้าหากต้องการคำตอบที่ถูกต้อง ต้องเพิ่มขนาดของบิตให้มีจำนวนมากกว่า 8 บิต

(2) $11011001_2 + 10101101_2$

การบวก	ตรวจสอบในรูปของเลขฐานสิบ
1101 1001+	– 3 9+
1010 1101	– 8 3
11000 0110	–1 2 2

จะเห็นว่าการบวกเลข Signed 2's complement เมื่อทำการตรวจสอบในรูปของเลขฐานสิบแล้ว ผลลัพธ์ที่ได้มีค่าเท่ากับ -122 ซึ่งเป็นค่าตัวเลขที่สามารถเก็บได้ในข้อมูลขนาด 8 บิต เมื่อพิจารณาการบวกเลข Signed 2's complement ผลลัพธ์ที่ได้มีค่าเท่ากับ 10000110_2 เมื่อแปลงเป็นเลขฐานสิบมีค่าเท่ากับ –122 ซึ่งเป็นคำตอบที่ถูกต้อง

(3) $10101010_2 - 10011010_2$

การลบโดยตรง	การลบแบบ $\overline{1s}$	การลบแบบ $\overline{2s}$	ตรวจสอบ
1010 1010–	1010 1010+	1010 1010+	– 8 6–
1001 1010	0110 0101	0110 0110	–1 0 2
0001 0000	10000 1111+	10001 0000	1 6
	↳ 1		
	0001 0000		

จะเห็นว่าการลบเลข Signed 2's complement เมื่อทำการตรวจสอบในรูปของเลขฐานสิบแล้ว ผลลัพธ์ที่ได้มีค่าเท่ากับ +16 ซึ่งเป็นค่าตัวเลขที่สามารถเก็บได้ในข้อมูลขนาด 8 บิต เมื่อ

พิจารณาการลบเลข Signed 2's complement ผลลัพธ์ที่ได้มีค่าเท่ากับ $0001\ 0000_2$ เมื่อแปลงเป็นเลขฐานสิบจะมีค่าเท่ากับ +16 ซึ่งเป็นคำตอบที่ถูกต้อง

(4) $10001011_2 - 01110101_2$

การลบโดยตรง	การลบแบบ $\overline{1s}$	การลบแบบ $\overline{2s}$	ตรวจสอบ
1000 1011 −	1000 1011 +	1000 1011 +	−1 1 7 −
0111 0101	1000 1010	1000 1011	1 1 7
0001 0110	10001 0101 +	10001 0110	−2 3 4
	1		
	0001 0110		

จะเห็นว่าการลบเลข Signed 2's complement เมื่อตรวจสอบในรูปของเลขฐานสิบแล้ว ผลลัพธ์ที่ได้มีค่าเท่ากับ −234 ซึ่งเป็นค่าตัวเลขที่ไม่สามารถเก็บได้ในข้อมูลขนาด 8 บิต เมื่อพิจารณาการลบเลข Signed 2's complement ผลลัพธ์ที่ได้มีค่าเท่ากับ $0001\ 0110_2$ เมื่อแปลงเป็นเลขฐานสิบจะมีค่าเท่ากับ +22 ซึ่งเป็นคำตอบที่ไม่ถูกต้อง

แบบฝึกหัดท้ายบท

1. จงเขียนลำดับของเลขฐานแปดตั้งแต่ 66_8 จนถึง 110_8 และเลขฐานสิบหกตั้งแต่ DD_{16} จนถึง 101_{16}
2. จงแปลงเลขฐานสิบที่กำหนดให้ต่อไปนี้ เป็นเลขฐานสอง เลขฐานแปด และเลขฐานสิบหก

   (ก) 131.5625 (ข) 27/32 (ค) 2048.0625 (ง) 0.4375
3. จงแปลงเลขฐานที่กำหนดให้ ไปเป็นเลขฐานสิบ

   (ก) 1011011.001101_2 (ข) 111011.1011_2 (ค) 6405_8

   (ง) 127_8 (จ) $A6F0_{16}$ (ฉ) $3C0E_{16}$
4. จงแปลงเลขฐานที่กำหนดให้ ไปเป็นเลขฐานสอง

   (ก) 3062_8 (ข) 640715_8 (ค) $C3A6_{16}$ (ง) $1FC02_{16}$
5. จงลบเลขฐานสองต่อไปนี้ และตรวจสอบคำตอบในรูปของเลขฐานสิบ

   (ก) 111.11_2-101.1_2 (ข) 1101_2-1000_2
6. จงลบเลขฐานสองต่อไปนี้ โดยใช้การลบแบบ 1's complement และ 2's complement พร้อมตรวจสอบคำตอบในรูปของเลขฐานสิบ

(ก) 94.5_{10}-43.75_{10} (ข) 32_{10}-128_{10}

7. จงลบเลขฐานสิบต่อไปนี้ โดยใช้การลบแบบ 7's complement และ 8's complement พร้อมตรวจสอบคำตอบในรูปของเลขฐานสิบ

(ก) 94.5_{10}-43.75_{10} (ข) 32_{10}-128_{10}

8. จงลบเลขฐานสิบต่อไปนี้ โดยใช้การลบแบบ 15's complement และ 16's complement พร้อมตรวจสอบคำตอบในรูปของเลขฐานสิบ

(ก) 94.5_{10}-43.75_{10} (ข) 32_{10}-128_{10}

9. จงหาผลคูณของเลขที่กำหนดให้ ในรูปของเลขฐานสอง เลขฐานแปด เลขฐานสิบหก

(ก) $24_{10} \times 12_{10}$ (ข) $49.5_{10} \times 51.75_{10}$

10. จงหาผลหารของเลขที่กำหนดให้ในรูปของเลขฐานสอง

(ก) $27_{10} \div 18_{10}$ (ข) $58.75_{10} \div 23.5_{10}$

11. จงแปลงเลขฐานสิบต่อไปนี้ ไปเป็นเลขที่มีเครื่องหมายแบบ Signed-magnitude (SM), Signed 1's complement และ Signed 2's complement

(ก) –1024 (ข) 948

12. จงหาผลบวกของเลข Signed 2's complement ขนาด 8 บิต และตรวจสอบคำตอบในรูปของเลขฐานสิบ ว่าผลบวกที่ได้มีค่าถูกต้องหรือไม่

(ก) 00111101+11010110 (ข) 10011100+10011011

13. จงหาผลลบของเลข Signed 2's complement ขนาด 8 บิต และตรวจสอบคำตอบในรูปของเลขฐานสิบ ว่าผลบวกที่ได้มีค่าถูกต้องหรือไม่

(ก) 10101010-10011010 (ข) 10001011-01110101

เฉลยแบบฝึกหัดท้ายบท

1. เลขฐานแปด 66 67 70 71 72 73 74 75 76 77 100 101 102 103 104 105 106 107 110

   เลขฐานสิบหก DD DE DF E0 E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF 100 101

2. (ก) $131.5625=10000011.1010_2=203.44_8=83.9_{16}$ (ข) $27/32=0.110110_2=0.66_8=0.D8_{16}$

   (ค) $2048.0625=1000000000000.0001_2=4000.04_8=800.1_{16}$

   (ง) $0.4375=0.01110_2=0.34_8=0.7_{16}$

3. (ก) 1011011.001101_2=91.203125 (ข) 111011.1011_2=59.6875 (ค) 6405_8=3333
   (ง) 127_8=87 (จ) $A6F0_{16}$=42736 (ฉ) $3C0E_{16}$=15374
4. (ก) 3062_8=011 000 110 010_2 (ข) 640715_8=110 100 000 111 001 101_2
   (ค) $C3A6_{16}$=1100 0011 1010 0110_2 (ง) $1FC02_{16}$=0001 1111 1100 0000 0010_2
5. (ก) 111.11_2-101.1_2=10.01_2(2.25) (ข) 1101_2-1000_2=101_2(5)
6. (ก) 1011110.10_2-101011.11_2=110010.11_2 (ข) 100000_2-10000000_2=-01100000_2
7. (ก) 136.4_8-53.6_8=62.6_8 (ข) 40_8-200_8=-140_8
8. (ก) $5E.8_{16}$-$2B.C_{16}$=$32.C_{16}$ (ข) 20_{16}-80_{16}=-60_{16}
9. (ก)10010000_2, 440_8, 120_{16} (ข) 101000000001.1010_2, 5001.50_8, $A01.A0_{16}$
10. (ก) 1.1_2 (ข) 10.1_2
11. (ก) SM ของ –1024=1100 0000 0000_2
    Signed 1s ของ –1024=1011 1111 1111_2
    Signed 1s ของ –1024 =1100 0000 0000_2
    (ข) 01110110100_2
12. (ก) 00111101_2+11010110_2=00010011_2, ถูกต้อง (+19)
    (ข) 10011100_2+10011011_2=00110111_2, ไม่ถูกต้อง (-201)
13. (ก) 10101010_2-10011010_2=00010000_2, ถูกต้อง (+16)
    (ข) 10001011_2-01110101_2=00010110_2, ไม่ถูกต้อง (-234)

บทที่ 3
รหัส (Codes)

วัตถุประสงค์

1. สามารถเข้าใจหลักการและความสำคัญของรหัสบีซีดีแบบต่างๆ ที่ใช้แทนตัวเลขฐานสิบ
2. สามารถเข้าใจหลักการของรหัสที่นำมาใช้แทนข้อมูลในระบบดิจิตอล
3. สามารถเข้าใจหลักการและความสำคัญของการตรวจสอบความผิดพลาดของข้อมูลดิจิตอล

แม้ว่าในชีวิตประจำวันของเราจะเกี่ยวข้องกับระบบเลขฐานสิบที่ประกอบด้วยเลข 0 ถึง 9 ก็ตาม แต่ระบบดิจิตอลจะเกี่ยวข้องกับเลขฐานสองอย่างมากเนื่องจากระบบดิจิตอลทำการประมวลผลข้อมูลด้วยเลข 0 และ 1 เท่านั้น ถ้าข้อมูลเลข 0 และ 1 มีจำนวนมาก ทำให้สามารถเข้าใจข้อมูลเหล่านี้ได้ยาก ดังนั้นการออกแบบระบบดิจิตอลจะใช้กลุ่มรหัส (Code group) ของเลข 0 และ 1 แทนตัวเลข ตัวอักษร เครื่องหมาย หรือสัญลักษณ์ต่างๆ การแปลงจากตัวเลข ตัวอักษร เครื่องหมาย หรือสัญลักษณ์ต่างๆ ไปเป็นเลขเลขฐานสองหรือกลุ่มรหัสของเลขฐานสองเรียกว่า **การเข้ารหัส** (Encoding) และตรงกันข้ามการแปลงจากเลขฐานสองหรือกลุ่มรหัสของเลขฐานสองไปเป็นตัวเลข ตัวอักษร เครื่องหมาย หรือสัญลักษณ์ต่างๆ ที่คนเราสามารถเข้าใจได้ เรียกว่า **การถอดรหัส** (Decoding) การทำงานของอุปกรณ์ดิจิตอลบางอย่างจะใช้รูปแบบกลุ่มรหัสของเลขฐานสองที่มีลักษณะเฉพาะขึ้นอยู่กับวัตถุประสงค์ของการใช้งานนั้นๆ กลุ่มรหัสของเลขฐานสองมีหลายรูปแบบ ดังนี้

3.1 รหัสบีซีดี (Binary-Coded Decimal, BCD)

เมื่อต้องการแปลงข้อมูลเลขฐานสองจำนวนหลายบิตไปเป็นเลขฐานสิบ หรือตรงกันข้ามถ้าต้องการแปลงเลขฐานสิบที่มีจำนวนหลายหลักไปเป็นเลขฐานสองแล้ว หากไม่ได้ใช้เครื่องคำนวณช่วยในการแปลงเลขฐาน จะทำให้ใช้เวลาในการแปลงข้อมูลนาน ดังนั้นเลขฐานสิบที่มีตัวเลขตั้งแต่ 0 ถึง 9 สามารถแทนด้วยกลุ่มรหัสของเลขฐานสองได้ ทำให้การแปลงข้อมูลระหว่างเลขฐานสองและเลขฐานสิบสะดวกและเร็วขึ้น กลุ่มรหัสของเลขฐานสองที่ใช้แทนเลขฐานสิบเรียกว่า BCD (Binary-coded Decimal) หรือเรียกสั้นๆ ว่า รหัสเลขฐานสิบ (Decimal Codes) จากความสัมพันธ์ของระบบเลขฐานสอง ฐานแปด ฐานสิบหก จะเห็นว่า กลุ่มรหัสของเลขฐานสองจำนวน 3 บิต สามารถแสดงค่าได้ทั้งหมด 8 ค่า (2^3=8, เลข 0-7) และกลุ่มรหัสของเลขฐานสองจำนวน 4 บิต สามารถแสดงค่าได้ทั้งหมด 16 ค่า (2^4=16, เลข 0-15) ดังนั้นถ้าใช้รหัส BCD แทนเลข 0-9 จะต้อง

ใช้เลขฐานสองจำนวนอย่างน้อยที่สุด 4 บิต กลุ่มรหัสของเลขฐานสองจำนวน 4 บิตใช้แทนเลข 0-9 ส่วน 6 ค่าที่เหลือคือ 10-15 ไม่ได้นำมาใช้ สามารถคำนวณรูปแบบกลุ่มรหัสเลขฐานสองจำนวน 4 บิต ที่เป็นไปได้ทั้งหมด ดังนี้

$$\binom{16}{10} x10! = \frac{16!}{6!} = 29,059,430,400$$

จะเห็นว่ากลุ่มรหัสเลขฐานสองจำนวน 4 บิต ที่ใช้แทนเลขฐานสิบมีประมาณ 30 พันล้านรูปแบบ ในที่นี้จะกล่าวถึงรูปแบบของกลุ่มรหัสเลขฐานสองที่นิยมใช้เท่านั้น รหัส BCD สามารถแบ่งออกเป็น 2 ประเภทคือ รหัสที่มีค่าน้ำหนักและรหัสที่ไม่มีค่าน้ำหนัก รายละเอียดมีดังนี้

3.1.1 รหัสที่มีค่าน้ำหนัก (Weighted Codes)

กลุ่มรหัสเลขฐานสองแต่ละบิตจะมีค่าน้ำหนักในแต่ละตำแหน่งแตกต่างกันไป ผลรวมของค่าน้ำหนักในตำแหน่งบิตที่เป็น 1 มีค่าเท่ากับเลขฐานสิบ ค่าน้ำหนักของบิตในแต่ละตำแหน่งจะมีค่าตามชื่อรูปแบบของรหัสนั้นๆ รูปแบบของรหัสที่มีน้ำหนักมีหลายรูปแบบ ดังตารางที่ 3.1

ตารางที่ 3.1 ตัวอย่างของรหัสที่มีค่าน้ำหนัก

Decimal digit	8421 code	2421 code	5421 code	7536 code	Biquinary code 5043210
0	0000	0000	0000	0000	0100001
1	0001	0001	0001	1001	0100010
2	0010	0010	0010	0111	0100100
3	0011	0011	0011	0010	0101000
4	0100	0100	0100	1011	0110000
5	0101	1011	1000	0100	1000001
6	0110	1100	1001	1101	1000010
7	0111	1101	1010	1000	1000100
8	1000	1110	1011	0110	1001000
9	1001	1111	1100	1111	1010000

รหัส 8421 เป็นรหัสที่นิยมใช้ในคอมพิวเตอร์มาก เนื่องจากเป็นรหัสที่เข้าใจได้ง่ายเมื่อเปรียบเทียบกับเลขฐานสอง บางครั้งจะเรียกว่ารหัส BCD ตัวเลข 8421 แสดงค่าน้ำหนักในแต่ละตำแหน่งของเลขฐานสองขนาด 4 บิต โดยที่ตำแหน่งบิตนัยสำคัญน้อยที่สุดมีค่าเท่ากับ 1 และที่ตำแหน่งบิตนัยสำคัญมากที่สุดมีค่าเท่ากับ 8 การนำรหัส 8421 แทนเลขฐานสิบ แสดงดังตารางที่ 3.1 และตารางที่ 3.2 โดยเลขฐานสิบแต่ละหลักจะแทนด้วยรหัส 8421 จำนวน 4 บิต

ถ้ากำหนดให้ w คือค่าน้ำหนักของแต่ละบิต ($w_3w_2w_1w_0$) และ b คือกลุ่มรหัสเลขฐานสองที่ใช้แทนเลขฐานสิบในแต่ละหลัก ($b_3b_2b_1b_0$) ดังนั้นเลขฐานสิบ N ในแต่ละหลัก หาได้จาก

$$N = (b_3 \times w_3) + (b_2 \times w_2) + (b_1 \times w_1) + (b_0 \times w_0)$$

เลขฐานสิบในแต่ละหลักสามารถแทนด้วยรหัส 8421 ตัวอย่างเช่น เลข 392_{10} แทนด้วยรหัส 8421 จะได้ว่า 392_{10}= (0011 1001 0010) $_{8421}$ เป็นต้น กลุ่มรหัสเลขฐานสองจะมีจำนวนบิตที่แน่นอน เพื่อไม่ให้เกิดปัญหาในการแปลงเลขระหว่างเลขฐานสิบและกลุ่มรหัสเลขฐานสองเหล่านี้ ทั้งนี้ต้องเข้าใจว่าการเข้ารหัสของเลขฐานสิบด้วยกลุ่มรหัสเลขฐานสองมีค่าไม่เท่ากับเลขฐานสอง ดังตารางที่ 3.2

ตารางที่ 3.2 การแทนเลขฐานสิบด้วยรหัส BCD

Decimal	Binary	8421	5421	7536
0	0000	0000	0000	0000
1	0001	0001	0001	1001
2	0010	0010	0010	0111
3	0011	0011	0011	0010
4	0100	0100	0100	1011
5	0101	0101	1000	0100
6	0110	0110	1001	1101
7	0111	0111	1010	1000
8	1000	1000	1011	0110
9	1001	1001	1100	1111
10	1010	0001 0000	0001 0000	1001 0000
11	1011	0001 0001	0001 0001	1001 1001
12	1100	0001 0010	0001 0010	1001 0111
13	1101	0001 0011	0001 0011	1001 0010
14	1110	0001 0100	0001 0100	1001 1011
15	1111	0001 0101	0001 1000	1001 0100

การที่รหัส 8421 เป็นที่นิยมใช้กันมาก ดังนั้นจำเป็นต้องเข้าใจหลักการบวกเลขของรหัส 8421 มีขั้นตอนการบวกดังนี้ การบวกรหัส 8421 ทำเช่นเดียวกับการบวกของเลขฐานสอง แต่ถ้าผลบวกของเลขฐานสองมีค่ามากกว่า 9 หรือผลบวกมีตัวทดสุดท้ายเกิดขึ้น ต้องทำการบวก 0110 (6) เข้าไป เพื่อให้ผลบวกของเลขรหัส 8421 มีคำตอบที่ถูกต้อง ตัวอย่างเช่น

Decimal	BCD	Decimal	BCD	Decimal	BCD
6+	0110+	7+	0111+	8+	1000+
3	0011	5	0101	9	1001
9	1001	12	1100+	17	1 0001+
			0110		0110
			0001 0010		0001 0111

จากตัวอย่างการบวกเลข 7+5 จะเห็นว่าผลบวกมีค่าเท่ากับ 1100 (12) ซึ่งมีค่ามากกว่า 9 ดังนั้นต้องทำการบวก 0110 (6) เพื่อให้ได้ผลบวกมีค่าเท่ากับ 0010 (2) และตัวทด การบวกเลข 8+9 จะเห็นว่าผลบวกที่ได้มีตัวทดเกิดขึ้น ดังนั้นต้องทำการบวก 0110 (6) เพื่อให้ได้ผลบวกมีค่าเท่ากับ 0111 (7) และตัวทด การบวกเลขรหัส BCD ที่มีตัวเลขจำนวนหลายหลัก จะทำการบวกทีละหลัก โดยเริ่มการบวกเลขที่อยู่หลักทางขวาสุดก่อนเสมอ ตัวอย่างเช่น

Decimal	BCD			
	1	1		ตัวทด
184 +	0001	1000	0100+	ตัวตั้ง
576	0101	0111	0110	ตัวบวก
760	0111	0000	1010+	ผลบวก
		0110	0110	บวก 6
	0111	0110	0000	ผลบวกของเลข BCD

รหัส 2421 เป็นรหัสที่มีคุณสมบัติของการคอมพลีเมนต์ในตัวเอง (Self-complement) ซึ่งรหัส 8421 จะไม่มีคุณสมบัตินี้ การคอมพลีเมนต์ในตัวเองมีลักษณะดังนี้ 9's complement ของเลขฐานสิบใดๆ สามารถเขียนแทนโดยใช้ 1's complement ของรหัส 2421 ได้ ตัวอย่างเช่น 9's complement ของเลข 7_{10} มีค่าเท่ากับ 2 เมื่อเขียนเลข 7 ในรูปรหัส 2421 จะมีค่าเท่ากับ 1101 ดังนั้น 1's complement ของ 1101 (7) มีค่าเท่ากับ 0010 (2) หรือ 9's complement ของเลข 328_{10} มีค่าเท่ากับ 671 เมื่อเขียนเลข 328_{10} ในรูปรหัส 2421 จะมีค่าเท่ากับ $(0011\ 0010\ 1110)_{2421}$ ดังนั้น 1's complement ของ $(0011\ 0010\ 1110)_{2421}$ (328) มีค่าเท่ากับ $(1100\ 1101\ 0001)_{2421}$ (671) สรุปได้ว่า 9's complement ของเลขฐานสิบสามารถเขียนแทนด้วย 1's complement ของเลขรหัส 2421

รหัส $753\bar{6}$ เป็นรหัสที่คอมพลีเมนต์ในตัวเองและค่าน้ำหนักบางตัวมีค่าเป็นลบ เครื่องหมายบาร์บนเลข 6 แสดงค่าน้ำหนักเท่ากับ –6 ตัวอย่างเช่น รหัส $753\bar{6}$ ของเลข 9 มีค่าเท่ากับ $(1111)_{753\bar{6}}$ หรือเลข $(1111)_{753\bar{6}}$ มีค่าเท่ากับเลขฐานสิบดังนี้

$$(1111)_{753\bar{6}} = (1\times 7)+(1\times 5)+(1\times 3)-(1\times 6) = 7+5+3-6 = 9$$

รหัสไบควินารีหรือรหัส 5043210 เป็นกลุ่มรหัสเลขฐานสองที่มีจำนวนบิตมากกว่า 4 บิต ลักษณะที่สำคัญของรหัสนี้ คือมีบิตที่เป็น 1 จำนวน 2 บิตเท่านั้น โดยบิตแรกอยู่ในกลุ่ม 50 (Bi) และบิตที่สองอยู่ในกลุ่ม 43210 (Quinary) รหัสนี้มีประโยชน์ในการตรวจสอบความผิดพลาดของข้อมูล เนื่องจากมีบิตที่เป็น 1 เพียง 2 ตัวเท่านั้น

ตัวอย่างที่ 3.1 จงแปลงเลข 35_{10} ต่อไปนี้ไปเป็นรหัส 8421, 5421, $753\bar{6}$ และ Biquinary

<u>วิธีทำ</u>

$$35_{10} = 0011\ \ 0101_{8421} = 0011\ \ 1000_{5421} = 0010\ \ 0100_{753\bar{6}} = 0101000\ \ 1000001_{Biquinary}$$

ตัวอย่างที่ 3.2 จงแปลงรหัสต่อไปนี้ เป็นเลขฐานสิบ

(1) $(1001\ 0101)_{8421}$ (2) $(0011\ 1100\ 1010)_{5421}$ (3) $(1111\ 1100\ 0100)_{2421}$

<u>วิธีทำ</u>

(1) $(1001\ 0101)_{8421}$

1001 = (1x8)+(1x1) = 8+1 = 9

0101 = (1x4)+(1x1) = 4+1 = 5

$\therefore (1001\ 0101)_{8421} = 95_{10}$

(2) $(0011\ 1100\ 1010)_{5421}$

0011 = (1x2)+(1x1) = 2+1 = 3

1100 = (1x5)+(1x4) = 5+4 = 9

1010 = (1x5)+(1x2) = 5+2 = 7

$\therefore (0011\ 1100\ 1010)_{5421} = 397_{10}$

(3) $(1111\ 1100\ 0100)_{2421}$

1111 = (1x2)+(1x4)+(1x2)+(1x1) = 2+4+2+1 = 9

1100 = (1x2)+(1x4) = 2+4 = 6

0100 = (1x4) = 4

$\therefore (1111\ 1100\ 0100)_{2421} = 964_{10}$

3.1.2 รหัสที่ไม่มีค่าน้ำหนัก (Non-weighted Codes)

รหัสที่ไม่มีค่าน้ำหนักจะไม่ระบุค่าน้ำหนักในแต่ละตำแหน่งของบิตเหมือนกับรหัสที่มีค่าน้ำหนัก รหัสที่ไม่มีค่าน้ำหนักที่น่าสนใจ ได้แก่ รหัสเกิน 3 (Excess-3) รหัส 2 ใน 5 (2-out-of-5) รหัสจอห์นสัน (Johnson) รหัสเกรย์ (Gray) เป็นต้น การแทนค่าเลขฐานสิบด้วยรหัสที่ไม่มีค่าน้ำหนัก แสดงตัวอย่างดังตารางที่ 3.3

ตารางที่ 3.3 ตัวอย่างของรหัสที่ไม่มีค่าน้ำหนัก

Decimal	Binary	Excess-3	2-out-of-5	Johnson	Gray
0	0000	0011	11000	00000	0000
1	0001	0100	00011	00001	0001
2	0010	0101	00101	00011	0011
3	0011	0110	00110	00111	0010
4	0100	0111	01001	01111	0110
5	0101	1000	01010	11111	0111
6	0110	1001	01100	11110	0101
7	0111	1010	10001	11100	0100
8	1000	1011	10010	11000	1100
9	1001	1100	10100	10000	1101

รหัสเกิน 3 เป็นรหัสที่คอมพลีเมนต์ในตัวเองและมีประโยชน์อย่างมากในการดำเนินการทางคณิตศาสตร์ รหัสนี้สามารถหาได้จากรหัส BCD บวกด้วย 0011 (3) เช่น เลข 4 มีรหัส BCD เท่ากับ 0100 สามารถหารหัสเกิน 3 ของเลข 4 ได้จาก 0100+0011 เท่ากับ 0111 การบวกของรหัสเกิน 3 สามารถพิจารณาได้ 2 กรณีคือ กรณีแรก ถ้าผลบวกของเลขมีค่ามากกว่า 9 ให้ทำการบวก 3 จะได้ผลบวกในรูปของรหัสเกิน 3 ที่ถูกต้อง เช่น

```
Decimal        Excess-3
  7+              1010+
  6               1001
 --               ----
 13             1 0011+
              0011 0011      บวกด้วย 3 ทั้งผลบวกและตัวทด
              ----------
              0100 0110
               ⇓    ⇓
               1    3
```

กรณีที่สอง ถ้าผลบวกของเลขมีค่าน้อยกว่าหรือเท่ากับ 9 ให้ทำการลบ 3 จะได้ผลบวกในรูปของรหัสเกิน 3 ที่ถูกต้อง เช่น

```
Decimal        Excess-3
  6+              1001+
  2               0101
 --               ----
  8               1110−
                  0011      ลบด้วย 3
                  ----
                  1011
                   ⇓
                   8
```

การลบเลขของรหัสเกิน 3 ผลลบที่ได้ต้องทำการบวกด้วย 3 เข้าไป จะได้ผลลบในรูปของรหัสเกิน 3 ที่ถูกต้อง เช่น

Decimal	Excess-3	
17 –	0100 1010 –	
11	0100 0100	
6	0000 0110 +	
	0011	บวกด้วย 3
	1001	
	⇓	
	6	

รหัส 2 ใน 5 จะมีบิตที่เป็น 1 จำนวน 2 บิต ที่เหลือจะเป็นบิต 0 รหัสนี้มีข้อดีเช่นเดียวกับรหัสไบควินารีคือสามารถตรวจสอบความผิดพลาดของข้อมูลได้ รหัส 2 ใน 5 ได้นำไปใช้ในระบบบาร์โค้ด (Barcode) เช่น การให้บริการทางไปรษณีย์ของอเมริกา (U.S. Postal Service) จากรูปที่ 3.1 แสดงรหัสบาร์โค้ดของจดหมายที่มีรหัสไปรษณีย์ (ZIP code) เป็น 14263-1045 ในตัวเลขแต่ละตัวใช้รหัส 2 ใน 5 โดยเส้นสั้นแทนบิต 0 และเส้นยาวแทนบิต 1 เส้นยาวที่อยู่หัวและท้ายของเลขรหัสไปรษณีย์เรียกว่าเฟรมบาร์ (Frame bar) เพื่อใช้กำหนดจุดเริ่มต้นและจุดสิ้นสุดของรหัสบาร์โค้ดของตัวเลขไปรษณีย์ชุดนั้น การพิจารณาว่าในแต่ละกลุ่มของรหัส 2 ใน 5 แทนเลขฐานสิบตัวเลขใด สามารถหาได้จากการเปรียบเทียบจากตารางที่ 3.3 เช่น รหัส 2 ใน 5 ของเลข 4 คือ 01001 เป็นต้น

รูปที่ 3.1 รหัสบาร์โค้ดของการให้บริการทางไปรษณีย์ในอเมริกา

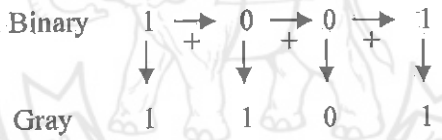
รหัสจอห์นสัน เป็นรหัสที่นำไปประยุกต์ใช้ในการออกแบบวงจรนับ พิจารณาจากตารางที่ 3.3 จะเห็นว่า ลักษณะของรหัสนี้ที่ตัวเลขตั้งแต่ 0-5 จะมีบิตที่เป็น 1 ถูกเลื่อนจากขวาไปซ้ายและตัวเลขตั้งแต่ 6-9 จะมีบิตที่เป็น 0 ถูกเลื่อนจากขวาไปซ้าย

รหัสเกรย์ เป็นรหัสที่มีลักษณะพิเศษคือกลุ่มรหัสของตัวเลขที่อยู่ติดกันจะแตกต่างกันเพียงบิตเดียวเท่านั้น เช่น รหัสเกรย์ของเลข 8 และ 9 คือ 1100 และ 1101 ตามลำดับ จะเห็นว่ารหัสเกรย์ของเลข 8 และ 9 จะมีบิตนัยสำคัญที่น้อยที่สุดต่างกันเพียงบิตเดียวเท่านั้น การใช้รหัสเกรย์แทนเลขฐานสิบในคอมพิวเตอร์ จำเป็นต้องแปลงเลขฐานสิบเป็นเลขฐานสองก่อน จากนั้นจึงทำการแปลงจากเลขฐานสองเป็นรหัสเกรย์ ถ้าหากต้องการหาเลขฐานสิบจากรหัสเกรย์ สามารถทำได้ในขั้นตอนตรงกันข้าม การแปลงระหว่างเลขฐานสองเป็นรหัสเกรย์ มีขั้นตอนดังนี้

<u>การแปลงจากเลขฐานสองเป็นรหัสเกรย์</u>

1. บิตซ้ายสุดของรหัสเกรย์มีค่าเท่ากับบิตซ้ายสุดของเลขฐานสองเสมอ
2. ทำการบวกเลขฐานสองตรงตำแหน่งที่ต้องการหากับบิตที่อยู่ทางซ้ายมือ แล้วเก็บผลลัพธ์ที่ได้เป็นคำตอบของรหัสเกรย์ในตำแหน่งบิตที่ตรงกัน ถ้าการบวกแต่ละครั้งมีตัวทดเกิดขึ้น ให้ตัดตัวทดทิ้งไป เช่น 1+1= 10 เก็บค่า 0 และตัด 1 ทิ้งไป
3. รหัสเกรย์ที่ได้จะต้องมีจำนวนบิตเท่ากับเลขฐานสองเสมอ

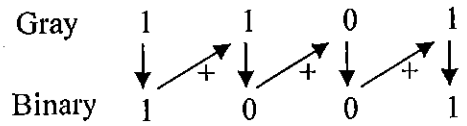
จากตารางที่ 3.3 ตัวอย่างการแปลงเลข 1001_2 (9_{10}) เป็นรหัสเกรย์ มีลักษณะดังนี้



<u>การแปลงจากรหัสเกรย์เป็นเลขฐานสอง</u>

1. บิตซ้ายสุดของเลขฐานสองมีค่าเท่ากับบิตซ้ายสุดของรหัสเกรย์เสมอ
2. นำบิตของเลขฐานสองที่ได้ ไปบวกกับบิตของรหัสเกรย์ในตำแหน่งถัดไปที่ต้องการหา แล้วเก็บผลลัพธ์ที่ได้เป็นคำตอบของเลขฐานสองในตำแหน่งบิตที่ตรงกัน ในการบวกแต่ละครั้ง ถ้ามีตัวทดให้ตัดตัวทดทิ้งไป
3. เลขฐานสองที่ได้จะต้องมีจำนวนบิตเท่ากับรหัสเกรย์เสมอ

จากตารางที่ 3.3 ตัวอย่างการแปลง 1101_{gray} (9_{10}) เป็นเลขฐานสอง มีลักษณะดังนี้



การหาผลลัพธ์ของการแปลงระหว่างเลขฐานสองและรหัสเกรย์ สามารถใช้การเปรียบเทียบแทนการบวกได้เช่นเดียวกัน ถ้าบิตเหมือนกันจะได้ผลลัพธ์เป็น 0 และถ้าบิตต่างกันจะได้ผลลัพธ์เป็น 1

ตัวอย่างที่ 3.3 จงแปลงเลข 88_{10} ไปเป็นรหัสเกรย์

<u>วิธีทำ</u>

การแปลงเลข 88_{10} เป็นเลขฐานสองโดยวิธีการหารเอาเศษ

$88 \div 2 = 44$ เศษ 0 (LSB)

$44 \div 2 = 22$ เศษ 0

$22 \div 2 = 11$ เศษ 0

$11 \div 2 = 5$ เศษ 1

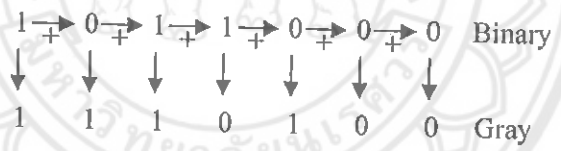
$5 \div 2 = 2$ เศษ 1

$2 \div 2 = 1$ เศษ 0

$1 \div 2 = 0$ เศษ 1 (MSB)

$\therefore\ 88_{10} = 1011000_2$

การแปลงเลขฐานสองเป็นรหัสเกรย์

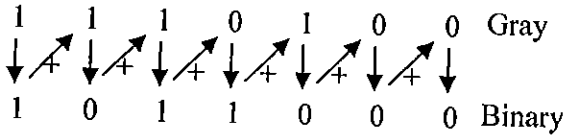


$\therefore\ 88_{10} = 1011000_2 = 1110100_{Gray}$

ตัวอย่างที่ 3.4 จงแปลงรหัสเกรย์ต่อไปนี้ 1110100_{Gray} เป็นเลขฐานสิบ

<u>วิธีทำ</u>

การแปลงรหัสเกรย์เป็นเลขฐานสอง



$\therefore\ 1110100_{Gray} = 1011000_2$

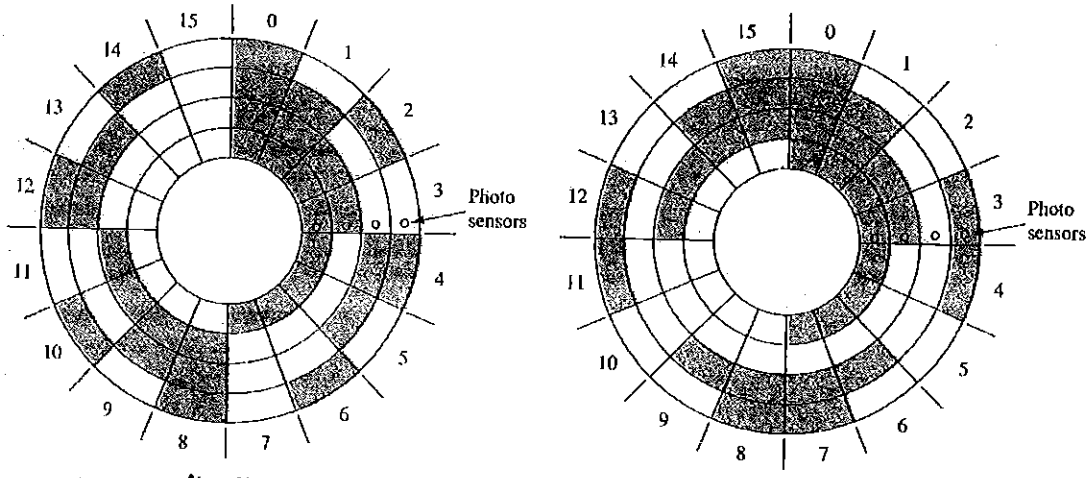
การแปลงเลข 1011000_2 เป็นเลขฐานสิบ

$$1011000_2 = (1\times64)+(1\times16)+(1\times8)$$
$$= 64+16+8$$
$$= 88$$
$$\therefore\ 1110100_{Gray} = 10110002 = 88_{10}$$

การใช้ประยุกต์ใช้งานของรหัสเกรย์นิยมใช้กันมากในการเข้ารหัสตำแหน่งทางแสงหรือตำแหน่งของแกนหมุนเพลาในทางเครื่องกล จากรูปที่ 3.2 สมมุติว่าการหาตำแหน่งมุมของเพลาสามารถหาได้จากเลขฐานสองจำนวน 4 บิต ดังนั้นสามารถแบ่งเพลาออกเป็น 2^4=16 เซกเตอร์ (Sector) ตำแหน่งของมุมในแต่ละเซกเตอร์มีค่าเพิ่มขึ้นทีละ 360/16 = 22.5° ถ้าหากต้องการความละเอียดของมุมมากกว่านี้ ให้เพิ่มจำนวนบิตของเลขฐานสอง จะได้จำนวนของเซกเตอร์เพิ่มขึ้นแต่ค่ามุมในแต่ละเซกเตอร์จะมีค่าน้อยลง เช่น การหาตำแหน่งมุมของเพลาด้วยเลขฐานสองจำนวน 5 บิต สามารถแบ่งเพลาออกเป็น 2^5=32 เซกเตอร์ ในแต่ละเซกเตอร์มีขนาดของมุม 360/32 = 11.25°

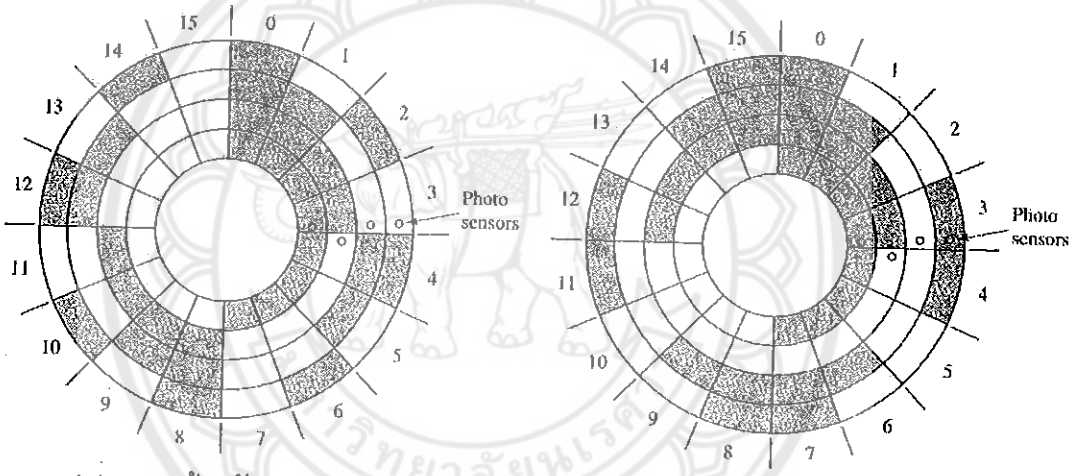
ถ้ากำหนดให้ตัวรับแสง (Photo sensors) อยู่ในแนวใดแนวหนึ่งที่มีตำแหน่งแน่นอน เพื่อทำการตรวจจับแสงในแต่ละเซล ในแต่ละเซกเตอร์จะให้แต่ละเซลทำการเข้ารหัสเลขฐานสองหรือรหัสเกรย์ ในที่นี้สมมุติว่าเซลที่แรงเงามีค่าเป็น 0 และเซลที่ไม่ได้แรงเงามีค่าเป็น 1 กรณีที่เพลามีการหาตำแหน่งมุมด้วยการเข้ารหัสเลขฐานสองดังรูปที่ 3.2(ก) เมื่อเพลามีการหมุนจากตำแหน่งเซกเตอร์ที่ 3 ไป 4 เลขฐานสองจะมีการเปลี่ยนจาก 0011_2 เป็น 0100_2 ซึ่งมีการเปลี่ยนค่า 0 และ 1 จำนวนถึง 3 บิต แต่ถ้าเพลามีการหาตำแหน่งมุมด้วยการเข้ารหัสเกรย์ดังรูปที่ 3.2(ข) เมื่อเพลามีการหมุนจากตำแหน่งเซกเตอร์ที่ 3 ไป 4 รหัสเกรย์จะมีการเปลี่ยนจาก 0010_{GRAY} เป็น 0110_{GRAY} ซึ่งมีการเปลี่ยนค่า 0 และ 1 เพียงบิตเดียวเท่านั้น

กรณีที่ตำแหน่งของรูรับแสงไม่ได้อยู่ในแนวเดียวกัน ตัวอย่างเช่น จากรูปที่ 3.3 รูรับแสงที่รับค่าของบิตที่สองจากซ้ายของรหัสมีการเลื่อนตำแหน่งไปเล็กน้อย ทำให้รูรับแสงไม่อยู่ในแนวเดียวกัน เมื่อมีการเข้ารหัสเพื่อหาตำแหน่งมุมของเพลาในเซกเตอร์ที่ 3 ตำแหน่งมุมที่ได้จะแตกต่างจากรูปที่ 3.2 เมื่อพิจารณาจากรูปที่ 3.3 (ก) การหาตำแหน่งมุมของเซกเตอร์ที่ 3 จากรูรับแสงด้วยการเข้ารหัสเลขฐานสองจะเปลี่ยนจาก 0011_2 เป็น 0111_2 ซึ่งเป็นตำแหน่งมุมของเซกเตอร์ที่ 7 ซึ่งมุมจะมีค่าที่ผิดไปถึง 4 เซกเตอร์หรือ 90° แต่ถ้าเป็นการหาตำแหน่งมุมจากรูรับแสงด้วยการเข้ารหัสเกรย์ดังรูปที่ 3.3 (ข) ตำแหน่งมุมที่ได้จะเปลี่ยนจาก 0010_{GRAY} เป็น 0110_{GRAY} ซึ่งเป็นตำแหน่งมุมของเซกเตอร์ที่ 4 ซึ่งมุมจะมีค่าที่ผิดไปเพียง 1 เซกเตอร์หรือ 22.5° เท่านั้น เพราะรหัสเกรย์มีการเปลี่ยนแปลงค่าของเซกเตอร์ที่อยู่ติดกันเพียง 1 บิตเท่านั้น ดังนั้นจะเห็นว่าการหาตำแหน่งมุมด้วยการเข้ารหัสเกรย์จะเกิดความผิดพลาดน้อยกว่าการเข้ารหัสเลขฐานสอง



(ก) การเข้ารหัสเลขฐานสอง (ข) การเข้ารหัสเกรย์

รูปที่ 3.2 การเข้ารหัสตำแหน่งมุมของเพลา



(ก) การเข้ารหัสเลขฐานสอง (ข) การเข้ารหัสเกรย์

รูปที่ 3.3 การเข้ารหัสตำแหน่งมุมของเพลาเมื่อรูรับแสงไม่ได้อยู่ในแนวเดียวกัน

3.2 รหัสอักขระ (Alphanumeric codes)

ระบบดิจิตอลไม่ได้มีข้อมูลที่เป็นเลขฐานสิบเท่านั้น แต่ยังมีข้อมูลแบบอื่นด้วย เช่น ตัวอักษร เครื่องหมาย สัญลักษณ์ต่างๆ (@, =, +, #, ...) ตัวควบคุมการทำงาน (Backspace, Form feed, ...) ดังนั้นกลุ่มรหัสเลขฐานสองที่ใช้แทนข้อมูลเหล่านี้เรียกว่ารหัสอักขระ (Alphanumeric codes) รหัสอักขระได้มีการพัฒนาขึ้นมาหลายรูปแบบ แต่รหัสอักขระที่รู้จักกันดีที่สุดคือรหัสแอสกี (American Standard Code for Information Interchange, ASCII) ที่ใช้บิตข้อมูลจำนวน 7 บิต (2^7=128) เป็นมาตรฐานที่ใช้ในคีย์บอร์ด (Keyboard) ของเครื่องคอมพิวเตอร์ ดังตารางที่ 3.4

ตารางที่ 3.4 รหัสแอสกีขนาด 7 บิต

American Standard Code for Information Interchange (ASCII).

Control Characters				Printable Characters											
Name	Dec	Binary	Hex	Symbol	Dec	Binary	Hex	Symbol	Dec	Binary	Hex	Symbol	Dec	Binary	Hex
NUL	0	0000000	00	space	32	0100000	20	@	64	1000000	40	`	96	1100000	60
SOH	1	0000001	01	!	33	0100001	21	A	65	1000001	41	a	97	1100001	61
STX	2	0000010	02	"	34	0100010	22	B	66	1000010	42	b	98	1100010	62
ETX	3	0000011	03	#	35	0100011	23	C	67	1000011	43	c	99	1100011	63
EOT	4	0000100	04	$	36	0100100	24	D	68	1000100	44	d	100	1100100	64
ENQ	5	0000101	05	%	37	0100101	25	E	69	1000101	45	e	101	1100101	65
ACK	6	0000110	06	&	38	0100110	26	F	70	1000110	46	f	102	1100110	66
BEL	7	0000111	07	'	39	0100111	27	G	71	1000111	47	g	103	1100111	67
BS	8	0001000	08	(	40	0101000	28	H	72	1001000	48	h	104	1101000	68
HT	9	0001001	09	)	41	0101001	29	I	73	1001001	49	i	105	1101001	69
LF	10	0001010	0A	*	42	0101010	2A	J	74	1001010	4A	j	106	1101010	6A
VT	11	0001011	0B	+	43	0101011	2B	K	75	1001011	4B	k	107	1101011	6B
FF	12	0001100	0C	,	44	0101100	2C	L	76	1001100	4C	l	108	1101100	6C
CR	13	0001101	0D	-	45	0101101	2D	M	77	1001101	4D	m	109	1101101	6D
SO	14	0001110	0E	.	46	0101110	2E	N	78	1001110	4E	n	110	1101110	6E
SI	15	0001111	0F	/	47	0101111	2F	O	79	1001111	4F	o	111	1101111	6F
DLE	16	0010000	10	0	48	0110000	30	P	80	1010000	50	p	112	1110000	70
DC1	17	0010001	11	1	49	0110001	31	Q	81	1010001	51	q	113	1110001	71
DC2	18	0010010	12	2	50	0110010	32	R	82	1010010	52	r	114	1110010	72
DC3	19	0010011	13	3	51	0110011	33	S	83	1010011	53	s	115	1110011	73
DC4	20	0010100	14	4	52	0110100	34	T	84	1010100	54	t	116	1110100	74
NAK	21	0010101	15	5	53	0110101	35	U	85	1010101	55	u	117	1110101	75
SYN	22	0010110	16	6	54	0110110	36	V	86	1010110	56	v	118	1110110	76
ETB	23	0010111	17	7	55	0110111	37	W	87	1010111	57	w	119	1110111	77
CAN	24	0011000	18	8	56	0111000	38	X	88	1011000	58	x	120	1111000	78
EM	25	0011001	19	9	57	0111001	39	Y	89	1011001	59	y	121	1111001	79
SUB	26	0011010	1A	:	58	0111010	3A	Z	90	1011010	5A	z	122	1111010	7A
ESC	27	0011011	1B	;	59	0111011	3B	[	91	1011011	5B	{	123	1111011	7B
FS	28	0011100	1C	<	60	0111100	3C	\	92	1011100	5C	|	124	1111100	7C
GS	29	0011101	1D	=	61	0111101	3D	]	93	1011101	5D	}	125	1111101	7D
RS	30	0011110	1E	>	62	0111110	3E	^	94	1011110	5E	~	126	1111110	7E
US	31	0011111	1F	?	63	0111111	3F	—	95	1011111	5F	Del	127	1111111	7F

ตารางที่ 3.5 รายละเอียดของรหัสควบคุม (Control Character)

Control characters

NUL	Null	DLE	Data-link escape
SOH	Start of heading	DC1	Device control 1
STX	Start of text	DC2	Device control 2
ETX	End of text	DC3	Device control 3
EOT	End of transmission	DC4	Device control 4
ENQ	Enquiry	NAK	Negative acknowledge
ACK	Acknowledge	SYN	Synchronous idle
BEL	Bell	ETB	End-of-transmission block
BS	Backspace	CAN	Cancel
HT	Horizontal tab	EM	End of medium
LF	Line feed	SUB	Substitute
VT	Vertical tab	ESC	Escape
FF	Form feed	FS	File separator
CR	Carriage return	GS	Group separator
SO	Shift out	RS	Record separator
SI	Shift in	US	Unit separator
SP	Space	DEL	Delete

ตารางที่ 3.6 รหัสแอสกีขนาด 8 บิต(ตัวอักขระที่เพิ่มขึ้นมา)

Symbol	Dec	Hex	Symbol	Dec	Hex	Symbol	Dec	Hex	Symbol	Dec	Hex
Ç	128	80	á	160	A0	└	192	C0	α	224	E0
ü	129	81	í	161	A1	┴	193	C1	β	225	E1
é	130	82	ó	162	A2	┬	194	C2	Γ	226	E2
â	131	83	ú	163	A3	├	195	C3	π	227	E3
ä	132	84	ñ	164	A4	─	196	C4	Σ	228	E4
à	133	85	Ñ	165	A5	┼	197	C5	σ	229	E5
å	134	86	ª	166	A6	╞	198	C6	µ	230	E6
ç	135	87	º	167	A7	╟	199	C7	τ	231	E7
ê	136	88	¿	168	A8	╚	200	C8	Φ	232	E8
ë	137	89	⌐	169	A9	╔	201	C9	Θ	233	E9
è	138	8A	¬	170	AA	╩	202	CA	Ω	234	EA
ï	139	8B	½	171	AB	╦	203	CB	δ	235	EB
î	140	8C	¼	172	AC	╠	204	CC	∞	236	EC
ì	141	8D	¡	173	AD	═	205	CD	φ	237	ED
Ä	142	8E	«	174	AE	╬	206	CE	ε	238	EE
Å	143	8F	»	175	AF	╧	207	CF	∩	239	EF
É	144	90	▒	176	B0	╨	208	D0	≡	240	F0
æ	145	91	▓	177	B1	╤	209	D1	±	241	F1
Æ	146	92	│	178	B2	╥	210	D2	≥	242	F2
ô	147	93	┤	179	B3	╙	211	D3	≤	243	F3
ö	148	94	╡	180	B4	╘	212	D4	⌠	244	F4
ò	149	95	╢	181	B5	╒	213	D5	⌡	245	F5
û	150	96	╖	182	B6	╓	214	D6	÷	246	F6
ù	151	97	╕	183	B7	╫	215	D7	≈	247	F7
ÿ	152	98	╣	184	B8	╪	216	D8	°	248	F8
Ö	153	99	║	185	B9	┘	217	D9	∙	249	F9
Ü	154	9A	╗	186	BA	┌	218	DA	·	250	FA
¢	155	9B	╝	187	BB	█	219	DB	√	251	FB
£	156	9C	╜	188	BC	▄	220	DC	^{n}	252	FC
¥	157	9D	╛	189	BD	▌	221	DD	^{2}	253	FD
₧	158	9E	╞	190	BE	▐	222	DE	■	254	FE
ƒ	159	9F	┐	191	BF	▀	223	DF		255	FF

รหัสแอสกีสามารถพิจารณาเป็นขนาด 8 บิตหรือ 1 ไบต์ได้ โดยบิตที่นัยสำคัญที่มากที่สุด (MSB) จะเป็น 0 เสมอ ถ้าเขียนรหัสแอสกีในรูปของเลขฐานสิบหกจะมีค่าตั้งแต่ 00_{16}-$7F_{16}$ จากตารางที่ 3.4 แสดงรหัสแอสกีของอักขระแต่ละตัวในรูปของเลขฐานสอง เลขฐานสิบ และเลขฐานสิบหกตามลำดับ อักขระของรหัสแอสกี 32 ตัวแรก (00_{16}-$1F_{16}$) เป็นคำสั่งควบคุม รายละเอียดของคำสั่งควบคุมแสดงในตารางที่ 3.5 เช่น รหัสว่าง (null) การขึ้นบรรทัดใหม่ (line feed) การยกเลิก (escape) เป็นต้น ส่วนอักขระที่เหลือจะเป็นตัวอักษรตัวพิมพ์ใหญ่ ตัวพิมพ์เล็ก ตัวเลขฐานสิบและสัญลักษณ์กราฟิกต่างๆ นอกจากนี้รหัสแอสกีได้เพิ่มจำนวนอักขระอีก 128 ตัว ซึ่งเป็นที่ยอมรับในเครื่องคอมพิวเตอร์ส่วนบุคคล (PC) เนื่องจากเครื่องคอมพิวเตอร์ส่วนบุคคลเป็นที่นิยมใช้กันอย่างกว้างขวางทำให้ต้องมีอักขระที่จำเป็นในการใช้งานเพิ่มขึ้น อักขระที่เพิ่มขึ้นมาจะมีขนาด 8 บิต (80_{16}-FF_{16}) โดยเพิ่มตัวอักษรภาษาละติน สัญลักษณ์ของสกุลเงิน ตัวอักษรกรีก สัญลักษณ์ทางคณิตศาสตร์ กราฟิกต่างๆ ดังตารางที่ 3.6 ในปัจจุบันได้มีรหัสมาตรฐาน Unicode มีขนาด 16 บิตที่พัฒนาขึ้นมาเพื่อรองรับการใช้งานมากขึ้นโดยใช้แทนตัวอักษรของภาษาแถบเอเชียและตะวันออกกลางได้หรือภาษาอื่นๆ ๆด้

ตัวอย่างที่ 3.5 ข้อความที่เข้ารหัสแอสกีต่อไปนี้ แสดงข้อความว่าอะไร

1001000 1000101 1001100 1010000

<u>วิธีทำ</u>

จากตารางที่ 3.4 เมื่อทำการแปลงกลุ่มรหัส 7 บิตเป็นตัวอักษรดังนี้

1001000	1000101	1001100	1010000
⇓	⇓	⇓	⇓
48_{16}	45_{16}	$4C_{16}$	50_{16}
⇓	⇓	⇓	⇓
H	E	L	P

ตัวอย่างที่ 3.6 จงเขียนรหัสแอสกี ที่ได้จากการพิมพ์คำสั่งของโปรแกรมภาษาเบสิคต่อไปนี้ ผ่านทางคีย์บอร์ด โดยแสดงรหัสแอสกีในรูปของเลขฐานสองและเลขฐานสิบหก

20 PRINT “A=”;X

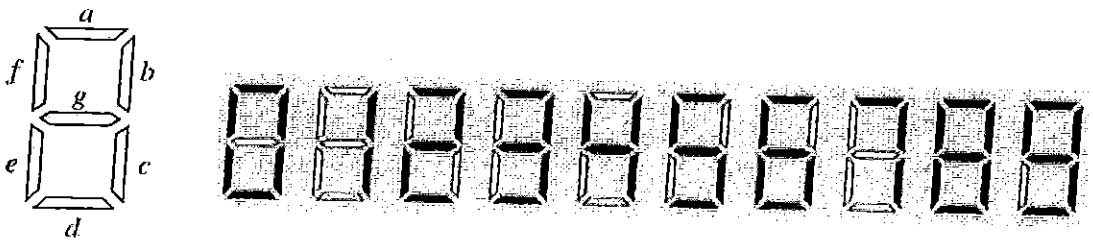
<u>วิธีทำ</u>

จากตารางที่ 3.4 คำสั่ง 20 PRINT “A=”;X สามารถเขียนแทนด้วยรหัสแอสกีได้ดังนี้

Symbol	Binary	Hexadecimal
2	0110010	32_{16}
0	0110000	30_{16}
space	0100000	20_{16}
P	1010000	50_{16}
R	1010010	52_{16}
I	1001001	49_{16}
N	1001110	$4E_{16}$
T	1010100	54_{16}
space	0100000	20_{16}
“	0100010	22_{16}
A	1000001	41_{16}
=	0111101	$3D_{16}$
“	0100010	22_{16}
;	0111011	$3B_{16}$
X	1011000	58_{16}

3.3 รหัสแสดงผล (Display code)

การแสดงผลตัวเลขที่เราสามารถเข้าใจได้ทันที ในเครื่องคอมพิวเตอร์หรืออุปกรณ์ดิจิตอล เช่น เครื่องคิดเลข เครื่องมือวัดแบบดิจิตอล นาฬิกาดิจิตอล เป็นต้น จะใช้ส่วนที่แสดงผลตัวเลขเป็นแบบรหัส 7 ส่วน (7 Segment code) ซึ่งมี 2 แบบคือแบบผลึกเหลว (Liquid Crystal Display, LCD) และแบบไดโอดเปล่งแสง (Light Emitting Diode, LED) รูปแบบของการแสดงผลตัวเลขของรหัส 7 ส่วนแบบไดโอดเปล่งแสง สามารถใช้แสดงผลเลข 0-9 โดยกำหนดให้แต่ละส่วนแสดงการติดหรือดับของไดโอดได้ รหัส 7 ส่วนจะมีชื่อเป็นตัวอักษรกำกับในแต่ละส่วนดังรูปที่ 3.4(ก) โดยเริ่มจาก a ถึง f ตัวอักษรเหล่านี้จะเรียงในทิศตามเข็มนาฬิกา จากรูปที่ 3.4 (ข) และตารางที่ 3.7 เป็นการแสดงผลตัวเลข 0-9 เมื่อกำหนดให้ LED ติดแสดงเลข 0 และ LED ดับแสดงเลข 1 ตัวอย่างเช่น ถ้าต้องการให้แสดงผลเป็นเลข 3 ต้องทำให้ LED ทุกส่วนติดยกเว้น LED ส่วนของ f และ e เป็นต้น



(ก) (ข)

รูปที่ 3.4 (ก) การกำหนดตัวอักษรของรหัส 7 ส่วน (ข) การแสดงผลตัวเลข 0-9

ตารางที่ 3.7 ค่า 0 และ 1 ของรหัส 7 ส่วนเมื่อแสดงตัวเลข 0-9

Decimal Digit	Seven segments						
	a	*b*	*c*	*d*	*e*	*f*	*g*
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1

3.4 การตรวจสอบความผิดพลาดของข้อมูล (Error detection)

การรับส่งข้อมูลในระบบดิจิตอลระหว่างจุด 2 จุดใดๆ หรือระหว่างด้านรับและด้านส่งนั้น ขณะที่อุปกรณ์หรือภายในวงจรดิจิตอลเองทำการรับส่งข้อมูลกัน ข้อมูลที่ได้รับมาหรือข้อมูลที่ส่งออกไปอาจมีความผิดพลาดของข้อมูลเกิดขึ้นได้ เนื่องจากข้อมูลที่ใช้มีเพียงเลข 0 และ 1 เท่านั้น เมื่อพิจารณาจากรหัสไบควินารีและรหัส 2 ใน 5 ที่ใช้แทนตัวเลข 0-9 จะเห็นว่ารหัสทั้งสองประกอบด้วยเลข 1 จำนวน 2 บิตเท่านั้น ลักษณะดังกล่าวสามารถช่วยตรวจสอบความถูกต้องของข้อมูลได้ง่ายกว่ารหัสแบบอื่นๆ ที่มีเลข 1 เป็นจำนวนบิตที่ไม่แน่นอน ข้อมูลที่ไม่ถูกต้องอาจเกิดขึ้นเมื่อข้อมูลมีการเปลี่ยนจาก 1 เป็น 0 หรือจาก 0 เป็น 1 ตัวอย่างเช่น รหัส 2 ใน 5 เมื่อด้านส่งทำการส่งบิตข้อมูล "00110" ไปยังด้านรับ เมื่อด้านรับได้รับบิตข้อมูลเป็น "01110" จะเห็นว่าบิตข้อมูลที่ได้รับมาไม่ถูกต้อง ดังนั้นระบบดิจิตอลจำเป็นต้องมีการออกแบบวงจรขึ้นมาเพื่อตรวจสอบความถูกต้องของข้อมูลหรือสามารถแก้ไขข้อมูลใหม่ให้ถูกต้องได้ วิธีการตรวจสอบข้อมูลที่นิยมใช้กันได้แก่ บิตพาริตี้เป็นการเพิ่มบิตเข้าไปในบิตข้อมูลเพื่อใช้ในการตรวจสอบและการเข้ารหัสแฮมมิ่งที่สามารถหาตำแหน่งของบิตที่ไม่ถูกต้องได้

3.4.1 บิตพาริตี้ (Parity bit)

บิตพาริตี้คือบิตเลข 0 หรือ 1 ที่เพิ่มเข้าไปในกลุ่มข้อมูลใดๆ แล้ว เมื่อรวมกลุ่มบิตข้อมูลและบิตพาริตี้จะได้บิตที่เป็นเลข 1 มีจำนวนเลขคู่เรียกว่าพาริตี้คู่ (Even parity) หรือบิตที่เป็นเลข 1 มีจำนวนเลขคี่เรียกว่าพารีตี้คี่ (Odd parity) ตัวอย่างการสร้างพารีตี้ของข้อมูลรหัส BCD 8421 และบิตพาริตี้แสดงได้ดังตารางที่ 3.8 จากตารางที่ 3.8 (ก) เป็นการสร้างพาริตี้คี่โดยบิตพาริตี้ (p) ที่เพิ่มเข้าไปอาจเป็นเลข 0 หรือ 1 แล้วทำให้จำนวนบิตที่เป็น 1 ทั้งหมดมีจำนวนเป็นเลขคี่ และตารางที่ 3.8 (ข) เป็นการสร้างพาริตี้คู่โดยบิตพาริตี้ (p) ที่เพิ่มเข้าไปอาจเป็นเลข 0 หรือ 1 แล้วทำให้จำนวน

บิตที่เป็น 1 ทั้งหมดมีจำนวนเป็นเลขคู่ เมื่อระบบดิจิตอลตรวจพบบิตข้อมูลที่ไม่ถูกต้อง ระบบอาจทำการร้องขอให้มีการส่งข้อมูลใหม่อีกครั้งหรือแจ้งเตือนว่าข้อมูลที่ได้รับมามีความผิดพลาดของข้อมูลเกิดขึ้น

ตารางที่ 3.8 การสร้างพาริตี้ของรหัส 8421 (ก) พาริตี้คี่ (ข) พาริตี้คู่

Decimal digit	8421*p*	Decimal digit	8421*p*
0	00001	0	00000
1	00010	1	00011
2	00100	2	00101
3	00111	3	00110
4	01000	4	01001
5	01011	5	01010
6	01101	6	01100
7	01110	7	01111
8	10000	8	10001
9	10011	9	10010

(ก) (ข)

การตรวจสอบความถูกต้องของข้อมูลโดยใช้บิตพาริตี้ สามารถตรวจสอบเมื่อมีข้อมูลไม่ถูกต้องเพียงบิตเดียวเท่านั้น (Single error) ถ้าข้อมูลมีบิตที่ไม่ถูกต้องจำนวน 2 บิต (Double error) จะไม่สามารถทำการตรวจสอบได้ ตัวอย่างเช่น เมื่อระบบตรวจสอบความถูกต้องของข้อมูลยอมรับพาริตี้คู่ และด้านส่งทำการส่งบิตข้อมูลและบิตพาริตี้ของเลข 9 (1001*0*) ถ้าด้านรับได้รับข้อมูลที่ไม่ถูกต้องเพียงบิตเดียวดังตารางที่ 3.9

ตารางที่ 3.9 ตัวอย่างการตรวจสอบความผิดพลาดโดยใช้บิตพาริตี้ของเลข 9

ด้านส่ง	ด้านรับ	
	ตัวอย่างข้อมูลผิดเพียง 1 บิต	ตัวอย่างข้อมูลผิด 2 บิต
1001*0*	0001*0*	0101*0*
	1101*0*	0011*0*
	1011*0*	0000*0*
	1000*0*	0001*1*
	1001*1*	1111*0*

เมื่อพิจารณาพาริตี้ของข้อมูลที่ได้รับจะเห็นว่าเป็นพาริตี้คี่ ซึ่งไม่สอดคล้องกับระบบที่ยอมรับพาริตี้คู่ ถ้าข้อมูลที่ได้รับมีบิตที่ไม่ถูกต้องจำนวน 2 บิต เมื่อพิจารณาพาริตี้ของข้อมูลที่ได้รับ

จะเห็นว่าเป็นพาริตี้คู่ซึ่งเหมือนกับที่ระบบกำหนดไว้ แต่ข้อมูลที่ได้รับไม่ใช่ข้อมูลที่ถูกต้อง จากตัวอย่างทั้งหมดที่กล่าวมาจะเห็นว่าการตรวจสอบความถูกต้องของข้อมูลด้วยบิตพาริตี้นั้น สามารถใช้ตรวจสอบข้อมูลที่ผิดเพียงบิตเดียวเท่านั้นและไม่สามารถหาตำแหน่งของบิตที่ไม่ถูกต้องได้

3.4.2 รหัสแฮมมิ่ง (Hamming Codes)

นาย R.W. Hamming เป็นผู้คิดค้นรหัสแฮมมิ่งขึ้นมา เป็นรูปแบบของรหัสที่รู้จักกันเป็นอย่างดีเพื่อใช้ในการตรวจสอบความถูกต้องของข้อมูล โดยตรวจสอบบิตข้อมูลที่ผิดเพียงบิตเดียวและบอกตำแหน่งของบิตข้อมูลที่ผิดได้ แต่ถ้าข้อมูลผิดหลายบิตจะไม่สามารถบอกตำแหน่งของบิตที่ผิดได้ ข้อมูลที่นำมาเข้ารหัสแฮมมิ่งจะมีบิตพาริตี้จำนวนหลายบิตและมีรูปแบบเป็นพาริตี้คู่ ตัวอย่างการตรวจสอบข้อมูลเลข 0-9 สมมุติให้ข้อมูลเลข 0-9 มีการเข้ารหัส BCD ดังนั้นบิตข้อมูล (data bit) ของเลข 0-9 มีจำนวน 4 บิต ($b_8b_4b_2b_1$) และมีบิตพาริตี้จำนวน 3 บิต ($p_4p_2p_1$) ดังนั้นการเข้ารหัสแฮมมิ่งมีจำนวนบิตทั้งหมด 7 บิต การจัดวางตำแหน่งของบิตข้อมูลและบิตพาริตี้ เป็นดังนี้

ตำแหน่งที่ (Position)	7	6	5	4	3	2	1
รูปแบบการจัดวางของบิตข้อมูลและบิตพาริตี้	b_8	b_4	b_2	p_4	b_1	p_2	p_1

การหาค่าบิตพาริตี้ของ $p_4p_2p_1$ สามารถหาได้จากความสัมพันธ์ของกลุ่มบิตต่างๆ ดังนี้

บิตพาริตี้ p_4 แสดงพาริตี้คู่ ร่วมกับบิตข้อมูลตำแหน่งที่ 5, 6, 7 (4, 5, 6, 7)

บิตพาริตี้ p_2 แสดงพาริตี้คู่ ร่วมกับบิตข้อมูลตำแหน่งที่ 3, 6, 7 (2, 3, 6, 7)

บิตพาริตี้ p_1 แสดงพาริตี้คู่ ร่วมกับบิตข้อมูลตำแหน่งที่ 3, 5, 7 (1, 3, 5, 7)

ตัวอย่างของการเข้ารหัสแฮมมิ่งของเลข 6 ที่มีบิตข้อมูล $b_8b_4b_2b_1$ = 0110 จะวางข้อมูลในตำแหน่งที่ 7, 6, 5, 3 ตามลำดับ จากนั้นทำการหาค่าของบิตพาริตี้ $p_4p_2p_1$ ดังนี้

p_4 แสดงพาริตี้คู่ร่วมกับบิตข้อมูลตำแหน่งที่ 5, 6, 7 ที่มีค่าเป็น 110 ดังนั้น p_4 ต้องมีค่าเป็น 0

p_2 แสดงพาริตี้คู่ร่วมกับบิตข้อมูลตำแหน่งที่ 3, 6, 7 ที่มีค่าเป็น 010 ดังนั้น p_2 ต้องมีค่าเป็น 1

p_1 แสดงพาริตี้คู่ร่วมกับบิตข้อมูลตำแหน่งที่ 3, 5, 7 ซึ่งมีค่าเป็น 010 ดังนั้น p_1 ต้องมีค่าเป็น 1

ดังนั้นเลข 6 เมื่อทำการเข้ารหัสแฮมมิ่งจะได้รูปแบบของรหัสดังนี้

ตำแหน่งที่ (Position)	7	6	5	4	3	2	1
รูปแบบการจัดวางของบิตข้อมูลและบิตพาริตี้	b_8	b_4	b_2	p_4	b_1	p_2	p_1
การเข้ารหัสแฮมมิ่งของเลข 6	0	1	1	0	0	1	1

ตัวเลข 0 ถึง 9 ที่เข้ารหัสแฮมมิ่งแล้ว จะมีรูปแบบของรหัสแตกต่างกันไปดังตารางที่ 3.10 เมื่อรหัส 8421 ของเลข 0-9 มีการเข้ารหัสแฮมมิ่ง บิตข้อมูลและบิตพาริตี้ของตัวเลขแต่ละตัวจะมีขนาด 7 บิต ข้อมูลขนาด 7 บิตนี้จะถูกส่งไปยังด้านรับ เมื่อด้านรับได้รับข้อมูลจะทำการตรวจสอบข้อมูลในแต่ละกลุ่มของบิตพาริตี้ ในที่นี้กำหนดให้ $C_4C_2C_1$ เป็นข้อมูลที่ได้จากการตรวจสอบพาริตี้ในกลุ่มบิตพาริตี้ $p_4p_2p_1$ ตามลำดับ ถ้ากลุ่มบิตเป็นพาริตี้คู่กำหนดให้ $C_i = 0$ และกลุ่มบิตเป็นพาริตี้คี่กำหนดให้ $C_i = 1$ จากนั้นให้นำค่า $C_4C_2C_1$ ที่ได้จากการตรวจสอบพาริตี้ของแต่ละกลุ่ม มาหาตำแหน่งของบิตที่ไม่ถูกต้อง ดังตารางที่ 3.11

ตารางที่ 3.10 รูปแบบของรหัส BCD 8421 ที่เข้ารหัสแฮมมิ่ง

Decimal Digit	7	6	5	4	3	2	1	ตำแหน่ง
	b_8	b_4	b_2	p_4	b_1	p_2	p_1	บิตข้อมูลและบิตพาริตี้
0	0	0	0	0	0	0	0	
1	0	0	0	0	1	1	1	
2	0	0	1	1	0	0	1	
3	0	0	1	1	1	1	0	
4	0	1	0	1	0	1	0	
5	0	1	0	1	1	0	1	
6	0	1	1	0	0	1	1	
7	0	1	1	0	1	0	0	
8	1	0	0	1	0	1	1	
9	1	0	0	1	1	0	0	

ตารางที่ 3.11 การหาตำแหน่งของบิตที่ไม่ถูกต้อง

C_4	C_2	C_1	ตำแหน่งของบิตที่ผิดพลาด
0	0	0	ไม่มี
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

ตัวอย่างที่ 3.7 เมื่อด้านรับได้รับข้อมูลมาดังนี้ 0100011 จงตรวจสอบว่าข้อมูลที่ได้รับถูกต้องหรือไม่ ถ้าข้อมูลที่ได้รับไม่ถูกต้องมีจงแก้ไขข้อมูลนั้น

<u>วิธีทำ</u>

(1) การจัดวางตำแหน่งของข้อมูลที่ได้รับ เป็นดังนี้

ตำแหน่งที่ (Position)	7	6	5	4	3	2	1
รูปแบบการจัดวางของบิตข้อมูลและบิตพาริตี้	b_8	b_4	b_2	p_4	b_1	p_2	p_1
ข้อมูลที่ได้รับ	0	1	0	0	0	1	1

(2) ตรวจสอบกลุ่มบิตพาริตี้ดังนี้

กลุ่มบิตพาริตี้ p_4 (ตำแหน่งที่ 4, 5, 6, 7) $\Rightarrow$ 0 0 1 0 เป็นพาริตี้คี่ ดังนั้น $C_4 = 1$

กลุ่มบิตพาริตี้ p_2 (ตำแหน่งที่ 2, 3, 6, 7) $\Rightarrow$ 1 0 1 0 เป็นพาริตี้คู่ ดังนั้น $C_2 = 0$

กลุ่มบิตพาริตี้ p_1 (ตำแหน่งที่ 1, 3, 5, 7) $\Rightarrow$ 1 0 0 0 เป็นพาริตี้คี่ ดังนั้น $C_1 = 1$

(3) ดังนั้นตำแหน่งของบิตที่ไม่ถูกต้องคือ $C_4C_2C_1$ =101 =5 เมื่อแก้ไขข้อมูลในตำแหน่งที่ 5 จะได้ข้อมูลเป็นเลข 6 (0110) จากเดิมที่รับมาเป็นเลข 4 (0100)

ตัวอย่างที่ 3.8 เมื่อด้านรับได้รับข้อมูลมาดังนี้ 0110011 จงตรวจสอบว่าข้อมูลที่ได้รับถูกต้องหรือไม่ ถ้าข้อมูลที่ได้รับไม่ถูกต้องมีจงแก้ไขข้อมูลนั้น

<u>วิธีทำ</u>

(1) การจัดวางตำแหน่งของข้อมูลที่ได้รับ เป็นดังนี้

ตำแหน่งที่ (Position)	7	6	5	4	3	2	1
รูปแบบการจัดวางของบิตข้อมูลและบิตพาริตี้	b_8	b_4	b_2	p_4	b_1	p_2	p_1
ข้อมูลที่ได้รับ	0	1	1	0	0	1	1

(2) ตรวจสอบกลุ่มบิตพาริตี้ดังนี้

กลุ่มบิตพาริตี้ p_4 (ตำแหน่งที่ 4, 5, 6, 7) $\Rightarrow$ 0 1 1 0 เป็นพาริตี้คู่ ดังนั้น $C_4 = 0$

กลุ่มบิตพาริตี้ p_2 (ตำแหน่งที่ 2, 3, 6, 7) $\Rightarrow$ 1 0 1 0 เป็นพาริตี้คู่ ดังนั้น $C_2 = 0$

กลุ่มบิตพาริตี้ p_1 (ตำแหน่งที่ 1, 3, 5, 7) $\Rightarrow$ 1 0 1 0 เป็นพาริตี้คู่ ดังนั้น $C_1 = 0$

(3) ดังนั้นตำแหน่งของบิตที่ไม่ถูกต้องคือบิตที่ $C_4C_2C_1$ =000 =0 หรือไม่มีตำแหน่งของบิตที่ผิด ดังนั้นข้อมูลด้านส่งและด้านรับตรงกันคือเลข 6 (0110)

ตัวอย่างที่ 3.9 กำหนดให้มีการส่งข้อมูลของเลข 6 ที่มีการเข้ารหัสแฮมมิ่งคือ 0110011 เมื่อด้านรับได้รับข้อมูลดังนี้ 1100011 ซึ่งมีบิตที่ผิดไปจำนวน 2 บิต จงตรวจสอบความผิดพลาดด้วยวิธีการของรหัสแฮมมิ่ง

<u>วิธีทำ</u>

(1) การจัดวางตำแหน่งของข้อมูลที่ได้รับ เป็นดังนี้

ตำแหน่งที่ (Position)	7	6	5	4	3	2	1
รูปแบบการจัดวางของบิตข้อมูลและบิตพาริตี้	b_8	b_4	b_2	p_4	b_1	p_2	p_1
ข้อมูลที่ได้รับ	1	1	0	0	0	1	1

(2) ตรวจสอบกลุ่มบิตพาริตี้ดังนี้

กลุ่มบิตพาริตี้ p_4 (ตำแหน่งที่ 4, 5, 6, 7) $\Rightarrow$ 0 0 1 1 เป็นพาริตี้คู่ ดังนั้น $C_4 = 0$

กลุ่มบิตพาริตี้ p_2 (ตำแหน่งที่ 2, 3, 6, 7) $\Rightarrow$ 1 0 1 1 เป็นพาริตี้คี่ ดังนั้น $C_2 = 1$

กลุ่มบิตพาริตี้ p_1 (ตำแหน่งที่ 1, 3, 5, 7) $\Rightarrow$ 1 0 0 1 เป็นพาริตี้คู่ ดังนั้น $C_1 = 0$

(3) ดังนั้นตำแหน่งของบิตที่ไม่ถูกต้องคือ $C_4C_2C_1$ =010 =2 เมื่อแก้ไขข้อมูลแล้ว จะได้ข้อมูลไม่ตรงกับข้อมูลด้านส่ง จะเห็นว่าการตรวจสอบความผิดพลาดของข้อมูลที่มีบิตไม่ถูกต้องจำนวน 2 บิต สามารถบอกได้ว่าข้อมูลที่ได้รับไม่ถูกต้องแต่ไม่สามารถหาตำแหน่งของบิตที่ผิดได้

<u>หมายเหตุ</u>

การตรวจสอบและแก้ไขข้อมูลด้วยวิธีการเข้ารหัสแฮมมิ่งนั้น ถ้าบิตข้อมูลที่นำมาเข้ารหัสแฮมมิ่งจำนวน m บิต จะต้องใช้บิตพาริตี้จำนวน k บิต เขียนสมการแสดงความสัมพันธ์ได้ดังนี้

$$2^k - 1 \geq m + k$$

จากตารางที่ 3.12 แสดงความสัมพันธ์ระหว่างจำนวนบิตพาริตี้กับจำนวนบิตข้อมูลต่างๆ ข้อมูลที่เข้ารหัสแฮมมิ่งแล้วจะมีบิตข้อมูลและบิตพาริตี้รวมกันได้ทั้งหมดจำนวน $m+k$ บิต และการจัดวางตำแหน่งของกลุ่มบิตพาริตี้ที่ใช้ในการตรวจสอบ แสดงได้ดังตารางที่ 3.13 จะเห็นว่าบิตพาริตี้มีการวางตำแหน่งที่ 2^n เสมอ เมื่อ $n = 0,1,2,...$

ตารางที่ 3.12 ความสัมพันธ์ระหว่างจำนวนบิตพาริตี้กับจำนวนบิตข้อมูล

Range of m information bits	Number of parity bits k
2–4	3
5–11	4
12–26	5
27–57	6
58–120	7

ตารางที่ 3.13 ตำแหน่งบิตของแต่ละกลุ่มที่ใช้ในการตรวจสอบพาริตี้

Parity bit position	Positions checked
1	1, 3, 5, 7, 9, 11, 13, 15, 17, . . .
2	2, 3, 6, 7, 10, 11, 14, 15, 18, . . .
4	4, 5, 6, 7, 12, 13, 14, 15, 20, . . .
8	8, 9, 10, 11, 12, 13, 14, 15, 24, . . .
⋮	⋮

แบบฝึกหัดท้ายบท

1. จงเขียนเลขฐานสิบต่อไปนี้ในรูปของรหัส BCD 8421

   (ก) 98 (ข) 170 (ค) 2469

2. จงแปลงรหัส BCD 8421 ต่อไปนี้เป็นเลขฐานสิบ

   (ก) 10000110 (ข) 001101010001 (ค) 1001010001110000

3. จงบวกเลข BCD 8421 ต่อไปนี้

   (ก) 00100011+00010101

   (ข) 010001010000+010000010111

   (ค) 00010110+00010101

   (ง) 01100111+01010011

4. จงแปลงเลขฐานสอง 11000110 เป็นรหัสเกรย์

5. จงแปลงรหัสเกรย์ 10101111 เป็นเลขฐานสอง

6. จงเขียนรหัสแอสกีในรูปของเลขฐานสิบหกเมื่อพิมพ์คำสั่งของโปรแกรมดังนี้

   80 INPUT Y

7. จากกลุ่มรหัสที่กำหนดให้จงเติมบิตพาริตี้คู่

   (ก) 1001 (ข) 111000 (ค) 101101

8. จากรหัส BCD ที่เข้ารหัสแฮมมิ่งเมื่อได้รับข้อมูลเป็น 1001000 จงหาตำแหน่งของบิตที่ผิดและแก้ไขข้อมูลให้ถูกต้อง

เฉลยแบบฝึกหัดท้ายบท

1. (ก) 98=$(1001\ 1000)_{BCD}$

   (ข) 170 =$(0001\ 0111\ 0000)_{BCD}$

   (ค) 2469=$(0010\ 0100\ 0110\ 1001)_{BCD}$

2. (ก) 10000110 = 86

   (ข) 001101010001=351

   (ค) 1001010001110000=9470

3. (ก) 00111000

   (ข) 100001100111

   (ค) 00110001

   (ง) 000100100000

4. 10100101

5. 11001010

6. 80 INPUT Y = $38_{16}30_{16}20_{16}49_{16}4E_{16}50_{16}55_{16}54_{16}20_{16}59_{16}$

7. (ก) 1001 = 1001<u>0</u>

   (ข) 111000 = 111000<u>1</u>

   (ค) 101101 = 101101<u>0</u>

8. **บิตตำแหน่ง**ที่ 3 เปลี่ยนจาก 0 เป็น 1 ข้อมูลคือเลข 9

บทที่ 4
ลอจิกเกต (Logic Gates)

วัตถุประสงค์

1. สามารถเข้าใจและอธิบายการทำงานของเกตแต่ละชนิดในรูปของตารางความจริง ไดอะแกรมเวลาและสมการบูลีน
2. สามารถเข้าใจหลักการทำงานและความสำคัญของเกตเอนกประสงค์
3. สามารถเข้าใจความสำคัญของเทคโนโลยีไอซี

บทนี้อธิบายถึงลอจิกเกตชนิดต่างๆ ที่ใช้ในการออกแบบวงจรดิจิตอล ตั้งแต่เกตพื้นฐานจนถึงเกตซับซ้อนที่สร้างมาจากเกตพื้นฐาน รวมทั้งสัญลักษณ์มาตรฐาน (Logic symbol) ที่ใช้แทนลอจิกเกต การทำงานของลอจิกเกต (Logical operation) ความสัมพันธ์ของรูปคลื่น (Waveform) ระหว่างสัญญาณอินพุตและเอาต์พุตของเกตในไดอะแกรมเวลา (Timing diagram) และการประยุกต์ใช้งานของลอจิกเกต

4.1 บทนำ

เมื่อพิจารณาอินพุตและเอาต์พุตในทางลอจิกหรือตรรกศาสตร์แล้ว มีสองเงื่อนไขคือจริง (True) หรือเท็จ (False) เงื่อนไขนี้สามารถแทนด้วยเลขฐานสองคือ 1 หรือ 0 ได้ตามลำดับ โดยทั่วไปวงจรดิจิตอลจะใช้ค่าระดับของแรงดันไฟฟ้าเพื่อแสดงสถานะเป็น 0 หรือ 1 ดังนั้นจึงจำเป็นต้องมีวิธีการอธิบายการทำงานของวงจรที่มีค่าลอจิก 0 หรือ 1

พีชคณิตบูลีน (Boolean algebra) เป็นคณิตศาสตร์อย่างง่ายที่สามารถนำมาใช้อธิบายการทำงานของวงจรลอจิกหรือวงจรดิจิตอลและเป็นเครื่องมือที่นำมาใช้ในการออกแบบและวิเคราะห์วงจรมานานหลายปีแล้ว โดยใช้นิพจน์บูลีน (Boolean expression) แสดงความสัมพันธ์ระหว่างอินพุตและเอาต์พุตของวงจร พีชคณิตบูลีนมีการดำเนินการทางคณิตศาสตร์ที่สำคัญ 3 แบบคือ แอนด์ (AND) ออร์ (OR) นอต (NOT) และตัวเลขที่ใช้ในการดำเนินการทางคณิตศาสตร์มีเพียง 0 หรือ 1 เท่านั้น จะไม่มีเลขจำนวนจริง (Real number) เลขจินตภาพ (Imaginary number) เลขทศนิยม (Fraction) หรือเลขอื่นๆ พีชคณิตบูลีนได้นำมาใช้ในการสร้างวงจรขึ้นใหม่ให้อยู่ในรูปวงจรอย่างง่ายหรือวงจรที่ใช้จำนวนเกตที่น้อยกว่าวงจรเดิม นอกจากพีชคณิตบูลีนแล้วยังมีเทคนิคอื่นๆ ที่เกี่ยวข้องกับการออกแบบและวิเคราะห์วงจรดิจิตอล ได้แก่ ตารางความจริง สัญลักษณ์ของเกต และไดอะแกรมเวลา

ลอจิกเกต (Logic gates)

ลอจิกเกตคืออุปกรณ์ที่มีสัญญาณอินพุตตั้งแต่ 1 สัญญาณขึ้นไปและมีสัญญาณเอาต์พุตเพียง 1 สัญญาณเท่านั้น สัญญาณอินพุตและเอาต์พุตมีความสัมพันธ์กันแบบตรรกศาสตร์ ได้แก่ และ(AND) หรือ (OR) ไม่ใช่ (NOT) สามารถแสดงบล็อกไดอะแกรมของลอจิกเกตทั่วไปดังรูปที่ 4.1 วงจรดิจิตอลที่มีการทำงานต่างกัน เช่น การกระทำทางคณิตศาสตร์ (บวก ลบ คูณ หาร) หรือการกระทำทางพีชคณิตบูลีน เกิดจากการนำเอาลอจิกเกตมาประกอบกันเป็นวงจร เรียกว่าวงจรลอจิกเชิงจัดหมู่ (Combinational logic) หรือวงจรคอมบิเนชัน ลอจิกเกตได้ผลิตออกมาเป็นวงจรรวมหรือไอซี (Integrated circuits, IC) เพื่อให้ผู้ใช้สามารถนำไอซีมาประกอบเป็นวงจรที่มีการทำงานตามต้องการได้ วงจรภายในของอุปกรณ์ลอจิกเกตประกอบด้วยทรานซิสเตอร์ที่ทำหน้าที่เป็นเสมือนสวิตซ์ เช่น ไบโพลาร์ทรานซิสเตอร์ (Bipolar transistor) ซีมอส (CMOS) เป็นต้น การที่โครงสร้างภายในของลอจิกเกตแตกต่างกันขึ้นอยู่กับชนิดของทรานซิสเตอร์ ทำให้การใช้กระแสและแรงดันไฟฟ้าแตกต่างกันด้วย

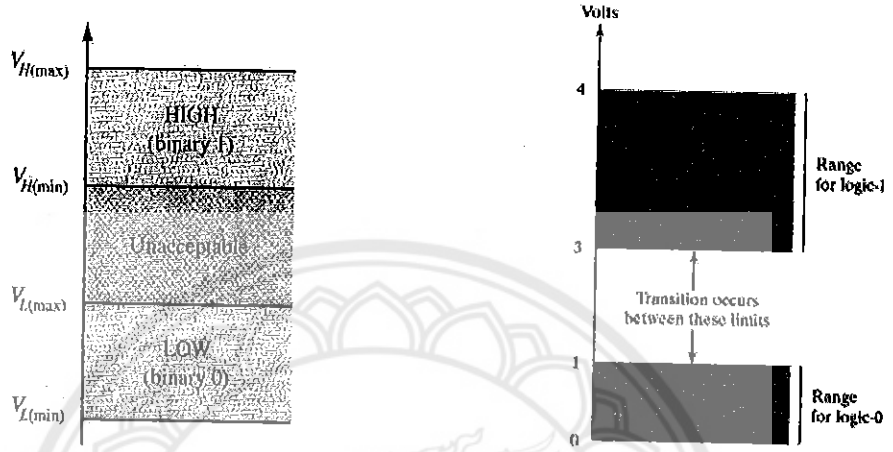
รูปที่ 4.1 บล็อกไดอะแกรมของลอจิกเกต

ค่าของระดับลอจิก (Logic level)

พีชคณิตบูลีนมีการกำหนดตัวแปรและค่าคงที่ ซึ่งใช้เพียง 2 ค่าคือ 0 หรือ 1 ตัวแปรจะมีค่าเป็น 0 หรือ 1 แตกต่างกันไปที่เวลาต่างๆ นอกจากนี้ค่าของตัวแปรที่ใช้อาจเป็นค่าของระดับแรงดันไฟฟ้า ตัวอย่างเช่น ค่าลอจิก 0 อาจมีค่าระดับแรงดันไฟฟ้าอยู่ในช่วง 0-0.8 โวลต์ หรือค่าลอจิก 1 อาจมีค่าระดับแรงดันไฟฟ้าอยู่ในช่วง 2-5 โวลต์ เป็นต้น ดังนั้นค่าลอจิก 0 หรือ 1 ของวงจรดิจิตอล จึงไม่ใช่ค่าแรงดันไฟฟ้าที่แท้จริงแต่เป็นเพียงค่าที่ใช้แทนค่าของระดับแรงดันไฟฟ้าเท่านั้น เรียกว่าค่าระดับลอจิก ระบบดิจิตอลจะมีคำในความหมายอื่นที่สามารถใช้เรียกแทนค่าลอจิก 0 หรือ 1 ดังตารางที่ 4.1 แต่ที่นิยมใช้ในการออกแบบและวิเคราะห์วงจรดิจิตอลคือ 1/0 และ HIGH/LOW อย่างไรก็ตามในทางปฏิบัติ HIGH จะมีขอบเขตของค่าแรงดันไฟฟ้าต่ำสุด ($V_{H(min)}$) และสูงสุด ($V_{H(max)}$) เช่นเดียวกับ LOW จะมีขอบเขตของค่าแรงดันไฟฟ้าต่ำสุด ($V_{L(min)}$) และสูงสุด ($V_{L(max)}$) เช่นกัน ค่าระดับแรงดันไฟฟ้าของ HIGH และ LOW จะมีค่าไม่เท่ากันขึ้นอยู่กับโครงสร้างภายในของวงจรเกตนั้นๆ จากรูปที่ 4.2 แสดงขอบเขตระดับแรงดันไฟฟ้าของ HIGH และ LOW ที่ใช้ในวงจรดิจิตอล สำหรับค่าแรงดันไฟฟ้าที่อยู่ระหว่าง $V_{L(max)}$ และ $V_{H(min)}$ จะเป็นช่วงของระดับแรงดันที่ไม่ใช้งาน (Unacceptable)

ตารางที่ 4.1 ความหมายของระดับลอจิก

ลอจิก 1	ลอจิก 0
TRUE	FALSE
ON	OFF
HIGH	LOW
YES	NO
Closed switch	Open switch



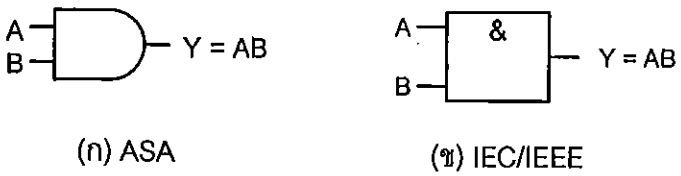
รูปที่ 4.2 ขอบเขตของระดับแรงดันไฟฟ้าเมื่อเทียบกับลอจิก 0 และ 1

4.2 สัญลักษณ์ลอจิกเกต (Logic symbol)

การเขียนวงจรดิจิตอลที่ทุกคนสามารถเข้าใจได้ตรงกัน ต้องมีการเขียนเป็นมาตรฐานเดียวกัน มาตรฐานการเขียนสัญลักษณ์ของลอจิกเกตมี 2 แบบ คือ

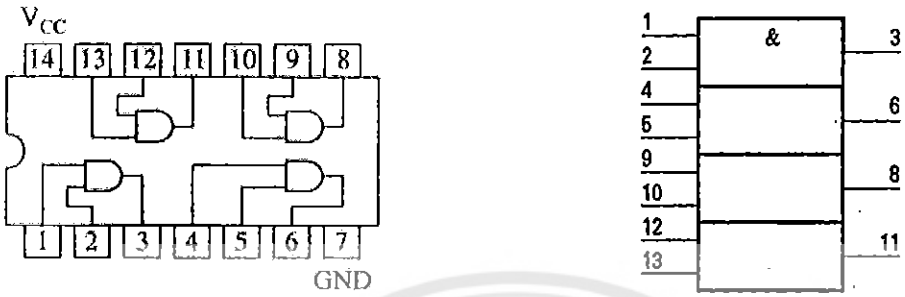
1. แบบ ASA (American Standard Association) หรือแบบมาตรฐาน ประกอบด้วยเส้นตรง เส้นโค้ง เป็นที่นิยมใช้กันทั่วไป แต่ข้อเสียของมาตรฐานนี้คือ การเขียนด้วยมืออาจทำให้เกิดการเข้าใจความหมายผิดได้ เช่น การเขียนเส้นตรงด้วยมือ อาจดูเป็นเส้นโค้ง ทำให้เข้าใจเป็นเกตอีกชนิดหนึ่งได้ เป็นต้น

2. แบบ IEC /IEEE (International Electrotechnical Commission/ Institute of Electrical and Electronic Engineering) เป็นรูปแบบที่มีมานานแล้ว แต่เริ่มเป็นที่นิยมใช้กันได้ไม่นานในทางอุตสาหกรรม หน่วยงานทหารของอเมริกาหรือบางบริษัท การเขียนสัญลักษณ์ลอจิกเกตแบบนี้จะแทนเกตด้วยรูปสี่เหลี่ยมและมีอักษรภายในรูปสี่เหลี่ยมเพื่อบอกคุณสมบัติของเกตชนิดนั้น เช่น ตัว "&" แทนแอนด์เกต เป็นต้น ข้อดีของการเขียนสัญลักษณ์แบบนี้คือสามารถใช้เขียนเป็นไอซีที่ประกอบด้วยเกตจำนวนมากกว่า 1 ตัวได้ โดยแบ่งพื้นที่สี่เหลี่ยมให้เรียงซ้อนกัน สี่เหลี่ยม 1 รูปใช้แทนเกต 1 ตัว ตัวอย่างรูปที่ 4.3 และ 4.4 เป็นการเขียนสัญลักษณ์แอนด์เกต 1 ตัวและ 4 ตัว ตามลำดับ



(ก) ASA (ข) IEC/IEEE

รูปที่ 4.3 สัญลักษณ์ของแอนด์เกตตามมาตรฐาน ASA และ IEC/IEEE



(ก) ASA (ข) IEC/IEEE

รูปที่ 4.4 ไอซีแอนด์เกตเบอร์ 7408 ตามมาตรฐาน ASA และ IEC/IEEE

4.3 การทำงานของลอจิกเกต (Logical operation)

การอธิบายการทำงานของลอจิกเกต เป็นการแสดงความสัมพันธ์ของค่าลอจิกระหว่างอินพุตและเอาต์พุตของเกต ซึ่งการอธิบายการทำงานของเกตสามารถนำไปใช้อธิบายการทำงานของวงจรดิจิตอลได้เช่นเดียวกัน สามารถอธิบายการทำงานได้ 2 แบบ ดังนี้

1. **ตารางความจริง** (Truth table)

ลอจิกเกตจะกำหนดตัวแปร (Variables) เป็นตัวอักษรให้กับจุดหรือตำแหน่งที่ต้องการแสดงสภาวะลอจิก แล้วนำตัวแปรเหล่านี้มาเขียนแสดงความสัมพันธ์กันในรูปของตารางความจริง การเขียนตารางความจริงนิยมให้ตัวแปรอินพุตอยู่ด้านซ้ายและตัวแปรเอาต์พุตอยู่ด้านขวาของตาราง ตัวแปรอินพุตสามารถแสดงค่าลอจิก 0 หรือ 1 ที่เป็นไปได้ทั้งหมดจำนวน 2^n กรณี เมื่อ n คือจำนวนตัวแปรอินพุต จากตารางที่ 4.2 เป็นตัวอย่างการเขียนค่าลอจิกของแอนด์เกต จะเห็นว่าตัวแปรอินพุต 2 ตัวคือ AB มีค่าลอจิกที่เป็นไปได้ทั้งหมด 4 กรณี ตัวแปรอินพุต 3 ตัวคือ ABC มีค่าลอจิกที่เป็นไปได้ทั้งหมด 8 กรณี และตัวแปรอินพุต 4 ตัวคือ ABCD มีค่าลอจิกที่เป็นไปได้ทั้งหมด 16 กรณี ตามลำดับ

ค่าลอจิกในแต่ละคอลัมน์ของตัวแปรอินพุตมีความสัมพันธ์กับค่า 2^n เช่น คอลัมน์แรกทางขวาสุดของตัวแปรมีค่าเท่ากับ $2^0=1$ ดังนั้นจะเขียนค่าลอจิก 0 และ 1 ทีละตัวสลับกันไป คอลัมน์ที่สองจากขวาจะมีค่าเท่ากับ $2^1=2$ ดังนั้นจะเขียนค่าลอจิก 0 และ 1 ทีละสองตัวสลับกันไป เป็นต้น เมื่อทำการแปลงค่าอินพุตของตัวแปรจากเลขฐานสองเป็นเลขฐานสิบจะได้ค่าตัวเลขเป็น 0, 1, 2, 3, 4,.... เรียงเป็นลำดับไป ค่าตัวเลขนี้เรียกว่าหมายเลขประจำแถวของตารางความจริง การ

เขียนในลักษณะนี้ทำให้สามารถตรวจสอบและพิจารณาค่าลอจิกได้สะดวก ค่าลอจิกที่เขียนในแถวเดียวกันแสดงว่าค่าลอจิกของตัวแปรแต่ละตัวเกิดขึ้นที่เวลาเดียวกัน จากตัวอย่างที่ 4.2 ตารางความจริงของอินพุต 2 ตัวแปร เมื่อพิจารณาค่าอินพุต A,B สัมพันธ์กับเอาต์พุต Y ในแถวแรกจะเห็นว่า A=0 และ B=0 จะให้เอาต์พุต Y=0 สำหรับค่าลอจิกที่นิยมนำมาเขียนในตารางความจริง คือ 1/0 จริง/เท็จ และ HIGH/LOW ค่าลอจิกของวงจรที่นำมาเขียนในตารางความจริงจะใช้เครื่องมือวัดค่าที่จุดของอินพุตและเอาต์พุตที่ต้องการ เครื่องมือที่ใช้วัด เช่น โวลต์มิเตอร์ ลอจิกโพรบ (Logic probe) เป็นต้น

ตารางที่ 4.2 ตารางความจริงของอินพุต 2, 3 และ 4 ตัวแปร

Input		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

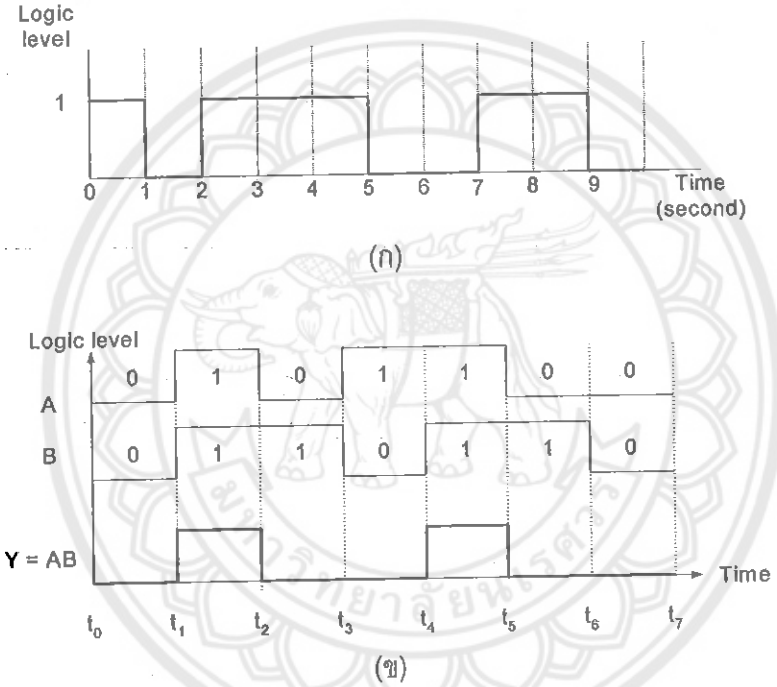
Input			Output
A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Input				Output
A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

2. ไดอะแกรมเวลา (Timing Diagram)

ไดอะแกรมเวลาคือกราฟที่ใช้แสดงความสัมพันธ์ของรูปคลื่น (Waveform) ตั้งแต่ 2 รูปคลื่นขึ้นไปที่เวลาใดๆ รูปคลื่นในที่นี้คือสัญญาณของอินพุตหรือเอาต์พุตที่มีค่าลอจิก 0 หรือ 1 ไดอะแกรมเวลาสามารถนำไปใช้แสดงความสัมพันธ์การทำงานของวงจรที่มีค่าหน่วงเวลา (Delay time) ได้ เนื่องจากการใช้งานจริง เมื่อทำการป้อนค่าลอจิกที่อินพุตของเกตหรือวงจรแล้ว เกตหรือวงจรต้องใช้เวลาช่วงหนึ่งจึงให้ค่าเอาต์พุตออกมา ค่าเวลาดังกล่าวเรียกว่าค่าหน่วงเวลา ลักษณะของไดอะแกรมแสดงดังรูปที่ 4.5 (ก) เมื่อแกน x แทนเวลาและแกน y แทนค่าระดับลอจิก 0 หรือ 1 จากรูปที่ 4.5 (ข) แสดงค่าลอจิกของอินพุต AB และเอาต์พุต Y ที่เวลาใดๆ จะเห็นว่าแกน y สามารถแสดงค่าลอจิกของอินพุตและเอาต์พุตได้หลายสัญญาณ แต่ละสัญญาณมีชื่อกำกับไว้ทางด้านซ้าย การพิจารณาว่าอินพุตและเอาต์พุตมีความสัมพันธ์กันอย่างไร ให้พิจารณาแต่ละสัญญาณที่เวลาเดียวกัน เช่น จากรูปที่ 4.5 (ก) แสดงค่าลอจิกของอินพุต AB และเอาต์พุต Y ที่เวลาใดๆ เมื่อ

Y แสดงค่าเอาต์พุตตามคุณสมบัติของแอนด์เกต จะเห็นว่าในช่วงเวลา t_0 ถึงเวลา t_1 อินพุต AB มีค่าลอจิก 0 และเอาต์พุต Y มีค่าลอจิก 0 ในช่วงเวลา t_1 ถึงเวลา t_2 อินพุต AB มีค่าลอจิก 1 และเอาต์พุต Y มีค่าลอจิก 1 ในช่วงเวลา t_2 ถึงเวลา t_3 อินพุต AB มีค่าลอจิก 0 และ 1 ตามลำดับ เอาต์พุต Y มีค่าลอจิก 0 เป็นต้น ค่าลอจิกที่เวลา t_0, t_1, t_2, ... จะมีการเปลี่ยนแปลงจากลอจิก 0 เป็น 1 หรือจากลอจิก 1 เป็น 0 การเปลี่ยนแปลงนี้จะเกิดขึ้นอย่างรวดเร็วทันทีทันใด ซึ่งไม่สามารถสังเกตเห็นการเปลี่ยนแปลงในระหว่างนั้นได้ แต่เมื่อเขียนไดอะแกรมเวลาจะเขียนเส้นต่อระหว่างลอจิก 0 และ 1 เพื่อให้เห็นการเปลี่ยนแปลงของสัญญาณอย่างต่อเนื่องชัดเจนและแสดงว่าสัญญาณที่เกิดขึ้นมีการเรียงต่อกัน



รูปที่ 4.5 ตัวอย่างของไดอะแกรมเวลา

4.4 ชนิดของลอจิกเกต (Logic gates)

ลอจิกเกตพื้นฐาน (Basic gate) ในวงจรดิจิตอลประกอบด้วยนอตเกต (NOT gate) แอนด์เกต (AND gate) และออร์เกต (OR gate) ลอจิกเกตเหล่านี้มีความสัมพันธ์ในทางตรรกศาสตร์คือ ไม่ใช่ (NOT) และ (AND) หรือ(OR) ตามลำดับ สำหรับค่า"จริง" ในทางตรรกศาสตร์จะแทนด้วยลอจิก 1 และค่า"เท็จ" จะแทนด้วยลอจิก 0 สำหรับเกตชนิดอื่นเป็นเกตที่ประกอบขึ้นจากเกตพื้นฐาน ได้แก่ แนนด์เกต (NAND gate) นอร์เกต (NOR gate) เอ็กคลูซีฟออร์เกต (Exclusive-OR gate) และเอ็กคลูซีฟนอร์เกต (Exclusive-NOR gate) การเข้าใจคุณสมบัติของเกตเหล่านี้ จะช่วยให้สามารถทำการออกแบบและวิเคราะห์การทำงานของวงจรดิจิตอลได้ คุณสมบัติของเกตแต่ละชนิดมีรายละเอียดดังนี้

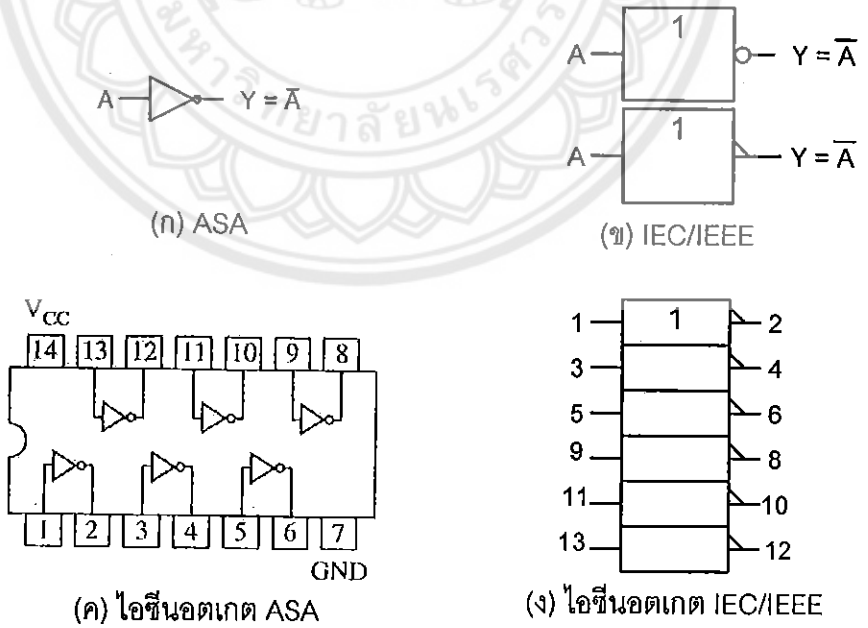
4.4.1 นอตเกตหรืออินเวอร์เตอร์ (NOT gate or INVERTER)

นอตเกตมีเพียง 1 อินพุตเท่านั้น ซึ่งต่างจากเกตชนิดอื่นที่มีตั้งแต่ 2 อินพุตขึ้นไป ถ้ากำหนดให้อินพุตของนอตเกตเป็นตัวแปร A และเอาต์พุตเป็นตัวแปร Y สามารถเขียนในรูปสมการบูลีน ดังนี้

$$Y = \overline{A} = \text{NOT} \quad A$$

จากสมการบูลีนอ่านว่า Y เท่ากับ A บาร์หรือคอมพลีเมนต์ของ A หรือนอต A หรืออินเวอร์สของ A เครื่องหมายบาร์เหนือตัวอักษร A แสดงการกระทำแบบนอต จากตารางความจริงของนอตเกตจะเห็นได้ว่าเอาต์พุตมีค่าตรงข้ามกับอินพุตเสมอ เมื่ออินพุตมีค่าเป็น 1 จะให้ค่าเอาต์พุตเป็น 0 และเมื่ออินพุตมีค่าเป็น 0 จะให้ค่าเอาต์พุตเป็น 1 สัญลักษณ์วงกลม (Bubble,o) หรือสามเหลี่ยม (Triangle, ◣) แสดงการทำงานของลอจิกที่ให้ค่าตรงกันข้าม สัญลักษณ์แสดงคุณสมบัติ "1" ที่เขียนในมาตรฐานของ IEC/IEEE หมายความว่าถ้าอินพุตเป็น 1 จะให้ค่าเอาต์พุตเป็น 0

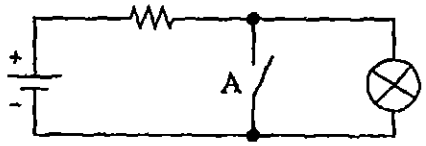
นอกจากนี้สามารถเปรียบเทียบการทำงานของนอตเกตกับการทำงานของวงจรสวิตซ์ได้ดังรูปที่ 4.8 เมื่อให้อินพุต A เป็นสวิตซ์ที่ต่อขนานกับหลอดไฟ เมื่อปิดสวิตซ์ (ลอจิก 1) จะเกิดการลัดวงจรทำให้ไม่มีกระแสไหลผ่านหลอดไฟ ทำให้หลอดไฟดับ (ลอจิก 0) แต่เมื่อเปิดสวิตซ์ (ลอจิก 0) จะมีกระแสไหลผ่านวงจร ทำให้หลอดไฟสว่าง (ลอจิก 1)



รูปที่ 4.6 สัญลักษณ์และตัวอย่างของไอซีนอตเกต

Input	Output
0	1
1	0

รูปที่ 4.7 ตารางความจริงของนอตเกต



รูปที่ 4.8 วงจรสวิตซ์

4.4.2 ออร์เกต (OR gate)

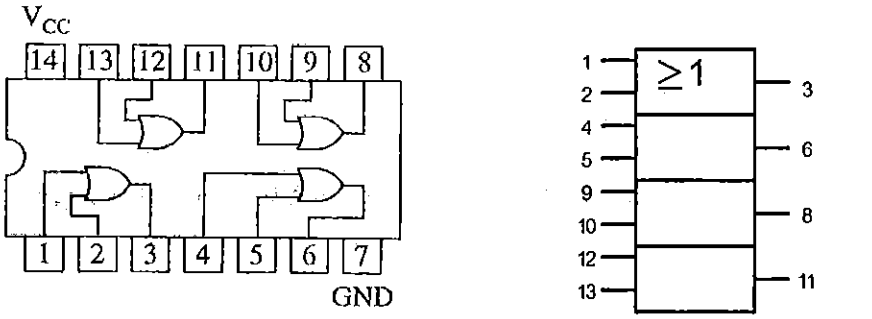
ออร์เกตประกอบด้วยอินพุตตั้งแต่ 2 อินพุตขึ้นไป ถ้ากำหนดให้อินพุตเป็นตัวแปร AB และเอาต์พุตเป็นตัวแปร Y สามารถเขียนในรูปสมการบูลีน ดังนี้

$$Y = A + B = A \text{ OR } B$$

จากสมการบูลีนอ่านว่า Y เท่ากับ A ออร์กับ B เครื่องหมาย "+" แสดงการกระทำแบบออร์ จากตารางความจริงของออร์เกตจะเห็นได้ว่า เอาต์พุตมีค่าลอจิก 0 เมื่ออินพุตมีค่าลอจิก 0 ทั้งหมด นอกนั้นเอาต์พุตมีค่าลอจิก 1 สัญลักษณ์แสดงคุณสมบัติ "≥1" ที่เขียนในมาตรฐานของ IEC/IEEE หมายความว่าอินพุตมีค่าลอจิก 1 ตั้งแต่ 1 อินพุตขึ้นไปจะให้เอาต์พุตมีค่าลอจิก 1

นอกจากนี้สามารถเปรียบเทียบการทำงานของออร์เกตกับการทำงานของวงจรสวิตซ์ได้ดังรูปที่ 4.11 เมื่อให้อินพุต A และ B เป็นสวิตซ์ที่ต่อขนานกัน เมื่อเปิดสวิตซ์ A และ B พร้อมกัน (ลอจิก 0) จะไม่มีกระแสไหลผ่านวงจร ทำให้หลอดไฟดับ (ลอจิก 0) แต่เมื่อปิดสวิตซ์ A หรือ B ตัวใดตัวหนึ่ง (ลอจิก 1) จะมีกระแสไหลผ่านวงจร ทำให้หลอดไฟสว่าง (ลอจิก 1)

(ก) ASA (ข) IEC/IEEE



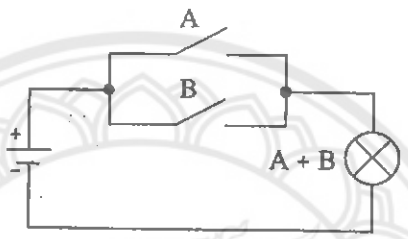
(ค) ไอซีออร์เกตแบบ ASA (ง) ไอซีออร์เกตแบบ IEC/IEEE

รูปที่ 4.9 สัญลักษณ์และตัวอย่างของไอซีออร์เกต

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Input			Output
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

รูปที่ 4.10 ตารางความจริงของออร์เกต



รูปที่ 4.11 วงจรสวิตซ์

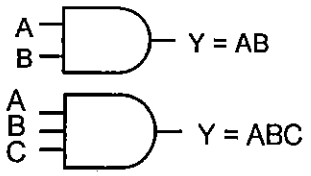
4.4.3 แอนด์เกต (AND gate)

แอนด์เกตประกอบด้วยอินพุตตั้งแต่ 2 อินพุตขึ้นไป ถ้ากำหนดให้อินพุตเป็นตัวแปร AB และเอาต์พุตเป็นตัวแปร Y สามารถเขียนในรูปสมการบูลีน ดังนี้

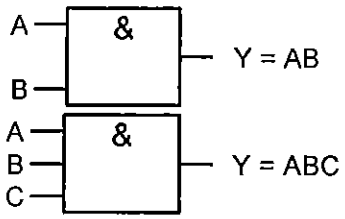
$$Y = A \cdot B = AB = A \quad AND \quad B$$

จากสมการบูลีนอ่านว่า Y เท่ากับ A แอนด์กับ B เครื่องหมาย " · " แสดงการกระทำแบบแอนด์ จากตารางความจริงของแอนด์เกตจะเห็นได้ว่า เอาต์พุตมีค่าลอจิก 1 เมื่ออินพุตมีค่าเป็นลอจิก 1 ทั้งหมด นอกนั้นเอาต์พุตมีค่าลอจิก 0 สัญลักษณ์แสดงคุณสมบัติ "&" ที่เขียนในมาตรฐานของ IEC/IEEE หมายความว่า ถ้าอินพุตมีค่าลอจิก 1 ทั้งหมด จะให้เอาต์พุตมีค่าลอจิก 1

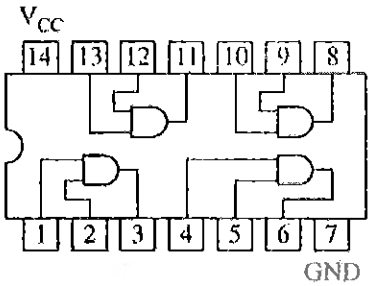
นอกจากนี้สามารถเปรียบเทียบการทำงานของแอนด์เกตกับการทำงานของวงจรสวิตซ์ได้ ดังรูปที่ 4.14 เมื่อให้อินพุต A และ B เป็นสวิตซ์ที่ต่ออนุกรมกัน เมื่อปิดสวิตซ์ทั้งหมด (ลอจิก 1) จะมีกระแสไหลผ่านวงจร ทำให้หลอดไฟสว่าง (ลอจิก 1) แต่เมื่อเปิดสวิตซ์ A หรือ B ตัวใดตัวหนึ่ง (ลอจิก 0) จะไม่มีกระแสไหลผ่านวงจร ทำให้หลอดไฟดับ (ลอจิก 0)



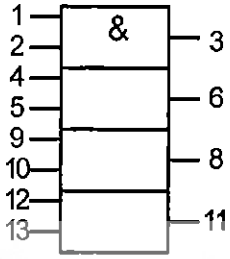
(ก) ASA



(ข) IEC/IEEE



(ค) ไอซีแอนด์เกตแบบ ASA



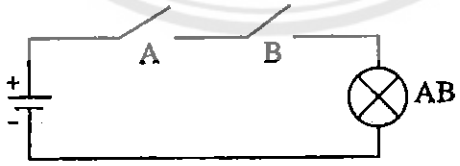
(ง) ไอซีแอนด์เกตแบบ IEC

รูปที่ 4.12 สัญลักษณ์และตัวอย่างของไอซีแอนด์เกต

Input		Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

Input			Output
A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

รูปที่ 4.13 ตารางความจริงของแอนด์เกต



รูปที่ 4.14 วงจรสวิตซ์

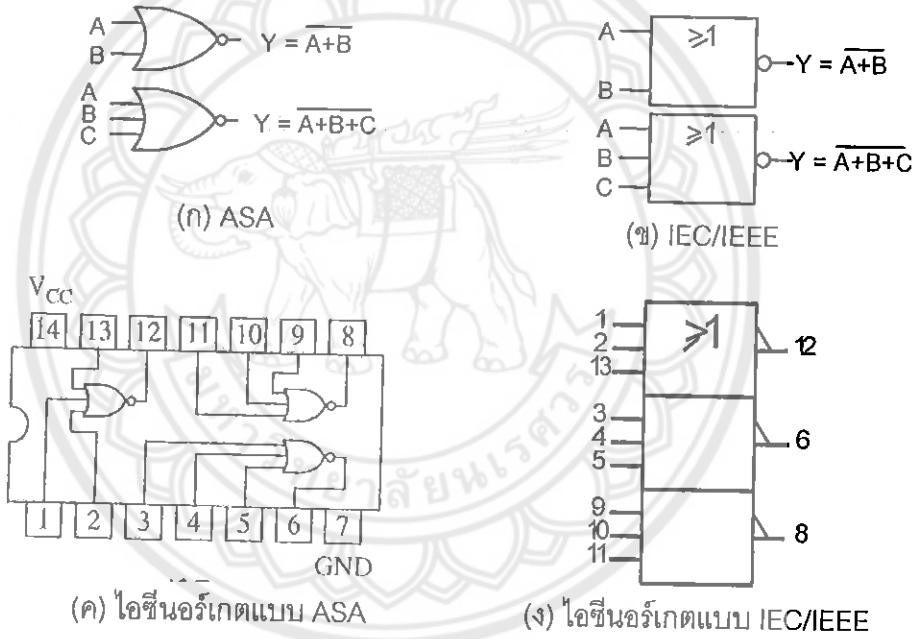
4.4.4 นอร์เกต (NOR gate)

นอร์เกตประกอบด้วยอินพุตตั้งแต่ 2 อินพุตขึ้นไป ถ้ากำหนดให้อินพุตเป็นตัวแปร AB และเอาต์พุตเป็นตัวแปร Y สามารถเขียนในรูปสมการบูลีน ดังนี้

$$Y = \overline{A + B} = \text{NOT}(A \quad \text{OR} \quad B)$$

จากสมการบูลีนอ่านว่า Y เท่ากับ A นอร์กับ B สมการบูลีนจะคล้ายกับออร์เกต แต่จะมีเครื่องหมายบาร์เหนือ A+B แสดงว่าค่าลอจิกของ A+B จะถูกทำให้เป็นค่าที่ตรงกันข้าม จากตารางความจริงของนอร์เกตจะเห็นได้ว่า เอาต์พุตมีค่าลอจิก 1 เมื่ออินพุตมีค่าลอจิก 0 ทั้งหมดซึ่งค่าเอาต์พุตของนอร์เกตมีค่าตรงข้ามกับเอาต์พุตของออร์เกตนั่นเอง วงจรของนอร์เกตที่ได้จากการนำเอานอตเกตมาต่อกับเอาต์พุตของออร์เกต จึงเรียกว่า NOT-OR หรือ NOR สัญลักษณ์แสดงคุณสมบัติ "≥1" และ "°" ที่เขียนในมาตรฐานของ IEC/IEEE หมายความว่าถ้าอินพุตมีค่าลอจิก 1 ตั้งแต่ 1 อินพุตขึ้นไป จะให้เอาต์พุตมีค่าลอจิก 0

รูปที่ 4.15 นอร์เกตที่สร้างจากเกตพื้นฐาน



รูปที่ 4.16 สัญลักษณ์และตัวอย่างของไอซีนอร์เกต

Input		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Input			Output
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

รูปที่ 4.17 ตารางความจริงของนอร์เกต

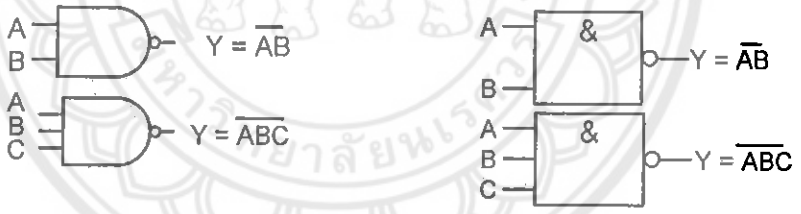
4.4.5 แนนด์เกต (NAND gate)

แนนด์เกตประกอบด้วยอินพุตตั้งแต่ 2 อินพุตขึ้นไป ถ้ากำหนดให้อินพุตเป็นตัวแปร AB และเอาต์พุตเป็นตัวแปร Y สามารถเขียนในรูปสมการบูลีน ดังนี้

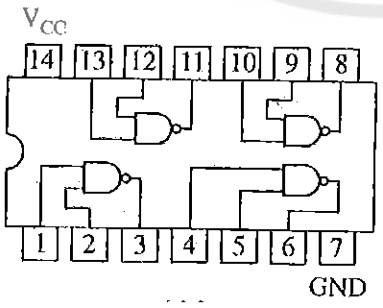
$$Y = \overline{A \cdot B} = \overline{AB} = \text{NOT}(A \quad \text{AND} \quad B)$$

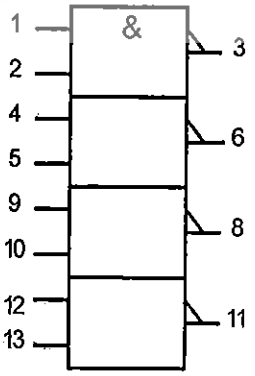
จากสมการบูลีนอ่านว่า Y เท่ากับ A แนนด์กับ B สมการบูลีนจะคล้ายกับแอนด์เกตแต่จะมีเครื่องหมายบาร์เหนือ A · B แสดงว่าค่าลอจิกของ A · B จะถูกทำให้มีค่าที่ตรงกันข้าม จากตารางความจริงของแนนด์เกตจะเห็นได้ว่า เอาต์พุตมีค่าลอจิก 0 เมื่ออินพุตมีค่าลอจิก 1 ทั้งหมด ค่าเอาต์พุตของแนนด์เกตมีค่าตรงกันข้ามกับเอาต์พุตของแอนด์เกต วงจรแนนด์เกตที่ได้จากการนำเอานอตเกตมาต่อกับเอาต์พุตของแอนด์เกต จึงเรียกว่า NOT- AND หรือ NAND สัญลักษณ์แสดงคุณสมบัติ "& " และ "^{o}" ที่เขียนในมาตรฐานของ IEC/IEEE หมายความว่า ถ้าอินพุตมีค่าลอจิก 1 ทั้งหมด จะให้เอาต์พุตมีค่าลอจิก 0

รูปที่ 4.18 แนนด์เกตที่สร้างจากเกตพื้นฐาน



(ก) ASA (ข) IEC/IEEE





(ค) ไอซีแนนด์เกตแบบ ASA (ง) ไอซีแนนด์เกตแบบ IEC/IEEE

รูปที่ 4.19 สัญลักษณ์และตัวอย่างของไอซีแนนด์เกต

Input		Output
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Input			Output
A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

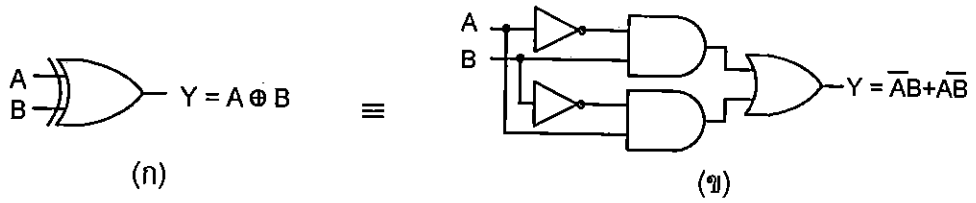
รูปที่ 4.20 ตารางความจริงของแนนด์เกต

4.4.6 เอ็กซ์คลูซีฟออร์เกต (Exclusive OR gate, XOR)

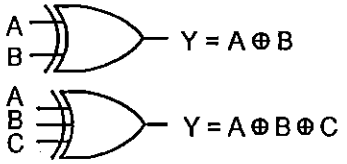
เอ็กซ์คลูซีฟออร์เกตประกอบด้วยอินพุตตั้งแต่ 2 อินพุตขึ้นไป ถ้ากำหนดให้อินพุตเป็นตัวแปร AB และเอาต์พุตเป็นตัวแปร Y สามารถเขียนในรูปสมการบูลีน ดังนี้

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

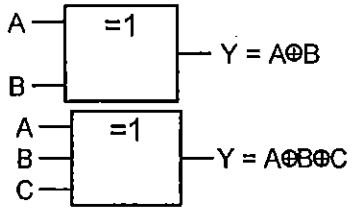
จากสมการบูลีนอ่านว่า Y เท่ากับ A เอ็กซ์คลูซีฟออร์กับ B เครื่องหมาย "$\oplus$" เป็นการกระทำแบบเอ็กซ์คลูซีฟออร์ จากตารางความจริงของเอ็กซ์คลูซีฟออร์เกตจะเห็นได้ว่า เอาต์พุตจะมีค่าลอจิก 1 เมื่ออินพุตมีค่าลอจิก 1 เป็นจำนวนคี่ และเอาต์พุตมีค่าลอจิก 0 เมื่ออินพุตมีค่าลอจิก 1 เป็นจำนวนคู่ วงจรเอ็กซ์คลูซีฟออร์เกตที่ได้จากการนำเอาเกตพื้นฐานมาต่อกันดังรูปที่ 4.21 (ก) แต่เนื่องจากไม่สะดวกนำมาใช้ในการออกแบบและวิเคราะห์วงจร จึงใช้สัญลักษณ์ตามมาตรฐานดังรูปที่ 4.21(ก) สัญลักษณ์ของเอ็กซ์คลูซีฟออร์เกตมีลักษณะคล้ายกับออร์เกต แต่มีเส้นโค้งเพิ่มเข้าไปทางด้านอินพุต สัญลักษณ์แสดงคุณสมบัติ "=1" ที่เขียนในมาตรฐานของ IEC/IEEE หมายความว่าถ้าอินพุตมีค่าลอจิก 1 เพียง 1 อินพุตเท่านั้น จะให้เอาต์พุตมีค่าลอจิก 1



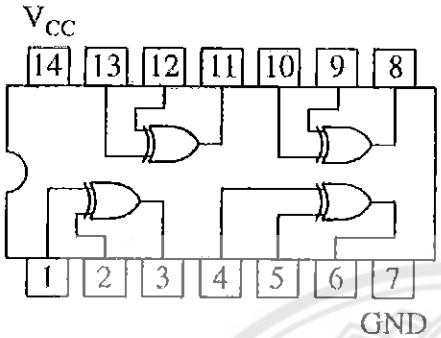
รูปที่ 4.21 เอ็กซ์คลูซีฟออร์เกตที่สร้างจากเกตพื้นฐาน



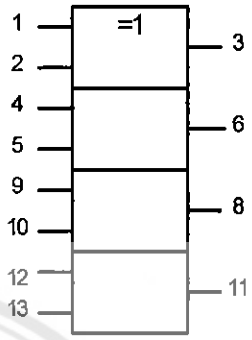
(ก) ASA



(ข) IEC/IEEE



(ค) ไอซีเอ็กซ์คลูซีฟออร์เกตแบบ ASA



(ง) ไอซีเอ็กซ์คลูซีฟออร์เกตแบบ IEC/IEEE

รูปที่ 4.22 สัญลักษณ์และตัวอย่างของไอซีเอ็กซ์คลูซีฟออร์เกต

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Input			Output
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

รูปที่ 4.23 ตารางความจริงของเอ็กซ์คลูซีฟออร์เกต

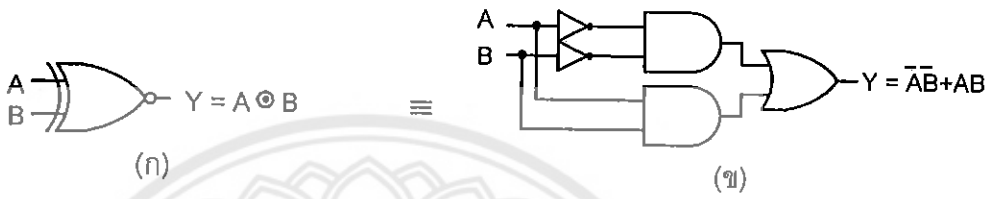
4.4.7 เอ็กซ์คลูซีฟนอร์เกต (Exclusive NOR gate, XNOR)

เอ็กซ์คลูซีฟนอร์เกตประกอบด้วยอินพุตตั้งแต่ 2 อินพุตขึ้นไป ถ้ากำหนดให้อินพุตเป็นตัวแปร AB และเอาต์พุตเป็นตัวแปร Y สามารถเขียนในรูปสมการบูลีน ดังนี้

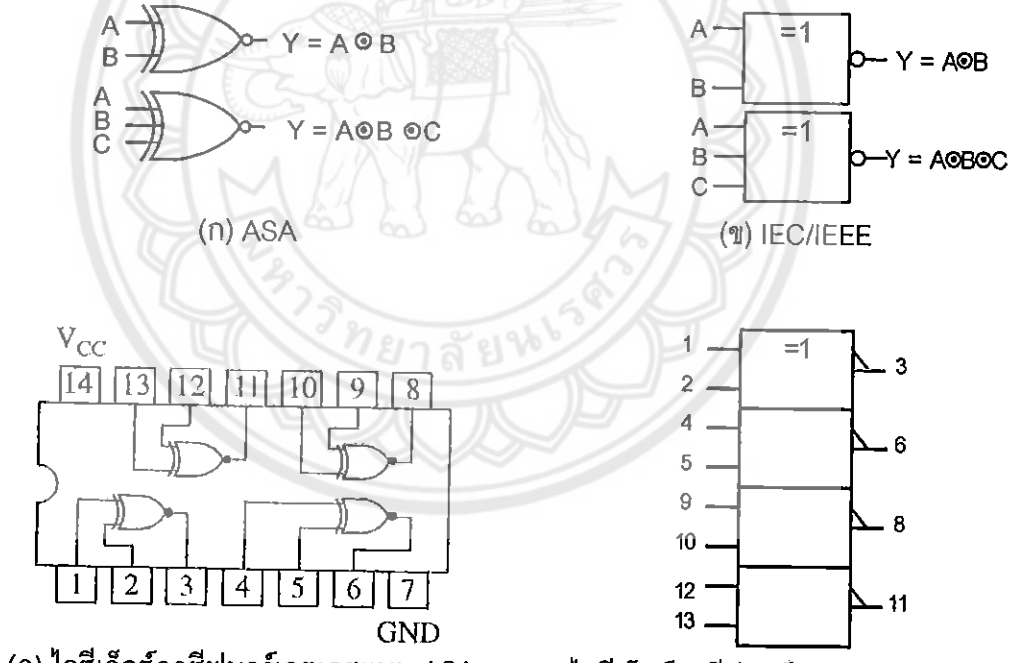
$$Y = A \odot B = \overline{A \oplus B} = \overline{\overline{A}B + A\overline{B}} = \overline{AB} + AB$$

จากสมการบูลีนอ่านว่า Y เท่ากับ A เอ็กซ์คลูซีฟนอร์กับ B เครื่องหมาย "⊙" แสดงการกระทำแบบเอ็กซ์คลูซีฟนอร์ สมการบูลีนสามารถเขียนคล้ายกับเอ็กซ์คลูซีฟออร์เกต แต่จะมีเครื่องหมายบาร์เหนือ A ⊕ B แสดงว่าค่าลอจิกของ A ⊕ B จะถูกทำให้เป็นค่าที่ตรงกันข้าม จากตารางความจริงของเอ็กซ์คลูซีฟนอร์เกตจะเห็นได้ว่า เมื่ออินพุตมีค่าลอจิก 1 เป็นจำนวนคี่ เอาต์พุต

จะมีค่าลอจิก 0 และเมื่ออินพุตมีค่าลอจิก 1 เป็นจำนวนคู่ เอาต์พุตจะมีค่าลอจิก 1 วงจรของเอ็กซ์คลูซีฟนอร์เกตที่ได้จากการนำเอาเกตพื้นฐานมาต่อกันดังรูปที่ 4.24 (ข) แต่เนื่องจากไม่สะดวกนำมาใช้ในการออกแบบและวิเคราะห์วงจร จึงใช้สัญลักษณ์ตามมาตรฐานดังรูปที่ 4.24(ก) วงจรเอ็กซ์คลูซีฟนอร์เกตมีลักษณะเหมือนกับเอ็กซ์คลูซีฟออร์เกตแต่เพิ่มวงกลมเข้าไปทางด้านเอาต์พุต สัญลักษณ์แสดงคุณสมบัติ "=1" และ"$^{\circ}$" ที่เขียนในมาตรฐานของ IEC/IEEE หมายความว่าถ้าอินพุตมีค่าลอจิก 1 หรือ 0 ทั้งคู่ จะให้เอาต์พุตมีค่าลอจิก 1



รูปที่ 4.24 เอ็กซ์คลูซีฟนอร์เกตเกตที่สร้างจากเกตพื้นฐาน



(ค) ไอซีเอ็กซ์คลูซีฟนอร์เกตเกตแบบ ASA (ง) ไอซีเอ็กซ์คลูซีฟนอร์เกตเกตแบบ IEC/IEEE

รูปที่ 4.25 สัญลักษณ์และตัวอย่างของไอซีเอ็กซ์คลูซีฟนอร์เกตเกต

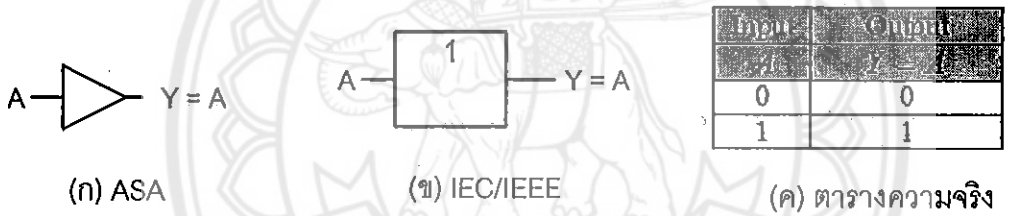
Input		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Input			Output
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

รูปที่ 4.26 ตารางความจริงของเอ็กซ์คลูซีฟนอร์เกต

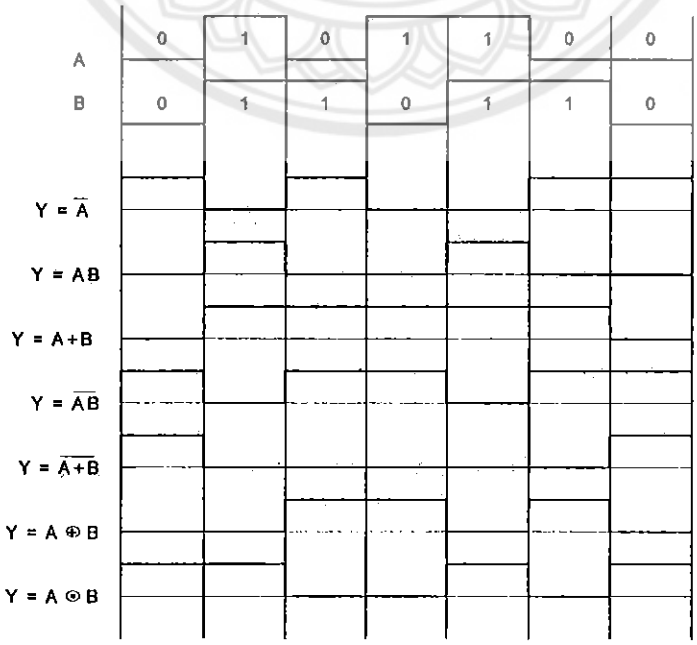
4.4.8 บัฟเฟอร์ (Buffer)

สัญลักษณ์ของบัฟเฟอร์จะคล้ายกับนอตเกต แต่จะไม่มีวงกลมที่ด้านเอาต์พุต จากตารางความจริงจะเห็นได้ว่าเอาต์พุตจะมีค่าเหมือนอินพุตทุกกรณี แต่ระดับของค่าแรงดันจะแตกต่างกัน นำไปใช้ในการขับค่าแรงดันไฟฟ้าในวงจร



Input	Output
A	Y = A
0	0
1	1

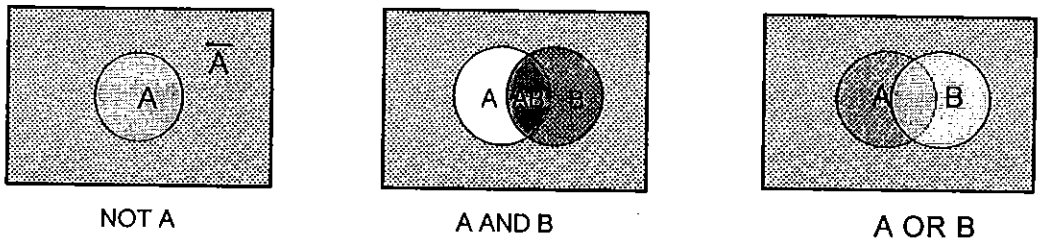
รูปที่ 4.27 สัญลักษณ์และตารางความจริงของบัฟเฟอร์



รูปที่ 4.28 ตัวอย่างไดอะแกรมเวลาของเกตแต่ละชนิด

4.5 แผนภาพเวนน์ (Venn Diagram)

เกตพื้นฐานสามารถอธิบายโดยใช้พื้นที่ได้เรียกว่าแผนภาพเวนน์ สามารถนำแผนภาพเวนน์ไปใช้กับแผนผังคาร์โนห์ (Kanuagh map) เพื่อหาสมการบูลีนอย่างง่ายได้



รูปที่ 4.29 แผนภาพเวนน์

ลักษณะของแผนภาพเวนน์แสดงดังรูปที่ 4.29 อธิบายได้ดังนี้

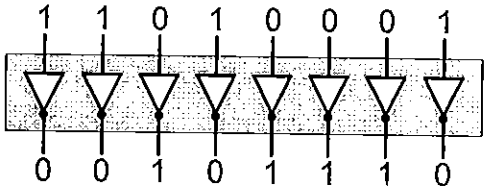
(1) นอตเกต มีความหมายว่า "ไม่ใช่" ดังนั้นจากรูป ถ้าพื้นที่ของวงกลมคือ A แสดงว่าพื้นที่นอกวงกลมไม่ใช่ A สามารถเขียนแทนด้วย $\overline{A}$

(2) แอนด์เกต มีความหมายว่า "และ" ดังนั้นจากรูป $A \cdot B$ คือพื้นที่ของวงกลม A และ B ที่ซ้อนทับกัน

(3) ออร์เกต มีความหมายว่า "หรือ" ดังนั้นจากรูป $A + B$ คือพื้นที่ของวงกลม A และ B ทั้งหมดทั้งที่ซ้อนทับกันและไม่ซ้อนกัน

4.6 ตัวอย่างการประยุกต์ใช้งานของลอจิกเกต

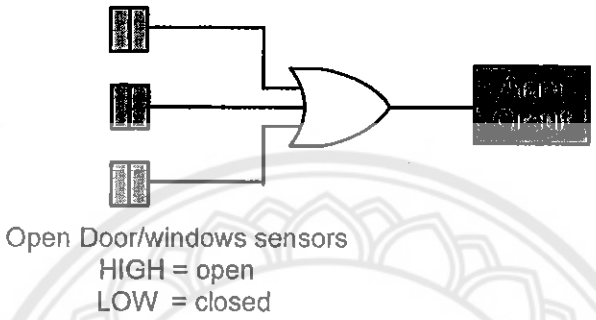
จากรูปที่ 4.30 เป็นการประยุกต์ใช้อินเวอร์เตอร์สร้างเป็นวงจรการหาค่า 1's complement ของเลขฐานสองขนาด 8 บิต สมมุติป้อนข้อมูลอินพุต $1101\ 0001_2$ ให้กับอินเวอร์เตอร์ จะได้เอาต์พุตเป็น $0010\ 1110_2$ ซึ่งคือค่า 1's complement $1101\ 0001_2$ นั่นเอง



รูปที่ 4.30 การใช้อินเวอร์เตอร์สร้างวงจร1's complement

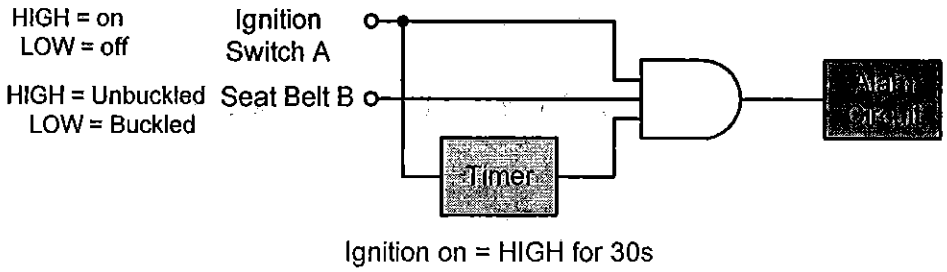
จากรูปที่ 4.31 เป็นการประยุกต์ใช้ออร์เกตมาสร้างเป็นวงจรตรวจจับผู้บุกรุกอย่างง่าย (Intrusion detection system) วงจรนี้สามารถนำไปใช้กับห้องต่างๆ ภายในบ้าน สมมุติว่าห้องที่ต้องการติดตั้งระบบตรวจจับผู้บุกรุกประกอบด้วยประตูจำนวน 1 ประตูและหน้าต่างจำนวน 2 บาน

ที่ประตูและหน้าต่างมีตัวเซนเซอร์ (Sensors) ทำหน้าที่ตรวจจับสัญญาณเมื่อมีการเปิดปิดประตูหน้าต่างเกิดขึ้น ถ้าประตูหรือหน้าต่างถูกเปิด ตัวเซนเซอร์จะส่งสัญญาณเอาต์พุตเป็น HIGH และถ้าประตูหรือหน้าต่างปิดอยู่ ตัวเซนเซอร์จะส่งสัญญาณเอาต์พุตเป็น LOW ตัวเซนเซอร์จะส่งสัญญาณเอาต์พุตที่ได้ เพื่อป้อนไปที่อินพุตของออร์เกต ถ้าประตูหรือหน้าต่างถูกเปิด ตัวเซนเซอร์จะส่งสัญญาณเอาต์พุตเป็น HIGH สัญญาณนี้จะป้อนที่อินพุตของออร์เกต ทำให้เอาต์พุตของออร์เกตเป็น HIGH สัญญาณจะถูกส่งไปให้วงจรส่งสัญญาณเตือนว่ามีการบุกรุกเกิดขึ้น



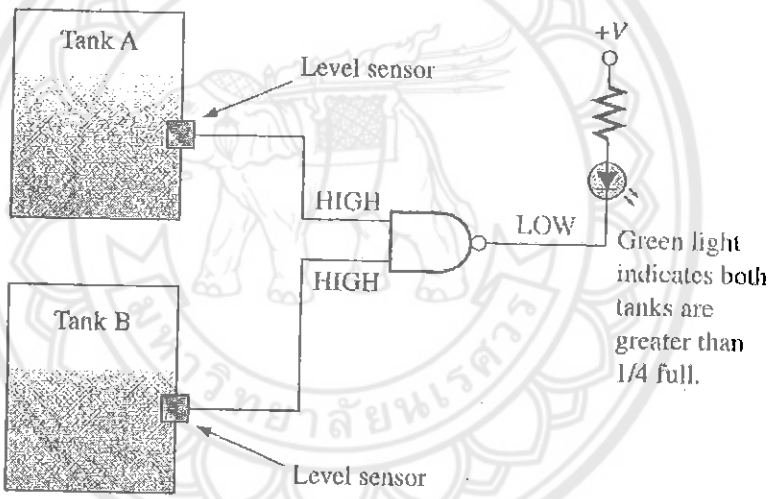
รูปที่ 4.31 การใช้ออร์เกตสร้างวงจรระบบตรวจจับผู้บุกรุกอย่างง่าย

จากรูปที่ 4.32 เป็นการประยุกต์ใช้แอนด์เกตมาสร้างเป็นวงจรเตือนการคาดเข็มขัดนิรภัยในรถอย่างง่าย (Seat belt alarm circuit) เมื่อทำการติดเครื่องยนต์ รถจะให้อินพุต A เป็น HIGH ส่วนคนขับหรือคนที่นั่งภายในรถ ถ้ายังไม่ได้คาดเข็มขัดนิรภัยจะให้อินพุต B เป็น HIGH ระหว่างที่เครื่องยนต์ติดนั้น จะมีตัวจับเวลา (Timer) ทำหน้าที่นับเวลาว่าหลังจากติดเครื่องยนต์ไปแล้วเป็นเวลา 30 วินาที จะให้เอาต์พุตเป็น HIGH ดังนั้นจากเงื่อนไขทั้ง 3 กรณีคือเมื่อทำการติดเครื่องยนต์ คนขับไม่ได้คาดเข็มขัดนิรภัยและตัวจับเวลาทำงานเป็นเวลา 30 วินาที จะให้อินพุตของแอนด์เกตเป็น HIGH ทั้งหมด และเอาต์พุตของแอนด์เกตเป็น HIGH ด้วย สัญญาณเอาต์พุตนี้จะทำให้วงจรส่งสัญญาณเตือนการคาดเข็มขัดนิรภัยดังขึ้น



รูปที่ 4.32 การใช้แอนด์เกตสร้างวงจรเตือนการคาดเข็มขัดนิรภัยในรถอย่างง่าย

จากรูปที่ 4.33 เป็นการประยุกต์ใช้แนนด์เกตมาสร้างเป็นวงจรตรวจวัดระดับของน้ำหรือสารเคมีที่บรรจุในถังที่ใช้สำหรับในโรงงานอุตสาหกรรม ถังน้ำจะมีตัวเซนเซอร์เป็นตัวตรวจวัดระดับน้ำ โดยติดตั้งให้อยู่ในตำแหน่งของระดับน้ำที่ต้องการตรวจสอบ ถ้าระดับน้ำในถังอยู่สูงกว่าระดับตัวเซนเซอร์ ตัวเซนเซอร์จะให้สัญญาณค่าแรงดันไฟฟ้า 5 โวลต์ (HIGH) ถ้าระดับน้ำอยู่ที่ระดับเดียวกับตัวเซนเซอร์ ตัวเซนเซอร์จะให้สัญญาณค่าแรงดันไฟฟ้า 0 โวลต์ (LOW) ดังนั้นการทำงานของวงจร ถ้าระดับน้ำในถัง A และ B มีค่าสูงกว่าระดับของเซนเซอร์ ตัวเซนเซอร์ของทั้ง 2 ถังจะให้สัญญาณเอาต์พุตเป็น HIGH และส่งไปยังอินพุตของแนนด์เกต เอาต์พุตของแนนด์เกตที่ได้จะมีค่าเป็น LOW มีผลทำให้ LED สว่าง ถ้าระดับน้ำในถัง A หรือ B หรือทั้งสองถัง มีระดับน้ำในถังอยู่ที่ระดับเดียวกับเซนเซอร์ ตัวเซนเซอร์ของทั้ง 2 ถังจะให้สัญญาณเอาต์พุตอย่างน้อย 1 เอาต์พุตเป็น LOW และส่งไปยังอินพุตของแนนด์เกต เอาต์พุตของแนนด์เกตที่ได้จะมีค่าเป็น HIGH มีผลทำให้ LED ดับ

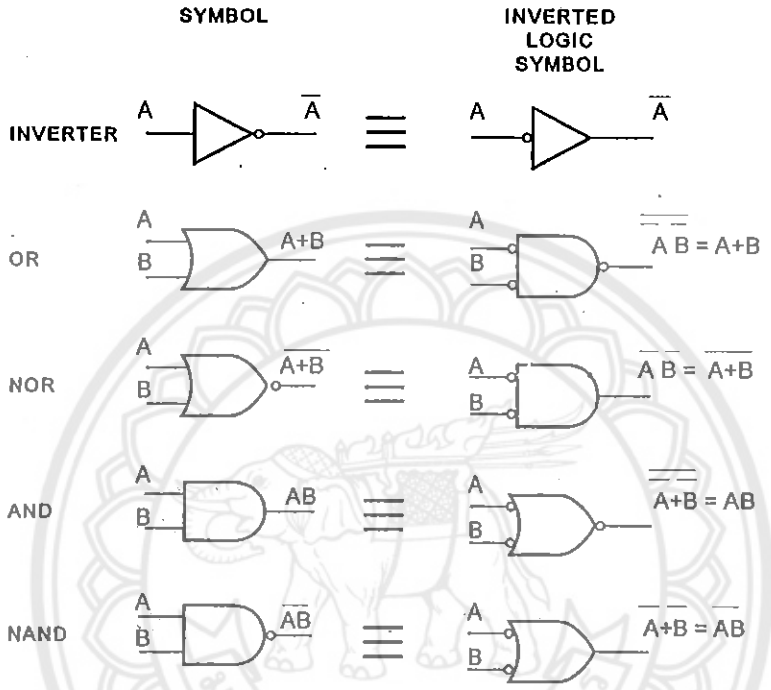


รูปที่ 4.33 การใช้แนนด์เกตสร้างวงจรตรวจวัดระดับของน้ำในถัง

4.7 เกตสมมูลย์ (Equivalent circuit)

สัญลักษณ์ของเกต ได้แก่ อินเวอร์เตอร์ แอนด์เกต ออร์เกต แนนด์เกตและนอร์เกต ที่ใช้ในการออกแบบวงจรดิจิตอล จะใช้สัญลักษณ์มาตรฐานดังที่กล่าวในหัวข้อที่ 4.4 แต่บางครั้งในการออกแบบวงจรเพื่อใช้งานจริงนั้น จะมีสัญลักษณ์ของเกตในรูปแบบอื่นเพิ่มขึ้นมาด้วยเรียกว่าเกตสมมูลย์หรืออินเวิร์ตของเกต (Inverted logic symbol) ดังรูปที่ 4.34 เพื่อใช้อธิบายการทำงานของวงจรดิจิตอลได้ง่ายขึ้น โดยพิจารณาจากวงจรที่ออกแบบได้โดยตรง การแปลงจากเกตมาตรฐานเป็นเกตสมมูลย์มีขั้นตอนดังนี้

1. ทำการแปลงอินพุตและเอาต์พุตของสัญลักษณ์เกตแบบ ASA หรือ IEC ดังนี้ ถ้าอินพุตหรือเอาต์พุตของเกตไม่มีวงกลมให้ใส่วงกลมเพิ่มเข้าไป แต่ถ้าอินพุตหรือเอาต์พุตของเกตนั้นมีวงกลมอยู่แล้ว ให้ตัดวงกลมทิ้งไป
2. ทำการเปลี่ยนสัญลักษณ์จากแอนด์เกตเป็นออร์เกต หรือจากออร์เกตเป็นแอนด์เกต ยกเว้นอินเวอร์เตอร์เท่านั้นที่ยังคงเหมือนเดิม



รูปที่ 4.34 สัญลักษณ์อินเวิร์ตของลอจิกเกตแบบ ASA

ตัวอย่างเช่น สัญลักษณ์มาตรฐานของแนนด์เกตสามารถเขียนในรูปอินเวิร์ตของลอจิกเกตได้ดังนี้ ใส่วงกลมที่ด้านอินพุตและวงกลมที่เอาต์พุตของแนนด์เกตให้ตัดทิ้งไป จากนั้นทำการเปลี่ยนจากแอนด์เกตเป็นออร์เกต เป็นต้น สมการบูลีนอินเวิร์ตของลอจิกเกตจะต่างจากเกตที่ใช้สัญลักษณ์มาตรฐาน การเปลี่ยนรูปสามารถใช้ทฤษฎีของดีมอร์แกน (DeMorgan theorem) ขั้นตอนมีดังนี้

1. ใส่ดับเบิ้ลบาร์หรือดับเบิ้ลคอมพลีเมนต์ของสมการบูลีน
2. เปลี่ยนรูปโดยใช้ทฤษฎีดีมอร์แกน เมื่อทฤษฎีดีมอร์แกนกล่าวว่า

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

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

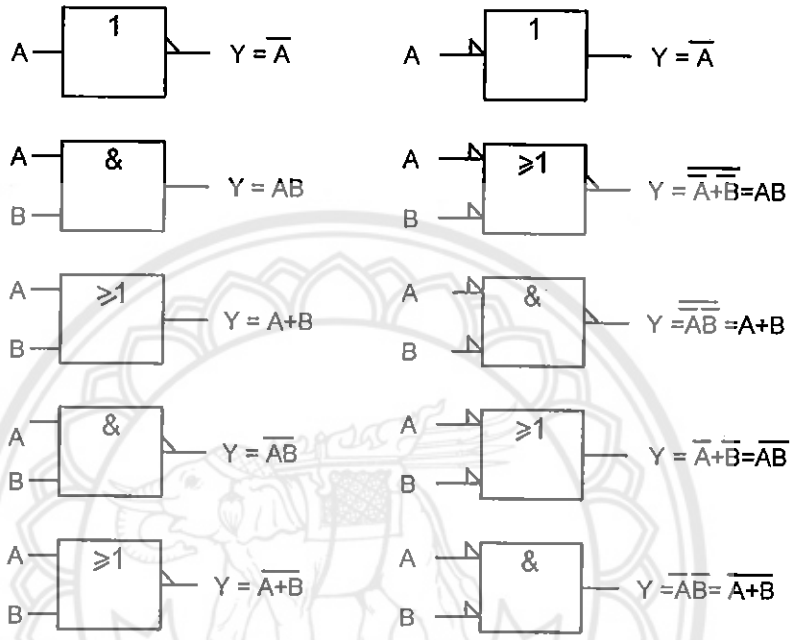
ตัวอย่างเช่น การเปลี่ยนรูปสมการบูลีนจากสัญลักษณ์มาตรฐานเป็นสัญลักษณ์เกตสมมูลย์ของ แอนด์เกต ออร์เกต แนนด์เกต นอร์เกต มีขั้นตอนดังนี้

(1) $Y = AB = \overline{\overline{AB}} = \overline{\overline{A} + \overline{B}}$

(2) $Y = A + B = \overline{\overline{A + B}} = \overline{\overline{A} \cdot \overline{B}}$

(3) $Y = \overline{AB} = \overline{A} + \overline{B}$

(4) $Y = \overline{A + B} = \overline{A} \cdot \overline{B}$



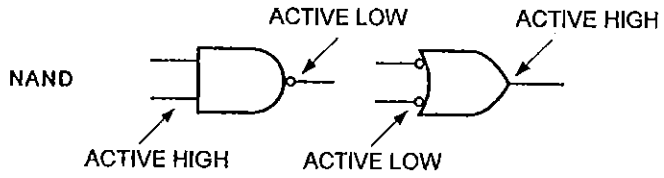
รูปที่ 4.35 สัญลักษณ์อินเวิร์ตของลอจิกเกตแบบ IEC/IEEE

การแปลความหมายจากสัญลักษณ์ของเกต (Logic symbol interpretation)

สัญลักษณ์ของเกตในรูปที่ 4.34 และ 4.35 สามารถอธิบายการทำงานของเกต โดยพิจารณาการตอบสนองต่อค่าระดับลอจิก (Active logic level) ดังนี้

1. สัญลักษณ์ของเกตที่ด้านอินพุตหรือเอาต์พุตที่ไม่มีวงกลมหรือไม่มีสามเหลี่ยมหมายถึงการตอบสนองต่อสัญญาณลอจิกที่เป็น 1 หรือ ACTIVE HIGH
2. สัญลักษณ์ของเกตที่ด้านอินพุตหรือเอาต์พุตที่มีวงกลมหรือมีสามเหลี่ยมหมายถึงการตอบสนองต่อสัญญาณลอจิกที่เป็น 0 หรือ ACTIVE LOW

จากรูปที่ 4.36 แสดงสัญลักษณ์ของแนนด์เกตจะมีวงกลมที่ด้านเอาต์พุตแต่ด้านอินพุตจะไม่มีวงกลม แสดงว่าอินพุตเป็น ACTIVE HIGH และเอาต์พุตเป็น ACTIVE LOW จากสัญลักษณ์ของแนนด์เกตอาจกล่าวได้ว่า "เมื่ออินพุตทั้งหมดเป็น HIGH เอาต์พุตเป็น LOW เท่านั้น" ส่วนสัญลักษณ์อินเวิร์ตของแนนด์เกต จะมีอินพุตเป็น ACTIVE LOW และเอาต์พุตเป็น ACTIVE HIGH หรืออาจกล่าวได้ว่า "เมื่ออินพุตใดๆ เป็น LOW เอาต์พุตจะเป็น HIGH"

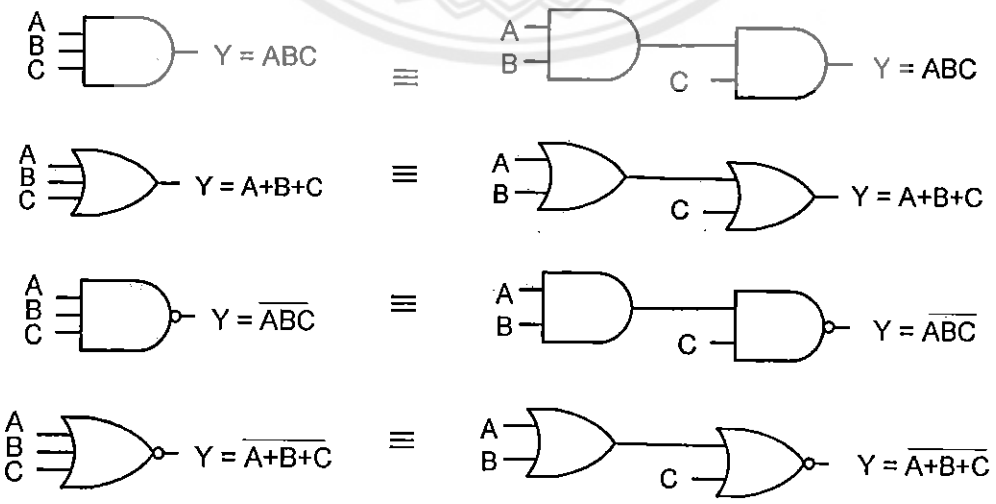


รูปที่ 4.36 ลักษณะ ACTIVE HIGH และ ACTIVE LOW ของแนนด์เกต

โดยทั่วไปในทางปฏิบัติ การเขียนสัญลักษณ์เพื่อแสดงการตอบสนองต่อสัญญาณว่าเป็น ACTIVE LOW หรือ ACTIVE HIGH ของวงจร สัญลักษณ์ที่มีวงกลมจะใส่เครื่องหมายบาร์ที่ตัวแปรเพื่อแสดงการตอบสนองต่อสัญญาณแบบ ACTIVE LOW ตัวอย่างเช่น จากรูปที่ 4.34 สัญลักษณ์อินเวิร์ตของนอร์เกตจะเห็นได้ว่าอินพุต A, B เป็น ACTIVE LOW ดังนั้นสามารถเขียนเป็น $\overline{A}, \overline{B}$ เป็นต้น

4.8 การต่อขยายเกต (Expanding gates)

ในทางปฏิบัติการนำเกตมาใช้ต่อวงจร อาจไม่มีเกตตามที่ต้องการโดยเฉพาะเกตที่มีตั้งแต่ 3 อินพุตขึ้นไป ดังนั้นสามารถประยุกต์โดยนำเกตแบบ 2 อินพุตมาสร้างเป็นเกตแบบ 3 อินพุตได้ จากรูปที่ 4.37 จะเห็นว่าแอนด์เกตแบบอินพุต 3 ตัวแปร สามารถนำแอนด์เกตแบบอินพุต 2 ตัวแปรจำนวน 2 ตัวมาต่อกันได้ ส่วนออร์เกตแบบอินพุต 3 ตัวแปร สามารถนำออร์เกตแบบอินพุต 2 ตัวแปรจำนวน 2 ตัวมาต่อกันได้ แนนด์เกตอินพุต 3 ตัวแปรสามารถนำแอนด์เกตอินพุต 2 ตัวแปรและแนนด์เกตอินพุต 2 ตัวแปรมาต่อกัน นอร์เกตอินพุต 3 ตัวแปรสามารถนำออร์เกตอินพุต 2 ตัวแปรและนอร์เกตอินพุต 2 ตัวแปรมาต่อกัน ลักษณะการต่อดังกล่าวจะให้เอาต์พุตสุดท้ายเหมือนกัน



รูปที่ 4.37 ตัวอย่างเกตอินพุต 3 ตัวแปรที่สร้างจากเกตอินพุต 2 ตัวแปร

4.9 เกตเอนกประสงค์ (Universal gates)

โดยทั่วไปสมการบูลีนประกอบด้วยตัวแปรที่มีการดำเนินการแบบนอต แอนด์และออร์ ดังนั้นในการออกแบบจะนำแอนด์เกต ออร์เกตและนอตเกตมาสร้างเป็นวงจร อย่างไรก็ตามการออกแบบวงจรสามารถใช้แนนด์เกตหรือนอร์เกตเพียงชนิดเดียวมาสร้างเป็นวงจรได้ เนื่องจากแนนด์และนอร์เกตสามารถนำไปสร้างวงจรที่มีการทำงานเหมือนเกตชนิดอื่นได้ ทำให้ช่วยประหยัดเกตหรือไอซีที่นำมาใช้ในการออกแบบวงจร การแปลงรูปสมการบูลีนจากเกตใดๆ ไปเป็นนอร์เกตหรือแนนด์เกต สามารถใช้ทฤษฎีของดีมอร์แกนช่วยในการแปลงรูปสมการ ขั้นตอนการแปลงมีดังนี้

1. ใส่ดับเบิ้ลบาร์หรือดับเบิ้ลคอมพลีเมนต์ของสมการบูลีน
2. เปลี่ยนรูปโดยใช้ทฤษฎีดีมอร์แกน เมื่อทฤษฎีดีมอร์แกนกล่าวว่า

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

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

4.9.1 การแปลงเกตใด ๆ เป็นนอร์เกต

(1) $Y = \overline{A} = \overline{A+A}$

นั่นคือการนำอินพุตทั้งสองของนอร์เกตมาต่อกัน จะให้เอาต์พุตของนอร์เกตเหมือนกับนอตเกตทุกประการ

(2) $Y = A+B = \overline{\overline{A+B}}$

นั่นคือการนำอินพุตทั้งสองมานอร์กันก่อน แล้วทำการกลับลอจิกอีกครั้ง โดยใช้นอตเกตที่สร้างจากนอร์เกต

(3) $Y = A\cdot B = \overline{\overline{A\cdot B}} = \overline{\overline{A}+\overline{B}}$

นั่นคือทำการกลับลอจิกของอินพุตทั้งสองก่อน แล้วจึงนำมานอร์กัน

(4) $Y = A \oplus B$

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

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

$$= \overline{\overline{(\overline{A}B)(A\overline{B})}}$$

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

นั่นคือนำอินพุตมากลับลอจิก แล้วนำอินพุตในวงเล็บมานอร์กันก่อน จากนั้นนำแต่ละวงเล็บมานอร์กัน

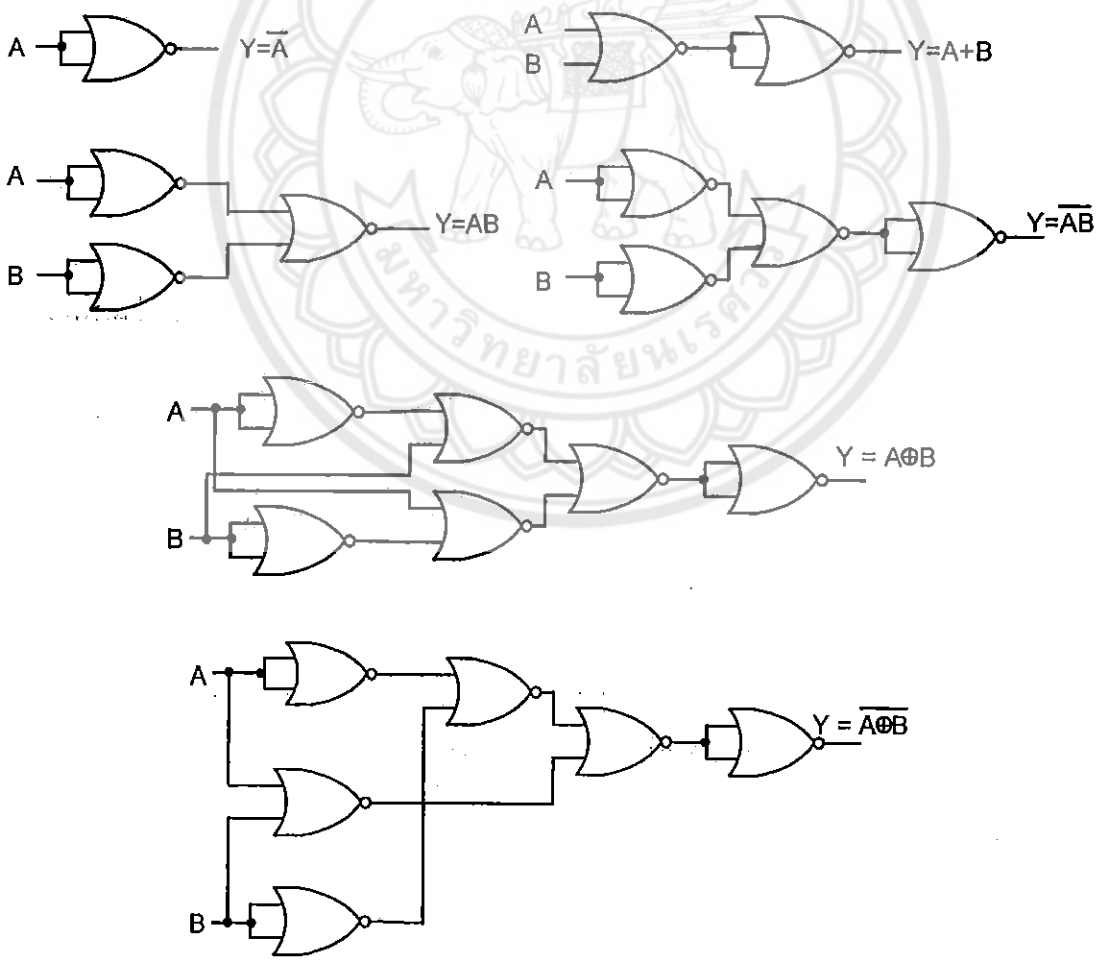
(5) $Y = \overline{A \oplus B}$

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

นั่นคือนำอินพุตมากลับลอจิก แล้วนำอินพุตในวงเล็บมานอร์กันก่อน จากนั้นนำแต่ละวงเล็บมานอร์กัน แล้วทำการกลับลอจิกอีกครั้ง

(6) $Y = \overline{AB} = \overline{\overline{AB}} = \overline{\overline{\overline{A} + \overline{B}}}$

นั่นคือกลับลอจิกของอินพุตทั้งสองก่อน จากนั้นนำมานอร์กันและกลับค่าลอจิกอีกครั้ง



รูปที่ 4.38 การนำนอร์เกตมาสร้างเป็นเกตชนิดอื่น

4.9.2 การแปลงเกตใด ๆ เป็นแนนด์เกต

(1) $Y = \overline{A} = \overline{A \cdot A}$

นั่นคือนำอินพุตทั้งสองของแนนด์เกตมาต่อกัน จะให้เอาต์พุตของแนนด์เกตเหมือนกับนอตเกตทุกประการ

(2) $Y = AB = \overline{\overline{AB}}$

นั่นคือนำอินพุตทั้งสอง มาแนนด์กันก่อน แล้วกลับค่าลอจิกอีกครั้ง โดยใช้นอตเกตที่สร้างจากแนนด์เกต

(3) $Y = A + B = \overline{\overline{A + B}} = \overline{\overline{A} \cdot \overline{B}}$

นั่นคือกลับค่าลอจิกของอินพุตทั้งสอง แล้วนำมาแนนด์กัน

(4) $Y = A \oplus B = \overline{A}B + A\overline{B}$

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

$= \overline{(\overline{\overline{A}B})(\overline{A\overline{B}})}$

นั่นคือกลับค่าลอจิกของอินพุตทั้งสอง แล้วนำอินพุตในวงเล็บมาแนนด์กันก่อน จากนั้นนำเอาต์พุตของแต่ละวงเล็บมาแนนด์กันอีกครั้ง

(5) $Y = \overline{A \oplus B} = \overline{A}\overline{B} + AB$

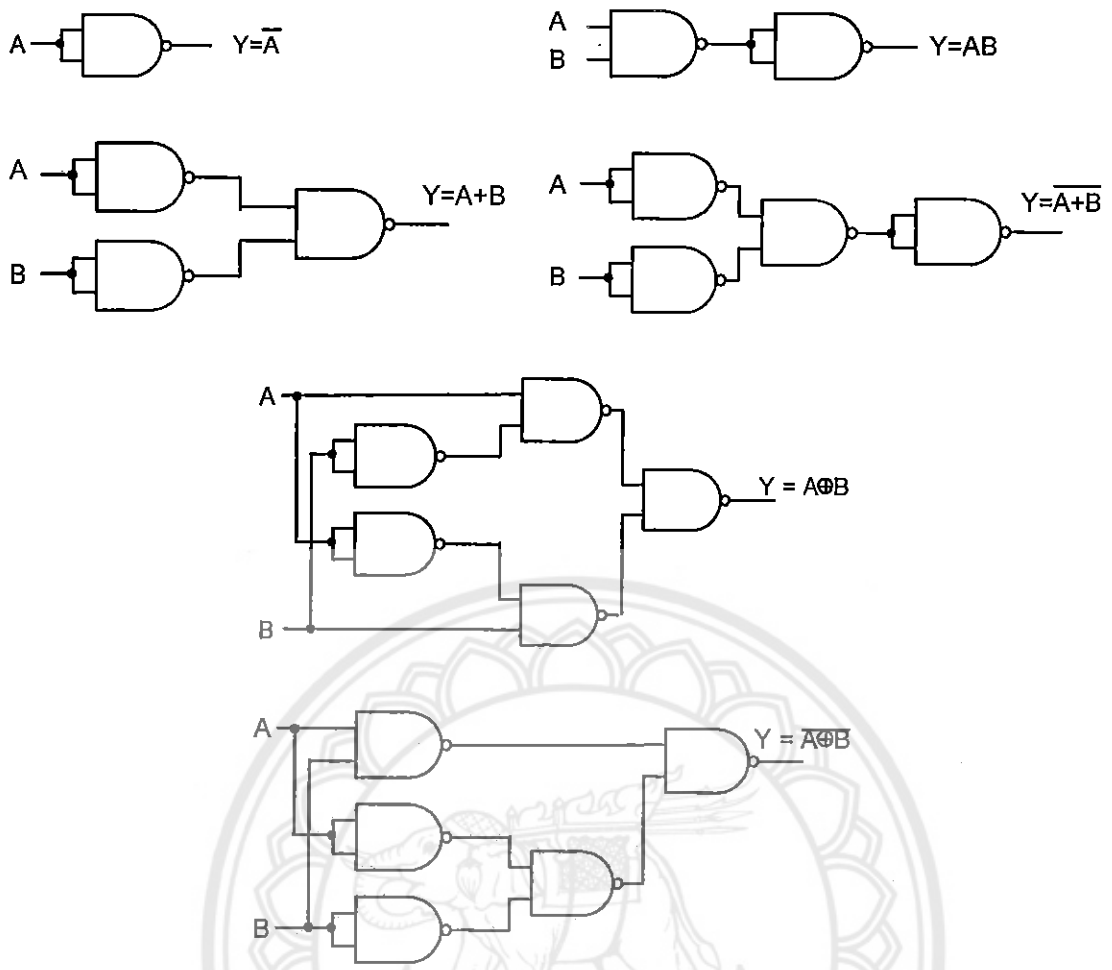
$= \overline{\overline{\overline{A}\overline{B} + AB}}$

$= \overline{(\overline{\overline{A}\overline{B}})(\overline{AB})}$

นั่นคือกลับค่าลอจิกของอินพุตทั้งสอง แล้วนำมาแนนด์กันจะได้วงเล็บแรก ส่วนวงเล็บที่สองนำอินพุตทั้งสองมาแนนด์กัน จากนั้นนำเอาต์พุตของทั้งสองวงเล็บมาแนนด์กัน

(6) $Y = \overline{A + B} = \overline{\overline{A + B}} = \overline{\overline{A} \cdot \overline{B}}$

นั่นคือกลับค่าลอจิกของอินพุตทั้งสองแล้วนำมาแนนด์กัน จากนั้นกลับค่าลอจิกอีกครั้งโดยใช้นอตเกตที่สร้างจากแนนด์เกต



รูปที่ 4.39 การนำแนนด์เกตมาสร้างเป็นเกตชนิดอื่น

4.10 เทคโนโลยีของวงจรรวมหรือไอซี (Integrated circuits, IC)

ลอจิกเกตได้ผลิตออกมาในรูปของวงจรรวมหรือไอซี วงจรภายในของลอจิกเกตประกอบด้วยทรานซิสเตอร์ที่ทำหน้าที่เป็นเสมือนสวิตซ์ ทรานซิสเตอร์ที่ใช้ในการออกแบบวงจรลอจิกเกตมี 2 ตระกูลใหญ่ๆ คือ TTL และ CMOS ในแต่ละตระกูลจะแบ่งออกเป็นหลายตระกูลย่อยขึ้นอยู่กับคุณสมบัติของไอซี ไอซีตระกูล TTL จะมีเบอร์ขึ้นต้นด้วยเลข 74 และไอซีตระกูล CMOS จะมีเบอร์ขึ้นต้นด้วยเลข 40 ไอซีแต่ละเบอร์ประกอบด้วยเกตตั้งแต่ 1 เกตขึ้นไป โดยทั่วไปไอซีที่ผลิตออกมาของแต่ละบริษัทจะมีมาตรฐานขั้นต่ำที่เหมือนกัน เช่น TEXAS, Instrument, National Semiconductor เป็นต้น รายละเอียดของค่าระดับแรงดันไฟฟ้าที่แสดงระดับลอจิก 0 หรือ 1 ของ TTL และ CMOS มีดังนี้

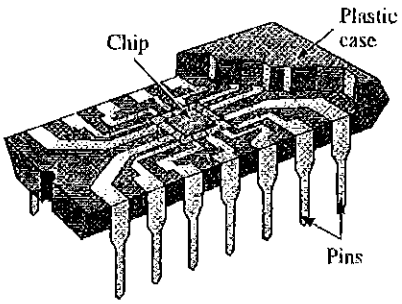
1. Transistor Transistor Logic(TTL, T^2L) ลักษณะของทรานซิสเตอร์ที่ใช้เป็นแบบไบโพลาร์ ทรานซิสเตอร์ (Bipolar Transistor) เมื่อป้อนไฟเลี้ยงที่ 5 โวลต์เข้าที่ไอซี TTL จะมีระดับ

แรงดันไฟฟ้าเมื่อเทียบกับค่าลอจิก 0 หรือ 1 ดังนี้ สัญญาณอินพุตที่มีค่าระดับแรงดันไฟฟ้าน้อยกว่า 0.8 โวลต์ แสดงลอจิก 0 หรือถ้ามีค่ามากกว่า 2 โวลต์ แสดงลอจิก 1 สำหรับสัญญาณเอาท์พุตที่มีค่าระดับแรงดันไฟฟ้าน้อยกว่า 0.4 โวลต์ แสดงลอจิก 0 หรือถ้ามีค่ามากกว่า 2.4 โวลต์แสดงลอจิก 1

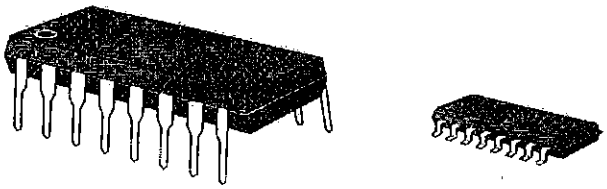
2. Complementary Metal Oxide Semiconductor (CMOS) เมื่อป้อนไฟเลี้ยงที่ 5 โวลต์เข้าที่ไอซี TTL จะมีระดับแรงดันไฟฟ้าเมื่อเทียบกับค่าลอจิก 0 หรือ 1 สัญญาณอินพุตที่มีค่าระดับแรงดันไฟฟ้าน้อยกว่า 1.5 โวลต์ แสดงลอจิก 0 หรือถ้ามีค่ามากกว่า 3.5 โวลต์แสดงลอจิก 1 สำหรับสัญญาณเอาท์พุตที่มีค่าระดับแรงดันไฟฟ้าน้อยกว่า 0.1 โวลต์ แสดงลอจิก 0 หรือถ้ามีค่ามากกว่า 4.9 โวลต์แสดงลอจิก 1

โครงสร้างของวงจรรวม

จากรูปที่ 4.40 แสดงโครงสร้างภายในของไอซีที่ประกอบด้วยชิพซิลิกอน (Silicon) ที่บรรจุอุปกรณ์อิเล็กทรอนิกส์ต่างๆ เช่น ทรานซิสเตอร์ ไดโอด ตัวต้านทานและตัวเก็บประจุไว้ บนชิพเพียงตัวเดียว แต่ละส่วนของชิพจะมีขา (Pins) ออกมาจากตัวถัง ลักษณะตัวถังของไอซีมีหลายแบบขึ้นอยู่กับการนำเอาไอซีไปใช้เพื่อต่อวงจรบนแผ่นวงจรพิมพ์ (Printed circuit board, PCB) ตัวอย่างเช่น แบบ DIP (Dual inline-package) มีขาออกจากไอซีจำนวน 2 แถว แต่ละแถวมีจำนวนขาเท่ากัน จำนวนขาที่มีได้แก่ 14, 16, 18, 20, ...เป็นต้น แบบ SMT (Surface-mount technology) จะใช้ในการติดตั้งไอซีบนผิวหน้าของแผ่นวงจรพิมพ์ ไอซีเป็นเทคโนโลยีแบบใหม่ที่ประกอบด้วยวงจรที่มีความซับซ้อนมากแต่ตัวไอซีมีขนาดเล็กมากเช่นกัน จากรูปที่ 4.42 แสดงตัวอย่างไอซีแบบ SMT ที่นิยมใช้กันในปัจจุบันมี 4 แบบคือ แบบ SOIC (Small-outline IC) แบบ PCCC (Plastic leaded chip carrier) แบบ LCCC (Leadless ceramic chip carrier) แบบ Flat pack (FP)

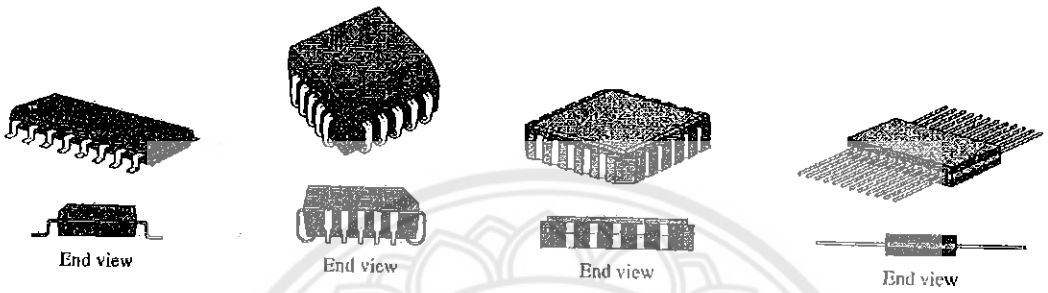


รูปที่ 4.40 โครงสร้างภายในของไอซี



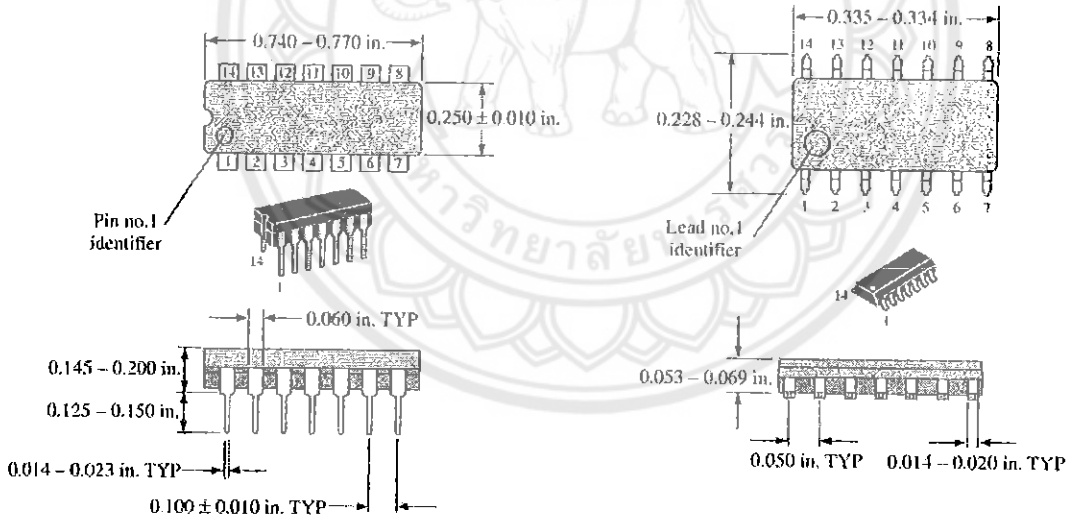
(ก) แบบ DIP (ข) แบบ SOIC

รูปที่ 4.41 ตัวอย่างของไอซีแบบต่างๆ



(ก) SOIC (ข) PLCC (ค) LCCC (ง) FP

รูปที่ 4.42 ไอซีแบบ SMT

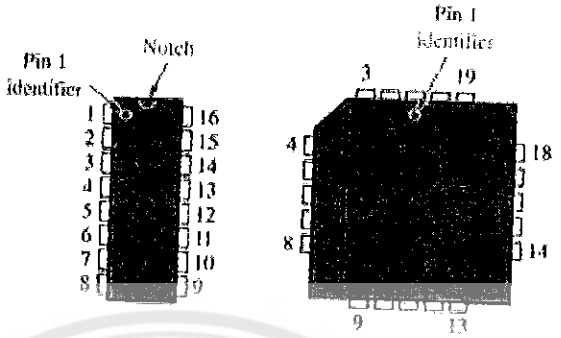


(ก) แบบ DIP (ข) แบบ SOIC

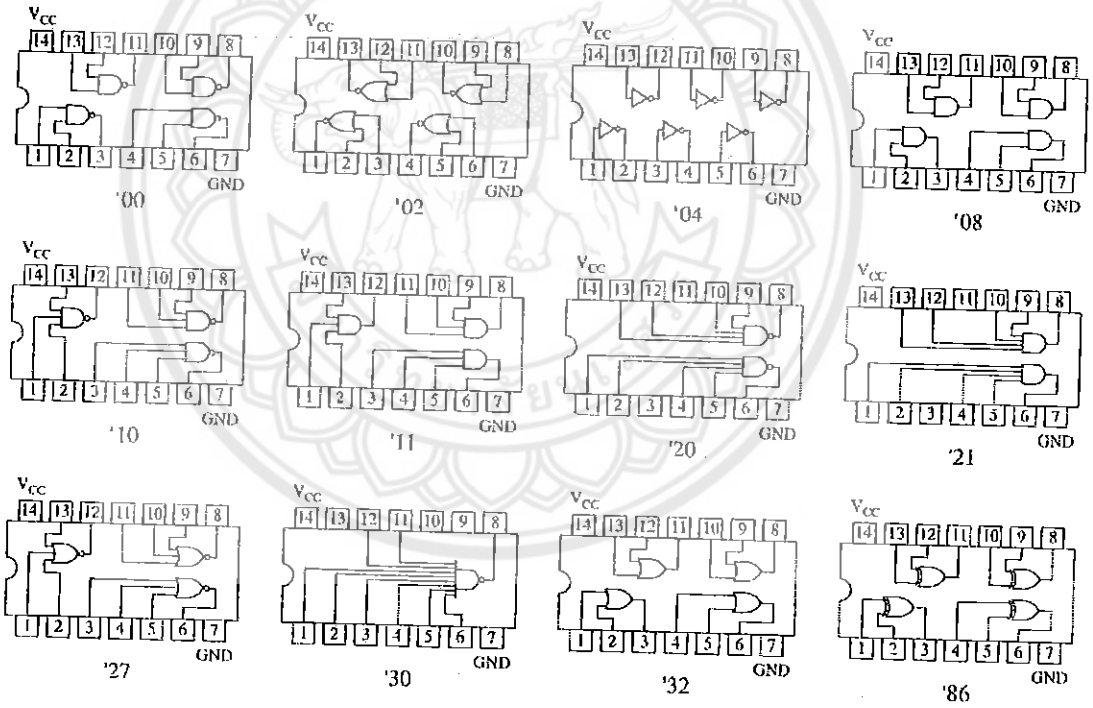
รูปที่ 4.43 ขนาดของแต่ละส่วนในไอซี

การแสดงตำแหน่งขาไอซี เริ่มจากพิจารณาทางด้านบนของตัวถัง (Top view) ดังรูปที่ 4.44 ขอบด้านหนึ่งที่ไม่มีขาจะมีลักษณะเว้าเข้าไปและไม่เป็นเส้นตรง จากนั้นพิจารณาขาในทิศทางทวนเข็มนาฬิกาจากจุดที่เว้าเข้าไป ขาแรกที่พบจะเป็นขาหมายเลข 1 และขาที่อยู่ต่อมาจะเป็นขาหมายเลข 2, 3, 4, ... ตามลำดับจนถึงขาสุดท้าย ในไอซีบางตัวจะมีเครื่องหมายเป็นจุดเป็นวงกลมลึกไปในตัวถังหรือมีสีแต้มบนตัวถังที่ตำแหน่งขาใดขาหนึ่งแสดงว่าขานั้นเป็นขาหมายเลข 1 จากนั้น

ขาอื่นๆ ให้ดูในทิศทางทวนเข็มนาฬิกาเช่นเดียวกัน จากรูปที่ 4.45 แสดงตำแหน่งขาที่สัมพันธ์กับโครงสร้างภายในของไอซีที่บรรจุลอจิกเกต สำหรับรูปที่ 4.46 เป็นบางส่วนของดาต้าชีท (Data sheet) ที่แสดงข้อมูลเฉพาะของไอซีเบอร์นั้น



รูปที่ 4.44 ตำแหน่งของขาไอซี



รูปที่ 4.45 ตำแหน่งขาและโครงสร้างภายในของไอซี

ประเภทของไอซี สามารถจำแนกประเภทไอซีจากโครงสร้างของวงจรที่บรรจุภายในชิพที่มีความซับซ้อนจากน้อยไปหามาก ดังนี้

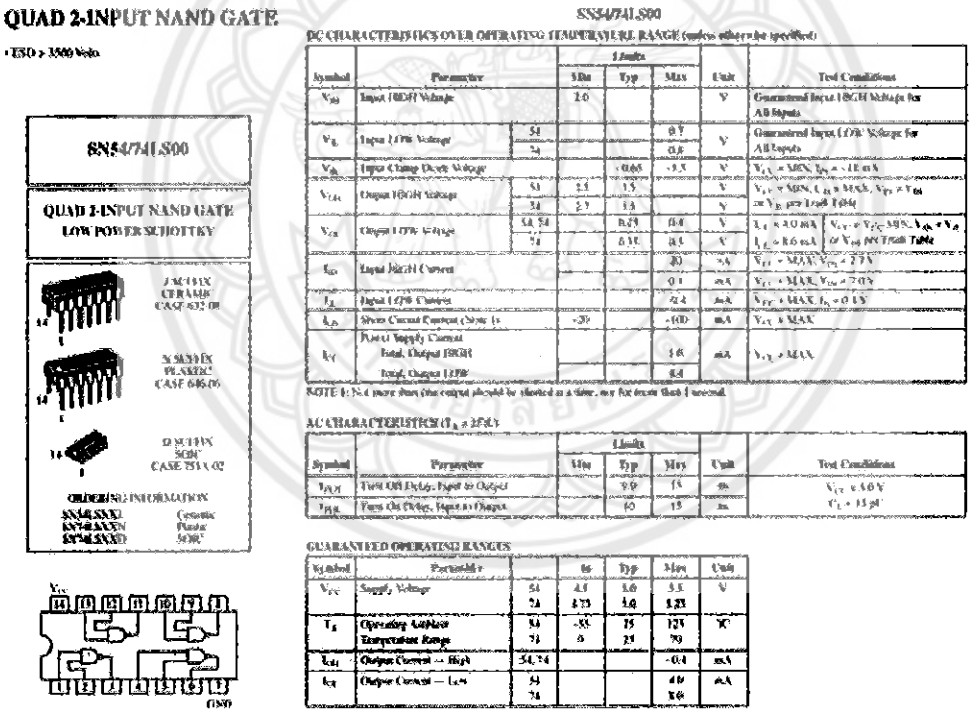
1. Small-scale integration (SSI) ชิพของไอซีประกอบด้วยเกตที่มีจำนวนไม่เกิน 12 ตัว ได้แก่ ไอซีของลอจิกเกตและฟลิปฟลอป เป็นต้น

2. Medium-scale integration (MSI) ชิพของไอซีประกอบด้วยเกตที่มีจำนวนตั้งแต่ 12 – 99 ตัว ได้แก่ไอซีของวงจรบวก (Adder) วงจรเข้ารหัส (Encoder) วงจรถอดรหัส (Decoder) วงจรนับ (Counter) วงจรรีจิสเตอร์ (Register) วงจรมัลติเพล็กซ์ (Multiplexer) วงจรคณิตศาสตร์ หน่วยความจำขนาดเล็ก เป็นต้น

3. Large-scale integration (LSI) ชิพของไอซีประกอบด้วยเกตที่มีจำนวนตั้งแต่ 100 –9999 ตัว ได้แก่ นาฬิกาดิจิตอล ชิพหน่วยความจำขนาดเล็ก เครื่องคำนวณอย่างง่าย เป็นต้น

4. Very large-scale integration (VLSI) ชิพของไอซีประกอบด้วยเกตที่มีจำนวนตั้งแต่ 10,000 – 99,999 ตัว ได้แก่ ไมโครโปรเซสเซอร์ ชิพหน่วยความจำขนาดใหญ่ เครื่องคำนวณขั้นสูง เป็นต้น

5. Ultra large-scale integration (ULSI) ชิพของไอซีประกอบด้วยเกตที่มีจำนวนตั้งแต่ 100,000 ตัวขึ้นไป ได้แก่ ชิพหน่วยความจำขนาดใหญ่ ไมโครโปรเซสเซอร์ขั้นสูง เป็นต้น

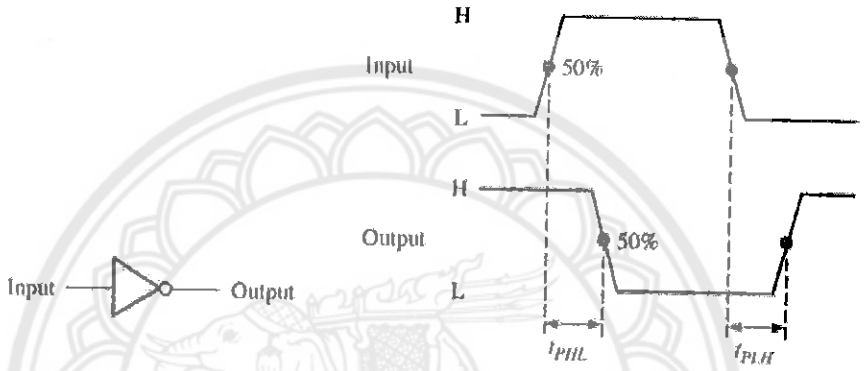


รูปที่ 4.46 รายละเอียดบางส่วนของดาต้าชีท

คุณสมบัติเฉพาะของไอซี

ประสิทธิภาพของไอซีสามารถวัดได้จากความเร็วในการทำงานของไอซี พิจารณาจากองค์ประกอบหรือพารามิเตอร์ที่สำคัญของไอซี ได้แก่

1. Propagation Delay Time (tp) คือเวลาที่ใช้ในการทำงานของลอจิกเกต เริ่มตั้งแต่เวลาที่ป้อนสัญญาณอินพุตจนกระทั่งได้สัญญาณเอาต์พุต การวัดค่าดังกล่าวสามารถพิจารณาจากการเปลี่ยนสถานะของสัญญาณได้ 2 แบบดังนี้ t_{PHL} คือช่วงเวลาของสัญญาณที่มีการเปลี่ยนจาก HIGH เป็น LOW (HL) และ t_{PLH} คือช่วงเวลาของสัญญาณที่มีการเปลี่ยนจาก LOW เป็น HIGH (LH) ตัวอย่างเช่น จากรูปที่ 4.47 แสดง Propagation delay time ของอินเวอร์เตอร์ การวัดเวลา t_{PHL} และ t_{PLH} จะวัดที่จุดกึ่งกลางที่มีการเปลี่ยนสถานะจาก HIGH เป็น LOW หรือจาก LOW เป็น HIGH



รูปที่ 4.47 Propagation Delay Time ของอินเวอร์เตอร์

2. DC Supply Voltage (Vcc) เป็นระดับของแรงดันไฟฟ้าที่ใช้เลี้ยงวงจร โดยปกติจะใช้แรงดันไฟฟ้า 5 โวลต์ แต่ขึ้นกับชนิดของไอซีด้วย ถ้าป้อนแรงดันสูงเกินไปไอซีอาจเสียหายได้หรือถ้าป้อนแรงดันต่ำเกินไปจะทำให้ไอซีทำงานไม่ได้ ตัวอย่างเช่น ไอซี TTL ใช้ไฟเลี้ยงวงจรต่ำสุดที่ 4.5 โวลต์ไปจนถึงสูงสุดที่ 5.5 โวลต์ ไอซี CMOS ใช้ไฟเลี้ยงวงจรต่ำสุดที่ 2.0 โวลต์ไปจนถึงสูงสุดที่ 6 โวลต์ เป็นต้น จะเห็นว่าแรงดันไฟเลี้ยงของไอซี CMOS จะมีขอบเขตของแรงดันที่กว้างกว่าไอซี TTL

3. Power Dissipation (P_D) เป็นกำลังไฟฟ้าของลอจิกเกตที่หาได้จากผลคูณของแรงดันไฟเลี้ยง DC และกระแสไฟฟ้าเฉลี่ย โดยปกติทางบริษัทผู้ผลิตจะให้ข้อมูลมาพร้อมกับ data sheet และโดยทั่วไปไอซี CMOS จะมีกำลังไฟฟ้าต่ำกว่า TTL สำหรับปริมาณของกระแสไฟฟ้าที่ไหลในวงจรที่จุดของสัญญาณเอาต์พุตทุกจุดที่มีสภาวะเป็น LOW จะใช้เป็นค่า I_{CCL} แต่ถ้าจุดของสัญญาณเอาต์พุตทุกจุดที่มีสภาวะเป็น HIGH จะใช้เป็นค่า I_{CCH} ดังนั้นกำลังไฟฟ้าเฉลี่ยของลอจิกเกตหาได้จาก

$$P_D = V_{CC}\left(\frac{I_{CCH} + I_{CCL}}{2}\right)$$

4. Input and Output Logic Level เป็นระดับของแรงดันไฟฟ้าที่ป้อนให้กับลอจิกเกตทางด้านอินพุตและเอาต์พุต กำหนดได้ดังนี้

V_{IL} เป็นระดับแรงดันไฟฟ้าที่อินพุตของลอจิกเกตแสดงสภาวะ LOW และ V_{IH} เป็นระดับแรงดันไฟฟ้าที่อินพุตของลอจิกเกตแสดงสภาวะ HIGH

V_{OL} เป็นระดับแรงดันไฟฟ้าที่เอาต์พุตของลอจิกเกตแสดงสภาวะ LOW และ V_{OH} เป็นระดับแรงดันไฟฟ้าที่เอาต์พุตของลอจิกเกตแสดงสภาวะ HIGH

สามารถสรุปค่าระดับแรงดันไฟฟ้าทางด้านอินพุตเอาต์พุตของไอซี CMOS และ TTL ได้ดังตารางที่ 4.3

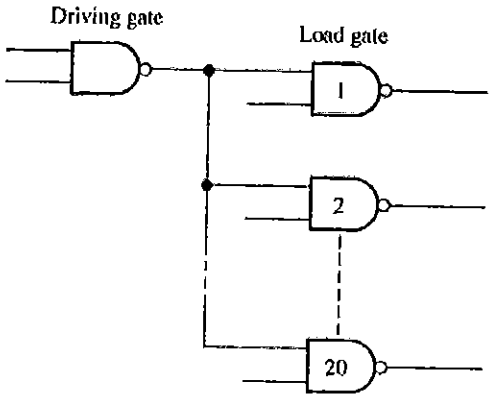
ตารางที่ 4.3 ระดับแรงดันไฟฟ้าทางด้านอินพุตเอาต์พุตของไอซี CMOS และ TTL

Voltage (Volts)		CMOS	TTL
V_{IL}	maximum	1.5	0.8
V_{IH}	minimum	3.5	2
V_{OL}	maximum	0.33	0.4
V_{OH}	minimum	4.4	2.4

5. Speed-Power Product (SPP) เป็นการวัดประสิทธิภาพของลอจิกเกตที่ขึ้นกับ Propagation delay time และกำลังไฟฟ้า หาได้จากสมการดังนี้

$$SPP = t_p P_D$$

6. Fan-out ของลอจิกเกตคือจำนวนอินพุตสูงสุดที่ต่อกับเอาต์พุตของเกต 1 ตัวที่เป็นไอซีชนิดเดียวกัน โดยระดับแรงดันไฟฟ้าด้านเอาต์พุตยังคงอยู่ในระดับที่กำหนดไว้ จากรูปที่ 4.48 จะเห็นว่าเอาต์พุตแนนด์เกตสามารถต่อเข้ากับอินพุตของแนนด์เกตได้จำนวนสูงสุดถึง 20 อินพุต



รูปที่ 4.48 Fan-out ของแนนด์เกต

บทที่ 5
พีชคณิตบูลีน (Boolean algebra)

วัตถุประสงค์

1. สามารถเข้าใจหลักการและความสำคัญของทฤษฎีพีชคณิตบูลีน
2. สามารถเข้าใจความสัมพันธ์ของรูปแบบมาตรฐานของสมการบูลีน
3. สามารถออกแบบและวิเคราะห์วงจรดิจิตอลด้วยพีชคณิตบูลีน

เนื้อหาบทนี้จะกล่าวถึงทฤษฎีของพีชคณิตบูลีน การพิสูจน์ทฤษฎีโดยใช้ตารางความจริง หรือการนำทฤษฎีอื่นมาช่วยในการพิสูจน์ รูปแบบมาตรฐานของการเขียนสมการบูลีน การออกแบบและวิเคราะห์วงจร รวมทั้งการลดรูปสมการบูลีนให้อยู่ในรูปสมการอย่างง่ายโดยใช้พีชคณิตบูลีน

5.1 บทนำ

ในปี 1854 นาย George Boole นักคณิตศาสตร์ชาวอังกฤษได้เขียนหนังสือเรื่อง การแสดงแนวคิดความเป็นเหตุเป็นผลของมนุษย์ (An Investigation of the Laws of Thought) โดยใช้สัญลักษณ์แทนเหตุการณ์ต่างๆ ที่มากระทำต่อกันในทางลอจิกหรือตรรกศาสตร์ (Logic) นาย George Boole กล่าวว่าระบบใดๆ สามารถแสดงความเป็นเหตุเป็นผลในรูปของข้อความที่เป็น "จริง" หรือ"เท็จ" ได้ ในปัจจุบันแนวคิดนี้เป็นที่รู้จักกันอย่างดีจึงเรียกว่าพีชคณิตบูลีน เพื่อเป็นเกียรติแก่นาย George Boole ต่อมาในปี 1938 พีชคณิตบูลีนได้ถูกนำมาประยุกต์ใช้งานเกี่ยวกับการแก้ปัญหาทางวิศวกรรม โดยนาย Claude Shannon จากมหาวิทยาลัยเอ็มไอที (MIT) ได้นำแนวคิดของ George Boole มาใช้ในการวิเคราะห์และออกแบบวงจรลอจิก โดยแสดงให้เห็นว่าพีชคณิตบูลีนสามารถนำมาใช้ในการออกแบบวงจรสวิทชิ่งที่สร้างจากรีเลย์ในระบบโทรศัพท์จนถึงระบบคอมพิวเตอร์ และได้เขียนวิทยานิพนธ์เรื่อง การวิเคราะห์สัญลักษณ์ของวงจรสวิทชิ่งและรีเลย์ (A Symbolic Analysis of Relay and Switching Circuit) ซึ่งเป็นพื้นฐานที่สำคัญของการออกแบบวงจรดิจิตอล การศึกษาเกี่ยวกับการนำพีชคณิตบูลีนไปประยุกต์ใช้ในการออกแบบวงจรลอจิกที่รู้จักกันดี เรียกว่าทฤษฎีวงจรสวิทชิ่ง (Switching circuit theory) พีชคณิตบูลีนจึงเป็นเครื่องมือสำคัญที่นำมาใช้อธิบายการทำงานของระบบดิจิตอลในรูปของสมการบูลีนได้เป็นอย่างดีและสามารถทำสมการบูลีนให้อยู่ในรูปสมการอย่างง่ายได้ โดยที่การทำงานของวงจรยังคงเหมือนกัน วงจรดิจิตอลที่ใช้พีชคณิตบูลีนมาอธิบายการทำงาน สามารถแบ่งวงจรออกเป็น 2 ประเภทคือวงจรลอจิกเชิงจัดหมู่หรือวงจรคอมบิเนชัน (Combinational Logic Circuit) และวงจรลอจิกเชิงลำดับหรือวงจรซีเควนเชียล (Sequential Logic Circuit) วงจรลอจิกเชิงจัดหมู่จะให้เอาต์พุตขึ้นอยู่กับอินพุตที่ป้อน

ให้กับวงจรในขณะนั้น แต่วงจรลอจิกเชิงลำดับจะให้เอาต์พุตขึ้นอยู่กับอินพุตที่ป้อนให้กับวงจรในขณะนั้นและอินพุตก่อนหน้านี้ด้วย วงจรนี้จึงต้องมีหน่วยความจำที่ใช้ในการเก็บอินพุตก่อนหน้านี้ไว้ด้วย

5.2 ทฤษฎีของพีชคณิตบูลีน (Boolean algebra theorems)

สมการบูลีนและการดำเนินการ (Boolean operations and expression)

พีชคณิตบูลีนประกอบด้วยตัวแปร ปกติใช้อักษรตัวพิมพ์ใหญ่ ที่มีการดำเนินการทางลอจิก 3 แบบคือ ไม่ใช่ (NOT) หรือ (OR) และ (AND) **การคอมพลีเมนต์ของบูลีน** (Complement) แสดงการกลับค่าลอจิกของตัวแปรเป็นค่าตรงข้าม ตัวแปรจะเขียนเครื่องหมายบาร์ไว้เหนือตัวอักษร ตัวอย่างการคอมพลีเมนต์ของตัวแปรใดๆ เช่น $\overline{A},\overline{B},\overline{C}$ การบวกแบบบูลีน (Boolean addition) ใช้ตัวดำเนินการออร์ (+) เทอมของผลบวกมีค่าเท่ากับ 1 เสมอ ถ้าตัวตั้งหรือตัวบวกมีค่าลอจิก 1 ตั้งแต่ 1 ค่าขึ้นไปและเทอมของผลบวกมีค่าเท่ากับ 0 ถ้าตัวตั้งและตัวบวกมีค่าลอจิก 0 ทั้งหมด ซึ่งเป็นไปตามคุณสมบัติของออร์เกต ตัวอย่างเทอมผลบวกแบบบูลีน เช่น $A+B, A+\overline{B}+C$ **การคูณแบบบูลีน** (Boolean multiplication) ใช้ตัวดำเนินการแอนด์ (•) เทอมของผลคูณมีค่าเป็นลอจิก 0 เมื่อตัวตั้งหรือตัวคูณมีค่าลอจิก 0 นอกนั้นให้ค่าผลคูณเป็นลอจิก 1 ซึ่งเป็นไปตามคุณสมบัติของแอนด์เกต ตัวอย่างเทอมผลคูณแบบบูลีน เช่น $AB, A\overline{B}, \overline{A}B\overline{C}D$ เป็นต้น

AND	OR	NOT
$0\cdot 0=0$	$0+0=0$	$\overline{0}=1$
$0\cdot 1=0$	$0+1=1$	$\overline{1}=0$
$1\cdot 0=0$	$1+0=1$	
$1\cdot 1=1$	$1+1=1$	

รูปที่ 5.1 การดำเนินการของพีชคณิตบูลีน

ทฤษฎีของพีชคณิตบูลีน

ทฤษฎีพีชคณิตบูลีนมีลักษณะเป็นคู่ซึ่งกันและกัน (Duality) โดยการเปลี่ยนตัวดำเนินการแอนด์หรือออร์ นอกจากนี้อาจมีการสลับที่กันระหว่างลอจิก 0 หรือ 1 ทฤษฎีพีชคณิตบูลีนที่สำคัญที่ใช้ในการออกแบบวงจรดิจิตอล มีดังนี้

(1) ทฤษฎีการสลับที่ (Commutative theorem)

(a) $A\cdot B=B\cdot A$

(b) $A+B=B+A$

(2) ทฤษฎีการจัดหมู่ (Associative theorem)

(a) $(A \cdot B) \cdot C = A \cdot (B \cdot C)$

(b) $(A + B) + C = A + (B + C)$

(3) ทฤษฎีการกระจาย (Distributive theorem)

(a) $A \cdot (B + C) = AB + AC$

(b) $A + (B \cdot C) = (A + B)(A + C)$

(4) ทฤษฎีเอกลักษณ์ (Identity theorem)

(a) $A \cdot 1 = A$

(b) $A + 1 = 1$

(5) ทฤษฎีว่าง (Null theorem)

(a) $A \cdot 0 = 0$

(b) $A + 0 = A$

(6) ทฤษฎีการนิเสธ (Negation theorem)

(a) $A\overline{A} = 0$

(b) $A + \overline{A} = 1$

(c) $\overline{\overline{A}} = A$

(7) ทฤษฎีไอเดมโพเทนท์ (Idempotent theorem)

(a) $A \cdot A \cdot A \cdot A ... = A$

(b) $A + A + A + A + ... = A$

(8) ทฤษฎีดีมอร์แกน (DeMorgan theorem)

(a) $\overline{A + B} = \overline{A} \cdot \overline{B}$

(b) $\overline{AB} = \overline{A} + \overline{B}$

(9) ทฤษฎีการซับซ้อน (Redundant theorem)

(a) $A + AB = A$

(b) $A \cdot (A + B) = A$

(10) ทฤษฎีการลดทอน

(a) $A + \overline{A}B = A + B$

(b) $A \cdot (\overline{A} + B) = AB$

การพิสูจน์ทฤษฎีพีชคณิตบูลีน

การพิสูจน์ทฤษฎีพีชคณิตบูลีนที่ใช้กันทั่วไปมี 2 วิธี วิธีแรกเป็นการพิสูจน์โดยใช้ตารางความจริงเป็นวิธีการที่ง่ายและเห็นได้ชัดเจนที่สุด โดยเขียนค่าในแต่ละเทอมของสมการทั้งสองด้านและพิจารณาผลลัพธ์สุดท้ายที่ได้ของทั้งสองด้านว่าเท่ากันหรือไม่ วิธีที่สองเป็นการนำทฤษฎีอื่นมาช่วยในการพิสูจน์ โดยการนำทฤษฎีมาแทนที่ในตัวแปรหรือเทอมของสมการนั้นเพื่อพิสูจน์ว่าทั้งสองด้านเท่ากัน ตัวอย่างเช่น

<u>การพิสูจน์โดยใช้ตารางความจริง</u>

(1) จงพิสูจน์ทฤษฎีดีมอร์แกน $\overline{A+B}=\overline{A}\cdot\overline{B}$

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

(2) จงพิสูจน์ทฤษฎีดีมอร์แกน $\overline{AB}=\overline{A}+\overline{B}$

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

(3) จงพิสูจน์ทฤษฎีการลดทอน $A+\overline{A}B=A+B$

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

(4) จงพิสูจน์ทฤษฎีการลดทอน $A\cdot(\overline{A}+B)=AB$

A	B	$\overline{A}$	$\overline{A}+B$	$A\cdot(\overline{A}+B)$	AB
0	0	1	1	0	0
0	1	1	1	0	0
1	0	0	0	0	0
1	1	0	1	1	1

(5) จงพิสูจน์ทฤษฎีการกระจาย $A\cdot(B+C)=AB+AC$

A	B	C	$(B+C)$	$A(B+C)$	AB	AC	$AB+AC$
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

(6) จงพิสูจน์ทฤษฎีการกระจาย $A+(B\cdot C)=(A+B)(A+C)$

A	B	C	$B\cdot C$	$A+(B\cdot C)$	$A+B$	$A+C$	$(A+B)(A+C)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

<u>การพิสูจน์โดยใช้ทฤษฎีพีชคณิตบูลีน</u>

(1) จงพิสูจน์ว่า $A+1=1$

<u>พิสูจน์</u>

$A+1=1\cdot(A+1)$ (ทฤษฎีที่ 4a)

$=(A+\overline{A})(A+1)$ (ทฤษฎีที่ 6b)

$=A+(\overline{A}\cdot 1)$ (ทฤษฎีที่ 3b)

$=A+\overline{A}$ (ทฤษฎีที่ 4a)

$=1$ (ทฤษฎีที่ 6b)

(2) จงพิสูจน์ว่า $A\cdot 0=0$

<u>พิสูจน์</u>

$A\cdot 0=0+(A\cdot 0)$ (ทฤษฎีที่ 5b)

$=(A\cdot\overline{A})+(A\cdot 0)$ (ทฤษฎีที่ 6a)

$=A\cdot(\overline{A}+0)$ (ทฤษฎีที่ 3a)

$=A\cdot\overline{A}$ (ทฤษฎีที่ 5b)

$=0$ (ทฤษฎีที่ 6a)

(3) จงพิสูจน์ว่า $A + A = A$

<u>พิสูจน์</u>

$A + A = (A + A) \cdot 1$ (ทฤษฎีที่ 4a)

$= (A + A)(A + \overline{A})$ (ทฤษฎีที่ 6b)

$= A + A\overline{A}$ (ทฤษฎีที่ 3b)

$= A + 0$ (ทฤษฎีที่ 6a)

$= A$ (ทฤษฎีที่ 5b)

(4) จงพิสูจน์ว่า $\overline{\overline{A}} = A$

<u>พิสูจน์</u>

$\overline{\overline{A}} = \overline{\overline{A}} + 0$ (ทฤษฎีที่ 5b)

$= \overline{\overline{A}} + A\overline{A}$ (ทฤษฎีที่ 6a)

$= (\overline{\overline{A}} + A)(\overline{\overline{A}} + \overline{A})$ (ทฤษฎีที่ 3b)

$= (\overline{\overline{A}} + A) \cdot 1$ (ทฤษฎีที่ 6b)

$= (\overline{\overline{A}} + A)(A + \overline{A})$ (ทฤษฎีที่ 6b)

$= A + \overline{\overline{A}}\overline{A}$ (ทฤษฎีที่ 3b)

$= A + 0$ (ทฤษฎีที่ 6a)

$= 0$ (ทฤษฎีที่ 5b)

(5) จงพิสูจน์ว่า $A + \overline{A}B = A + B$

<u>พิสูจน์</u>

$A + \overline{A}B = (A + \overline{A})(A + B)$ (ทฤษฎีที่ 3b)

$= 1 \cdot (A + B)$ (ทฤษฎีที่ 6b)

$= A + B$ (ทฤษฎีที่ 4a)

(6) จงพิสูจน์ว่า $A(\overline{A} + B) = AB$

<u>พิสูจน์</u>

$A(\overline{A} + B) = A\overline{A} + AB$ (ทฤษฎีที่ 3a)

$= 0 + AB$ (ทฤษฎีที่ 6a)

$= AB$ (ทฤษฎีที่ 5b)

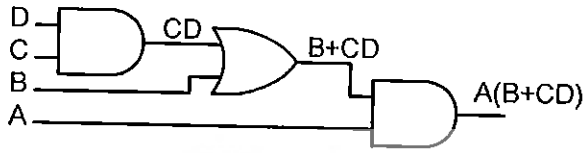
5.3 การวิเคราะห์วงจรลอจิกด้วยสมการบูลีน (Boolean analysis of logic circuit)

สมการบูลีนสามารถนำมาใช้อธิบายการทำงานของวงจรลอจิก ที่เกิดจากการนำเกตแต่ละตัวมาต่อเข้าด้วยกันเป็นวงจร เอาต์พุตของวงจรสามารถเขียนในรูปสมการที่แสดงความสัมพันธ์ของตัวแปรอินพุตแต่ละตัว การเขียนสมการบูลีนจากวงจรลอจิกนั้น เริ่มจากการเขียนเอาต์พุตของเกตแต่ละตัว ตั้งแต่เกตที่อยู่ทางด้านซ้ายสุดจนกระทั่งถึงเกตตัวสุดท้ายในวงจร ตัวอย่างเช่น จากวงจรรูปที่ 5.2 สามารถเขียนสมการบูลีนของวงจรได้ดังนี้

(1) แอนด์เกตที่อยู่ด้านซ้ายสุดมีอินพุต C และ D ดังนั้นสมการเอาต์พุตของแอนด์เกตคือ CD

(2) เอาต์พุตของแอนด์เกตจาก (1) และอินพุต B จะป้อนเป็นอินพุตให้กับออร์เกต ดังนั้นสมการเอาต์พุตของออร์เกต คือ $B+CD$

(3) เอาต์พุตของออร์เกตจาก (2) และอินพุต A จะป้อนเป็นอินพุตให้กับแอนด์เกตที่อยู่ด้านขวาสุด ดังนั้นสมการเอาต์พุตของแอนด์เกต คือ $A(B+CD)$



รูปที่ 5.2 การเขียนสมการบูลีนจากวงจรลอจิก

นอกจากนี้สมการบูลีนของวงจรลอจิก สามารถนำไปเขียนเป็นตารางความจริงเพื่อหาความสัมพันธ์ระหว่างเอาต์พุตของวงจรและค่าของตัวแปรอินพุตที่เป็นไปได้ทั้งหมด จากวงจรรูปที่ 5.2 ประกอบด้วยตัวแปรอินพุตทั้งหมด 4 ตัวคือ A,B,C,D ดังนั้นค่าลอจิกของตัวแปรอินพุตที่เป็นไปได้มีจำนวนทั้งหมด 2^4=16 กรณี จากนั้นแทนค่าลอจิก 0 หรือ 1 ของตัวแปรอินพุตที่เป็นไปได้ในสมการ $Y = A(B+CD)$ โดยใช้หลักการบวกและการคูณแบบบูลีน จะได้ตารางความจริงของวงจรรูปที่ 5.2 ดังนี้

ตารางที่ 5.1 ตารางความจริงของวงจรรูปที่ 5.2

A	B	C	D	$Y=A(B+CD)$
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

ตัวอย่างเช่น ถ้าตัวแปรอินพุต A,B,C,D มีค่าเป็นลอจิก 0 ทั้งหมด จะได้ $Y = A(B+CD) = 0\cdot(0+0\cdot 0) = 0$ หรือถ้าตัวแปรอินพุต A,B,C,D มีค่าเป็นลอจิก 1011 ตามลำดับ จะได้ $Y = A(B+CD) = 1\cdot(0+1\cdot 1) = 1$ เป็นต้น

5.4 รูปแบบมาตรฐานของสมการบูลีน (Standard forms of Boolean expression)

สมการบูลีนมีรูปแบบมาตรฐาน 2 แบบคือ ผลบวกของผลคูณ (Sum-of-Product, SOP) และผลคูณของผลบวก (Product-of-Sum, POS) ในที่นี้คำว่าผลบวกหรือผลคูณไม่ได้หมายถึงการบวกหรือการคูณของตัวเลข แต่หมายถึงการดำเนินการออร์ (+) และการดำเนินการแอนด์ (•) เนื่องจากมีเครื่องหมายเหมือนกับการบวกและการคูณ จึงนำคำว่าผลบวกหรือผลคูณมาใช้แทนการออร์หรือการแอนด์ โดยทั่วไปในการออกแบบวงจรลอจิกมักนิยมเขียนสมการ SOP มากกว่าสมการ POS แต่การเลือกสมการรูปแบบใดนั้นขึ้นอยู่กับจำนวนเกตและลักษณะของวงจรที่ทำการออกแบบเพื่อนำไปใช้งานด้วย

รูปแบบผลบวกของผลคูณหรือ SOP

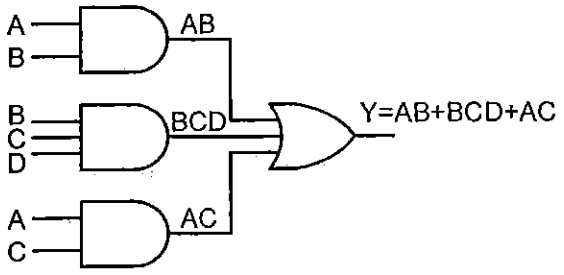
สมการ SOP จะเขียนตัวแปรของแต่ละเทอมหรือนิพจน์ในรูปของการแอนด์กัน ในแต่ละเทอมประกอบด้วยตัวแปรตั้งแต่ 1 ตัวขึ้นไปและตัวแปรแต่ละตัวอาจอยู่ในรูปของการคอมพลีเมนต์หรือไม่มีการคอมพลีเมนต์ก็ได้ จากนั้นนำแต่ละเทอมมาทำการออร์กัน เช่น

$$Y = \overline{A}\cdot B\cdot C + A\cdot B + \overline{A}\cdot\overline{B}$$

$$Y = A\cdot B + \overline{A}\cdot B\cdot\overline{C} + \overline{C}\cdot\overline{D} + C$$

$$Y = \overline{A}\cdot B + C\cdot\overline{D} + E\cdot F + G\cdot K + H\cdot\overline{L}$$

จากตัวอย่างจะเห็นว่าสมการ SOP ไม่มีการคอมพลีเมนต์ตัวแปรพร้อมกันมากกว่า 1 ตัว เช่น $\overline{ABC}$ หรือ $\overline{RST}$ เป็นต้น การเขียนวงจรอย่างง่ายของสมการ SOP จะนำเอาต์พุตของแอนด์เกตตั้งแต่ 2 ตัวขึ้นไป มาทำการออร์กัน จำนวนของแอนด์เกตมีค่าเท่ากับจำนวนเทอมของผลคูณ ดังนั้นเอาต์พุตของออร์เกตที่ได้มีค่าเท่ากับสมการ SOP ดังรูปที่ 5.3



รูปที่ 5.3 วงจรลอจิกของสมการ SOP เมื่อ $Y = AB + BCD + AC$

ตัวอย่างที่ 5.1 จงแปลงสมการบูลีนต่อไปนี้ให้อยู่ในรูปสมการ SOP

(1) $Y = AB + B(CD + EF)$

(2) $Y = (A + B)(B + C + D)$

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

<u>วิธีทำ</u>

(1) $Y = AB + B(CD + EF) = AB + BCD + BEF$

(2) $Y = (A + B)(B + C + D) = AB + AC + AD + BB + BC + BD$

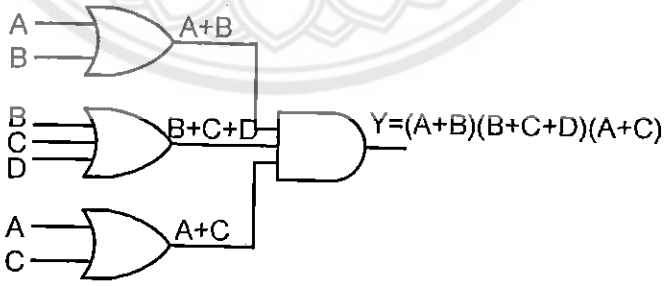
(3) $Y = \overline{(\overline{A + B}) + C} = (\overline{\overline{A + B}})\overline{C} = (A + B)\overline{C} = A\overline{C} + B\overline{C}$

รูปแบบผลคูณของผลบวกหรือ POS

สมการ POS จะเขียนตัวแปรของแต่ละเทอมหรือนิพจน์ในรูปของการออร์กัน ในแต่ละเทอมประกอบด้วยตัวแปรตั้งแต่ 1 ตัวขึ้นไปและตัวแปรแต่ละตัวอาจอยู่ในรูปของการคอมพลีเมนต์หรือไม่มีการคอมพลีเมนต์ก็ได้ จากนั้นนำแต่ละเทอมมาทำการแอนด์กัน เช่น

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

การเขียนวงจรอย่างง่ายของสมการ POS จะนำเอาต์พุตของออร์เกตตั้งแต่ 2 ตัวขึ้นไปมาทำการแอนด์กัน จำนวนของออร์เกตจะเท่ากับจำนวนเทอมของผลบวก ดังนั้นเอาต์พุตของแอนด์เกตที่ได้มีค่าเท่ากับสมการ POS ดังรูปที่ 5.4



รูปที่ 5.4 วงจรลอจิกของสมการ POS เมื่อ $Y = (A + B)(B + C + D)(A + C)$

การเขียน Minterm และ Maxterm

ตัวแปรของเลขฐานสองสามารถเขียนให้อยู่ในรูปของตัวแปรปกติหรือตัวแปรคอมพลีเมนต์ได้ การนำตัวแปรมาแอนด์หรือออร์กัน มีรูปแบบการเขียนดังนี้

(1) Minterm คือเทอมที่ตัวแปรอยู่ในรูปของการแอนด์กันครบทุกตัวแปร โดยที่ตัวแปรปกติ ($A,B,C,...$) จะเขียนแทนลอจิก 1 และตัวแปรคอมพลีเมนต์ ($\overline{A},\overline{B},\overline{C},...$) จะเขียนแทนลอจิก 0 ถ้าสมมุติให้สมการมีตัวแปรทั้งหมด n ตัว สามารถเขียน Minterm ได้ทั้งหมด 2^n เทอม ในแต่ละ Minterm มีตัวแปรจำนวน n ตัวมาทำการแอนด์กัน จะใช้สัญลักษณ์ m_j แทน Minterm ใดๆ เมื่อ j คือเลขฐานสิบที่ได้จากการแปลงเลขฐานสองของตัวแปรใน Minterm นั้น

(2) Maxterm คือเทอมที่ตัวแปรอยู่ในรูปของการออร์กันครบทุกตัวแปร โดยที่ตัวแปรปกติ ($A,B,C,...$) จะเขียนแทนลอจิก 0 และตัวแปรคอมพลีเมนต์ ($\overline{A},\overline{B},\overline{C},...$) จะเขียนแทนลอจิก 1 ถ้าสมมุติให้สมการมีตัวแปรทั้งหมด n ตัว สามารถเขียน Maxterm ได้ทั้งหมด 2^n เทอม ในแต่ละ Maxterm มีตัวแปรจำนวน n ตัวมาทำการออร์กัน จะใช้สัญลักษณ์ M_j แทน Maxterm ใดๆ เมื่อ j คือเลขฐานสิบที่ได้จากการแปลงเลขฐานสองของตัวแปรใน Maxterm นั้น

ตัวอย่างการเขียน Minterm และ Maxterm ของ 2 และ 3 ตัวแปร แสดงดังตารางที่ 5.2 การเขียนสมการ SOP จะใช้สัญลักษณ์ $\sum m$ หมายถึงผลบวกของ Minterm และการเขียนสมการ POS จะใช้สัญลักษณ์ $\prod M$ หมายถึงผลคูณของ Maxterm

ตารางที่ 5.2 การเขียน Minterm และ Maxterm ของ 2 และ 3 ตัวแปร

A	B	Minterm		Maxterm	
0	0	$\overline{A}\cdot\overline{B}$	m_0	$A+B$	M_0
0	1	$\overline{A}\cdot B$	m_1	$A+\overline{B}$	M_1
1	0	$A\cdot\overline{B}$	m_2	$\overline{A}+B$	M_2
1	1	$A\cdot B$	m_3	$\overline{A}+\overline{B}$	M_3

A	B	C	Minterm		Maxterm	
0	0	0	$\overline{A}\cdot\overline{B}\cdot\overline{C}$	m_0	$A+B+C$	M_0
0	0	1	$\overline{A}\cdot\overline{B}\cdot C$	m_1	$A+B+\overline{C}$	M_1
0	1	0	$\overline{A}\cdot B\cdot\overline{C}$	m_2	$A+\overline{B}+C$	M_2
0	1	1	$\overline{A}\cdot B\cdot C$	m_3	$A+\overline{B}+\overline{C}$	M_3
1	0	0	$A\cdot\overline{B}\cdot\overline{C}$	m_4	$\overline{A}+B+C$	M_4
1	0	1	$A\cdot\overline{B}\cdot C$	m_5	$\overline{A}+B+\overline{C}$	M_5
1	1	0	$A\cdot B\cdot\overline{C}$	m_6	$\overline{A}+\overline{B}+C$	M_6
1	1	1	$A\cdot B\cdot C$	m_7	$\overline{A}+\overline{B}+\overline{C}$	M_7

รูปแบบ Canonical (Canonical form)

ถ้าสมการ SOP หรือ POS ที่มีตัวแปรครบทุกเทอม จะเรียกสมการบูลีนนี้ว่า Canonical ซึ่งรูปแบบ Canonical มี 2 แบบ คือ

(1) Canonical Sum คือสมการ SOP ที่มีทุกเทอมเป็น Minterm เช่น

$$f(A,B,C) = \overline{A}BC + ABC + \overline{A}\overline{B}C$$

(2) Canonical Product คือสมการ POS ที่มีทุกเทอมเป็น Maxterm เช่น

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

สมการบูลีนที่อยู่ในรูปแบบ Canonical จะมีประโยชน์อย่างมากในการนำสมการบูลีนไปสร้างตารางความจริงและการเขียนแผนผังคาร์โนห์

ตัวอย่างที่ 5.2 จงเขียนสมการต่อไปนี้ในรูปสัญลักษณ์ $\sum m$ และ $\prod M$

(1) $f(A,B,C) = A\overline{B}C + AB\overline{C} + \overline{A}BC + A\overline{B}\overline{C}$

(2) $f(A,B,C) = (A+\overline{B}+C)(A+B+\overline{C})(\overline{A}+B+C)(A+\overline{B}+\overline{C})$

<u>วิธีทำ</u>

(1) $f(A,B,C) = A\overline{B}C + AB\overline{C} + \overline{A}BC + A\overline{B}\overline{C}$

101 + 110+ 011+ 100

$= m_5 + m_6 + m_3 + m_4$

$= \sum m(3,4,5,6)$

(2) $f(A,B,C) = (A+\overline{B}+C)(A+B+\overline{C})(\overline{A}+B+C)(A+\overline{B}+\overline{C})$

(010) (001) (100) (011)

$= M_2 \cdot M_1 \cdot M_4 \cdot M_3$

$= \prod M(1,2,3,4)$

ตัวอย่างที่ 5.3 จงเขียนสมการต่อไปนี้ในรูปของ Canonical Sum

(1) $f(A,B) = \sum m(0,2)$

(2) $f(A,B,C) = \sum m(0,2,4,6)$

<u>วิธีทำ</u>

(1) $f(A,B) = \sum m(0,2) \quad = m_0 + m_2 \quad = \overline{A}\overline{B} + A\overline{B}$

(2) $f(A,B,C) = \sum m(0,2,4,6) \quad = m_0 + m_2 + m_4 + m_6 \quad = \overline{A}\overline{B}\overline{C} + \overline{A}B\overline{C} + A\overline{B}\overline{C} + AB\overline{C}$

ตัวอย่างที่ 5.4 จงเขียนสมการต่อไปนี้ในรูปของ Canonical Product

(1) $f(A,B) = \prod M(0,2)$

(2) $f(A,B,C) = \prod M(0,2,4,6)$

<u>วิธีทำ</u>

(1) $f(A,B)=\prod M(0,2) = M_0 \cdot M_2 = (A+B)(\overline{A}+B)$

(2) $f(A,B,C)=\prod M(0,2,4,6)$

$$= M_0 \cdot M_2 \cdot M_4 \cdot M_6$$

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

การแปลงสมการบูลีนให้อยู่ในรูปของ Canonical

กรณีสมการบูลีนที่แต่ละเทอมมีตัวแปรไม่ครบจำนวน n **ตัวหรือสมการไม่อยู่ในรูปของ** Canonical **สามารถแปลงสมการให้อยู่ในรูปของ** Canonical **ได้ ดังนี้**

(1) กรณีของสมการ SOP ถ้าแต่ละเทอมของสมการขาดตัวแปรใดๆ สมมุติว่าเป็นตัวแปร x ให้นำ $x+\overline{x}$ มาแอนด์กับเทอมนั้น เนื่องจาก $x+\overline{x}$ มีค่าเท่ากับ 1 เมื่อนำ 1 มาแอนด์กับเทอมใดๆ จะได้ผลลัพธ์เป็นเทอมนั้นเหมือนเดิม เช่น

$$f(A,B,C) = A\overline{B} + AC + BC$$

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

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

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

$$= m_5 + m_4 + m_7 + m_3$$

$$= \sum m(3,4,5,7)$$

จากสมการ $f(A,B,C) = A\overline{B} + AC + BC$ จะเห็นว่าเทอมแรกขาดตัวแปร C ดังนั้นจึงนำ $C+\overline{C}$ มาแอนด์กับเทอมแรก ส่วนเทอมที่สองและสามขาดตัวแปร B และ A ให้นำ $B+\overline{B}$ และ $A+\overline{A}$ มาแอนด์ตามลำดับ จากนั้นใช้ทฤษฎีการกระจาย (3a) จะได้สมการที่อยู่ในรูป Canonical Sum

(2) กรณีของสมการ POS ถ้าแต่ละเทอมของสมการขาดตัวแปรใดๆ สมมุติว่าเป็นตัวแปร x ให้นำ $x\overline{x}$ มาออร์กับเทอมนั้น เนื่องจาก $x\overline{x}$ มีค่าเท่ากับ 0 เมื่อนำ 0 มาออร์กับเทอมใดๆ จะได้ผลลัพธ์เป็นเทอมนั้นเหมือนเดิม เช่น

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

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

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

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

$$= M_2 \cdot M_3 \cdot M_0 \cdot M_4$$
$$= \prod M(0,2,3,4)$$

จากสมการบูลีน $f(A,B,C) = (A+\overline{B})(A+C)(B+C)$ จะเห็นได้ว่าเทอมแรกขาดตัวแปร C ดังนั้นจึงนำ $C\overline{C}$ มาออร์กับเทอมแรก ส่วนเทอมที่สองและสามขาดตัวแปร B และ A ให้นำ $B\overline{B}$ และ $A\overline{A}$ มาออร์ตามลำดับ จากนั้นใช้ทฤษฎีการกระจาย (3b) จะได้สมการที่อยู่ในรูป Canonical Product

ความสัมพันธ์ระหว่าง Canonical Sum และ Canonical Product

โดยทั่วไปสมการ SOP และ POS ที่อยู่ในรูป Canonical สามารถเขียนความสัมพันธ์ได้ดังนี้ $M_j = \overline{m}_j$ หรือ $m_j = \overline{M}_j$ นั่นคือผลคูณของ Maxterm มีค่าเท่ากับคอมพลีเมนต์ของผลบวกของ Minterm หรือผลบวกของ Minterm มีค่าเท่ากับคอมพลีเมนต์ของผลคูณของ Maxterm สามารถนำทฤษฎีดีมอร์แกนมาช่วยในการแปลงรูปสมการเพื่อหาความสัมพันธ์ได้ เช่น

(1) $$f(A,B) = \sum m(0,2)$$
$$= \overline{A}\,\overline{B} + A\overline{B}$$
$$= \overline{\overline{\overline{A}\,\overline{B} + A\overline{B}}}$$
$$= \overline{(\overline{\overline{A}\,\overline{B}}) \cdot (\overline{A\overline{B}})}$$
$$= \overline{(\overline{\overline{A}} + \overline{\overline{B}})(\overline{A} + \overline{\overline{B}})}$$
$$= \overline{(A+B)(\overline{A}+B)}$$
$$= \overline{M_0 \cdot M_2}$$
$$= \overline{\prod M(0,2)}$$
$$= \prod M(1,3)$$

(2) $$f(A,B) = \prod M(1,3)$$
$$= \overline{\overline{(A+\overline{B})(\overline{A}+\overline{B})}}$$
$$= \overline{(\overline{A+\overline{B}}) + (\overline{\overline{A}+\overline{B}})}$$
$$= \overline{(\overline{A}B) + (AB)}$$
$$= \overline{m_1 + m_3}$$
$$= \overline{\sum m(1,3)}$$
$$= \sum m(0,2)$$

จาก (1) และ (2) สรุปได้ว่า

$$f(A,B) = \sum m(0,2) = \overline{\sum m(1,3)} = \prod M(1,3) = \overline{\prod M(0,2)}$$

5.5 สมการบูลีนและตารางความจริง (Boolean expression and Truth tables)

จากหัวข้อที่ 5.3 ได้กล่าวถึงการสร้างตารางความจริงจากสมการบูลีนของวงจรลอจิก โดยการแทนค่าลอจิก 0 หรือ 1 ของตัวแปรอินพุตที่เป็นไปได้ทุกกรณีลงในสมการโดยตรงและจะได้ค่าเอาต์พุตออกมาเป็นค่าลอจิก 0 หรือ 1 ถ้าสมการมีตัวแปรที่เกี่ยวข้องจำนวนมาก การสร้างตารางความจริงของสมการนั้นจะใช้เวลานานมาก แต่ถ้านำ Minterm และ Maxterm ของสมการบูลีนมาใช้ จะทำให้การสร้างตารางความจริงทำได้ง่ายขึ้น โดยใช้ค่าลอจิก 0 หรือ 1 ของแต่ละเทอมในสมการมาเขียนแสดงความสัมพันธ์ในรูปของตารางความจริง

การแปลงสมการ SOP เป็นตารางความจริง (Converting SOP to Truth table)

การแปลงสมการ SOP ไปเป็นตารางความจริงนั้น จำเป็นต้องเขียนสมการ SOP ให้อยู่ในรูปของ Canonical Sum ก่อน นั่นคือสมการ SOP ต้องมีทุกเทอมในสมการเป็น Minterm จากนั้นแทนที่แต่ละ Minterm ของสมการในตารางความจริงที่คอลัมน์เอาต์พุตด้วยค่าลอจิก 1 เอาต์พุตที่เหลือในตารางความจริงมีค่าลอจิก 0

ตัวอย่างที่ 5.5 จงเขียนตารางความจริงของสมการ $Y = \overline{A}\overline{B}C + A\overline{B}\overline{C} + ABC$

<u>วิธีทำ</u>

จากสมการ $Y = \overline{A}\overline{B}C + A\overline{B}\overline{C} + ABC$ จะเห็นว่าสมการอยู่ในรูปของ Canonical Sum ตัวแปรในเทอมของผลคูณหรือ Minterm สามารถเขียนในรูปของเลขฐานสองได้ดังนี้ $\overline{A}\overline{B}C = 001$ $A\overline{B}\overline{C} = 100$ $ABC = 111$ จากนั้นแทนที่แต่ละ Minterm ในคอลัมน์เอาต์พุตของตารางความจริงเท่ากับ 1 นอกนั้นให้เอาต์พุตมีค่าเท่ากับ 0

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

การแปลงสมการ POS เป็นตารางความจริง (Converting POS to Truth table)

การแปลงสมการ POS ไปเป็นตารางความจริงนั้น จำเป็นต้องเขียนสมการ POS ให้อยู่ในรูปของ Canonical Product ก่อน นั่นคือสมการ POS ต้องมีทุกเทอมในสมการเป็น Maxterm จากนั้นแทนที่แต่ละ Maxterm ของสมการในตารางความจริงที่คอลัมน์เอาต์พุตด้วยค่าลอจิก 0 เอาต์พุตที่เหลือในตารางความจริงมีค่าลอจิก 1

ตัวอย่างที่ 5.6 จงเขียนตารางความจริงของสมการ

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

<u>วิธีทำ</u>

จากสมการที่กำหนดให้จะเห็นว่าสมการอยู่ในรูปของ Canonical Product ตัวแปรในเทอมของผลบวกหรือ Maxterm สามารถเขียนในรูปของเลขฐานสองได้ดังนี้

$A+B+C=000$ $\quad A+\overline{B}+C=010$ $\quad A+\overline{B}+\overline{C}=011$

$\overline{A}+B+\overline{C}=101$ $\quad \overline{A}+\overline{B}+C=110$

จากนั้นแทนที่แต่ละ Maxterm ในคอลัมน์เอาต์พุตของตารางความจริงเท่ากับ 0 นอกนั้นให้เอาต์พุตมีค่าเท่ากับ 1

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

การเขียนสมการบูลีนจากตารางความจริง (Determining expression from Truth table)

เมื่อกำหนดตารางความจริงที่ประกอบด้วยอินพุต 2 ตัวแปรคือ A,B และเอาต์พุตคือ $f(A,B)$ ดังนี้

A	B	$f(A,B)$
0	0	f_0
0	1	f_1
1	0	f_2
1	1	f_3

<u>กรณีการเขียนสมการ SOP</u>

$$f(A,B)=\sum_{i=0}^{3} m_i f_i$$

$$=m_0 f_0+m_1 f_1+m_2 f_2+m_3 f_3$$

จากสมการดังกล่าว เมื่อกำหนด f_i ตามเงื่อนไขดังนี้

ถ้า $f_i=0$ จะได้ $m_i f_i=0$

ถ้า $f_i = 1$ จะได้ $m_i f_i = m_i$

ดังนั้นการเขียนสมการ SOP จะเลือกเขียน Minterm ที่มีเอาต์พุต $f(A,B)$ เท่ากับลอจิก 1

<u>กรณีการเขียนสมการ POS</u>

$$f(A,B) = \prod_{i=0}^{3}(M_i + f_i)$$
$$= (M_0 + f_0)(M_1 + f_1)(M_2 + f_2)(M_3 + f_3)$$

จากสมการดังกล่าว เมื่อกำหนด f_i ตามเงื่อนไขดังนี้

ถ้า $f_i = 0$ จะได้ $M_i + f_i = M_i$

ถ้า $f_i = 1$ จะได้ $M_i + f_i = 1$

ดังนั้นการเขียนสมการ POS จะเลือกเขียน Maxterm ที่มีเอาต์พุต $f(A,B)$ เท่ากับลอจิก 0

ตัวอย่างที่ 5.7 จงเขียนสมการ SOP และ POS จากตารางความจริงที่กำหนดให้

A	B	C	f(A,B,C)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

<u>**วิธีทำ**</u>

$$f(A,B,C) = \sum m(1,2,4,5,7)$$
$$= m_1 + m_2 + m_4 + m_5 + m_7$$
$$= \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C + ABC$$

$$f(A,B,C) = \prod M(0,3,6)$$
$$= M_0 \cdot M_3 \cdot M_6$$
$$= (A+B+C)(A+\overline{B}+\overline{C})(\overline{A}+\overline{B}+C)$$

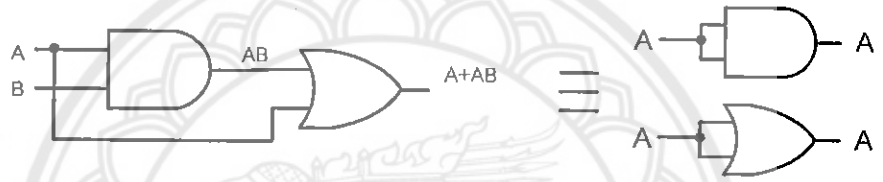
5.6 การลดรูปสมการโดยใช้พีชคณิตบูลีน (Simplification using Boolean algebra)

ในการออกแบบวงจรลอจิก ถ้าสมการบูลีนมีจำนวนเทอมที่มาก จะทำให้วงจรที่ออกแบบใช้เกตเป็นจำนวนมากด้วย นอกจากนี้การตรวจสอบการทำงานของวงจรสามารถทำได้ยากหรือ

วงจรอาจเกิดเสียหายได้ง่าย ดังนั้นการลดรูปสมการบูลีนจะช่วยลดจำนวนเทอมของสมการให้น้อยลงทำให้จำนวนเกตที่นำมาใช้ในวงจรมีจำนวนน้อยลงด้วย แต่การทำงานของวงจรยังคงเหมือนเดิมทุกอย่าง แต่การลดรูปสมการโดยใช้พีชคณิตบูลีนมีข้อจำกัดคือไม่มีกฎเกณฑ์การลดรูปสมการที่แน่นอน ทำให้ผลลัพธ์สุดท้ายของสมการที่ได้อาจแตกต่างกัน ขึ้นอยู่กับประสบการณ์ของผู้ออกแบบโดยตรง เนื่องจากสมการที่ลดรูปแล้ว ไม่สามารถบอกได้ว่าสมการนั้นมีจำนวนเทอมที่น้อยที่สุดหรือไม่ นอกจากนี้การลดรูปสมการที่มีจำนวนหลายตัวแปรหรือมีจำนวนเทอมของสมการมาก จะใช้เวลาในการลดรูปสมการที่นานอีกด้วย

ตัวอย่างที่ 5.8 จงเขียนวงจรลอจิกของสมการ $A + AB = A$

<u>วิธีทำ</u>



ตัวอย่างที่ 5.9 จงลดรูปสมการต่อไปนี้ให้สมการมีจำนวนเทอมที่น้อยที่สุด โดยใช้พีชคณิตบูลีน พร้อมทั้งเขียนวงจรลอจิก

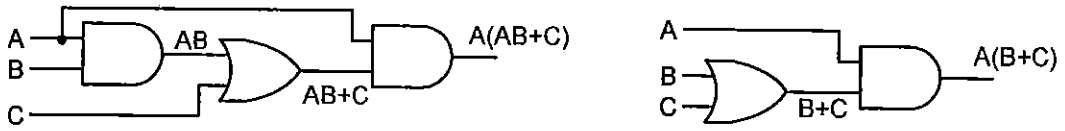
(1) $Y = A(AB + C)$

(2) $Y = \overline{AB} + \overline{C}$

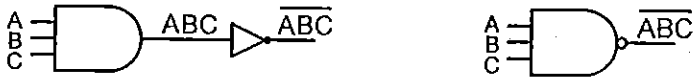
(3) $Y = AB(\overline{A}C + B)$

<u>วิธีทำ</u>

(1) $Y = A(AB + C) \;= AAB + AC \;= AB + AC \;= A(B + C)$



(2) $Y = \overline{AB} + \overline{C} \;= \overline{A} + \overline{B} + \overline{C} \;= \overline{ABC}$



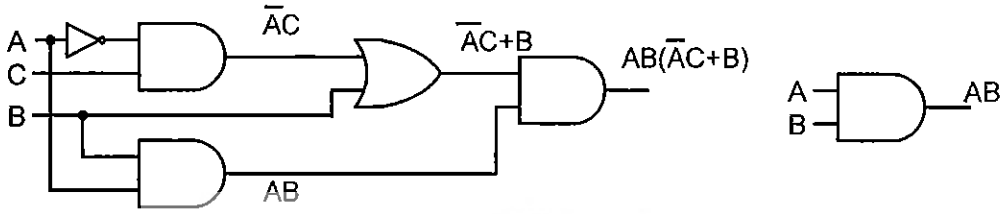
(3) $Y = AB(\overline{A}C + B)$

$= AB\overline{A}C + ABB$

$= (A\overline{A})BC + AB$

$= 0 \cdot BC + AB$

$= AB$



ตัวอย่างที่ 5.10 จงลดรูปสมการต่อไปนี้ โดยใช้พีชคณิตบูลีน

(1) $Y = \overline{\overline{A} + \overline{BC}}$

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

(3) $Y = A \cdot \overline{B} \cdot \overline{C} + A \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot C + \overline{A} \cdot \overline{B} \cdot C$

(4) $Y = \overline{A} \cdot B + A \cdot B + \overline{A} \cdot \overline{B}$

(5) $Y = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C}$

<u>วิธีทำ</u>

(1) $Y = \overline{\overline{A} + \overline{BC}}$

$= \overline{\overline{A}} \cdot \overline{\overline{BC}}$

$= ABC$

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

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

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

$= (A + B)CD$

(3) $Y = A \cdot \overline{B} \cdot \overline{C} + A \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot C + \overline{A} \cdot \overline{B} \cdot C$

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

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

หรือ $Y = A \cdot \overline{B} \cdot \overline{C} + A \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot C + \overline{A} \cdot \overline{B} \cdot C$

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

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

(4) $$Y = \overline{A} \cdot B + A \cdot B + \overline{A} \cdot \overline{B}$$

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

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

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

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

(5) $$Y = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C}$$

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

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

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

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

หรือ $$Y = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C}$$

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

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

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

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

หรือ $$Y = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C}$$

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

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

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

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

5.7 การออกแบบวงจรลอจิกโดยใช้พีชคณิตบูลีน

ขั้นตอนการออกแบบวงจรมีดังนี้

(1) เขียนสมการบูลีนจากตารางความจริง ไดอะแกรมเวลา หรือจากรูปวงจรลอจิก

(2) ลดรูปสมการให้มีจำนวนเทอมที่น้อยที่สุดหรือให้อยู่ในรูปสมการอย่างง่ายที่สุดโดยใช้พีชคณิตบูลีน

(3) เขียนวงจรลอจิกจากสมการที่ลดรูปแล้ว

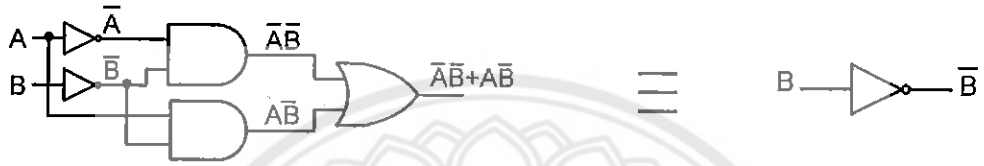
ตัวอย่างที่ 5.11 จงออกแบบวงจรลอจิกจากตารางความจริงที่กำหนดให้ โดยใช้พีชคณิตบูลีน

A	B	$f(A,B)$
0	0	1
0	1	0
1	0	1
1	1	0

<u>วิธีทำ</u>

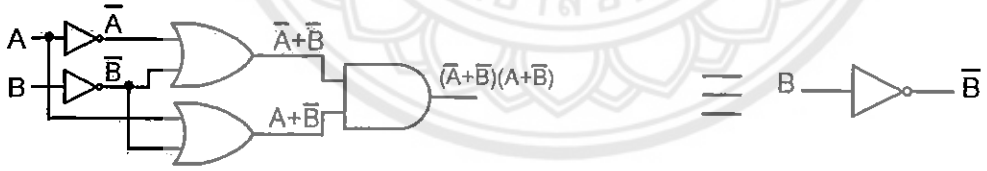
จากตารางความจริงเขียนสมการ SOP ได้ดังนี้

$$f(A,B)=\sum m(0,2) \quad =\overline{A}\overline{B}+A\overline{B} \quad =(\overline{A}+A)\overline{B} \quad =\overline{B}$$



จากตารางความจริงเขียนสมการ POS ได้ดังนี้

$$\begin{aligned} f(A,B) &= \prod M(1,3) \\ &= (A+\overline{B})(\overline{A}+\overline{B}) \\ &= A\cdot\overline{A}+A\cdot\overline{B}+\overline{B}\cdot\overline{A}+\overline{B}\cdot\overline{B} \\ &= A\overline{B}+\overline{A}\overline{B}+\overline{B} \\ &= \overline{B}(A+\overline{A}+1) \\ &= \overline{B} \end{aligned}$$



ตัวอย่างที่ 5.12 จงออกแบบวงจรลอจิกจากตารางความจริงที่กำหนดให้ โดยใช้พีชคณิตบูลีน

A	B	C	$f(A,B,C)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

<u>วิธีทำ</u>

จากตารางความจริงเขียนสมการ SOP ได้ดังนี้

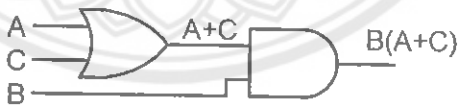
$$f(A,B,C) = \sum m(3,6,7)$$
$$= \overline{A}BC + AB\overline{C} + ABC$$
$$= B(\overline{A}C + A\overline{C} + AC)$$
$$= B(\overline{A}C + A(\overline{C} + C))$$
$$= B(\overline{A}C + A)$$
$$= B(C + A)$$

หรือ $f(A,B,C) = \sum m(3,6,7)$

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

หรือ $f(A,B,C) = \sum m(3,6,7)$

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



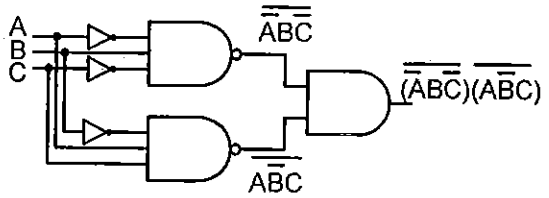
ตัวอย่างที่ 5.13 จงออกแบบวงจรลอจิกจากตารางความจริงที่กำหนดให้ โดยใช้พีชคณิตบูลีน

A	B	C	$f(A,B,C)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

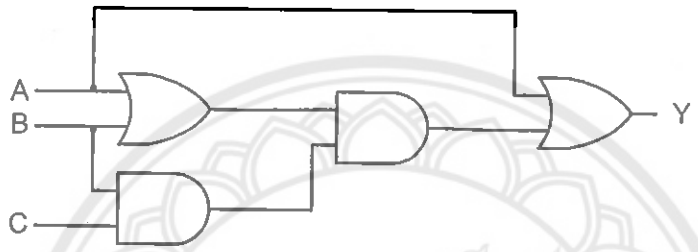
<u>วิธีทำ</u>

จากตารางความจริงเขียนสมการ POS ได้ดังนี้

$$f(A,B,C)=\prod M(2,5)$$
$$=(A+\overline{B}+C)(\overline{A}+B+\overline{C})$$
$$=(\overline{\overline{A}B\overline{C}})(\overline{A\overline{B}C})$$

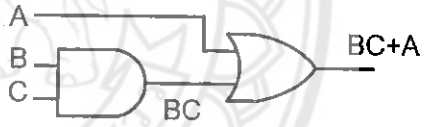


ตัวอย่างที่ 5.14 จงทำวงจรให้อยู่ในรูปอย่างง่าย โดยใช้พีชคณิตบูลีน

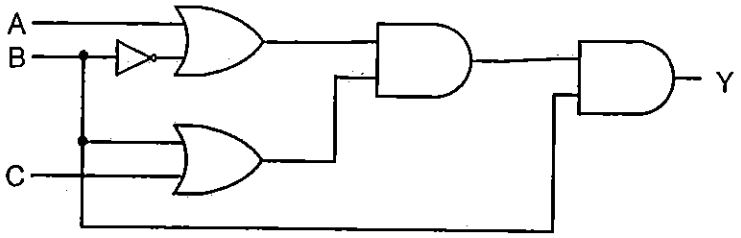


<u>วิธีทำ</u>

$$Y=BC(A+B)+A$$
$$=ABC+BBC+A$$
$$=ABC+BC+A$$
$$=BC(A+1)+A$$
$$=BC+A$$

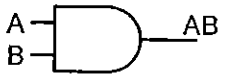


ตัวอย่างที่ 5.15 จงทำวงจรให้อยู่ในรูปอย่างง่าย โดยใช้พีชคณิตบูลีน



<u>วิธีทำ</u>

$$Y=(A+\overline{B})(B+C)B$$
$$=(AB+AC+\overline{B}B+\overline{B}C)B$$
$$=(AB+AC+\overline{B}C)B$$



$$= ABB + ABC + B\overline{B}C$$
$$= AB + ABC + 0$$
$$= AB(1 + C)$$
$$= AB$$

5.8 การออกแบบวงจรลอจิกโดยใช้แนนด์หรือนอร์เกต

สมการบูลีนมีการดำเนินการแอนด์ออร์นอต สามารถใช้เกตพื้นฐานมาสร้างเป็นวงจรลอจิกได้ สำหรับเกตอเนกประสงค์คือแนนด์เกตหรือนอร์เกต สามารถนำไปใช้สร้างแทนเกตชนิดอื่นได้ ดังที่กล่าวในเรื่องของเกตอเนกประสงค์ ดังนั้นการออกแบบวงจรลอจิกสามารถดัดแปลงเกตชนิดอื่นในวงจรให้เป็นวงจรที่มีแนนด์หรือนอร์เกตเพียงอย่างเดียวได้ โดยทำการแปลงสมการบูลีนให้อยู่ในรูปแนนด์หรือนอร์เกต มีขั้นตอนดังนี้

1. ถ้าต้องการสร้างวงจรที่ประกอบด้วยแนนด์เกตเพียงอย่างเดียวใช้สมการ SOP
   ถ้าต้องการสร้างวงจรที่ประกอบด้วยนอร์เกตเพียงอย่างเดียวใช้สมการ POS
2. ลดรูปสมการด้วยพีชคณิตบูลีน
3. ใส่ดับเบิ้ลบาร์ที่สมการบูลีน
4. ใช้ทฤษฎีดีมอร์แกนช่วยในการเปลี่ยนรูปสมการให้อยู่ในรูปของแนนด์หรือนอร์เกต

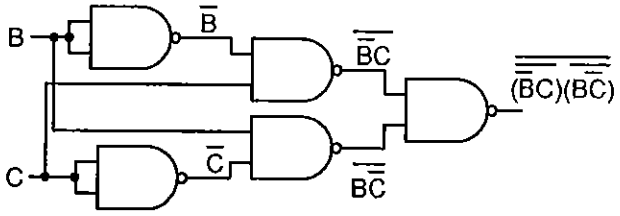
ตัวอย่างที่ 5.16 จงออกแบบวงจรจากตารางความจริงที่กำหนดให้ โดยใช้แนนด์เกตหรือนอร์เกตเพียงอย่างเดียว

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

<u>วิธีทำ</u>

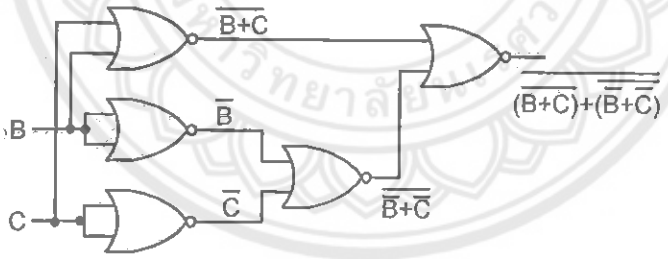
(1) วงจรที่ใช้แนนด์เกตเพียงอย่างเดียว

$$Y = \sum m(1,2,5,6)$$
$$= \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}C + AB\overline{C}$$
$$= \overline{B}(\overline{A}C + AC) + B(\overline{A}\overline{C} + A\overline{C})$$
$$= \overline{B}((\overline{A} + A)C) + B((\overline{A} + A)\overline{C})$$
$$= \overline{B}C + B\overline{C}$$
$$= \overline{\overline{\overline{B}C + B\overline{C}}}$$
$$= \overline{\overline{\overline{B}C} \cdot \overline{B\overline{C}}}$$



(2) วงจรที่ใช้นอร์เกตเพียงอย่างเดียว

$$Y = \prod M(0,3,4,7)$$
$$= (A + B + C)(A + \overline{B} + \overline{C})(\overline{A} + B + C)(\overline{A} + \overline{B} + \overline{C})$$
$$= (B + C + A\overline{A})(\overline{B} + \overline{C} + A\overline{A})$$
$$= (B + C)(\overline{B} + \overline{C})$$
$$= \overline{\overline{(B + C)(\overline{B} + \overline{C})}}$$
$$= \overline{\overline{(B + C)} + \overline{(\overline{B} + \overline{C})}}$$



แบบฝึกหัดท้ายบท

1. จากตารางความจริงที่กำหนดให้ จงเขียนสมการ SOP ในรูปอย่างง่าย

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

2. จงลดรูปสมการบูลีนที่กำหนดให้ต่อไปนี้

(1) $Y = A\overline{B}C + \overline{A}BC + \overline{A}\overline{B}C + \overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C}$

(2) $Y = \overline{A}\overline{B}\overline{C} + A\overline{B}C + \overline{A}BC + AB\overline{C}$

(3) $Y = AC\left[\overline{B} + B(B + \overline{C})\right]$

(4) $Y = DE\overline{F} + \overline{D}E\overline{F} + \overline{D}\overline{E}\overline{F}$

(5) $Y = X\overline{Y}Z + XY\overline{Z} + \overline{X}YZ + \overline{X}Y\overline{Z} + X\overline{Y}\overline{Z} + XYZ$

(6) $Y = \overline{W}\overline{X}\overline{Y}\overline{Z} + W\overline{X}YZ + W\overline{X}\overline{Y}Z + \overline{W}YZ + W\overline{X}\overline{Y}\overline{Z}$

(7) $Y = A\overline{B} + A\overline{B}\overline{C}D + CD + B\overline{C}D + ABCD$

(8) $f(A,B,C) = \sum m(1,2,3,5,7)$

(9) $f(A,B,C) = \sum m(0,2,5,7)$

(10) $f(A,B,C,D) = \sum m(0,2,4,6,15)$

(11) $f(A,B,C,D) = \sum m(3-7,9,11-15)$

3. จงทำสมการบูลีนต่อไปนี้ให้อยู่ในรูปอย่างง่าย

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

(2) $Y = (X+\overline{Y}+Z)(X+\overline{Y}+\overline{Z})(\overline{X}+\overline{Y}+Z)(\overline{X}+Y+Z)$

(3) $Y = (B+C+D)(A+B+\overline{C}+D)(\overline{A}+B+C+\overline{D})(A+\overline{B}+C+D)(\overline{A}+\overline{B}+C+D)$

(4) $Y = (X+\overline{Y})(W+\overline{Z})(\overline{X}+\overline{Y}+\overline{Z})(W+X+Y+Z)$

(5) $f(A,B,C) = \prod M(0,4,6)$

(6) $f(A,B,C) = \prod M(1,5-7)$

เฉลยแบบฝึกหัดท้ายบท

1. $Y = \overline{B} + C$

2. (1) $Y = \overline{B} + \overline{A}C$

   (2) $Y = BC + A$

   (3) $Y = AC$

   (4) $Y = E\overline{F} + \overline{D}\overline{F}$

   (5) $Y = X + Y$

   (6) $Y = \overline{XYZ} + W\overline{X}Z + \overline{W}YZ$

   (7) $Y = CD + BD + A\overline{B}$

   (8) $f(A,B,C) = A\overline{B} + C$

   (9) $f(A,B,C) = AC + \overline{A}\overline{C}$

   (10) $f(A,B,C,D) = \overline{A}\overline{D} + ABCD$

   (11) $f(A,B,C,D) = B + AD + CD$

3. (1) $Y = A(\overline{B} + C)$

   (2) $Y = (X + \overline{Y})(\overline{X} + Z)$

   (3) $Y = (C + D)(\overline{A} + B + C)(A + B + D)$

   (4) $Y = (W + X)(\overline{Y} + Z)(X + \overline{Y})(W + \overline{Z})$

   (5) $f(A,B,C) = (B + C)(\overline{A} + C)$

   (6) $f(A,B,C) = (B + \overline{C})(\overline{A} + \overline{B})$

บทที่ 6
แผนผังคาร์โนห์ (Karnaugh Map)

วัตถุประสงค์

1. สามารถเข้าใจหลักการและความสำคัญของแผนผังคาร์โนห์
2. สามารถออกแบบและวิเคราะห์วงจรโดยใช้แผนผังคาร์โนห์
3. เข้าใจหลักการและความสำคัญของเทอม don't care

6.1 บทนำ

การออกแบบวงจรลอจิกเพื่อทำการลดรูปวงจรให้อยู่ในรูปวงจรอย่างง่ายมีหลายวิธี สำหรับวิธีที่ได้กล่าวในบทที่ 5 เป็นการนำพีชคณิตบูลีนมาช่วยลดรูปสมการเพื่อให้ได้วงจรอย่างง่าย เป็นวิธีที่ขึ้นอยู่กับประสบการณ์และความสามารถของผู้ออกแบบวงจร ในการที่จะนำเอาทฤษฎีพีชคณิตบูลีนต่างๆ มาประยุกต์ใช้เพื่อลดรูปสมการได้มากน้อยเพียงใด แต่วิธีที่นิยมใช้กันทั่วไปคือแผนผังคาร์โนห์ (Karnaugh Map, KM) หรือที่เรียกกันว่า K-map เป็นการลดรูปสมการให้มีจำนวนเทอมน้อยที่สุด มีขั้นตอนที่แน่นอนและสะดวกกว่าวิธีการใช้ทฤษฎีพีชคณิตบูลีน แต่ KM มีข้อจำกัดคือสามารถใช้ลดรูปสมการที่มีตัวแปรตั้งแต่ 2 ถึง 6 ตัวแปรเท่านั้น ถ้าจำนวนตัวแปรมากกว่านี้ จะต้องใช้วิธีควินแม็กคลัสกี้ (Quine-McClusky, QM) มาช่วยลดรูปสมการ

KM มีลักษณะคล้ายกับตารางความจริง ตารางความจริงแสดงความสัมพันธ์ของตัวแปรอินพุตกับตัวแปรเอาต์พุตในลักษณะของแถวและคอลัมน์ แต่ KM แสดงความสัมพันธ์ของตัวแปรอินพุตกับตัวแปรเอาต์พุตในลักษณะเป็นช่อง (Cell) โดยพื้นที่แต่ละช่องแทนค่าเลขฐานสองของตัวแปรอินพุตที่อยู่ในรูปของ Minterm หรือ Maxterm การกำหนดค่าลอจิก 0 หรือ 1 ของแต่ละช่องเพื่อใช้แทนค่าเอาต์พุต เลขฐานสองในแต่ละช่องของ KM ที่อยู่ติดกันจะมีบิตแตกต่างกันเพียงบิตเดียวเท่านั้น ทำให้สามารถรวมพื้นที่ของช่องที่อยู่ติดกันได้ KM ที่มีตัวแปรอินพุตจำนวน n ตัวจะมีจำนวนช่องเท่ากับ 2^n ช่อง เช่น KM ที่มีตัวแปรอินพุต 3 ตัวมีจำนวนช่องเท่ากับ $2^3=8$ ช่อง หรือตัวแปรอินพุต 4 ตัวมีจำนวนช่องเท่ากับ $2^4=16$ ช่อง เป็นต้น

6.2 รูปแบบของ KM

KM ชนิด 2 ตัวแปร (A,B)

KM ชนิด 2 ตัวแปร แสดงดังรูปที่ 6.1 ในที่นี้กำหนดให้ตัวแปรอินพุตคือ A และ B ประกอบด้วยพื้นที่จำนวน 4 ช่อง ($2^2=4$) ในแต่ละช่องถูกกำหนดด้วยเลขฐานสอง เมื่อนำเลขฐานสองมาแปลงเป็นเลขฐานสิบจะได้เป็นหมายเลขประจำช่องนั้น ในแต่ละช่องของ KM จะมี

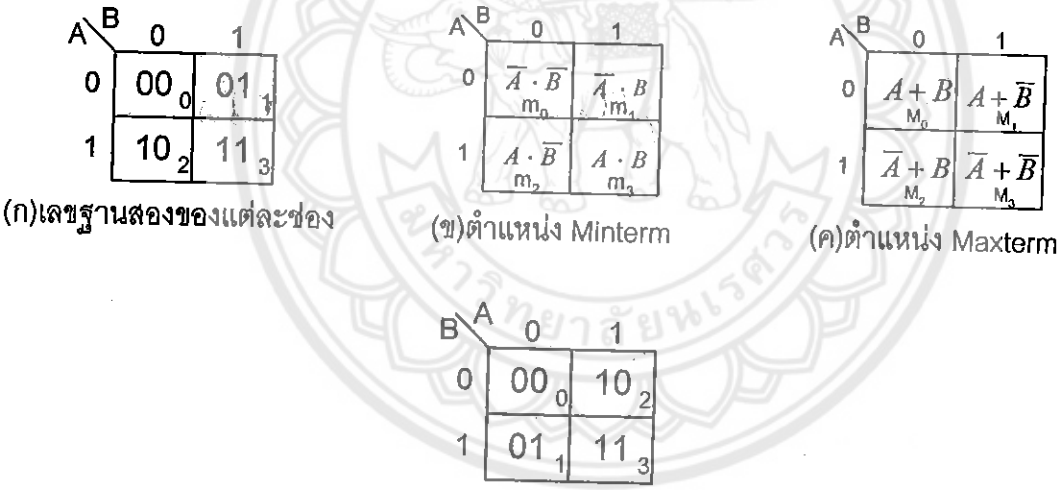
ค่าเลขฐานสองแตกต่างกันขึ้นอยู่กับการวางตำแหน่งของตัวแปร จากรูปที่ 6.1 (ก) และ (ง) จะเห็นว่าการวางตำแหน่งของตัวแปร A และ B แตกต่างกัน ทำให้หมายเลขประจำช่องแตกต่างกันด้วย ถ้าหมายเลขประจำช่องเป็นดังรูปที่ 6.1 (ก) ดังนั้นในแต่ละช่องสามารถแสดงพื้นที่ของ Minterm และ Maxterm ได้ดังรูปที่ 6.1 (ข) และ (ค) ตามลำดับ

ช่องที่ 0 กำหนดตัวแปร $A=0$ และ $B=0$ แสดงค่า Minterm คือ $\overline{A}\overline{B}$ (m_0) หรือ Maxterm คือ $A+B$ (M_0)

ช่องที่ 1 กำหนดตัวแปร $A=0$ และ $B=1$ แสดงค่า Minterm คือ $\overline{A}B$ (m_1) หรือ Maxterm คือ $A+\overline{B}$ (M_1)

ช่องที่ 2 กำหนดตัวแปร $A=1$ และ $B=0$ แสดงค่า Minterm คือ $A\overline{B}$ (m_2) หรือ Maxterm คือ $\overline{A}+B$ (M_2)

ช่องที่ 3 กำหนดตัวแปร $A=1$ และ $B=1$ แสดงค่า Minterm คือ AB (m_3) หรือ Maxterm คือ $\overline{A}+\overline{B}$ (M_3)



รูปที่ 6.1 KM ชนิด 2 ตัวแปร

KM ชนิด 3 ตัวแปร (A,B,C)

KM ชนิด 3 ตัวแปร แสดงดังรูปที่ 6.2 ในที่นี้กำหนดให้ตัวแปรอินพุตคือ A , B และ C ประกอบด้วยพื้นที่จำนวน 8 ช่อง (2^3=8) จาก KM จะเห็นว่าลำดับของเลขฐานสองจะเป็น 00, 01, 11, 10 เสมอ และตำแหน่งช่องที่ขอบด้านซ้ายกับขอบด้านขวาในแถวเดียวกันจะมีบิตแตกต่างกัน

เพียงบิตเดียว เมื่อนำขอบด้านซ้ายและขวามาชนกันจะได้เป็นรูปทรงกระบอก ดังนั้นจึงถือว่าเป็นช่องที่อยู่ติดกัน สามารถนำมารวมกลุ่มกันได้

A \ BC	00	01	11	10
0	000 (0)	001 (1)	011 (3)	010 (2)
1	100 (4)	101 (5)	111 (7)	110 (6)

(ก) เลขฐานสองในแต่ละช่อง

A \ BC	00	01	11	10
0	$\overline{A}\cdot\overline{B}\cdot\overline{C}$	$\overline{A}\cdot\overline{B}\cdot C$	$\overline{A}\cdot B\cdot C$	$\overline{A}\cdot B\cdot\overline{C}$
1	$A\cdot\overline{B}\cdot\overline{C}$	$A\cdot\overline{B}\cdot C$	$A\cdot B\cdot C$	$A\cdot B\cdot\overline{C}$

(ข) ตำแหน่ง Minterm

A \ BC	00	01	11	10
0	$A+B+C$	$A+B+\overline{C}$	$A+\overline{B}+\overline{C}$	$A+\overline{B}+C$
1	$\overline{A}+B+C$	$\overline{A}+B+\overline{C}$	$\overline{A}+\overline{B}+\overline{C}$	$\overline{A}+\overline{B}+C$

(ค) ตำแหน่ง Maxterm

C \ AB	00	01	11	10
0	000 (0)	010 (2)	110 (6)	100 (4)
1	001 (1)	011 (3)	111 (7)	101 (5)

C \ AB	0	1
00	000 (0)	001 (1)
01	010 (2)	011 (3)
11	110 (6)	111 (7)
10	100 (4)	101 (5)

A \ BC	0	1
00	000 (0)	100 (4)
01	001 (1)	101 (5)
11	011 (3)	111 (7)
10	010 (2)	110 (6)

(ง) เลขฐานสองของแต่ละช่องในรูปแบบอื่น

รูปที่ 6.2 KM ชนิด 3 ตัวแปร

KM ชนิด 4 ตัวแปร (A,B,C,D)

KM ชนิด 4 ตัวแปร แสดงดังรูปที่ 6.3 ในที่นี้กำหนดให้ตัวแปรอินพุตคือ A, B, C และ D ประกอบด้วยพื้นที่จำนวน 16 ช่อง (2^4=16) จาก KM จะเห็นว่าตำแหน่งช่องที่ขอบด้านซ้ายกับขอบด้านขวาในแถวเดียวกัน และช่องที่ขอบด้านบนกับขอบด้านล่างในคอลัมน์เดียวกัน จะมีบิตแตกต่างกันเพียงบิตเดียวเท่านั้น เมื่อนำขอบแนวนอนและแนวตั้งมาชนกันจะได้รูปทรงกระบอกที่มีหัวท้ายติดกันหรือคล้ายกับโดนัท ดังนั้นจึงถือว่าเป็นช่องที่อยู่ติดกัน สามารถรวมกลุ่มกันได้

AB \ CD	00	01	11	10
00	0000 0	0001 1	0011 3	0010 2
01	0100 4	0101 5	0111 7	0110 6
11	1100 12	1101 13	1111 15	1110 14
10	1000 8	1001 9	1011 11	1010 10

(ก) เลขฐานสองในแต่ละช่อง

AB \ CD	00	01	11	10
00	$\bar{A}\cdot\bar{B}\cdot\bar{C}\cdot\bar{D}$	$\bar{A}\cdot\bar{B}\cdot\bar{C}\cdot D$	$\bar{A}\cdot\bar{B}\cdot C\cdot D$	$\bar{A}\cdot\bar{B}\cdot C\cdot\bar{D}$
01	$\bar{A}\cdot B\cdot\bar{C}\cdot\bar{D}$	$\bar{A}\cdot B\cdot\bar{C}\cdot D$	$\bar{A}\cdot B\cdot C\cdot D$	$\bar{A}\cdot B\cdot C\cdot\bar{D}$
11	$A\cdot B\cdot\bar{C}\cdot\bar{D}$	$A\cdot B\cdot\bar{C}\cdot D$	$A\cdot B\cdot C\cdot D$	$A\cdot B\cdot C\cdot\bar{D}$
10	$A\cdot\bar{B}\cdot\bar{C}\cdot\bar{D}$	$A\cdot\bar{B}\cdot\bar{C}\cdot D$	$A\cdot\bar{B}\cdot C\cdot D$	$A\cdot\bar{B}\cdot C\cdot\bar{D}$

(ข) ตำแหน่ง Minterm

AB \ CD	00	01	11	10
00	$A+B+C+D$	$A+B+C+\bar{D}$	$A+B+\bar{C}+\bar{D}$	$A+B+\bar{C}+D$
01	$A+\bar{B}+C+D$	$A+\bar{B}+C+\bar{D}$	$A+\bar{B}+\bar{C}+\bar{D}$	$A+\bar{B}+\bar{C}+D$
11	$\bar{A}+\bar{B}+C+D$	$\bar{A}+\bar{B}+C+\bar{D}$	$\bar{A}+\bar{B}+\bar{C}+\bar{D}$	$\bar{A}+\bar{B}+\bar{C}+D$
10	$\bar{A}+B+C+D$	$\bar{A}+B+C+\bar{D}$	$\bar{A}+B+\bar{C}+\bar{D}$	$\bar{A}+B+\bar{C}+D$

(ค) ตำแหน่ง Maxterm

CD \ AB	00	01	11	10
00	0000 0	0100 4	1100 12	1000 8
01	0001 1	0101 5	1101 13	1001 9
11	0011 3	0111 7	1111 15	1011 11
10	0010 2	0110 6	1110 14	1010 10

(ง) ค่าเลขฐานสองในแต่ละช่องในรูปแบบอื่น

รูปที่ 6.3 KM ชนิด 4 ตัวแปร

KM ชนิด 5 ตัวแปร (A,B,C,D,E)

KM ชนิด 5 ตัวแปร แสดงดังรูปที่ 6.4 ในที่นี้กำหนดให้ตัวแปรอินพุตคือ A , B , C , D และ E ประกอบด้วยพื้นที่จำนวน 32 ช่อง (2^5=32) จากรูปที่ 6.4 (ก) เป็น KM ที่มีโครงสร้างแบบชั้น

(Layer structure) ที่ใช้ KM ชนิด 4 ตัวแปรจำนวน 2 แผนผัง แต่ละแผนผังแทนค่าลอจิกของตัวแปรตัวแรกในที่นี้คือตัวแปร A ถ้านำแผนผังทั้งสองมาวางซ้อนกัน ช่องที่อยู่ตรงกันจะมีบิตต่างกัน

BC \ DE	00	01	11	10
00	00000 0	00001 1	00011 3	00010 2
01	00100 4	00101 5	00111 7	00110 6
11	01100 12	01101 13	01111 15	01110 14
10	01000 8	01001 9	01011 11	01010 10

A = 0

BC \ DE	00	01	11	10
00	10000 16	10001 17	10011 19	10010 18
01	10100 20	10101 21	10111 23	10110 22
11	11100 28	11101 29	11111 31	11110 30
10	11000 24	11001 25	11011 27	11010 26

A = 1

(ก) เลขฐานสองในแต่ละช่อง

AB \ CDE	000	001	011	010	110	111	101	100
00	00000	00001	00011	00010	00110	00111	00101	00100
01	01000	01001	01011	01010	01110	01111	01101	01100
11	11000	11001	11011	11010	11110	11111	11101	11100
10	10000	10001	10011	10010	10110	10111	10101	10100

(ข) เลขฐานสองในแต่ละช่อง

DE \ ABC	000	001	011	010	110	111	101	100
00	00000	00100	01100	01000	11000	11100	10100	10000
01	00001	00101	01101	01001	11001	11101	10101	10001
11	00011	00111	01111	01011	11011	11111	10111	10011
10	00010	00110	01110	01010	11010	11110	10110	10010

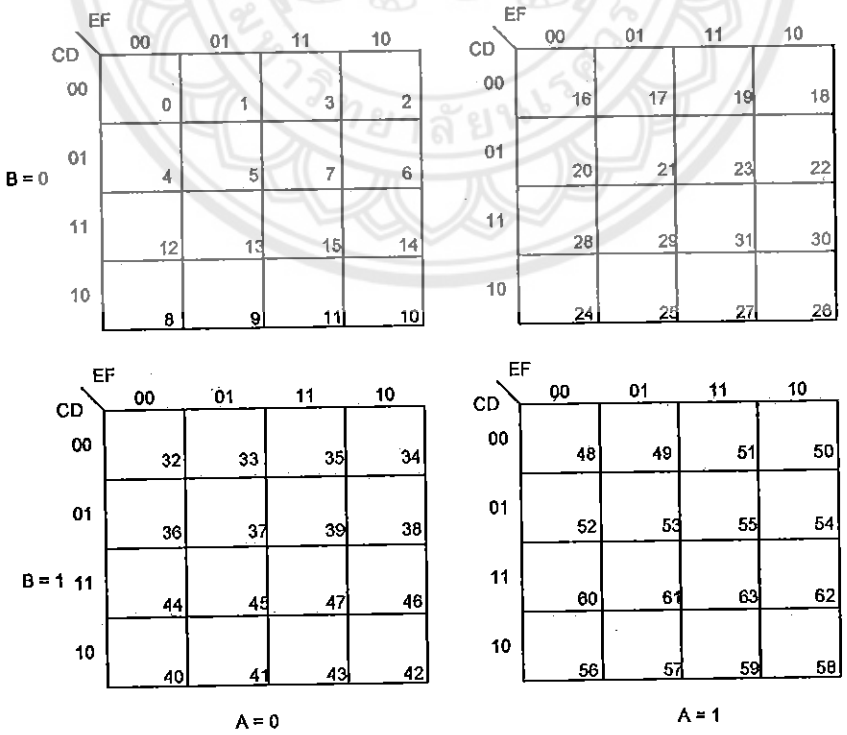
(ค) เลขฐานสองของแต่ละช่องในรูปแบบอื่น

รูปที่ 6.4 KM ชนิด 5 ตัวแปร

เพียงบิตเดียว ดังนั้นจึงถือว่าเป็นช่องที่อยู่ติดกันด้วยและสามารถรวมกลุ่มกันได้ เช่น เมื่อแผนผังทั้งสองวางซ้อนกัน ช่องหมายเลข 15 และ 31 จะอยู่ตรงกัน ดังนั้นถือว่าเป็นช่องที่อยู่ติดกัน สำหรับรูปที่ 6.4 (ข) เป็น KM ที่มีโครงสร้างแบบสะท้อน (Reflective structure) ที่ใช้ KM ชนิด 4 ตัวแปร จำนวน 2 แผนผังวางเรียงกัน แผนผังทั้งสองจะเสมือนเป็นภาพสะท้อนซึ่งกันและกัน โดยมีเส้นหนาทึบเป็นเส้นแบ่งตรงกลาง แต่ละช่องของภาพสะท้อนที่ตรงกัน จะมีบิตที่แตกต่างกันเพียงบิตเดียว จึงถือว่าเป็นช่องที่อยู่ติดกันด้วยและสามารถรวมกลุ่มกันได้ เช่น ช่องหมายเลข 9 และ 13 จะถือว่าสองช่องนี้ติดกัน เป็นต้น

KM ชนิด 6 ตัวแปร (A,B,C,D,E,F)

KM ชนิด 6 ตัวแปร แสดงดังรูปที่ 6.5 ในที่นี้กำหนดให้ตัวแปรอินพุตคือ A, B, C, D, E และ F ประกอบด้วยพื้นที่จำนวน 64 ช่อง (2^6=64) จะใช้ KM ชนิด 4 ตัวแปรจำนวน 4 แผนผัง แต่ละแผนผังจะแทนค่าลอจิกของตัวแปรสองตัวแรกในที่นี้คือตัวแปร A และ B แผนผังที่ติดกันในแนวนอนและแนวตั้ง ถ้านำมาวางซ้อนกัน ช่องที่อยู่ตรงกันจะมีบิตต่างกันเพียงบิตเดียว ดังนั้นจึงถือว่าเป็นช่องที่อยู่ติดกันและสามารถรวมกลุ่มกันได้ เช่น ช่องหมายเลข 5 (000101) และ 37 (010101) ของแผนผังในแนวตั้ง ถือว่าสองช่องนี้อยู่ติดกัน หรือช่องหมายเลข 37 (010101) และหมายเลข 53 (110101) ของแผนผังในแนวนอน ถือว่าสองช่องนี้อยู่ติดกัน สามารถรวมเป็นกลุ่มเดียวกันได้



รูปที่ 6.5 KM ชนิด 6 ตัวแปร

6.3 การเขียนตารางความจริงและสมการบูลีนลงใน KM

6.3.1 การเขียนตารางความจริงลงใน KM

การเขียนข้อมูลจากตารางความจริงลงใน KM สามารถเขียนได้โดยตรงเนื่องจากตารางความจริงและ KM มีลักษณะคล้ายกัน ตัวอย่างเช่น จากรูปที่ 6.6 จะเห็นว่าจากตารางความจริงเอาต์พุต $f(A,B,C)$ ที่มีค่าลอจิก 1 จะมีอินพุตที่แตกต่างกันจำนวน 3 กรณี และเอาต์พุต $f(A,B,C)$ ที่มีค่าลอจิก 0 จะมีอินพุตที่แตกต่างกันจำนวน 5 กรณี ให้นำค่าเอาต์พุตจากตารางความจริงเขียนลงใน KM โดยค่าตัวแปรอินพุตของตารางความจริงจะต้องตรงกันกับค่าเลขฐานสองในแต่ละช่องของ KM หรือหมายเลขในแต่ละแถวของตารางความจริงกับหมายเลขประจำช่องของ KM ต้องตรงกัน

A	B	C	f(A,B,C)
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

A \ BC	00	01	11	10
0	1	1	0	0
1	0	0	1	0

รูปที่ 6.6 ตัวอย่างการเขียนตารางความจริงลงใน KM

ตัวอย่างที่ 6.1 จงเขียนตารางความจริงที่กำหนดให้ ลงใน KM

A	B	f
0	0	1
0	1	0
1	0	1
1	1	0

<u>วิธีทำ</u>

A \ B	0	1
0	1	0
1	1	0

ตัวอย่างที่ 6.2 จงเขียนตารางความจริงที่กำหนดให้ ลงใน KM

	A	B	C	D	F(A,B,C,D)
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

<u>วิธีทำ</u>

AB \ CD	00	01	11	10
00	1 (0)	0 (1)	0 (3)	1 (2)
01	0 (4)	1 (5)	0 (7)	0 (6)
11	1 (12)	1 (13)	1 (15)	1 (14)
10	1 (8)	0 (9)	0 (11)	1 (10)

6.3.2 การเขียนสมการบูลีนลงใน KM

กรณีที่สมการอยู่ในรูป Canonical

ถ้าสมการ SOP อยู่ในรูปของ Canonical Sum ให้เขียน Minterm ที่มีอยู่ในสมการลงใน KM ที่ช่องของ Minterm นั้น โดยเขียนแทนด้วยลอจิก 1 ช่องที่เหลือแทนด้วยลอจิก 0

ถ้าสมการ POS อยู่ในรูปของ Canonical Product ให้เขียน Maxterm ที่มีอยู่ในสมการลงใน KM ที่ช่องของ Maxterm นั้น โดยเขียนแทนด้วยลอจิก 0 ช่องที่เหลือแทนด้วยลอจิก 1

ตัวอย่างที่ 6.3 จงเขียนสมการบูลีนต่อไปนี้ ลงใน KM

(1) $f(A,B,C) = \overline{A}BC + ABC + \overline{A}\overline{B}C$

(2) $f(A,B,C) = (\overline{A} + B + C)(A + B + C)(\overline{A} + \overline{B} + C)$

<u>วิธีทำ</u>

(1) $f(A,B,C) = \overline{A}BC + ABC + \overline{A}\overline{B}C$

011 111 001

$= m_3 + m_7 + m_1$

$= \sum m(1,3,7)$

A \ BC	00	01	11	10
0	0 (0)	1 (1)	1 (3)	0 (2)
1	0 (4)	0 (5)	1 (7)	0 (6)

(2) $f(A,B,C) = (\overline{A}+B+C)(A+B+C)(\overline{A}+\overline{B}+C)$

100 000 110

$= M_4 \cdot M_0 \cdot M_6$

$= \prod M(0,4,6)$

A \ BC	00	01	11	10
0	0 (0)	1 (1)	1 (3)	1 (2)
1	0 (4)	1 (5)	1 (7)	0 (6)

กรณีที่สมการไม่อยู่ในรูป Canonical

สามารถทำได้ 2 วิธี คือวิธีการแปลงสมการบูลีนให้อยู่ในรูป Canonical **แล้วเขียน** Minterm **หรือ** Maxterm ที่มีอยู่ในสมการลงใน KM และวิธีการโดยใช้แผนภาพเวนน์ (Venn diagram) **มีหลักการ**ดังนี้

เลข 0 แทนตัวแปรคอมพลีเมนท์ และ เลข 1 แทนตัวแปรปกติ

ตัวแปรที่นำมาแอนด์กัน คือพื้นที่ของตัวแปรทั้งหมดที่ซ้อนทับกัน

ตัวแปรที่นำมาออร์กัน คือพื้นที่ทั้งหมดของตัวแปรทั้งที่ซ้อนทับกันและไม่**ซ้อนทับกัน**

ตัวอย่างที่ 6.4 จงเขียนสมการบูลีนต่อไปนี้ ลงใน KM

(1) $Y = \overline{A}BC + AB + \overline{A}\overline{B}$

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

<u>วิธีทำ</u>

<u>วิธีที่ 1</u> **การแปลงสมการให้อยู่ในรูป** Canonical

(1) $Y = \overline{A}BC + AB + \overline{A}\overline{B}$

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

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

011 111 110 001 000

$= m_3 + m_7 + m_6 + m_1 + m_0$

$$= \sum m(0,1,3,6,7)$$

A \ BC	00	01	11	10
0	1 (0)	1 (1)	1 (3)	0 (2)
1	0 (4)	0 (5)	1 (7)	1 (6)

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

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

$$= \underset{100}{(\overline{A}+B+C)}\underset{000}{(A+B+C)}\underset{001}{(A+B+\overline{C})}\underset{110}{(\overline{A}+\overline{B}+C)}\underset{111}{(\overline{A}+\overline{B}+\overline{C})}$$

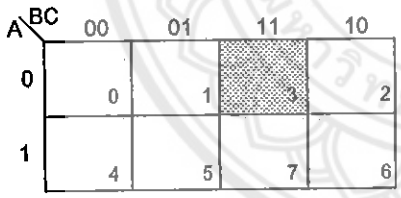
$$= M_4 \cdot M_0 \cdot M_1 \cdot M_6 \cdot M_7$$

$$= \prod M(0,1,4,6,7)$$

A \ BC	00	01	11	10
0	0 (0)	0 (1)	1 (3)	1 (2)
1	0 (4)	1 (5)	0 (7)	0 (6)

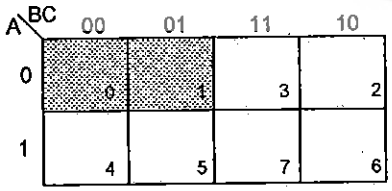
<u>วิธีที่ 2</u> การใช้แผนภาพเวนน์

(1) $Y = \overline{A}BC + AB + \overline{A}\overline{B}$

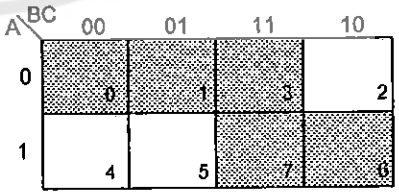


$\overline{A}BC$

AB



$\overline{A}\overline{B}$

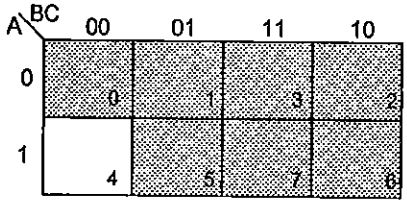


$Y = \overline{A}BC + AB + \overline{A}\overline{B}$

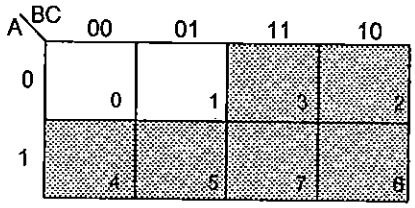
A \ BC	00	01	11	10
0	1 (0)	1 (1)	1 (3)	0 (2)
1	0 (4)	0 (5)	1 (7)	1 (6)

$Y = \overline{A}BC + AB + \overline{A}\overline{B}$

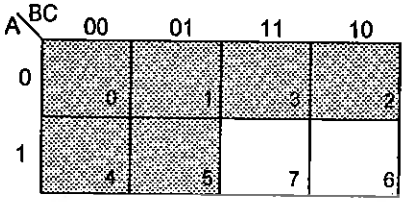
(2) $Y=(\overline{A}+B+C)(A+B)(\overline{A}+\overline{B})$



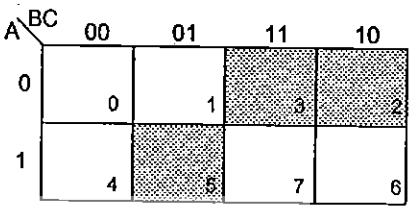
$\overline{A}+B+C$



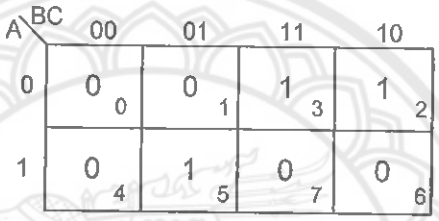
$A+B$



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



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



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

6.4 การรวมกลุ่มใน KM เพื่อหาคำตอบ

การรวมกลุ่มใน KM มีหลักการดังนี้

1. เลือกรวมกลุ่มช่องที่เป็นลอจิก 0 หรือ 1 อย่างใดอย่างหนึ่ง ถ้าต้องการคำตอบในรูปสมการ SOP ให้รวมช่องที่เป็นลอจิก 1 หรือถ้าต้องการคำตอบในรูปสมการ POS ให้รวมช่องที่เป็นลอจิก 0

2. จำนวนช่องต่อกลุ่ม ต้องมีจำนวนช่องให้มากที่สุดเท่าที่เป็นไปได้ เพราะจำนวนช่องต่อกลุ่มยิ่งมีจำนวนมาก จะทำให้จำนวนตัวแปรในเทอมนั้นน้อยลง ความสัมพันธ์ของตัวแปรจำนวน n ตัว ที่ได้จากการรวมกลุ่มของ KM แสดงดังตารางที่ 6.1 ตัวอย่างเช่น ถ้าใช้ KM ชนิด 3 ตัวแปร การรวมช่องที่อยู่ติดกันในแนวนอนหรือแนวตั้งจำนวน 4 ช่องสามารถเขียนแทนด้วยตัวแปร 1 ตัว หรือการรวมช่องที่อยู่ติดกันในแนวนอนหรือแนวตั้งจำนวน 2 ช่องสามารถเขียนแทนด้วยตัวแปร 2 ตัว

3. จำนวนกลุ่มที่ได้นั้น ต้องมีจำนวนน้อยที่สุดเท่าที่เป็นไปได้เพราะ 1 กลุ่มมีค่าเท่ากับ 1 เทอมของสมการบูลีน หรือจำนวนกลุ่มเท่ากับจำนวนเทอมของสมการบูลีน

4. ช่องที่รวมกลุ่มไปแล้วสามารถนำมารวมกลุ่มกับช่องอื่นได้อีกถ้าจำเป็น โดยที่กลุ่มนั้นจะต้องมีอย่างน้อย 1 ช่องที่ไม่เคยถูกรวมกลุ่มมาก่อน

ตารางที่ 6.1 ความสัมพันธ์ของการรวมช่องกับจำนวนแปรต่อกลุ่ม

จำนวนช่องต่อกลุ่ม (2^n)	1	2	4	8	16	32	64	...
จำนวนตัวแปรต่อกลุ่ม	$n-0$	$n-1$	$n-2$	$n-3$	$n-4$	$n-5$	$n-6$	...

จำนวนช่องต่อกลุ่ม (2^n)	จำนวนตัวแปรในแต่ละกลุ่ม (เทอม)			
	$n=2$	$n=3$	$n=4$	$n=5$
1	2	3	4	5
2	1	2	3	4
4	0	1	2	3
8		0	1	2
16			0	1
32				0

การหาคำตอบของสมการบูลีนจาก KM

การหาคำตอบสมการ SOP มีวิธีการดังนี้

1. รวมกลุ่มช่องที่มีลอจิก 1
2. ในกลุ่มเดียวกันต้องเขียนตัวแปรในรูปการคูณกัน (AND) การรวมช่องเป็น 1 กลุ่มจะได้เป็น 1 เทอมของสมการ การเขียนคำตอบในแต่ละกลุ่มจะนำตัวแปรที่มีค่าลอจิกเหมือนกันทุกช่องมาเขียน สำหรับตัวแปรที่มีค่าลอจิก 1 ให้เขียนในรูปของตัวแปรปกติ ($A,B,C,...$) และตัวแปรที่มีค่าลอจิก 0 ให้เขียนในรูปของตัวแปรคอมพลีเมนท์ ($\overline{A},\overline{B},\overline{C},...$)
3. นำแต่ละเทอมที่ได้ในข้อ 2 มาบวกกัน (OR)

การหาคำตอบสมการ POS มีวิธีการดังนี้

1. รวมกลุ่มช่องที่มีลอจิก 0
2. ในกลุ่มเดียวกันต้องเขียนตัวแปรในรูปการบวกกัน (OR) การเขียนคำตอบในแต่ละกลุ่มจะนำตัวแปรที่มีค่าลอจิกเหมือนกันทุกช่องมาเขียน สำหรับตัวแปรที่มีค่าลอจิก 0 ให้เขียนในรูปของตัวแปรปกติ ($A,B,C,...$) และตัวแปรที่มีค่าลอจิก 1 จะเขียนในรูปของตัวแปรคอมพลีเมนท์ ($\overline{A},\overline{B},\overline{C},...$)
3. นำแต่ละเทอมที่ได้ในข้อ 1.2 มาคูณกัน (AND)

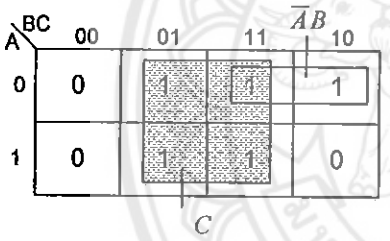
จากที่กล่าวข้างต้น สามารถสรุปการหาคำตอบของสมการจาก KM ได้ดังนี้

การหาคำตอบของสมการในรูป	SOP	POS
การเลือกรวมกลุ่มที่เป็นลอจิก 0 หรือ 1	1	0
ตัวแปรที่ได้จากการรวมกลุ่ม ในกลุ่มเดียวกันจะเขียนตัวแปรอยู่ในรูป	AND	OR
การนำแต่ละกลุ่ม(เทอม) มารวมกัน	OR	AND
การรวมกลุ่มทุกช่องใน KM จะให้ผลลัพธ์เป็น	1	0

ตัวอย่างที่ 6.5 จงลดรูปสมการจาก KM ที่กำหนดให้ในรูปสมการ SOP และ POS

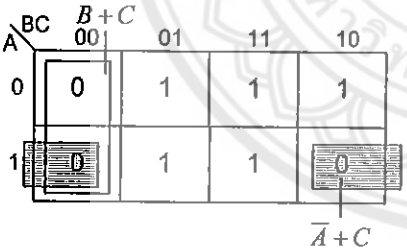
A \ BC	00	01	11	10
0	0 (0)	1 (1)	1 (3)	1 (2)
1	0 (4)	1 (5)	1 (7)	0 (6)

<u>วิธีทำ</u>



คำตอบสมการ SOP

$Y = \overline{A}B + C$



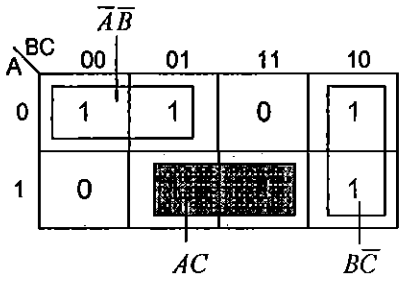
คำตอบสมการ POS

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

ตัวอย่างที่ 6.6 จงลดรูปสมการจาก KM ที่กำหนดให้ในรูปสมการ SOP และ POS

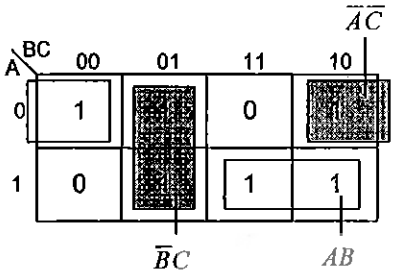
A \ BC	00	01	11	10
0	1	1	0	1
1	0	1	1	1

<u>วิธีทำ</u>



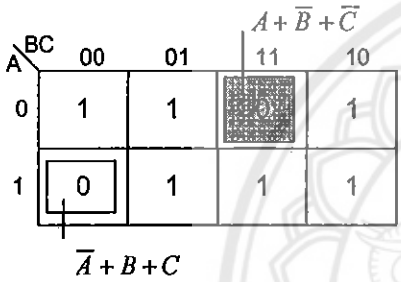
คำตอบสมการ SOP (แบบที่ 1)

$$Y = \overline{A}\overline{B} + AC + B\overline{C}$$



คำตอบสมการ SOP (แบบที่ 2)

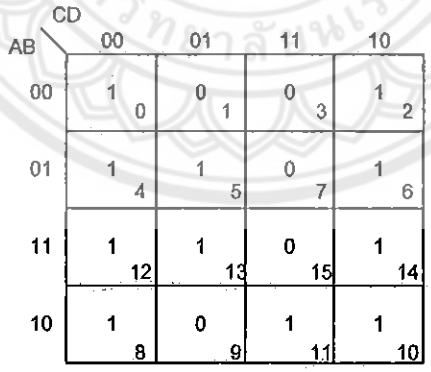
$$Y = \overline{A}\overline{C} + AB + \overline{B}C$$



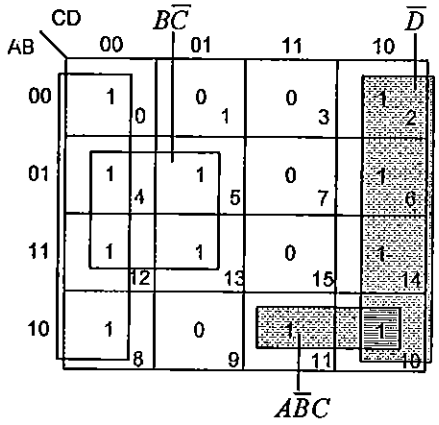
คำตอบสมการ POS

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

ตัวอย่างที่ 6.7 จงลดรูปสมการ SOP และ POS จาก KM ที่กำหนดให้

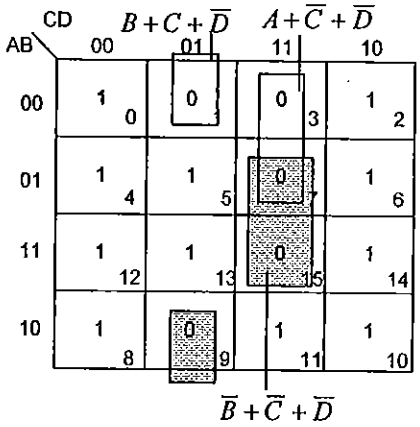


<u>**วิธีทำ**</u>



คำตอบสมการ SOP

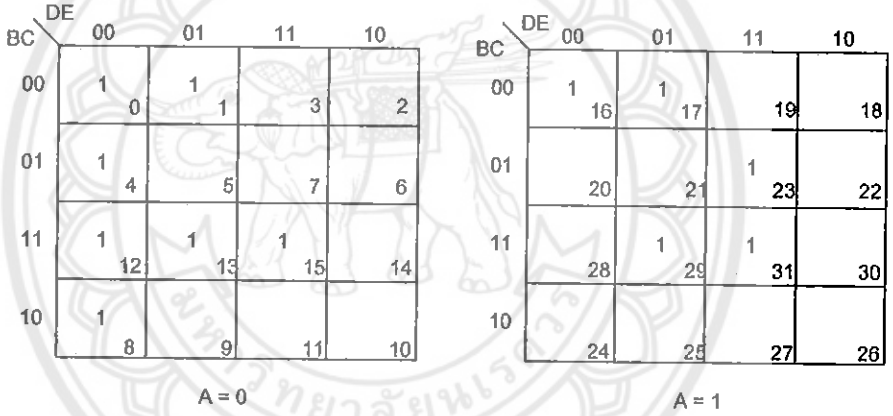
$$Y = A\overline{B}C + B\overline{C} + \overline{D}$$



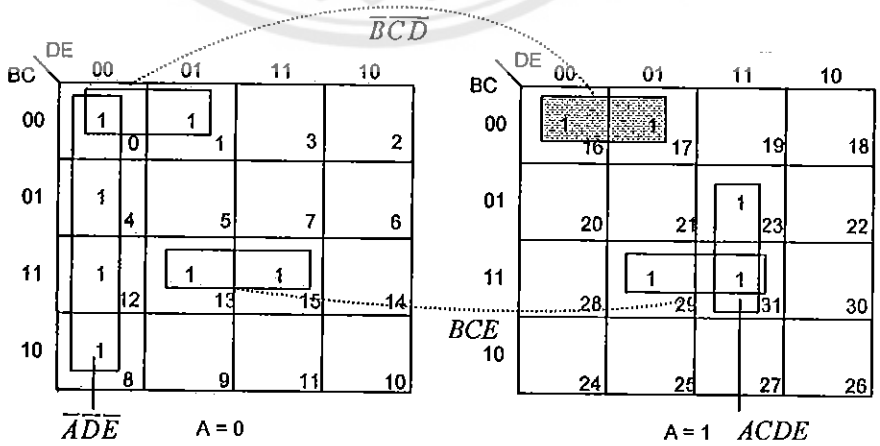
คำตอบสมการ POS

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

ตัวอย่างที่ 6.8 จงลดรูปสมการ SOP จาก KM ที่กำหนดให้



<u>วิธีทำ</u>

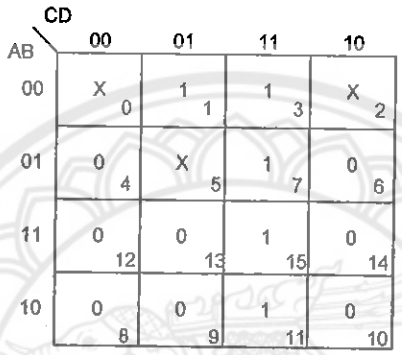


คำตอบสมการ SOP $\quad Y = ACDE + \overline{A}\overline{D}\overline{E} + \overline{B}\overline{C}\overline{D} + BCE$

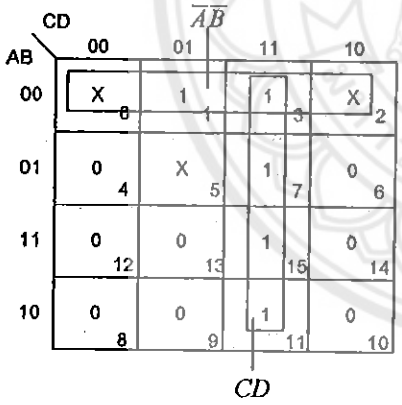
6.5 Don't care term

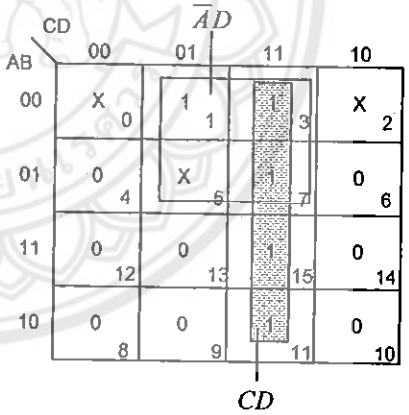
เทอม don't care หมายถึงเทอมที่มีค่าเป็นลอจิก 0 หรือ 1 ก็ได้ เนื่องจากในการออกแบบวงจรลอจิก บางครั้งค่าลอจิกของตัวแปรอินพุตบางกรณีจะไม่ถูกนำมาพิจารณา ดังนั้นเอาต์พุตสามารถกำหนดให้เป็นลอจิก 0 หรือ 1 ได้ สัญลักษณ์ของเทอม don't care จะใช้ตัวแปร d, X หรือเครื่องหมายกากบาท เมื่อเทอม Don't care ปรากฏใน KM สามารถกำหนดเทอม Don't care เป็นลอจิก 0 หรือ 1 ก็ได้ที่ทำให้คำตอบของสมการมีจำนวนเทอมที่น้อยที่สุด

ตัวอย่างที่ 6.9 จงลดรูปสมการ SOP จาก KM ที่กำหนดให้



<u>วิธีทำ</u>





คำตอบสมการ SOP (แบบที่ 1)

$$Y = \overline{A}\overline{B} + CD$$

คำตอบสมการ SOP (แบบที่ 2)

$$Y = \overline{A}D + CD$$

6.6 การออกแบบวงจรลอจิกโดยใช้ KM

ขั้นตอนมีดังนี้

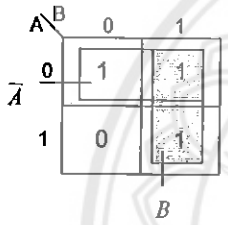
1. นิยามปัญหาของวงจรและกำหนดตัวแปรอินพุต/เอาต์พุตที่เกี่ยวข้อง

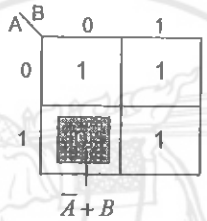
2. เขียนความสัมพันธ์ของตัวแปรอินพุตและเอาต์พุตในรูปตารางความจริง
3. นำข้อมูลจากตารางความจริงเขียนลงใน KM
4. ลดรูปสมการให้อยู่ในรูปอย่างง่ายตามกฎการรวมกลุ่มของ KM
5. จัดรูปสมการให้อยู่ในรูปแบบวงจรที่ต้องการ โดยใช้ทฤษฎีดีมอร์แกน เช่น วงจรต้องการใช้แนนด์หรือนอร์เกต เป็นต้น
6. เขียนวงจรลอจิก

ตัวอย่างที่ 6.10 จงลดรูปสมการ $Y = AB + \overline{A}B + \overline{A}\overline{B}$ โดยใช้ KM พร้อมทั้งเขียนวงจรลอจิก

<u>วิธีทำ</u>

$$Y = AB + \overline{A}B + \overline{A}\overline{B}$$
11 01 00





คำตอบสมการ SOP

$Y = \overline{A} + B$

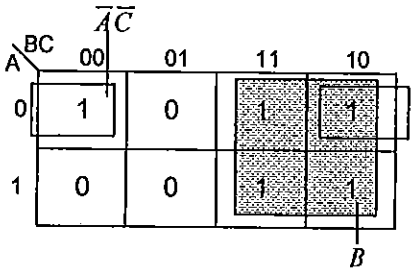
คำตอบสมการ POS

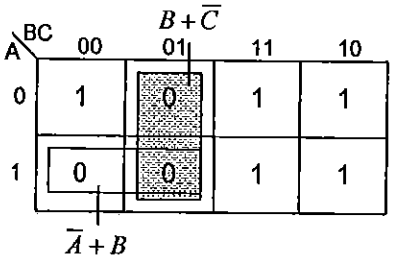
$Y = \overline{A} + B$

ตัวอย่างที่ 6.11 จงลดรูปสมการ $Y = AB + BC + \overline{A}\overline{C}$ โดยใช้ KM พร้อมทั้งเขียนวงจรลอจิก

<u>วิธีทำ</u>

$$Y = AB + BC + \overline{A}\overline{C}$$
11 11 00



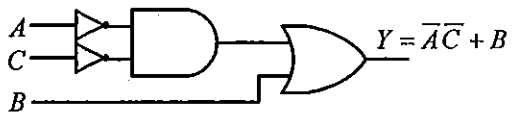


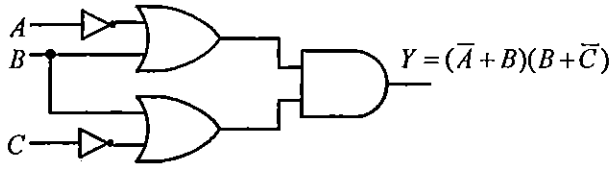
คำตอบสมการ SOP

$Y = \overline{A}\overline{C} + B$

คำตอบสมการ POS

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



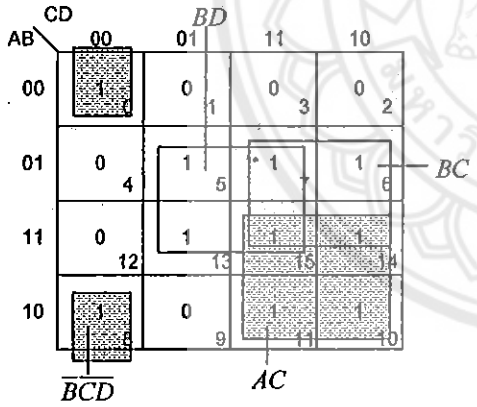


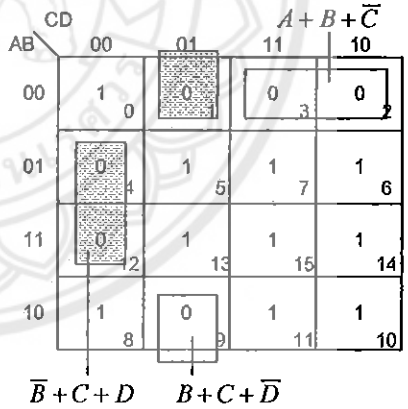
ตัวอย่างที่ 6.12 จงลดรูปสมการต่อไปนี้ ให้อยู่ในรูปสมการ SOP และ POS อย่างง่ายโดยใช้ KM

$$f(A,B,C,D)=(\overline{A}+\overline{B}+C+D)(A+\overline{B}+C+D)(A+B+C+\overline{D})$$
$$(A+B+\overline{C}+\overline{D})(\overline{A}+B+C+\overline{D})(A+B+\overline{C}+D)$$

<u>วิธีทำ</u>

$$f(A,B,C,D)=\underset{1100}{(\overline{A}+\overline{B}+C+D)}\underset{0100}{(A+\overline{B}+C+D)}\underset{0001}{(A+B+C+\overline{D})}$$
$$\underset{0011}{(A+B+\overline{C}+\overline{D})}\underset{1001}{(\overline{A}+B+C+\overline{D})}\underset{0010}{(A+B+\overline{C}+D)}$$
$$=M_{12}\cdot M_4\cdot M_1\cdot M_3\cdot M_9\cdot M_2$$
$$=\prod M(1,2,3,4,9,12)$$
$$=\overline{\sum m(1,2,3,4,9,12)}$$
$$=\sum m(0,5-8,10,11,13-15)$$





คำตอบสมการ SOP

$$f(A,B,C,D)=BD+BC+AC+\overline{B}\overline{C}\overline{D}$$

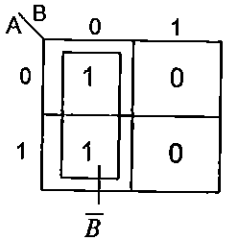
คำตอบสมการ POS

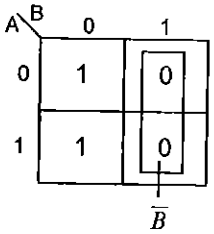
$$f(A,B,C,D)=(A+B+\overline{C})(\overline{B}+C+D)(B+C+\overline{D})$$

ตัวอย่างที่ 6.13 จากตารางความจริงต่อไปนี้ จงหาสมการอย่างง่ายในรูป SOP และ POS โดยใช้ KM พร้อมทั้งเขียนวงจรลอจิก

A	B	Y
0	0	1
0	1	0
1	0	1
1	1	0

วิธีทำ



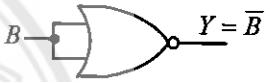


คำตอบสมการ SOP

$Y = \overline{B}$

คำตอบสมการ POS

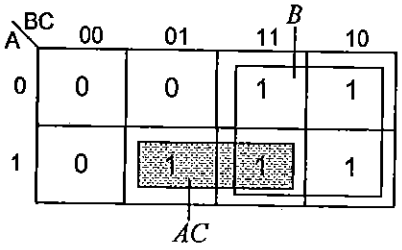
$Y = \overline{B}$

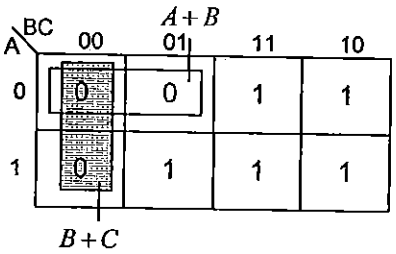


ตัวอย่างที่ 6.14 จงออกแบบวงจรลอจิกจากตารางความจริงที่กำหนด โดยใช้ KM

A	B	C	f(A, B, C)
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

วิธีทำ



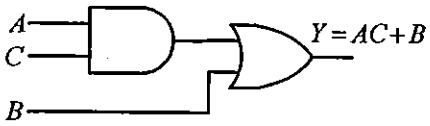


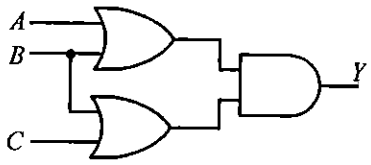
คำตอบสมการ SOP

$$Y = AC + B$$

คำตอบสมการ POS

$$Y = (B + C)(A + B)$$

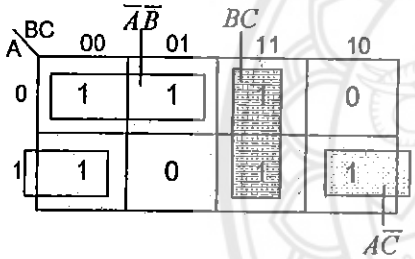




ตัวอย่างที่ 6.15 จงลดรูปสมการจากตารางความจริงที่กำหนดให้ โดยใช้ KM

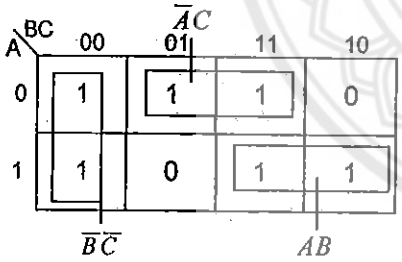
A	B	C	Y(A, B, C)
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

<u>วิธีทำ</u>



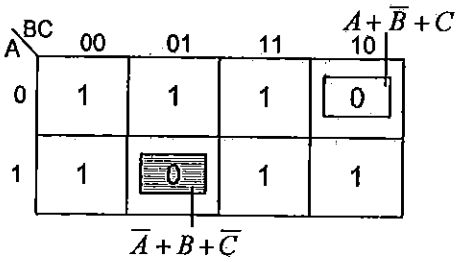
คำตอบสมการ SOP (แบบที่ 1)

$$Y = \overline{A}\overline{B} + BC + A\overline{C}$$



คำตอบสมการ SOP (แบบที่ 2)

$$Y = \overline{A}C + \overline{B}\overline{C} + AB$$



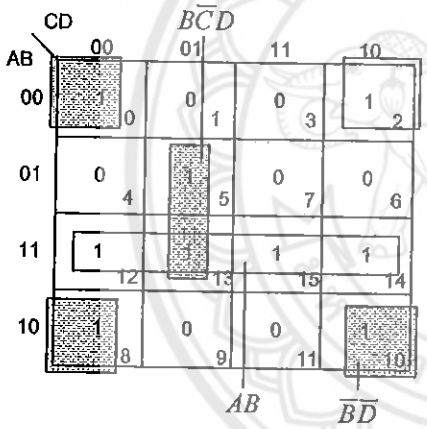
คำตอบสมการ POS

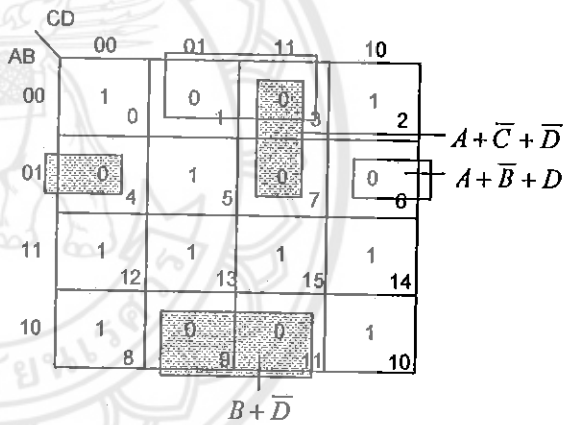
$$Y = (A + \overline{B} + C)(\overline{A} + B + \overline{C})$$

ตัวอย่างที่ 6.16 จงลดรูปสมการจากตารางความจริงที่กำหนด โดยใช้ KM

ลำดับที่	A	B	C	D	f(A,B,C,D)
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

<u>วิธีทำ</u>





คำตอบสมการ SOP

$Y = AB + \overline{B}\overline{D} + B\overline{C}D$

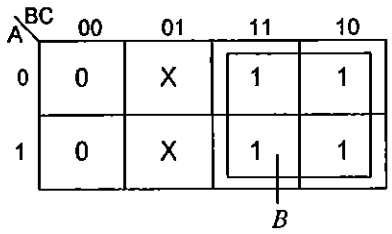
คำตอบสมการ POS

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

ตัวอย่างที่ 6.17 จงลดรูปสมการจากตารางความจริงที่กำหนดให้ โดยใช้ KM

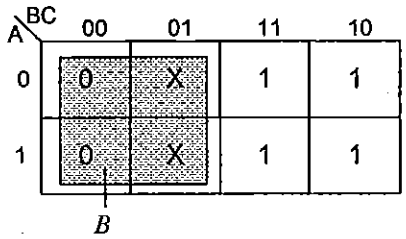
A	B	C	f(A,B,C)
0	0	0	0
0	0	1	X
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	X
1	1	0	1
1	1	1	1

วิธีทำ



คำตอบสมการ SOP

$$Y = B$$



คำตอบสมการ POS

$$Y = B$$

ตัวอย่างที่ 6.18 จงลดรูปสมการต่อไปนี้โดยใช้ KM

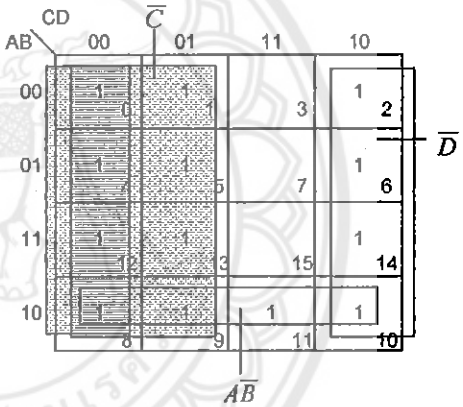
$$f(A,B,C,D) = \overline{C}(\overline{A}\overline{B}\overline{D} + D) + A\overline{B}C + \overline{D}$$

วิธีทำ

$$f(A,B,C,D) = \overline{C}(\overline{A}\overline{B}\overline{D} + D) + A\overline{B}C + \overline{D}$$

$$= \overline{A}\overline{B}\overline{C}\overline{D} + \overline{C}D + A\overline{B}C + \overline{D}$$

0000 01 101 0

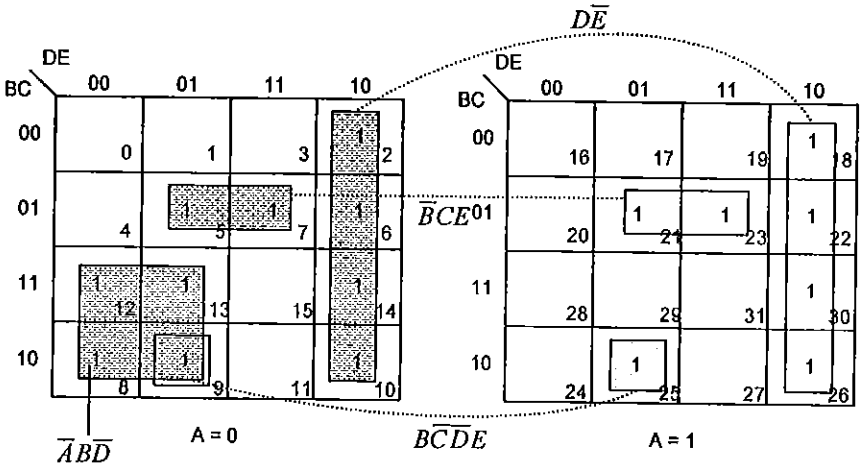


$$f(A,B,C,D) = A\overline{B} + \overline{C} + \overline{D}$$

ตัวอย่างที่ 6.19 จงลดรูปสมการต่อไปนี้ โดยใช้ KM

$$f(A,B,C,D,E) = \sum m(2,5-10,12-14,18,21-23,25,26,30)$$

วิธีทำ

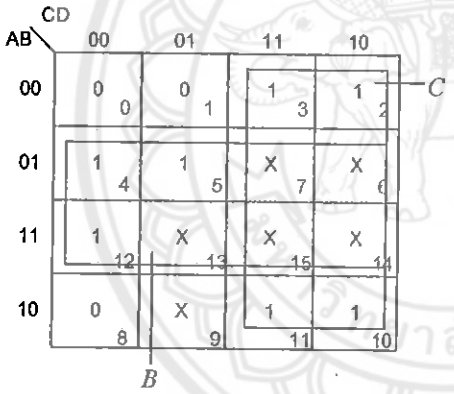


$$f(A,B,C,D,E) = D\overline{E} + \overline{A}B\overline{D} + \overline{B}CE + B\overline{C}\overline{D}E$$

ตัวอย่างที่ 6.20 จงลดรูปสมการต่อไปนี้ ให้อยู่ในรูปสมการ SOP อย่างง่าย โดยใช้ KM

$$f(A,B,C,D) = \sum m(2-5,10-12) + dc(6,7,9,13-15)$$

<u>วิธีทำ</u>



$$f(A,B,C,D) = B + C$$

แบบฝึกหัดท้ายบท

1. จากตารางความจริงที่กำหนดให้ จงเขียนสมการ SOP ในรูปอย่างง่าย โดยใช้KM

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

2. จงลดรูปสมการบูลีนที่กำหนดให้ต่อไปนี้ โดยใช้ KM

(1) $Y = A\overline{B}C + \overline{A}BC + \overline{A}\overline{B}C + \overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C}$

(2) $Y = A + BC$

(3) $Y = AC\left[\overline{B} + B(B + \overline{C})\right]$

(4) $Y = DE\overline{F} + \overline{D}E\overline{F} + \overline{D}\overline{E}\overline{F}$

(5) $Y = X\overline{Y}Z + XY\overline{Z} + \overline{X}YZ + \overline{X}Y\overline{Z} + X\overline{Y}\overline{Z} + XYZ$

(6) $Y = \overline{W}\overline{X}\overline{Y}\overline{Z} + W\overline{X}YZ + W\overline{X}\overline{Y}Z + \overline{W}YZ + W\overline{X}\overline{Y}\overline{Z}$

(7) $Y = A\overline{B} + A\overline{B}\overline{C}D + CD + B\overline{C}D + ABCD$

(8) $f(A,B,C) = \sum m(1,2,3,5,7)$

(9) $f(A,B,C) = \sum m(0,2,5,7)$

(10) $f(A,B,C,D) = \sum m(0,2,4,6,15)$

(11) $f(A,B,C,D) = \sum m(3-7,9,11-15)$

3. จงทำสมการบูลีนต่อไปนี้ให้อยู่ในรูปอย่างง่าย โดยใช้KM

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

(2) $Y = (X+\overline{Y}+Z)(X+\overline{Y}+\overline{Z})(\overline{X}+\overline{Y}+Z)(\overline{X}+Y+Z)$

(3) $Y = (B+C+D)(A+B+\overline{C}+D)(\overline{A}+B+C+\overline{D})(A+\overline{B}+C+D)(\overline{A}+\overline{B}+C+D)$

(4) $Y = (X+\overline{Y})(W+\overline{Z})(\overline{X}+\overline{Y}+\overline{Z})(W+X+Y+Z)$

(5) $f(A,B,C) = \prod M(0,4,6)$

(6) $f(A,B,C) = \prod M(1,5-7)$

4. จงเขียนสมการ SOP ในรูปอย่างง่าย จากตารางความจริงที่กำหนดให้ต่อไปนี้

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

5. จงลดรูปสมการต่อไปนี้ โดยใช้KM

(1) $$Y = \overline{A}\overline{B}\overline{C}D\overline{E} + \overline{A}\overline{B}C\overline{D}\overline{E} + \overline{A}BC\overline{D}\overline{E} + \overline{A}B\overline{C}\overline{D}\overline{E} + \overline{A}\overline{B}\overline{C}\overline{D}E + \overline{A}BC\overline{D}E + \overline{A}BCDE + A\overline{B}\overline{C}\overline{D}\overline{E} + A\overline{B}\overline{C}\overline{D}E + A\overline{B}CDE + ABCDE + A\overline{B}CD\overline{E}$$

(2) $f(A,B,C,D,E) = \sum m(2,5-10,12-14,18,21-23,25,26,30)$

(3) $f(A,B,C,D,E) = \sum m(2,3,10,11,16,17,22)$

(4) $f(A,B,C,D,E) = \sum m(2,3,6,7,10,11,14,15,17,25)$

(5) $f(A,B,C,D,E) = \sum m(5,7,13,15,26,27,30,31)$

เฉลยแบบฝึกหัดท้ายบท

1. $Y = \overline{B} + C$

2. (1) $Y = \overline{B} + \overline{A}C$

   (2) $Y = BC + A$

   (3) $Y = AC$

   (4) $Y = E\overline{F} + \overline{D}\overline{F}$

   (5) $Y = X + Y$

   (6) $Y = \overline{XYZ} + W\overline{X}Z + \overline{W}YZ$

   (7) $Y = CD + BD + A\overline{B}$

   (8) $f(A,B,C) = A\overline{B} + C$

   (9) $f(A,B,C) = AC + \overline{A}\overline{C}$

   (10) $f(A,B,C,D) = \overline{A}\overline{D} + ABCD$

   (11) $f(A,B,C,D) = B + AD + CD$

3. (1) $Y = A(\overline{B} + C)$

   (2) $Y = (X + \overline{Y})(\overline{X} + Z)$

   (3) $Y = (C + D)(\overline{A} + B + C)(A + B + D)$

   (4) $Y = (W + X)(\overline{Y} + Z)(X + \overline{Y})(W + \overline{Z})$

   (5) $f(A,B,C) = (B + C)(\overline{A} + C)$

   (6) $f(A,B,C) = (B + \overline{C})(\overline{A} + \overline{B})$

4. $Y = \sum m(7,8,9) + dc(10-15) = A + BCD$

   ถ้าไม่ใช้ Don't care term จะได้ $Y = A\overline{B}\overline{C} + \overline{A}BCD$

5. (1) $Y = \overline{A}\overline{D}\overline{E} + \overline{B}\overline{C}\overline{D} + BCE + ACDE$

   (2) $f(A,B,C,D,E) = D\overline{E} + \overline{B}CE + \overline{A}B\overline{D} + B\overline{C}\overline{D}E$

   (3) $f(A,B,C,D,E) = \overline{A}\overline{C}D + A\overline{B}\overline{C}\overline{D} + A\overline{B}CD\overline{E}$

   (4) $f(A,B,C,D,E) = \overline{A}D + A\overline{C}\overline{D}E$

   (5) $f(A,B,C,D,E) = \overline{A}CE + ABD$

บทที่ 7
วงจรคำนวณคณิตศาสตร์และลอจิก
(Arithmetic Logic Unit)

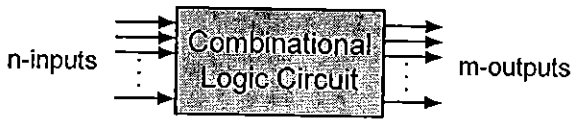
วัตถุประสงค์

1. เข้าใจหลักการทำงานและสามารถออกแบบวงจรบวกเลขฐานสองแบบครึ่งและแบบเต็มได้
2. เข้าใจหลักการทำงานและสามารถออกแบบวงจรลบเลขฐานสองแบบครึ่งและแบบเต็มได้
3. สามารถนำวงจรบวกเลขฐานสองแบบครึ่งและแบบเต็มไปสร้างเป็นวงจรบวกแบบขนาน วงจรบวกเลข BCD และวงจรการคูณได้
4. เข้าใจหลักการทำงานและสามารถออกแบบวงจรเปรียบเทียบเลขฐานสองได้

7.1 บทนำ

วงจรดิจิตอลแบ่งออกเป็น 2 ประเภทใหญ่ๆ คือ วงจรคอมบิเนชันหรือวงจรลอจิกเชิงจัดหมู่ (Combinational logic circuit) และวงจรซีเควนเชียลหรือวงจรลอจิกเชิงลำดับ (Sequential logic circuit) วงจรคอมบิเนชันประกอบด้วยตัวแปรอินพุต ตัวแปรเอาต์พุตและเกตที่นำมาต่อกันเพื่อให้ได้เอาต์พุตตามที่ต้องการ วงจรประเภทนี้จะไม่มีตัวเก็บข้อมูล (Storage) สามารถเขียนบล็อกไดอะแกรมของวงจรคอมบิเนชันดังรูปที่ 7.1 จะเห็นว่าบล็อกไดอะแกรมประกอบด้วยตัวแปรอินพุตจำนวน n ตัว ที่ให้ตัวแปรเอาต์พุตจำนวน m ตัว ตัวแปรอินพุตแต่ละตัวมีค่าเป็นลอจิก 0 หรือ 1 ถ้าตัวแปรอินพุตมีจำนวน n ตัวจะให้ค่าลอจิก 0 หรือ 1 ที่เป็นไปได้จำนวน 2^n กรณี และในแต่ละกรณีจะมีเอาต์พุต 1 ค่า ดังนั้นวงจรคอมบิเนชันสามารถเขียนความสัมพันธ์ของตัวแปรอินพุตและเอาต์พุตในรูปของตารางความจริงได้ นอกจากนี้สามารถเขียนความสัมพันธ์ในรูปของสมการบูลีนได้จำนวน m สมการ สมการเอาต์พุตแต่ละสมการจะเขียนอยู่ในเทอมของตัวแปรอินพุต n ตัว

วงจรคอมบิเนชันได้แก่ วงจรบวก (Adders) วงจรลบ (Subtractors) วงจรเปรียบเทียบ (Comparators) วงจรเข้ารหัส (Encoders) วงจรถอดรหัส (Decoders) วงจรแปลงรหัส (Code converters) วงจรพาริตี้ (Parity) วงจรมัลติเพล็กซ์ (Multiplexers) และวงจรดีมัลติเพล็กซ์ (Demultiplexers)



รูปที่ 7.1 บล็อกไดอะแกรมของวงจรคอมบิเนชัน

การออกแบบวงจรคอมบิเนชัน มีขั้นตอนที่สำคัญดังนี้

1. กำหนดหรือนิยามปัญหาของวงจรที่ต้องการออกแบบ กำหนดตัวแปรอินพุตและเอาต์พุตที่เกี่ยวข้องทั้งหมดพร้อมสัญลักษณ์ของตัวแปร
2. เขียนความสัมพันธ์ของตัวแปรอินพุตและเอาต์พุตในรูปตารางความจริง
3. ลดรูปสมการให้อยู่ในรูปสมการอย่างง่ายหรือให้มีจำนวนเทอมของสมการที่น้อยที่สุด โดยใช้ทฤษฎีพีชคณิตบูลีนหรือ KM
4. เปลี่ยนรูปเกตของวงจรตามความเหมาะสม โดยใช้ทฤษฎีดีมอร์แกน เช่น วงจรที่ต้องใช้แนนด์หรือนอร์เกตเท่านั้น
5. เขียนวงจรคอมบิเนชัน

ตารางความจริงของวงจรคอมบิเนชันประกอบด้วยอินพุตจำนวน n คอลัมน์และเอาต์พุตจำนวน m คอลัมน์ เอาต์พุตที่ได้จะเป็นตัวกำหนดการทำงานของวงจรที่แท้จริง ดังนั้นการเขียนตารางความจริงจึงเป็นสิ่งสำคัญอย่างมาก ถ้ากำหนดความสัมพันธ์ของตัวแปรอินพุตและเอาต์พุตในตารางความจริงไม่ถูกต้องสมบูรณ์ ทำให้วงจรที่ได้มีการทำงานไม่ถูกต้องเช่นกัน สมการเอาต์พุตจากตารางความจริงสามารถทำให้อยู่ในรูปอย่างง่าย โดยการลดรูปสมการด้วยวิธีการพีชคณิตบูลีนหรือ KM ในการออกแบบวงจรต้องคำนึงถึงองค์ประกอบที่เกี่ยวข้องด้วยเช่น จำนวนเกตที่ใช้ จำนวนอินพุตของเกตแต่ละตัว เวลาที่ใช้ในการทำงานของเกตในการเปลี่ยนสถานะหนึ่งไปยังอีกสถานะหนึ่ง (propagation delay time) ข้อจำกัดเกี่ยวกับคุณสมบัติของเกตแต่ละตัว รวมทั้งการออกแบบวงจรร่วมกับไอซี เป็นต้น อย่างไรก็ตามสิ่งสำคัญที่สุดของการออกแบบวงจรคือการลดรูปสมการให้อยู่ในรูปสมการอย่างง่ายก่อนที่จะพิจารณาองค์ประกอบอื่นตามมา

วงจรคำนวณทางคณิตศาสตร์และลอจิก (Arithmetic Logic Unit, ALU)

โดยทั่วไปคอมพิวเตอร์จะมีหน่วยประมวลผลกลางหรือที่เรียกสั้นๆ ว่า ซีพียู (Central Processing Unit, CPU) ทำหน้าที่เป็นเสมือนกับสมองของคอมพิวเตอร์ วงจรภายในซีพียูประกอบด้วย วงจรคำนวณทางคณิตศาสตร์และลอจิก (Arithmetic Logic Unit, ALU) หน่วยควบคุม (Control Unit) และรีจิสเตอร์ (Register) วงจรคำนวณทางคณิตศาสตร์และลอจิกทำหน้าที่เกี่ยวกับการบวก การลบ การคูณ การหารและการกระทำทางลอจิก เช่น แอนด์ ออร์ นอต เป็นต้น ในที่นี้จะกล่าวถึงการออกแบบวงจรบวก วงจรลบ วงจรเปรียบเทียบและวงจรคูณ การออกแบบวงจรจะใช้วิธีการเลียนแบบการบวก การลบ การคูณของเลขฐานสองที่ได้กล่าวไว้ในเรื่องระบบจำนวน

7.2 วงจรบวกเลขฐานสอง (Binary Adders)

การบวกเลขในวงจรดิจิตอลหมายถึงการนำเลขฐานสองจำนวน 2 ชุดมาบวกเข้าด้วยกัน เลขฐานสองแต่ละชุดมีขนาดตั้งแต่ 1 บิตขึ้นไป รูปที่ 7.2 แสดงตัวอย่างการบวกเลขฐานสองขนาด 1 บิตและขนาดหลายบิต รูปที่ 7.2 (ก) จะเห็นได้ว่าผลบวกที่ได้จากการบวก 0+0, 0+1 และ 1+0 จะให้ผลบวก 1 บิต แต่การบวก 1+1 จะให้ผลบวก 2 บิต บิตที่มีค่าสูงกว่าเรียกว่าบิตของตัวทด (Carry bit) และบิตที่มีค่าต่ำสุดเรียกว่าบิตของผลบวก (Sum bit) รูปที่ 7.2 (ข) แสดงการบวกเลขขนาดหลายบิต บิตของตัวทดที่ได้จากการบวกบิตแต่ละคู่จะถูกบวกเพิ่มไปในบิตที่สูงกว่า จากหลักการบวกเลขฐานสองนี้สามารถนำไปออกแบบวงจรบวกได้ดังนี้

```
0+      0+      1+      1+
0       1       0       1
0       1       1      10
```

(ก) การบวกเลขฐานสองขนาด 1 บิต

```
                1  1  1        ตัวทด
1  0  0+            1  0  1+
0  1  0             0  1  1
1  1  0          1  0  0  0
```

(ข) การบวกเลขฐานสองขนาด 3 บิต

รูปที่ 7.2 การบวกเลขฐานสอง

(1) วงจรบวกแบบครึ่ง (Half Adder, HA)

วงจร HA เป็นวงจรบวกขนาด 1 บิต โดยตัวตั้งและตัวบวกมีเพียงบิตเดียว ใช้สำหรับการบวกเลขฐานสองที่ตำแหน่งบิตต่ำสุด (LSB) เท่านั้น ผลลัพธ์ที่ได้จากการบวกมีจำนวน 2 บิตคือบิตผลบวกและบิตตัวทด สามารถเขียนบล็อกไดอะแกรมของวงจร HA ได้ดังรูปที่ 7.3 (ก) ประกอบด้วยตัวแปรอินพุตที่เป็นตัวตั้ง (A) และตัวบวก (B) ตัวแปรเอาต์พุตที่เป็นผลบวก (Σ) และตัวทดออก (C_{out} : Output carry) จากตารางความจริงของวงจร HA จะเห็นได้ว่า C_{out} มีค่าเป็น 1 เพียงค่าเดียวเท่านั้น เมื่ออินพุต A และ B มีค่าเป็น 1 ทั้งคู่

(ก) บล็อกไดอะแกรม

A	B	Σ	C_{out}
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

(ข) ตารางความจริง

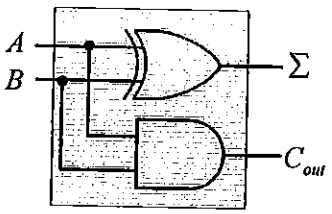
รูปที่ 7.3 บล็อกไดอะแกรมและตารางความจริงของวงจร HA

จากตารางความจริงสามารถเขียนสมการ SOP ได้ดังนี้

$$\Sigma = \sum m(1,2) = \overline{A}B + A\overline{B} = A \oplus B$$

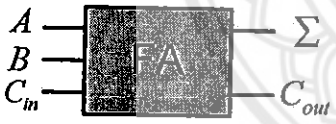
$$C_{out} = AB$$

นำสมการที่ได้ไปเขียนวงจร HA ได้ดังรูป



(2) **วงจรบวกแบบเต็ม** (Full Adder, FA)

วงจร FA เป็นวงจรบวกขนาด 1 บิต โดยตัวตั้งและตัวบวกมีเพียงบิตเดียว นอกจากนี้การออกแบบวงจร FA ยังคำนึงถึงตัวทดที่ได้จากการบวกของบิตที่ต่ำกว่าด้วย ผลลัพธ์ที่ได้จากการบวกมีจำนวน 2 บิต คือบิตของผลบวกและบิตตัวทด บล็อกไดอะแกรมของวงจร FA แสดงดังรูปที่ 7.4 (ก) ประกอบด้วยตัวแปรอินพุตที่เป็นตัวตั้ง (A) ตัวบวก (B) และตัวทดเข้า (C_{in} : Input carry) ตัวแปรเอาต์พุตที่เป็นผลบวก (Σ) และตัวทดออก (C_{out} : Output carry)



(ก) บล็อกไดอะแกรม

A	B	C_{in}	Σ	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

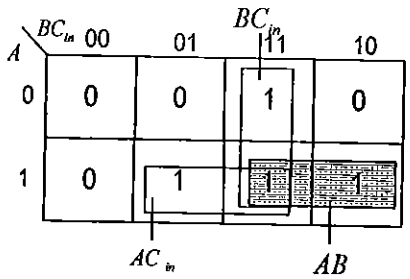
(ข) ตารางความจริง

รูปที่ 7.4 บล็อกไดอะแกรมและตารางความจริงของวงจร FA

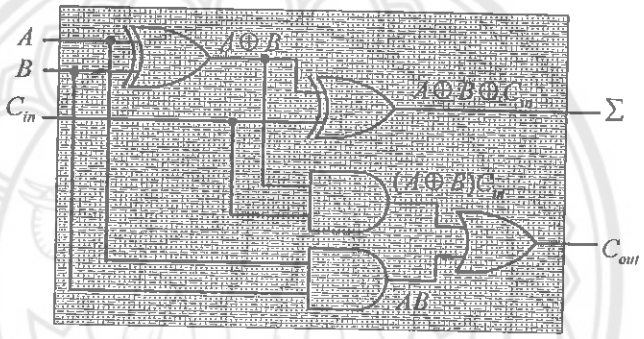
จากตารางความจริงสามารถเขียนสมการ SOP ได้ดังนี้

$$\begin{aligned}
\Sigma &= \sum m(1,2,4,7) \\
&= \overline{A}\overline{B}C_{in} + \overline{A}B\overline{C}_{in} + A\overline{B}\overline{C}_{in} + ABC_{in} \\
&= \overline{A}(\overline{B}C_{in} + B\overline{C}_{in}) + A(\overline{B}\overline{C}_{in} + BC_{in}) \\
&= \overline{A}(B \oplus C_{in}) + A(\overline{B \oplus C_{in}}) \\
&= A \oplus B \oplus C_{in} \\
C_{out} &= \sum m(3,5,6,7)
\end{aligned}$$

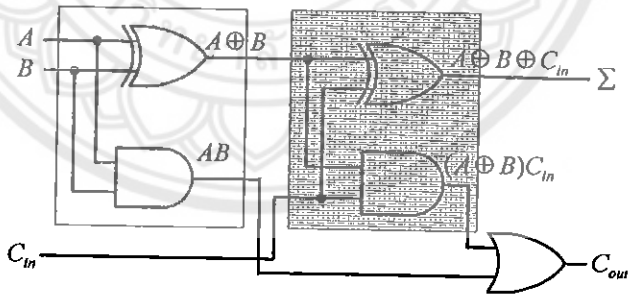
$$= \overline{A}BC_{in} + A\overline{B}C_{in} + AB\overline{C}_{in} + ABC_{in}$$
$$= (\overline{A}B + A\overline{B})C_{in} + AB(\overline{C}_{in} + C_{in})$$
$$= (A \oplus B)C_{in} + AB \quad \text{หรือ}$$
$$C_{out} = BC_{in} + AC_{in} + AB$$

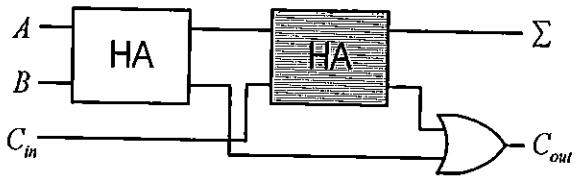


นำสมการ $\Sigma = A \oplus B \oplus C_{in}$ และ $C_{out} = (A \oplus B)C_{in} + AB$ เขียนวงจร FA ได้ดังรูป



เมื่อจัดรูปวงจรใหม่ จะได้ดังรูป

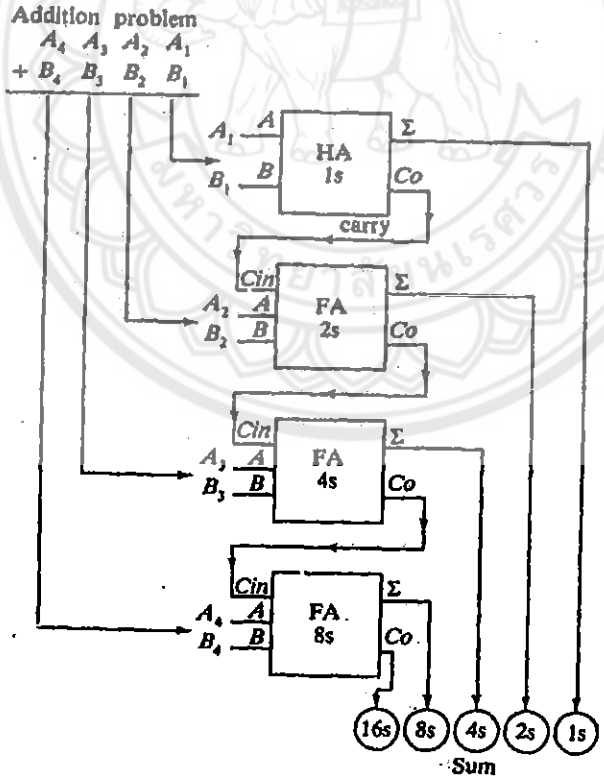




จากรูปจะเห็นได้ว่าวงจร FA สามารถนำวงจร HA จำนวน 2 วงจรและออร์เกตมาต่อร่วมกัน โดยที่วงจร HA วงจรแรกจะทำการบวกระหว่าง A และ B ส่วนวงจร HA ที่สองจะทำการบวกระหว่าง A, B และ C_{in}

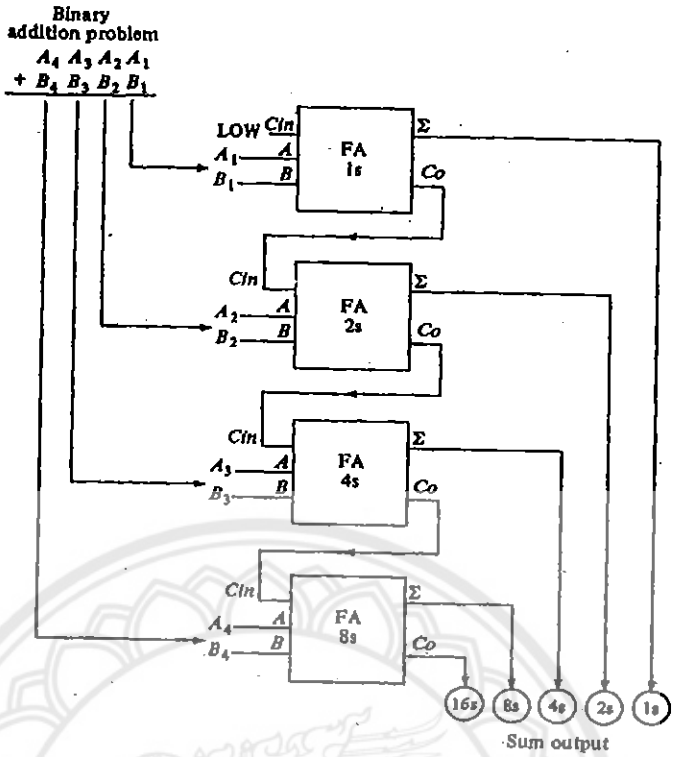
(3) **วงจรบวกแบบขนาน** (Parallel Adder)

การบวกเลขฐานสองจำนวนหลายบิต สามารถแสดงการบวกได้ 2 แบบ คือการบวกแบบอนุกรม (Serial Adder) และการบวกแบบขนาน (Parallel Adder) การบวกแบบอนุกรมจะเหมือนกับการบวกด้วยมือ โดยทำการบวกในคอลัมน์ของบิตต่ำสุดไปยังคอลัมน์ของบิตสูงสุด ถ้าข้อมูลมีจำนวนหลายบิต จะใช้เวลาในการบวกนาน แต่ถ้าเป็นการบวกแบบขนานจะให้ผลลัพธ์ที่เร็วกว่าเนื่องจากเวิร์ดข้อมูล (เวิร์ดข้อมูล (Data words) หมายถึงกลุ่มของข้อมูลที่มีจำนวนบิตต่างๆ กันเช่น 4, 8, 16 เป็นต้น) จะนำมาบวกพร้อมกันและให้ผลบวกทันที ดังนั้นวงจรบวกแบบอนุกรมจะเป็นวงจรที่ง่ายแต่ทำงานช้ากว่าวงจรบวกแบบขนาน ตัวอย่างวงจรบวกแบบขนานขนาด 4 บิต รูปที่ 7.5 (ก) เป็นวงจรบวกที่ประกอบด้วยวงจร HA จำนวน 1 วงจรและวงจร FA จำนวน 3 วงจร จะเห็นได้ว่าวงจร HA จะบวกบิตคอลัมน์แรก A_1 และ B_1 ส่วนคอลัมน์ที่สองจะใช้วงจร FA ทำการบวกบิต A_2 , B_2 และตัวทดที่ได้จากการบวกในคอลัมน์แรก (C_{out} ของคอลัมน์แรกป้อนเข้าที่ C_{in} ของคอลัมน์ที่สอง) สำหรับคอลัมน์ที่สามและสี่ใช้วงจร FA และทำการบวกเช่นเดียวกันกับคอลัมน์ที่สอง ผลบวกได้คือ $\sum$ ส่วน C_{out} ของวงจร FA ที่คอลัมน์สุดท้าย (8s) คือโอเวอร์โฟลว์ของผลบวก



(ก) วงจรบวกแบบขนานขนาด 4 บิตโดยใช้วงจร HA และ FA

รูปที่ 7.5 วงจรบวกแบบขนานขนาด 4 บิต (ในที่นี้ $Co = C_{out}$)

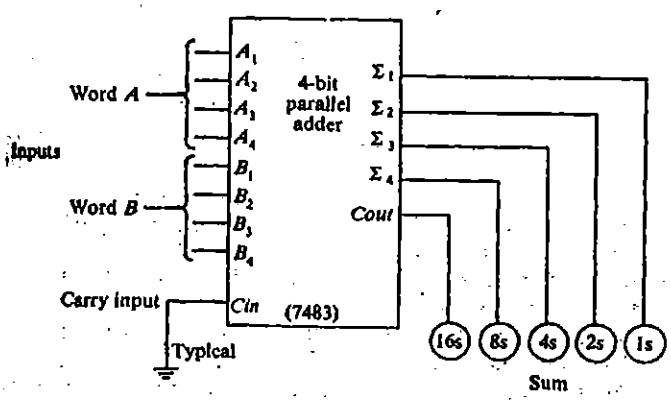


(ข) วงจรบวกแบบขนานขนาด 4 บิตโดยใช้วงจร FA

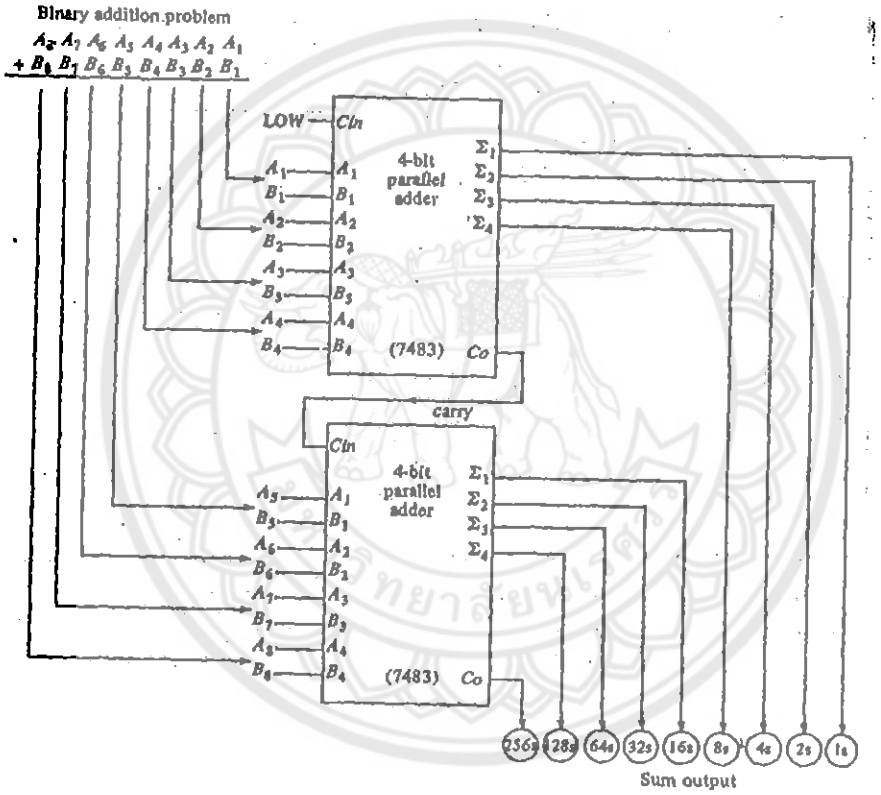
รูปที่ 7.5 วงจรบวกแบบขนานขนาด 4 บิต (ในที่นี้ $Co = C_{out}$)

รูปที่ 7.5 (ข) เป็นวงจรบวกแบบขนานขนาด 4 บิตที่ประกอบด้วยวงจร FA จำนวน 4 วงจร จะเห็นได้ว่าวงจร FA ที่ทำการบวก A_1 และ B_1 ในคอลัมน์แรก จะทำการบวกเหมือนกับวงจร HA โดยที่ C_{in} ของวงจร FA มีค่าเป็น LOW ส่วนการบวกบิตในคอลัมน์ถัดไปจะเหมือนกับวงจรในรูปที่ 7.5 (ก)

รูปที่ 7.6 เป็นตัวอย่างไอซีเบอร์ 7483 ของวงจรบวกแบบขนานขนาด 4 บิต การใช้งานของไอซีเบอร์นี้จะกำหนดให้ A_1, B_1 เป็นบิตต่ำสุด (LSB) ส่วน A_4, B_4 เป็นบิตสูงสุด (MSB) และ C_{in} จะต่อลงกราวด์ (GND) เสมอ ถ้าไม่ได้นำไอซีไปต่อกับวงจรบวกอื่นๆ จากรูปที่ 7.7 เป็นการประยุกต์ใช้ไอซีเบอร์ 7483 มาสร้างเป็นวงจรบวกแบบขนานขนาด 8 บิต โดยใช้ไอซีเบอร์ 7483 จำนวน 2 ตัวมาต่อเรียงกัน (Cascade) ไอซีตัวบนทำการบวก 4 บิตคือ $A_4A_3A_2A_1$ และ $B_4B_3B_2B_1$ ส่วนไอซีตัวล่างจะทำการบวก 4 บิตคือ $A_8A_7A_6A_5$ และ $B_8B_7B_6B_5$ สำหรับ C_{out} ของไอซีตัวบนจะต่อเข้ากับ C_{in} ของไอซีตัวล่าง



รูปที่ 7.6 ไอซีเบอร์ 7483 วงจรบวกแบบขนานขนาด 4 บิต



รูปที่ 7.7 วงจรบวกแบบขนานขนาด 8 บิตโดยใช้ไอซีเบอร์ 7483

รูปที่ 7.8 เป็นวงจรลบแบบขนานขนาด 4 บิตโดยใช้วงจร FA ต่อร่วมกับเกตอินเวอร์เตอร์ การลบเลขใช้วิธีการลบแบบ 2's complement เมื่อตัวตั้งคือ $A_4A_3A_2A_1$ ตัวลบคือ $B_4B_3B_2B_1$ ตัวลบจะกลับค่าลอจิกโดยผ่านเกตอินเวอร์เตอร์ก่อน (1's complementของ $B_4B_3B_2B_1$) แล้วที่ป้อนให้กับวงจร FA สำหรับ C_{in} ของวงจร FA ที่ 1s กำหนดค่าเป็น HIGH ซึ่งเป็นการบวก 1 กับค่า 1's complement ของ $B_4B_3B_2B_1$ (2's complement ของ $B_4B_3B_2B_1$) ผลต่างที่ได้จะปรากฏที่ตัวแสดงผลด้านขวา ตัวอย่างการลบเลขฐานสอง 1110_2 - 0111_2 โดยใช้ 2's complement

การลบโดยใช้ 2's complement

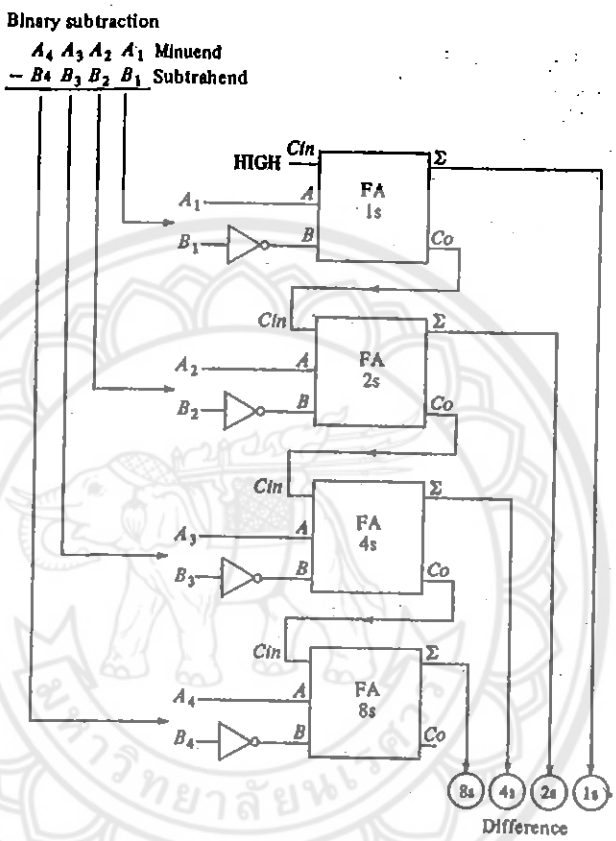
2's complement ของตัวลบ 0111= 1000+1

= 1001

```
      1 1 1 0+
      1 0 0 1
  ------------
    1 0 1 1 1
```

ตรวจสอบ

```
1 1 1 0-      1 4-
0 1 1 1       7
-------
0 1 1 1       7
```



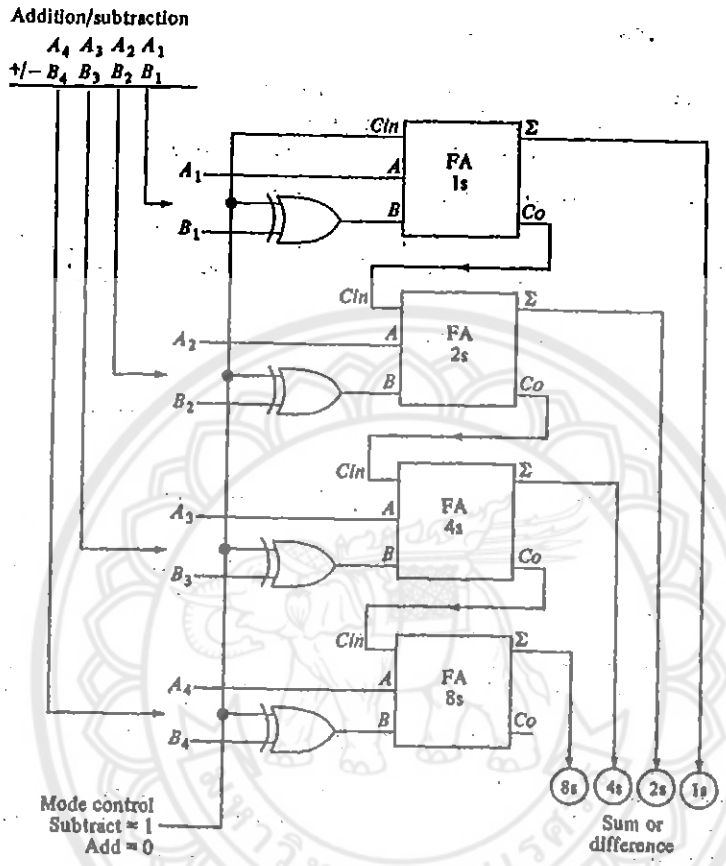
รูปที่ 7.8 วงจรลบแบบขนานขนาด 4 บิตโดยใช้วงจร FA

รูปที่ 7.9 เป็นทั้งวงจรบวกและลบแบบขนานขนาด 4 บิตโดยใช้วงจร FA ต่อร่วมกับเกต XOR วงจรนี้จะมีโหมดควบคุม (Mode control) เป็นอินพุตเพิ่มขึ้นมา ถ้าโหมดควบคุมเป็น LOW จะเป็นการบวกแบบขนานและถ้าเป็น HIGH จะเป็นการลบแบบขนาน ในที่นี้ตัวบวกหรือตัวลบคือ $B_4B_3B_2B_1$ จะเป็นอินพุตให้กับเกต XOR ทั้ง 4 ตัว การทำงานของวงจรมีดังนี้

วงจรบวกแบบขนาน กำหนดให้โหมดควบคุมเป็น 0 ดังนั้นเกต XOR จะไม่มีผลต่อตัวบวก $B_4B_3B_2B_1$ และ C_m ของวงจร FA ที่ 1s จะมีค่าเป็น 0 ดังนั้นวงจร FA นี้ จะทำงานเหมือนกับวงจร HA ผลบวกที่ได้จะปรากฏที่ตัวแสดงผลด้านขวา

วงจรลบแบบขนาน เป็นการลบแบบ 2's complement กำหนดให้โหมดควบคุมเป็น 1 ดังนั้นเกต XOR จะมีผลต่อตัวลบ $B_4B_3B_2B_1$ โดยเกต XOR ทำหน้าที่เหมือนกับอินเวอร์เตอร์คือทำ

การกลับค่าลอจิกของ $B_4B_3B_2B_1$ (1's complement ของ $B_4B_3B_2B_1$) และ C_{in} ของวงจร FA ที่ 1s จะมีค่าเป็น 1 ซึ่งเป็นการบวก 1 กับค่า 1's complement ของ $B_4B_3B_2B_1$ (2's complement ของ $B_4B_3B_2B_1$) ผลต่างที่ได้จะปรากฏที่ตัวแสดงผลด้านขวา



รูปที่ 7.9 วงจรบวกและลบแบบขนานขนาด 4 บิตโดยใช้วงจร FA

7.3 วงจรบวกเลข BCD (BCD Adder)

การบวกเลขฐานสิบ 1 หลัก เมื่อแทนเลขฐานสิบด้วยรหัส BCD การบวกสามารถทำได้เช่นเดียวกับการบวกของเลขฐานสอง แต่ถ้าผลบวกมีค่ามากกว่า 9 หรือมีตัวทดสุดท้ายเกิดขึ้น ต้องทำการบวก 0110 (6) เข้าไป เพื่อให้ผลบวกของเลข BCD มีค่าที่ถูกต้อง การบวกเลข BCD สามารถใช้วงจรบวกแบบขนานขนาด 4 บิตจำนวน 2 วงจร การบวกรหัส BCD 1 หลัก มีค่าผลบวกที่เป็นไปได้ตั้งแต่ 0-19 ดังตารางที่ 7.1

ตารางที่ 7.1 การเปรียบเทียบผลบวกของเลขฐานสองและเลข BCD

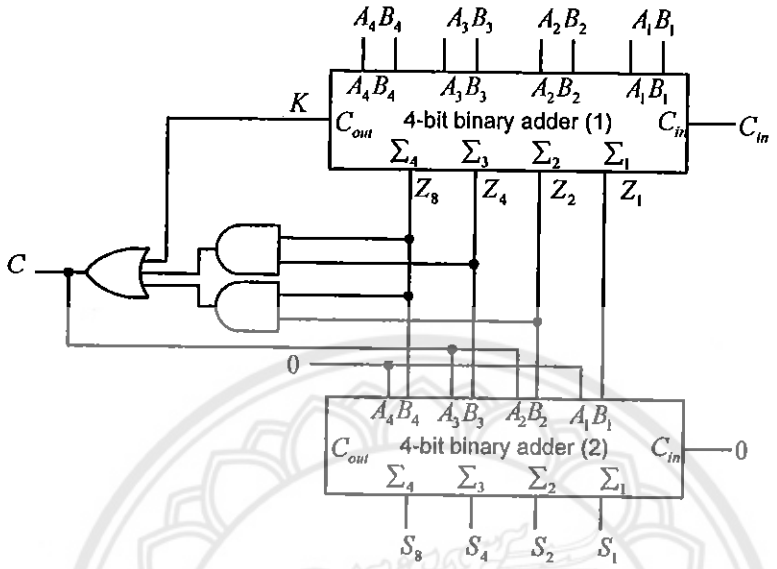
Binary Sum					BCD Sum					Decimal
K	Z_8	Z_4	Z_2	Z_1	C	S_8	S_4	S_2	S_1	
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	2
0	0	0	1	1	0	0	0	1	1	3
0	0	1	0	0	0	0	1	0	0	4
0	0	1	0	1	0	0	1	0	1	5
0	0	1	1	0	0	0	1	1	0	6
0	0	1	1	1	0	0	1	1	1	7
0	1	0	0	0	0	1	0	0	0	8
0	1	0	0	1	0	1	0	0	1	9
0	1	0	1	0	1	0	0	0	0	10
0	1	0	1	1	1	0	0	0	1	11
0	1	1	0	0	1	0	0	1	0	12
0	1	1	0	1	1	0	0	1	1	13
0	1	1	1	0	1	0	1	0	0	14
0	1	1	1	1	1	0	1	0	1	15
1	0	0	0	0	1	0	1	1	0	16
1	0	0	0	1	1	0	1	1	1	17
1	0	0	1	0	1	1	0	0	0	18
1	0	0	1	1	1	1	0	0	1	19

จากตารางที่ 7.1 กำหนดให้ K คือตัวทดของผลบวกเลขฐานสองและ $Z_8Z_4Z_2Z_1$ คือเลขฐานสองที่มีค่าน้ำหนักในแต่ละตำแหน่งของบิตเป็น 8, 4, 2, 1 ตามลำดับ ในคอลัมน์ผลบวกของเลขฐานสอง (Binary Sum) จะแสดงที่เอาต์พุตของวงจรบวกที่ (1) แต่การบวกเลข BCD ได้แสดงในคอลัมน์ผลบวกเลข BCD (BCD Sum) เมื่อ C คือตัวทดของผลบวกเลข BCD และ $S_8S_4S_2S_1$ คือเลข BCD ตามลำดับ ดังนั้นการออกแบบวงจรบวกเลข BCD ต้องมีตรวจสอบผลบวกของเลขฐานสองที่ได้ว่ามีค่ามากกว่า 9 หรือไม่ เมื่อพิจารณาจากคอลัมน์ผลบวกเลขฐานสองตั้งแต่ 01010 (10) ถึง 01111 (15) จะเห็นได้ว่าบิตตำแหน่งที่ Z_8 มีค่าเป็น 1 และตำแหน่งที่ Z_4 หรือ Z_2 ต้องมีค่าเป็น 1 ด้วย ส่วนเลขฐานสองตั้งแต่ 10000 (16) ถึง 10011 (19) จะมีค่าตัวทด $K=1$ ดังนั้นจากเงื่อนไขดังกล่าว ถ้าผลบวกเลขฐานสองมีค่ามากกว่า 9 หรือมีตัวทดสุดท้ายเกิดขึ้น วงจรจะต้องทำการบวก 6 สามารถเขียนสมการบูลีนแสดงความสัมพันธ์ได้ดังนี้

$$C(Add6) = K + Z_8(Z_4 + Z_2)$$
$$= K + Z_8Z_4 + Z_8Z_2$$

จากสมการบูลีน ถ้า $C=1$ ต้องทำการบวก 6 เข้าไป เพื่อให้ได้ผลบวกของเลข BCD ที่ถูกต้อง จากรูปที่ 7.10 ถ้าเอาต์พุต C ของออร์เกตมีค่าเป็น 0 จะไม่มีการบวก 6 แต่ถ้าเอาต์พุต C ของออร์เกตมีค่าเป็น 1 จะมีการบวก 6 ในวงจรบวกที่ (2) สำหรับ C_{out} ของวงจรบวกที่ (2) จะตัดทิ้ง

ไป เพราะวงจรจะใช้ค่าเอาต์พุต C ของออร์เกตแทน เมื่อต้องการบวกเลขฐานสิบจำนวน n หลัก จะใช้วงจรบวกเลข BCD จำนวน n วงจร และเอาต์พุต C ของออร์เกตจากวงจรแรกจะป้อนเข้าที่อินพุต C_{in} ของวงจรบวกเลข BCD ในหลักที่สูงกว่าต่อไป



รูปที่ 7.10 วงจรบวกเลข BCD 1 หลัก

7.4 วงจรลบเลขฐานสอง (Binary Subtractors)

จากรูปที่ 7.11 แสดงตัวอย่างการลบเลขฐานสองขนาด 1 บิตและขนาดหลายบิต รูปที่ 7.11 (ก) จะเห็นได้ว่าผลต่างที่ได้จากการลบ 0-0, 1-0 และ 1-1 จะให้ผลต่าง 1 บิต แต่การลบ 0-1 จะให้ค่าผลต่างและค่าของตัวยืมที่ยืมจากหลักถัดไป รูปที่ 7.11 (ข) แสดงการลบเลขที่มีขนาด 3 บิต กรณีที่บิตของตัวตั้งน้อยกว่าตัวลบจะต้องทำการยืมจากหลักที่อยู่ถัดไป จากหลักการลบเลขฐานสองนี้สามารถนำไปออกแบบวงจรลบได้ดังนี้

0 −	0 − *borrow* 1	1 −	1 −
0	1	0	1
0	1	1	0

(ก) การลบเลขฐานสองขนาด 1 บิต

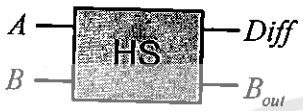
1 0 0 −	1 0 1 −
0 1 0	0 1 1
0 1 0	0 1 0

(ข) การลบเลขฐานสองขนาด 3 บิต

รูปที่ 7.11 การลบเลขฐานสอง

(1) **วงจรลบแบบครึ่ง** (Half Subtractor, HS)

วงจร HS เป็นวงจรลบขนาด 1 บิต โดยตัวตั้งและตัวลบมีเพียงบิตเดียว ผลลัพธ์ที่ได้มีจำนวน 2 บิตคือบิตของผลต่าง (Difference bit) และบิตตัวยืม (Borrow bit) สามารถเขียนบล็อกไดอะแกรมและตารางความจริงของวงจร HS ได้ดังรูปที่ 7.12 บล็อกไดอะแกรมของวงจร HS ประกอบด้วยตัวแปรอินพุตที่เป็นตัวตั้ง (A) และตัวลบ (B) ตัวแปรเอาต์พุตที่เป็นผลต่าง ($Diff$) และตัวยืม (B_{out})



(ก) บล็อกไดอะแกรม

A	B	$Diff$	B_{out}
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

(ข) ตารางความจริง

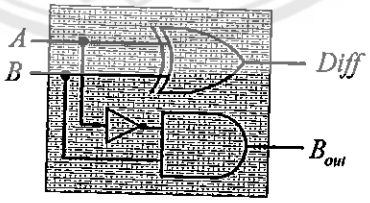
รูปที่ 7.12 บล็อกไดอะแกรมและตารางความจริงของวงจร HS

จากตารางความจริงสามารถเขียนสมการ SOP ได้ดังนี้

$$Diff = \sum m(1,2) = \overline{A}B + A\overline{B} = A \oplus B$$

$$B_{out} = \overline{A}B$$

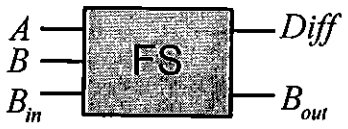
นำสมการที่ได้ไปเขียนวงจร HS ได้ดังรูป



(2) **วงจรลบแบบเต็ม** (Full Subtractor, FS)

วงจร FS เป็นวงจรลบขนาด 1 บิต โดยตัวตั้งและตัวลบมีเพียงบิตเดียว นอกจากนี้วงจรยังคำนึงถึงตัวยืมที่เกิดจากการลบของบิตที่ต่ำกว่าด้วย ผลลัพธ์ที่ได้มีจำนวน 2 บิตคือบิตของผลต่าง (Difference bit) และบิตของตัวยืม (Borrow bit) บล็อกไดอะแกรมและตารางความจริงของวงจร

FS แสดงดังรูปที่ 7.13 บล็อกไดอะแกรมประกอบด้วยตัวแปรอินพุตคือตัวตั้ง (A) ตัวลบ (B) และตัวยืม (B_{in}) ตัวแปรเอาต์พุตคือผลต่าง ($Diff$) และตัวยืม (B_{out})



(ก) บล็อกไดอะแกรม

A	B	B_{in}	$Diff$	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

(ข) ตารางความจริง

รูปที่ 7.13 บล็อกไดอะแกรมและตารางความจริงของวงจร FS

	$64s$	$32s$	$16s$	$8s$	$4s$	$2s$	$1s$	
$A-$	1	1	1	0	1	0	1–	117 –
B	0	0	1	1	1	0	0	28
$Diff$	1	0	1	1	0	0	1	89
B_{in}	0	1	1	0	0	0		
B_{out}	0	0	1	1	0	0	0	

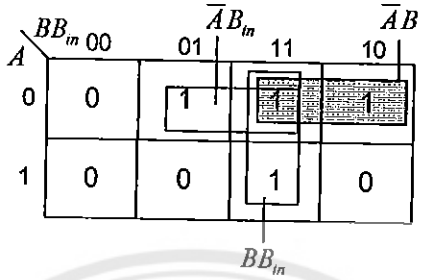
รูปที่ 7.14 ตัวอย่างการลบเลขฐานสองโดยใช้ตารางความจริง

จากตารางความจริงสามารถเขียนสมการ SOP ได้ดังนี้

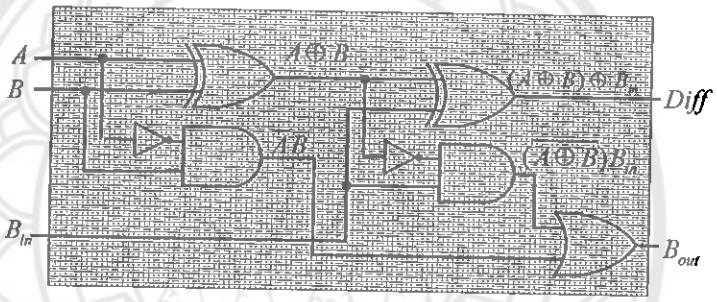
$$
\begin{aligned}
Diff &= \sum m(1,2,4,7) \\
&= \overline{A}\,\overline{B}B_{in} + \overline{A}B\overline{B}_{in} + A\overline{B}\,\overline{B}_{in} + ABB_{in} \\
&= \overline{A}(\overline{B}B_{in} + B\overline{B}_{in}) + A(\overline{B}\,\overline{B}_{in} + BB_{in}) \\
&= \overline{A}(B \oplus B_{in}) + A(\overline{B \oplus B_{in}}) \\
&= A \oplus B \oplus B_{in}
\end{aligned}
$$

$$
\begin{aligned}
B_{out} &= \sum m(1,2,3,7) \\
&= \overline{A}\,\overline{B}B_{in} + \overline{A}B\overline{B}_{in} + \overline{A}BB_{in} + ABB_{in} \\
&= \overline{A}(\overline{B}B_{in} + B\overline{B}_{in}) + BB_{in}(\overline{A} + A) \\
&= \overline{A}(B \oplus B_{in}) + BB_{in} \quad \text{หรือ}
\end{aligned}
$$

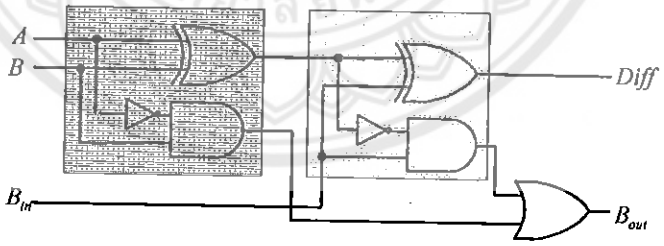
$$B_{out} = \sum m(1,2,3,7)$$
$$= \overline{A}\,\overline{B}B_{in} + \overline{A}B\overline{B}_{in} + \overline{A}BB_{in} + ABB_{in}$$
$$= B_{in}(\overline{A}\,\overline{B} + AB) + \overline{A}B(\overline{B}_{in} + B_{in})$$
$$= B_{in}(\overline{A \oplus B}) + \overline{A}B \text{ หรือ}$$
$$B_{out} = \overline{A}B_{in} + \overline{A}B + BB_{in}$$

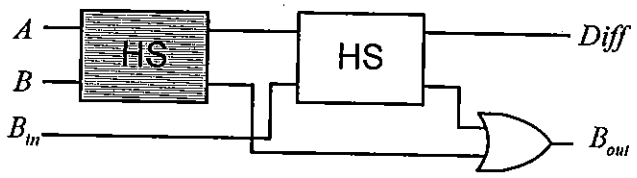


นำสมการ $Diff = A \oplus B \oplus B_{in}$ และ $B_{out} = B_{in}(\overline{A \oplus B}) + \overline{A}B$ เขียนวงจร FS ได้ดังรูป



เมื่อจัดรูปวงจร FS ใหม่จะได้ดังรูป

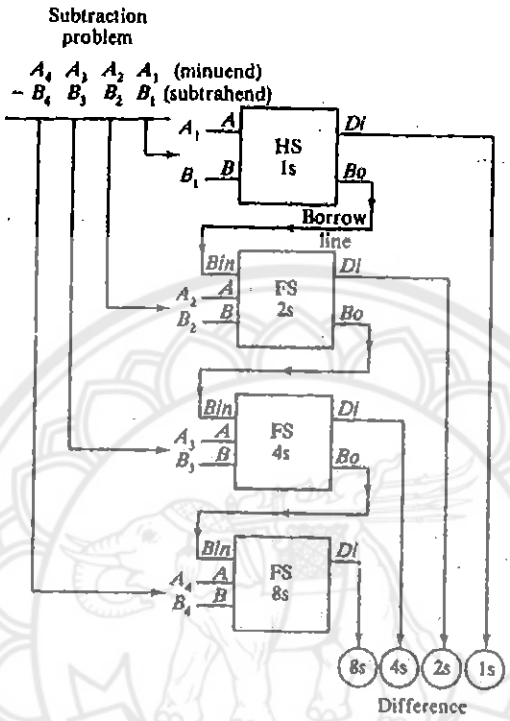




จะเห็นได้ว่าวงจร FS สามารถนำวงจร HS จำนวน 2 วงจรและออร์เกตมาต่อรวมกันได้ โดยที่วงจร HS วงจรแรกจะทำการลบระหว่าง A และ B ส่วนวงจร HS วงจรที่สองจะทำการลบระหว่าง A, B และ B_{in}

(3) **วงจรลบแบบขนาน** (Parallel Subtractor)

จากรูปที่ 7.15 เป็นวงจรลบแบบขนานขนาด 4 บิต โดยใช้วงจร HS และ FS เป็นการลบเลขฐานสองโดยใช้วิธีการลบโดยตรง เมื่อตัวตั้งคือ $A_4A_3A_2A_1$ ตัวลบคือ $B_4B_3B_2B_1$ การลบเลขฐานสองในคอลัมน์แรกจะใช้ HS ผลต่างที่ได้จะปรากฏที่ตัวแสดงผลด้านขวา



รูปที่ 7.15 วงจรลบแบบขนานขนาด 4 บิต โดยใช้วงจร HS และ FS (เมื่อ $Di = Diff$ และ $Bo = B_{out}$)

7.5 วงจรเปรียบเทียบ (Comparators)

การเปรียบเทียบเลขจำนวน 2 ชุดเพื่อแสดงว่าเลขชุดใดมีค่ามากกว่า น้อยกว่าหรือเท่ากับเลขอีกชุดหนึ่ง ถ้ากำหนดให้เลข 2 ชุดคืออินพุต A และ B เอาต์พุตที่ได้จากการเปรียบเทียบจะมี 3 ตัวแปรเพื่อแสดงว่า $A > B$, $A < B$ หรือ $A = B$ วงจรเปรียบเทียบเลขขนาดจำนวน n บิต อินพุตของตัวแปรที่เป็นไปได้ในตารางความจริงจะมีค่าเท่ากับ 2^{2n} กรณี การเปรียบเทียบจะพิจารณาบิตทีละคู่ที่ตำแหน่งตรงกัน ตัวอย่างเช่นเลขจำนวน 2 ชุดมีขนาด 4 บิต ดังนั้นตัวแปรของข้อมูลแต่ละชุดเป็นดังนี้

$$A = A_3A_2A_1A_0$$

$$B = B_3B_2B_1B_0$$

การเปรียบเทียบว่าเลขชุด A เท่ากับ B หรือไม่ จะทำการเปรียบเทียบทีละคู่ดังนี้ $A_3 = B_3$, $A_2 = B_2$, $A_1 = B_1$ และ $A_0 = B_0$ เมื่อเท่ากันทุกคู่แสดงว่า $A = B$

การเปรียบเทียบว่าเลขชุด A มากกว่าหรือน้อยกว่า B จะทำการเปรียบเทียบทีละคู่เช่นเดียวกัน โดยเริ่มที่ตำแหน่งของบิตที่สูงสุดก่อน (MSB) ไปยังคู่ที่ตำแหน่งของบิตต่ำกว่า ถ้าพบคู่ใดที่ $A = 1$ และ $B = 0$ แสดงว่าเลขชุด $A > B$ หรือคู่ใดที่ $A = 0$ และ $B = 1$ แสดงว่าเลขชุด $A < B$

ตัวอย่างที่ 7.1 จงออกแบบวงจรเปรียบเทียบความเท่ากันขนาด 1 บิต กำหนดให้ข้อมูลแต่ละชุดเป็น A และ B เอาต์พุตคือ Y และมีเงื่อนไขดังนี้

$$Y = \begin{cases} 1 & , A = B \\ 0 & , A \neq B \end{cases}$$

<u>วิธีทำ</u>

(1) ตารางความจริงของวงจรเปรียบเทียบขนาด 1 บิต

A	B	$Y_{(A=B)}$
0	0	1
0	1	0
1	0	0
1	1	1

(2) สมการ SOP

$$Y = \overline{A}\,\overline{B} + AB = \overline{A \oplus B}$$

(3) วงจรเปรียบเทียบความเท่ากันขนาด 1 บิต

ตัวอย่างที่ 7.2 จงออกแบบวงจรเปรียบเทียบความเท่ากันขนาด 2 บิต กำหนดให้ข้อมูลแต่ละชุดเป็น A_1A_0 และ B_1B_0 เอาต์พุตคือ Y และมีเงื่อนไขดังนี้

$$Y = \begin{cases} 1 & , A = B \\ 0 & , A \neq B \end{cases}$$

<u>วิธีทำ</u>

(1) ตารางความจริงของวงจรเปรียบเทียบขนาด 2 บิต

หมายเลขประจำแถว	A_1	B_1	A_0	B_0	$Y_{(A=B)}$
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	1

(2) สมการ SOP

$$Y_{(A=B)} = \sum m(0,3,12,15)$$
$$= \overline{A}_1\overline{B}_1\overline{A}_0\overline{B}_0 + \overline{A}_1\overline{B}_1A_0B_0 + A_1B_1\overline{A}_0\overline{B}_0 + A_1B_1A_0B_0$$

(3) เปลี่ยนรูปสมการ

<u>แบบที่ 1</u>

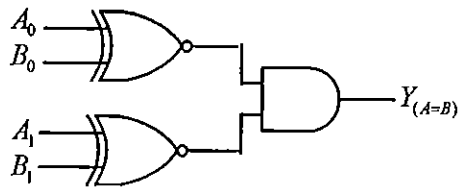
$$Y_{(A=B)} = \overline{A}_1\overline{B}_1\overline{A}_0\overline{B}_0 + \overline{A}_1\overline{B}_1A_0B_0 + A_1B_1\overline{A}_0\overline{B}_0 + A_1B_1A_0B_0$$
$$= \overline{A}_1\overline{B}_1(\overline{A}_0\overline{B}_0 + A_0B_0) + A_1B_1(\overline{A}_0\overline{B}_0 + A_0B_0)$$
$$= \overline{A}_1\overline{B}_1(\overline{A_0 \oplus B_0}) + A_1B_1(\overline{A_0 \oplus B_0})$$
$$= (\overline{A}_1\overline{B}_1 + A_1B_1)(\overline{A_0 \oplus B_0})$$
$$= (\overline{A_1 \oplus B_1})(\overline{A_0 \oplus B_0})$$

<u>แบบที่ 2</u>

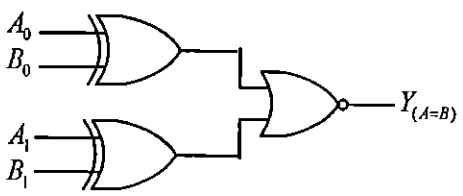
$$Y_{(A=B)} = (\overline{A_1 \oplus B_1})(\overline{A_0 \oplus B_0})$$
$$= \overline{\overline{(\overline{A_1 \oplus B_1})(\overline{A_0 \oplus B_0})}}$$
$$= \overline{\overline{(\overline{A_1 \oplus B_1})} + \overline{(\overline{A_0 \oplus B_0})}}$$
$$= \overline{(A_1 \oplus B_1) + (A_0 \oplus B_0)}$$

(4) วงจรเปรียบเทียบความเท่ากันขนาด 2 บิต

<u>แบบที่ 1</u>



<u>แบบที่ 2</u>



ตัวอย่างที่ 7.3 จงออกแบบวงจรเปรียบเทียบความเท่ากัน มากกว่าและน้อยกว่าของข้อมูลขนาด 1 บิต กำหนดให้ข้อมูลเป็น A และ B เอาต์พุตคือ Y และมีเงื่อนไขดังนี้

$$Y = \begin{cases} 1 & ,A = B, A < B, A > B \\ 0 & ,others \end{cases}$$

<u>วิธีทำ</u>

(1) ตารางความจริงของวงจรเปรียบเทียบของข้อมูล 2 ชุดขนาด 1 บิต

A	B	$Y_{(A=B)}$	$Y_{(A>B)}$	$Y_{(A<B)}$
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

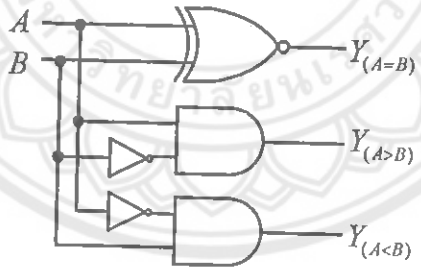
(2) สมการ SOP

$$Y_{(A=B)} = \overline{A}\,\overline{B} + AB = \overline{A \oplus B}$$

$$Y_{(A>B)} = A\overline{B}$$

$$Y_{(A<B)} = \overline{A}B$$

(3) วงจรเปรียบเทียบเท่ากัน มากกว่า น้อยกว่า ขนาด 1 บิต



ตัวอย่างที่ 7.4 จงออกแบบวงจรเปรียบเทียบความเท่ากัน มากกว่าและน้อยกว่าของข้อมูลขนาด 2 บิต กำหนดให้ข้อมูลแต่ละชุดเป็น A_1A_0 และ B_1B_0 เอาต์พุตคือ Y และมีเงื่อนไขดังนี้

$$Y = \begin{cases} 1 & ,A = B, A < B, A > B \\ 0 & ,others \end{cases}$$

<u>วิธีทำ</u>

(1) ตารางความจริงของวงจรเปรียบเทียบขนาด 2 บิต

หมายเลขประจำแถว	A_1	B_1	A_0	B_0	$Y_{(A=B)}$	$Y_{(A>B)}$	$Y_{(A<B)}$
0	0	0	0	0	1	0	0
1	0	0	0	1	0	0	1
2	0	0	1	0	0	1	0
3	0	0	1	1	1	0	0
4	0	1	0	0	0	0	1
5	0	1	0	1	0	0	1
6	0	1	1	0	0	0	1
7	0	1	1	1	0	0	1
8	1	0	0	0	0	1	0
9	1	0	0	1	0	1	0
10	1	0	1	0	0	1	0
11	1	0	1	1	0	1	0
12	1	1	0	0	1	0	0
13	1	1	0	1	0	0	1
14	1	1	1	0	0	1	0
15	1	1	1	1	1	0	0

(2) สมการ SOP

$$Y_{(A=B)} = \sum m(0,3,12,15) = (\overline{A_1 \oplus B_1})(\overline{A_0 \oplus B_0})$$

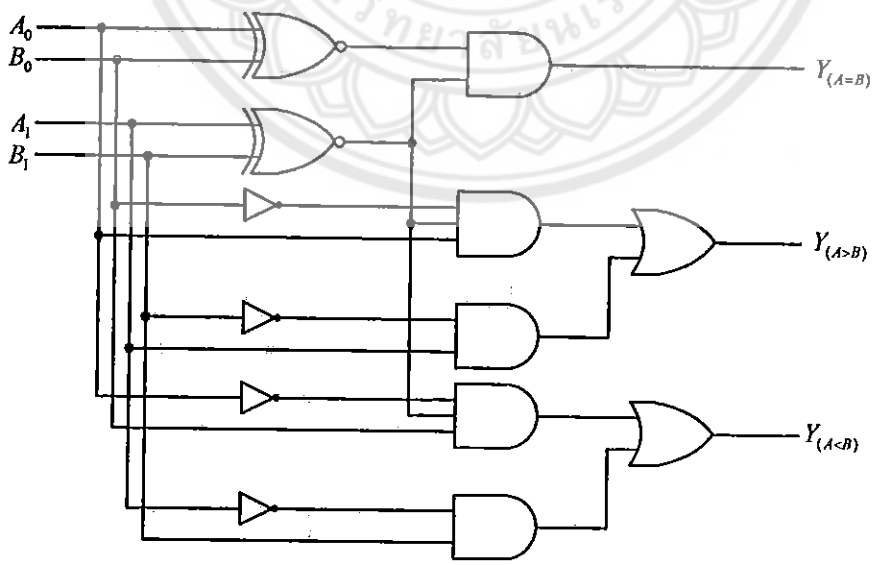
$$Y_{(A>B)} = \sum m(2,8,9,10,11,14)$$

$$= A_1\overline{B}_1 + \overline{B}_1 A_0 \overline{B}_0 + A_1 A_0 \overline{B}_0 \quad \text{หรือ} \quad A_1\overline{B}_1 + A_0\overline{B}_0(\overline{A_1 \oplus B_1})$$

$$Y_{(A<B)} = \sum m(1,4,6,7,13)$$

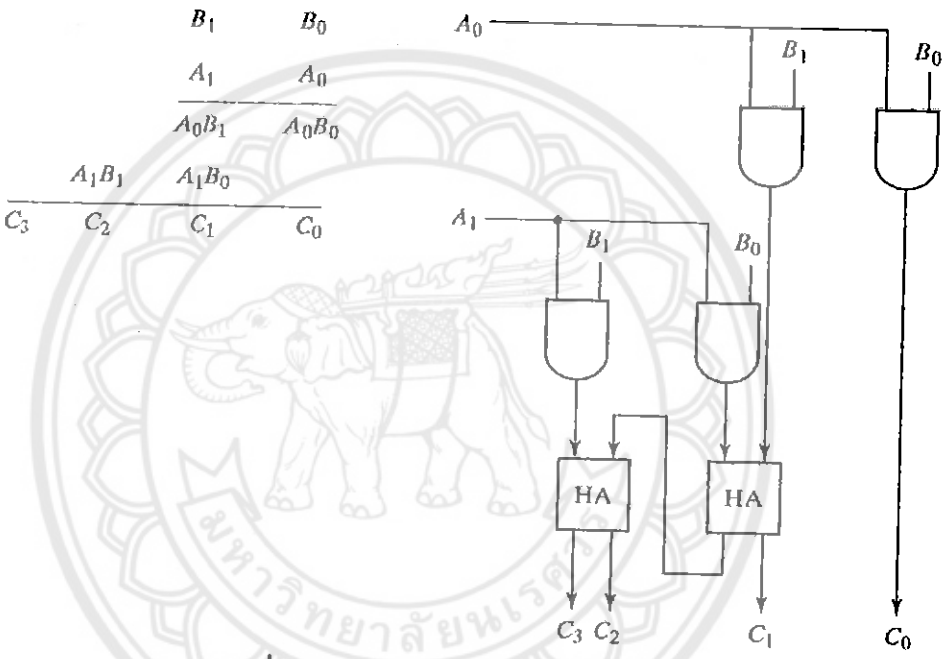
$$= \overline{A}_1 B_1 + \overline{A}_0 B_0(\overline{A}_1 + B_1) \quad \text{หรือ} \quad \overline{A}_1 B_1 + \overline{A}_0 B_0(\overline{A_1 \oplus B_1})$$

(3) วงจรเปรียบเทียบขนาด 2 บิต



7.6 วงจรคูณ (Multiplication)

วงจรคูณเลขฐานสองอย่างง่าย สามารถใช้หลักการคูณกันของเลขฐานสอง โดยนำตัวคูณไปคูณกับตัวตั้งทีละบิต เริ่มจากบิตต่ำสุดก่อน แล้วนำผลคูณที่ได้ในแต่ละบิตมาบวกกัน จะเห็นว่าจากขั้นตอนการคูณสามารถนำแอนด์เกตและวงจรบวก HA หรือ FA มาใช้ในการออกแบบวงจรได้ ถ้าต้องการออกแบบวงจรคูณ $J \times K$ บิต จะได้ผลคูณที่มีจำนวนบิตเท่ากับ $J+K$ บิต และวงจรจะใช้แอนด์เกตจำนวน $J \times K$ เกต ตัวอย่างเช่น การออกแบบวงจรคูณขนาด 2x2 บิต เมื่อกำหนดให้ตัวตั้งคือ B_1B_0 ตัวคูณคือ A_1A_0 และผลคูณที่ได้คือ $C_3C_2C_1C_0$ สามารถออกแบบวงจรคูณโดยใช้แอนด์เกตจำนวน 4 เกตและวงจรบวก HA จำนวน 2 ชุด ดังรูปที่ 7.16



รูปที่ 7.16 วงจรลบแบบขนานขนาด 4 บิต

แบบฝึกหัดท้ายบท

1. จงออกแบบวงจร FA โดยใช้แนนด์เกตเท่านั้น
2. จงออกแบบวงจร FA โดยใช้นอร์เกตเท่านั้น
3. จงออกแบบวงจร FS โดยใช้แนนด์เกตเท่านั้น
4. จงออกแบบวงจร FS โดยใช้นอร์เกตเท่านั้น
5. จงออกแบบวงจรบวกและลบเลข Signed 2's complement ขนาด 4 บิตโดยใช้วงจร FA
6. จงออกแบบวงจรเปรียบเทียบความเท่ากันของข้อมูลขนาด 3 บิต กำหนดให้ข้อมูล 2 ชุดคือ $A_2A_1A_0$ และ $B_2B_1B_0$
7. จงออกแบบวงจรการคูณเลขฐานสองขนาด 2x3 บิต

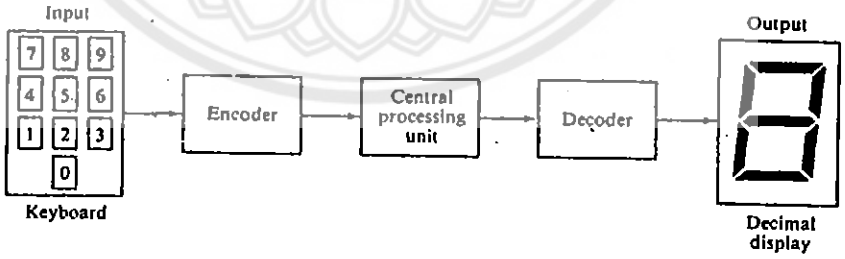
บทที่ 8
วงจรลอจิกเชิงจัดหมู่ (Combinational Circuit)

วัตถุประสงค์

1. สามารถเข้าใจหลักการทำงานของวงจรคอมบิเนชัน เช่น วงจรเข้ารหัส วงจรถอดรหัส วงจรมัลติเพล็กซ์ วงจรดีมัลติเพล็กซ์ และวงจรพาริตี้
2. สามารถออกแบบวงจรคอมบิเนชันได้

8.1 บทนำ

เมื่อระบบคอมพิวเตอร์ทำการประมวลผลข้อมูลด้วยเลขฐานสอง แต่มนุษย์เราสามารถเข้าใจข้อมูลเลขฐานสองได้ยาก ถ้ามีข้อมูลเลขฐานสองจำนวนมาก ดังนั้นคอมพิวเตอร์จึงต้องมีวงจรที่ใช้ในการแปลงภาษาเครื่อง (Machine Language) เป็นภาษามนุษย์ (Human Language) ที่เราสามารถเข้าใจได้ จากรูปที่ 8.1 แสดงบล็อกไดอะแกรมของเครื่องคำนวณอย่างง่าย โดยการป้อนตัวเลข 0 ถึง 9 ที่แป้นคีย์บอร์ด วงจรเข้ารหัส (Encoder) ที่อยู่ระหว่างคีย์บอร์ดและซีพียูจะทำหน้าที่แปลงจากเลขฐานสิบไปเป็นรหัสของเลขฐานสอง เช่น รหัส BCD 8421 เป็นต้น จากนั้นซีพียูทำหน้าที่ในการประมวลข้อมูลเลขฐานสองและส่งข้อมูลไปยังวงจรถอดรหัส (Decoder) เพื่อแปลงจากเลขฐานสองไปเป็นรหัส 7 ส่วนและแสดงผลเป็นตัวเลข 0-9 จะเห็นได้ว่าวงจรถอดรหัสและวงจรเข้ารหัสทำหน้าที่ในการแปลงรหัสข้อมูล โดยวงจรเข้ารหัสทำหน้าที่แปลงภาษามนุษย์ไปเป็นภาษาเครื่องและวงจรถอดรหัสทำหน้าที่ตรงกันข้าม



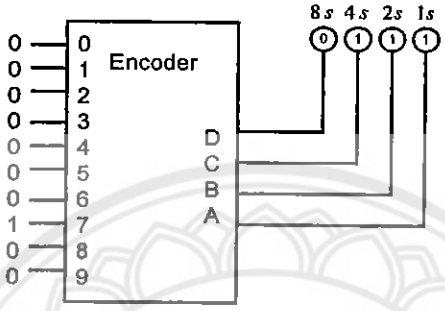
รูปที่ 8.1 บล็อกไดอะแกรมเครื่องคำนวณอย่างง่าย

8.2 วงจรเข้ารหัส (Encoder)

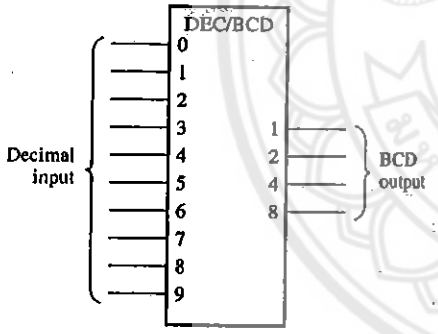
วงจรเข้ารหัสเป็นวงจรลอจิกเชิงจัดหมู่ทำหน้าที่รับข้อมูลที่เป็นตัวเลข สัญลักษณ์หรือตัวอักขระต่างๆ เพื่อแปลงข้อมูลเหล่านี้เป็นเลขฐานสอง เลขฐานสองที่ได้จะนำไปใช้งานต่อไป วงจรจะมีอินพุต ACTIVE เพียงครั้งละ 1 อินพุตเท่านั้น วงจรเข้ารหัสได้แก่

(1) วงจรเข้ารหัสเลขฐานสิบเป็น BCD (Decimal to BCD Encoder)

วงจรเข้ารหัสเลขฐานสิบเป็น BCD เป็นวงจรที่ใช้แปลงเลขฐานสิบเป็นรหัส BCD วงจรนี้มี 10 อินพุต แต่ละอินพุตคือเลขฐานสิบ 1 ตัว และเอาต์พุตคือรหัส BCD ดังนั้นอาจเรียกว่าวงจรเข้ารหัส 10:4 line โดยทั่วไปวงจรเข้ารหัสมีอินพุต ACTIVE HIGH เพียง 1 อินพุต ที่ให้เอาต์พุต 1 ค่า ค่าเอาต์พุตที่ได้จะไม่ซ้ำกัน ตัวอย่างเช่น จากรูปที่ 8.2 เมื่อให้อินพุตเลข 7 เป็นลอจิก 1 วงจรเข้ารหัสแสดงผลที่เอาต์พุตมีค่าเป็น 0111



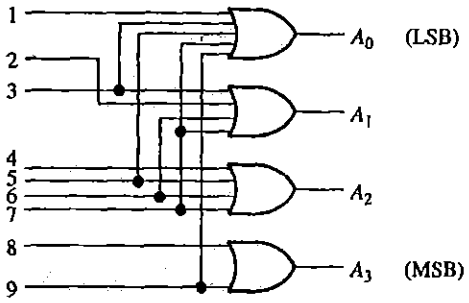
รูปที่ 8.2 ตัวอย่างการเข้ารหัสของเลข 7



(ก) สัญลักษณ์

Decimal Digit	รหัส BCD A_3(8)	A_2(4)	A_1(2)	A_0(1)
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

(ข) ตารางความจริง



(ค) วงจรเข้ารหัสเลขฐานสิบเป็นเลข BCD

รูปที่ 8.3 วงจรเข้ารหัสเลขฐานสิบเป็นเลข BCD

ตัวอย่างการออกแบบวงจรเข้ารหัสเลขฐานสิบเป็น BCD สามารถเขียนสัญลักษณ์และตารางความจริงได้ดังรูปที่ 8.3 (ก) และ (ข) ตามลำดับ จากตารางความจริงแสดงความสัมพันธ์ของเลขฐานสิบและเลข BCD โดยที่บิต A_3 เป็นบิตนัยสำคัญที่มากที่สุดและ A_0 เป็นบิตนัยสำคัญที่น้อยที่สุด เมื่อพิจารณาจากตารางความจริงจะเห็นได้ว่าบิต A_3 มีค่าลอจิก 1 ที่เลข 8 และ 9 สามารถเขียนสมการของ A_3 ในเทอมของตัวเลขฐานสิบได้ดังนี้

$$A_3 = 8 + 9$$

ส่วน A_2, A_1 และ A_0 เขียนในเทอมของตัวเลขฐานสิบได้เช่นเดียวกัน

$$A_2 = 4 + 5 + 6 + 7$$

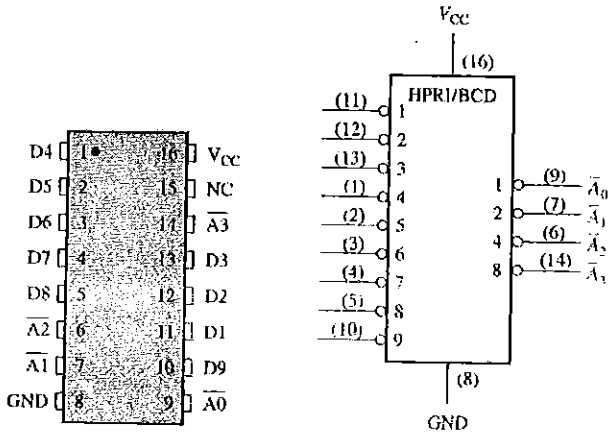
$$A_1 = 2 + 3 + 6 + 7$$

$$A_0 = 1 + 3 + 5 + 7 + 9$$

จากสมการของ A_3, A_2, A_1 และ A_0 เขียนวงจรในรูปอย่างง่ายได้ดังรูปที่ 8.3 (ค)

การทำงานของวงจรรูปที่ 8.3 (ค) วงจรมีอินพุต ACTIVE HIGH เพียง 1 อินพุตเท่านั้น ตัวอย่างเช่น ให้อินพุตของเลข 9 เป็น 1 (อินพุตที่เหลือเป็น 0) จะได้เอาต์พุตของวงจร $A_3A_2A_1A_0 = 1001$ แต่การใช้งานจริงบางครั้งอินพุตอาจมีค่าลอจิก 1 เข้ามาพร้อมกันมากกว่า 1 อินพุต ทำให้เอาต์พุตที่ได้มีค่าไม่ถูกต้อง เช่น เมื่อป้อนอินพุตที่ 3 และ 6 เป็นลอจิก 1 พร้อมกัน จะได้เอาต์พุตมีค่าเป็น 0111 เป็นต้น ดังนั้นจึงมีการออกแบบวงจรเข้ารหัสแบบลำดับความสำคัญ (Priority) เพื่อไม่ให้เกิดความผิดพลาดในการใช้งาน โดยเอาต์พุตจะเลือกแสดงผลอินพุตที่มีค่าเลขมากที่สุดเสมอ (Highest-order decimal digit) จึงเรียกวงจรนี้ว่าวงจรเข้ารหัสเลขฐานสิบเป็น BCD แบบลำดับความสำคัญ (Decimal to BCD Priority Encoder) ตัวอย่างเช่น ไอซีเบอร์ 74HC147 ทำหน้าที่แปลงเลขฐานสิบไปเป็นเลขฐานสองขนาด 4 บิต อินพุตและเอาต์พุตของไอซีเป็นแบบ ACTIVE LOW คุณสมบัติการทำงานดังรูปที่ 8.4 (ข)

รูปที่ 8.4(ก) สัญลักษณ์ไอซีมีวงกลมทั้งด้านอินพุตและเอาต์พุตแสดงว่าอินพุตและเอาต์พุตเป็นแบบ ACTIVE LOW ดังนั้นในการใช้งานจริง จะนำอินเวอร์เตอร์ต่อที่เอาต์พุตของไอซี จะได้เลขฐานสองของเลข 0-9 เมื่อพิจารณาตารางความจริง แถวแรกอินพุตของเลข 1-9 เป็น HIGH ทั้งหมดจะให้เอาต์พุต $\overline{A}_3\overline{A}_2\overline{A}_1\overline{A}_0$ เป็น 1111 แถวที่สองอินพุตของเลข 9 จะ ACTIVE LOW จะให้เอาต์พุตเป็น $LHHL$ เมื่อผ่านอินเวอร์เตอร์จะเป็น 1001 ซึ่งเป็นรหัส BCD ของเลข 9 สำหรับแถวอื่นๆ จะมีลักษณะเช่นเดียวกัน ตัวแปร X ในตารางความจริงคือเทอม don't care ที่เป็น HIGH หรือ LOW ก็ได้ วงจรนี้มีการเข้ารหัสตามลำดับความสำคัญ โดยตัวเลขที่มีค่ามากสุดที่ ACTIVE LOW จะแสดงผลที่เอาต์พุต เช่น ถ้าอินพุตของเลข 3 และ 6 เป็น ACTIVE LOW วงจรเข้ารหัสจะแสดงเอาต์พุตของตัวเลขที่มีค่าสูงสุดคือเลข 6 เท่านั้น



(ก) สัญลักษณ์

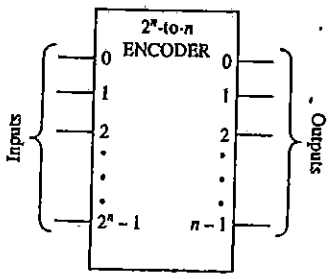
Input (Decimal Digit)									Output			
		3	4	5	6	7	8	9	A_3	A_2	A_1	A_0
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	X	L	H	H	H	H	L	L	H
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

(ข) ตารางการทำงาน

รูปที่ 8.4 ไอซีเบอร์ 74HC147 (HPRI: Highest value input has priority)

(2) **วงจร**เข้ารหัส 2^n line to n line

วงจรประกอบด้วยอินพุตจำนวน 2^n อินพุตและเอาต์พุตจำนวน n เอาต์พุต โดยอินพุตที่เป็น ACITVE HIGH มีเพียง 1 อินพุต และให้เอาต์พุตเป็นเลขฐานสอง



รูปที่ 8.5 สัญลักษณ์ของวงจรเข้ารหัส 2^n : n line

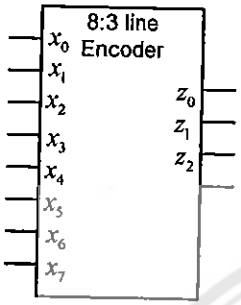
ตัวอย่างวงจรเข้ารหัส 8:3 line เมื่อเขียนสมการเอาต์พุตได้ดังนี้

$$z_0 = x_1 + x_3 + x_5 + x_7$$

$$z_1 = x_2 + x_3 + x_6 + x_7$$

$$z_2 = x_4 + x_5 + x_6 + x_7$$

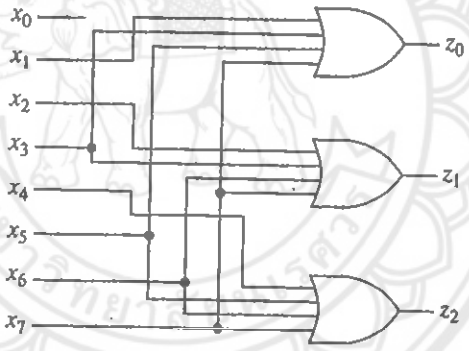
เมื่อนำไปเขียนวงจรจะได้ดังรูปที่ 8.5 (ค)



(ก) สัญลักษณ์

x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	z_2	z_1	z_0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

(ข) ตารางความจริง

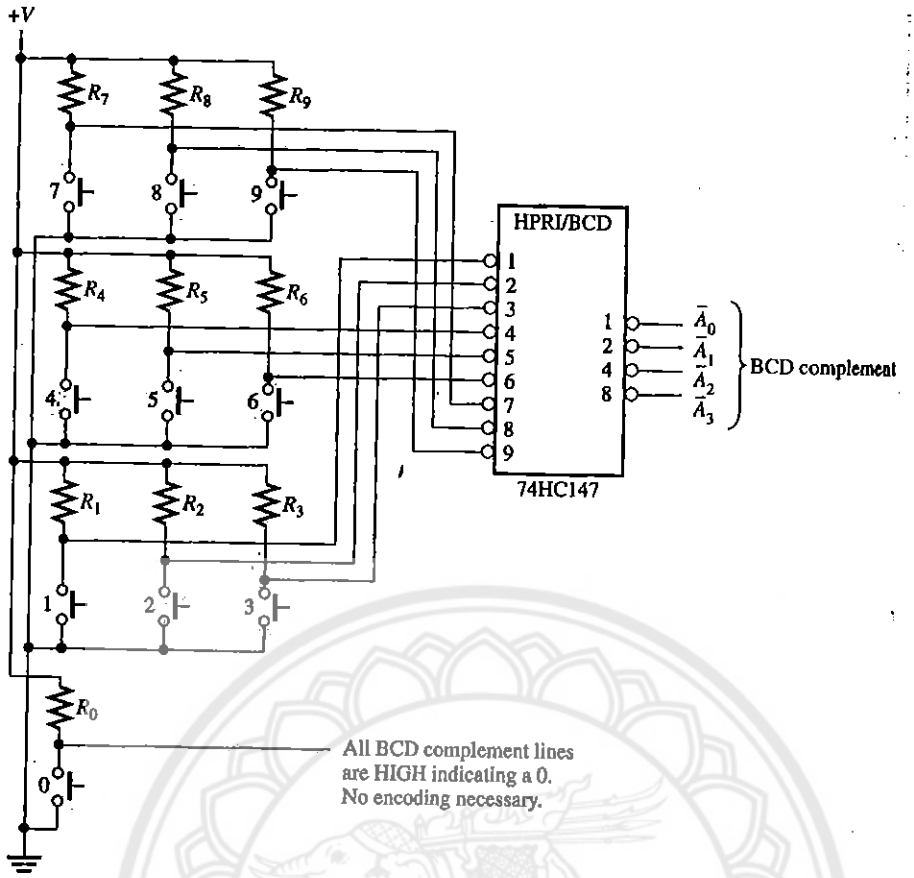


(ค) วงจรเข้ารหัส 8 : 3 line

รูปที่ 8.5 วงจรเข้ารหัส 8 : 3 line

(3) ตัวอย่างการประยุกต์ใช้งาน

จากรูปที่ 8.6 เป็นตัวอย่างการเข้ารหัสคีย์บอร์ดอย่างง่ายโดยใช้ไอซี 74HC147 คีย์บอร์ดมีตัวเลข 0-9 เมื่อกดคีย์เลขใดๆ วงจรจะทำแปลงเป็นเลข BCD สวิตซ์ของเลข 0-9 จะต่อกับแรงดันไฟฟ้า $+V$ เมื่อไม่ได้กดคีย์ใดๆ อินพุตของไอซีจะเป็น HIGH และเมื่อกดคีย์ใดๆ จะให้อินพุตเป็น LOW คีย์ของเลข 0 จะไม่ต่อเข้าไอซีเพราะเอาต์พุตจะเป็น 0 เมื่อไม่ได้กดคีย์ใดๆ

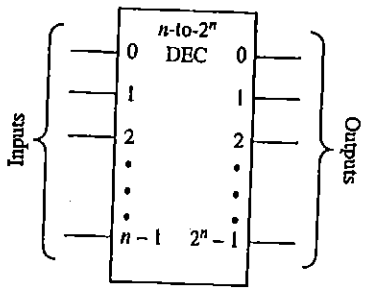


รูปที่ 8.6 ตัวอย่างการเข้ารหัสของคีย์บอร์ดอย่างง่าย

8.3 วงจรถอดรหัส (Decoder)

วงจรถอดรหัส ทำหน้าที่รับข้อมูลที่เป็นเลขฐานสองเพื่อแปลงเป็นข้อมูลในรูปแบบต่างๆ ที่สามารถเข้าใจได้ง่าย โดยทั่วไปวงจรถอดรหัสจะมีเอาต์พุต ACTIVE เพียง 1 เอาต์พุตเท่านั้น เอาต์พุตที่เหลือจะไม่ ACTIVE เอาต์พุตอาจเป็นแบบ ACTIVE HIGH หรือ ACTIVE LOW ก็ได้ ซึ่งมีลักษณะตรงกันข้ามกับวงจรเข้ารหัสที่มีอินพุต ACTIVE เพียงครั้งละ 1 อินพุต

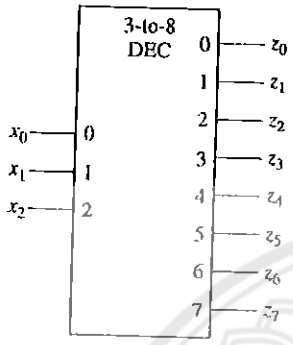
(1) วงจรถอดรหัส n line to 2^n line



รูปที่ 8.7 สัญลักษณ์วงจรถอดรหัส $n : 2^n$ line

วงจรถอดรหัส 3 : 8 line ที่มีเอาต์พุตแบบ ACTIVE HIGH

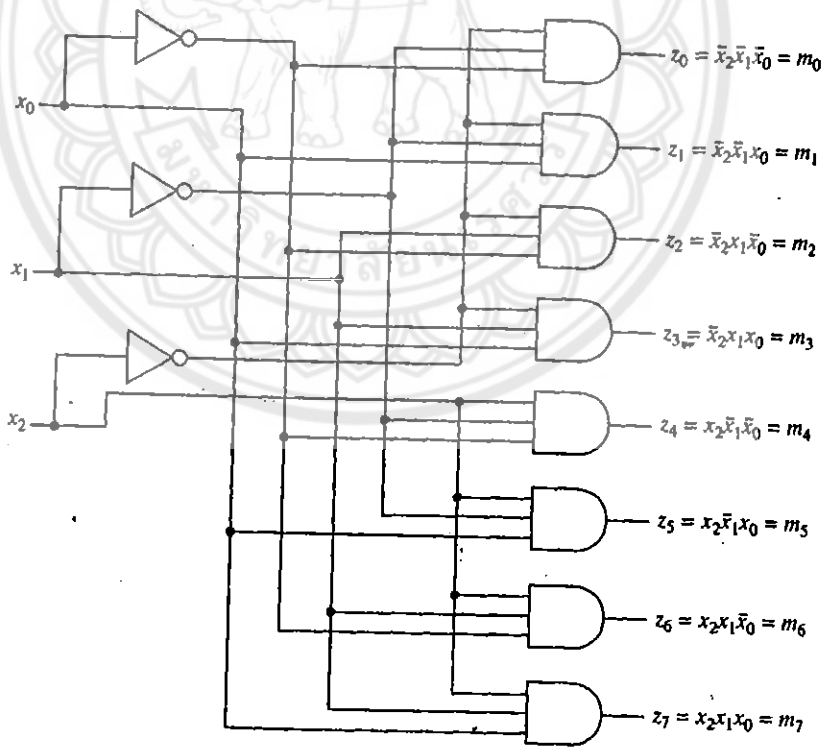
เมื่อกำหนดตัวแปรอินพุต/เอาต์พุตของวงจรคือ $x_2x_1x_0$ และ $z_7z_6z_5z_4z_3z_2z_1z_0$ ตามลำดับ สัญลักษณ์และตารางความจริงของวงจรแสดงดังรูปที่ 8.8 จากตารางความจริงเอาต์พุตเป็น HIGH เพียง 1 เอาต์พุต ส่วนเอาต์พุตที่เหลือเป็น LOW ดังนั้นสามารถเขียนสมการ SOP ของเอาต์พุตได้ดังนี้



(ก) สัญลักษณ์

Inputs			Outputs							
x_2	x_1	x_0	z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

(ข) ตารางความจริง



(ค) วงจรถอดรหัส

รูปที่ 8.8 วงจรถอดรหัส 3 : 8 line ที่มีเอาต์พุตแบบ ACTIVE HIGH

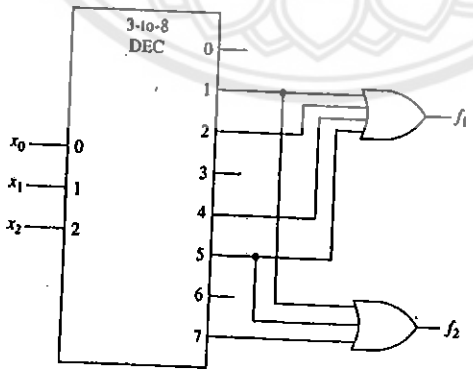
$$z_0 = \bar{x}_2\bar{x}_1\bar{x}_0 = m_0 \qquad z_1 = \bar{x}_2\bar{x}_1x_0 = m_1$$
$$z_2 = \bar{x}_2x_1\bar{x}_0 = m_2 \qquad z_3 = \bar{x}_2x_1x_0 = m_3$$
$$z_4 = x_2\bar{x}_1\bar{x}_0 = m_4 \qquad z_5 = x_2\bar{x}_1x_0 = m_5$$
$$z_6 = x_2x_1\bar{x}_0 = m_6 \qquad z_7 = x_2x_1x_0 = m_7$$

จะเห็นได้ว่าในแต่ละสมการของเอาต์พุตคือ Minterm นั่นเอง ดังนั้นสามารถเรียกวงจรนี้ว่า วงจรสร้าง Minterm (Minterm generator) เมื่อเขียนวงจรจะได้ดังรูปที่ 8.8 (ค) วงจรประกอบด้วยนอตและแอนด์เกต ถ้าต้องการใช้เกตชนิดอื่น สามารถเปลี่ยนรูปสมการโดยใช้ดีมอร์แกน จะได้สมการใหม่ที่สร้างจากนอตและนอร์เกตได้

$$z_0 = \bar{x}_2\bar{x}_1\bar{x}_0 = \overline{\overline{\bar{x}_2\bar{x}_1\bar{x}_0}} = \overline{x_2 + x_1 + x_0} \qquad z_1 = \bar{x}_2\bar{x}_1x_0 = \overline{\overline{\bar{x}_2\bar{x}_1x_0}} = \overline{x_2 + x_1 + \bar{x}_0}$$
$$z_2 = \bar{x}_2x_1\bar{x}_0 = \overline{\overline{\bar{x}_2x_1\bar{x}_0}} = \overline{x_2 + \bar{x}_1 + x_0} \qquad z_3 = \bar{x}_2x_1x_0 = \overline{\overline{\bar{x}_2x_1x_0}} = \overline{x_2 + \bar{x}_1 + \bar{x}_0}$$
$$z_4 = x_2\bar{x}_1\bar{x}_0 = \overline{\overline{x_2\bar{x}_1\bar{x}_0}} = \overline{\bar{x}_2 + x_1 + x_0} \qquad z_5 = x_2\bar{x}_1x_0 = \overline{\overline{x_2\bar{x}_1x_0}} = \overline{\bar{x}_2 + x +_1 \bar{x}_0}$$
$$z_6 = x_2x_1\bar{x}_0 = \overline{\overline{x_2x_1\bar{x}_0}} = \overline{\bar{x}_2 + \bar{x}_1 + x_0} \qquad z_7 = x_2x_1x_0 = \overline{\overline{x_2x_1x_0}} = \overline{\bar{x}_2 + \bar{x}_1 + \bar{x}_0}$$

วงจรถอดรหัสสามารถนำไปประยุกต์ใช้ในการสร้างสมการ SOP และ POS ได้ ตัวอย่างเช่น ถ้าต้องการสร้างสมการ SOP ของสมการต่อไปนี้ สามารถสร้างวงจรได้โดยใช้ออร์เกต ดังรูปที่ 8.9

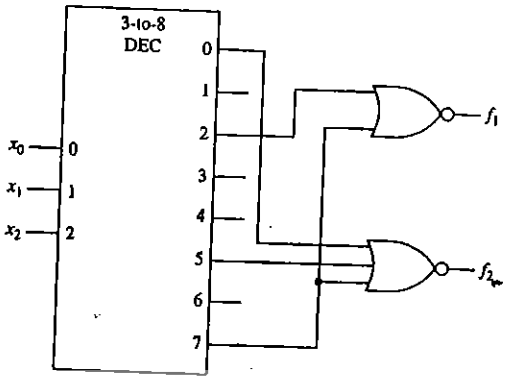
$$f_1(x_2, x_1, x_0) = \sum m(1,2,4,5) = m_1 + m_2 + m_4 + m_5$$
$$f_2(x_2, x_1, x_0) = \sum m(1,5,7) = m_1 + m_5 + m_7$$



รูปที่ 8.9 การประยุกต์ใช้วงจรถอดรหัสสร้างสมการ SOP

ถ้าต้องการสร้างสมการ SOP ต่อไปนี้ โดยใช้นอร์เกต ดังรูปที่ 8.10

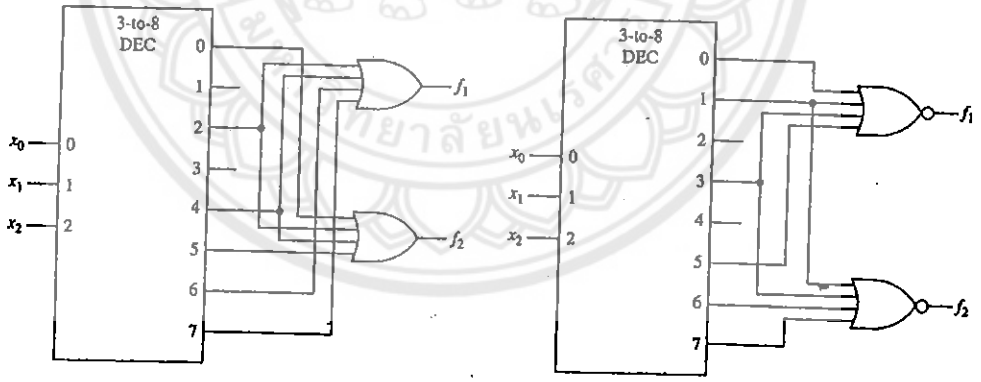
$$f_1(x_2,x_1,x_0) = \sum m(0,1,3,4,5,6) = \overline{\sum m(2,7)}$$
$$f_2(x_2,x_1,x_0) = \sum m(1,2,3,4,6) = \overline{\sum m(0,5,7)}$$



รูปที่ 8.10 การประยุกต์ใช้วงจรถอดรหัสสร้างสมการ SOP

ถ้าต้องการสร้างสมการ POS ของสมการต่อไปนี้ สามารถสร้างวงจรได้**ดังรูปที่** 8.11 **โดยใช้ออร์เกตหรือนอร์เกตตามลำดับ**

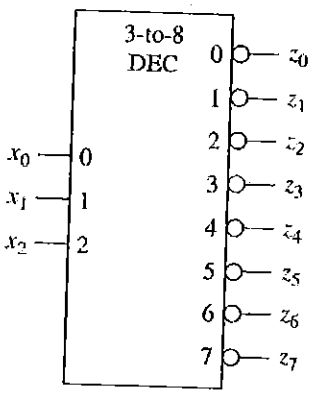
$$f_1(x_2,x_1,x_0) = \prod M(0,1,3,5) = \overline{\sum m(0,1,3,5)} = \sum m(2,4,6,7)$$
$$f_2(x_2,x_1,x_0) = \prod M(1,3,6,7) = \overline{\sum m(1,3,6,7)} = \sum m(0,2,4,5)$$



รูปที่ 8.11 การประยุกต์ใช้วงจรถอดรหัสสร้างสมการ POS

วงจรถอดรหัส 3 : 8 line ที่มีเอาต์พุตแบบ ACTIVE LOW

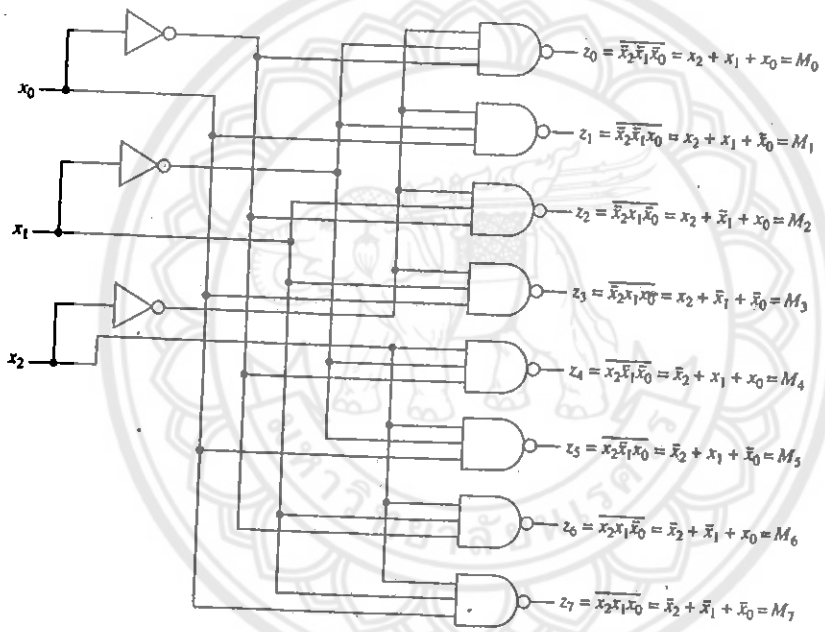
เมื่อกำหนดตัวแปรอินพุต/เอาต์พุตของวงจรคือ $x_2x_1x_0$ และ $z_7z_6z_5z_4z_3z_2z_1z_0$ ตามลำดับ สัญลักษณ์และตารางความจริงของวงจรแสดงดังรูปที่ 8.12 จากตารางความจริงเอาต์พุตเป็น LOW เพียง 1 เอาต์พุต ส่วนเอาต์พุตที่เหลือเป็น HIGH ดังนั้นสามารถเขียนสมการ POS ของเอาต์พุตได้ดังนี้



(ก) สัญลักษณ์

Inputs			Outputs							
x_2	x_1	x_0	z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_7
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

(ข) ตารางความจริง



(ค) วงจรถอดรหัส

รูปที่ 8.12 วงจรถอดรหัส 3: 8 line ที่มีเอาต์พุตแบบ ACTIVE LOW

$$z_0 = x_2 + x_1 + x_0 = M_0 \qquad z_1 = x_2 + x_1 + \bar{x}_0 = M_1$$

$$z_2 = x_2 + \bar{x}_1 + x_0 = M_2 \qquad z_3 = x_2 + \bar{x}_1 + \bar{x}_0 = M_3$$

$$z_4 = \bar{x}_2 + x_1 + x_0 = M_4 \qquad z_5 = \bar{x}_2 + x_1 + x_0 = M_5$$

$$z_6 = \bar{x}_2 + \bar{x}_1 + x_0 = M_6 \qquad z_7 = \bar{x}_2 + \bar{x}_1 + \bar{x}_0 = M_7$$

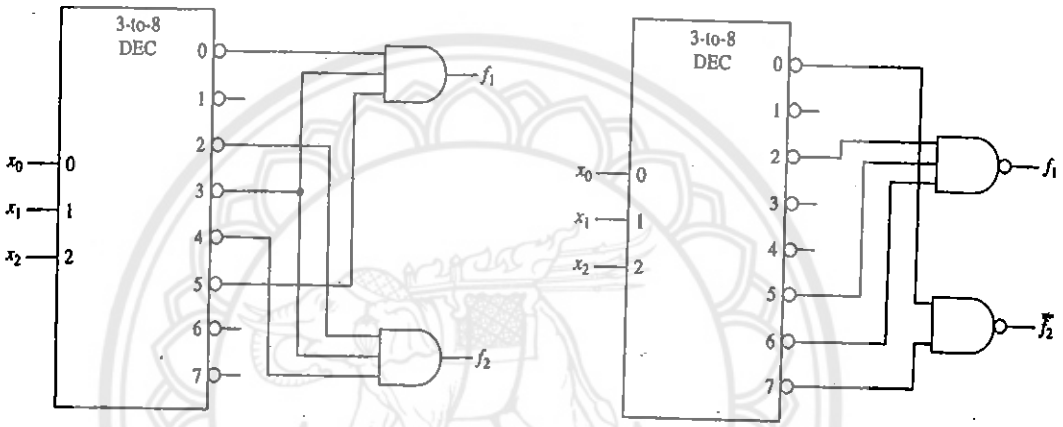
หรือ

$$z_0 = x_2 + x_1 + x_0 = \overline{\bar{x}_2\bar{x}_1\bar{x}_0} \qquad z_1 = x_2 + x_1 + \bar{x}_0 = \overline{\bar{x}_2\bar{x}_1 x_0}$$

$$z_2 = x_2 + \bar{x}_1 + x_0 = \overline{\bar{x}_2 x_1\bar{x}_0} \qquad z_3 = x_2 + \bar{x}_1 + \bar{x}_0 = \overline{\bar{x}_2 x_1 x_0}$$

$$z_4 = \bar{x}_2 + x_1 + x_0 = \overline{x_2\bar{x}_1\bar{x}_0} \qquad z_5 = \bar{x}_2 + x_1 + x_0 = \overline{\bar{x}_2 x_1 x_0}$$
$$z_6 = \bar{x}_2 + \bar{x}_1 + x_0 = \overline{x_2 x_1 \bar{x}_0} \qquad z_7 = \bar{x}_2 + \bar{x}_1 + \bar{x}_0 = \overline{x_2 x_1 x_0}$$

จะเห็นได้ว่าในแต่ละสมการของเอาต์พุตคือ Maxterm นั่นเอง ดังนั้นสามารถเรียกวงจรนี้ว่า วงจรสร้าง Maxterm (Maxterm generator) เมื่อเขียนวงจรจากสมการจะได้วงจร ที่ประกอบด้วยนอตเกตและออร์เกต นอกจากนี้สามารถเปลี่ยนรูปสมการโดยใช้ดีมอร์แกน จะได้สมการวงจรถอดรหัสที่ใช้นอตและแนนด์เกตดังรูปที่ 8.12 (ค) วงจรถอดรหัสนี้สามารถนำไปประยุกต์ใช้ในการสร้างสมการ SOP และ POS โดยใช้แอนด์เกตหรือแนนด์เกตได้ดังรูปที่ 8.13



$$f_1(x_2, x_1, x_0) = \prod M(0,3,5) \qquad f_1(x_2, x_1, x_0) = \prod M(0,1,3,4,7)$$
$$f_2(x_2, x_1, x_0) = \prod M(2,3,4) \qquad f_2(x_2, x_1, x_0) = \prod M(1,2,3,4,5,6)$$

รูปที่ 8.13 การประยุกต์ใช้วงจรถอดรหัสสร้างสมการ POS

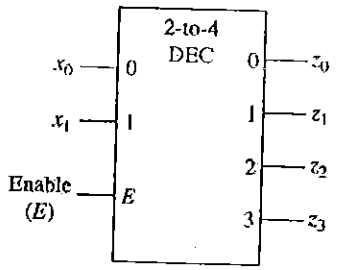
(2) **วงจรถอดรหัส** n line to 2^n line **ที่มีอินพุตอินาเบิล**

โดยปกติวงจรถอดรหัสจะมีขาอินพุตอินาเบิลเพิ่มขึ้นมาตั้งแต่ 1 อินพุตขึ้นไป ขาอินาเบิลนี้อาจเป็นแบบ ACTIVE LOW ($\bar{E}$) หรือแบบ ACTIVE HIGH (E) ทำหน้าที่ควบคุมวงจรถอดรหัสให้ทำงานหรือไม่ทำงาน เมื่อมีการนำวงจรหลายวงจรมาต่อกัน ทำให้การใช้งานวงจรมีความยืดหยุ่น

วงจรถอดรหัส 2 : 4 line ที่มีอินพุตอินาเบิล ACTIVE HIGH

เมื่อกำหนดให้อินพุตคือ x_1x_0 และ E (Enable) และเอาต์พุตคือ $z_3z_2z_1z_0$ สัญลักษณ์และตารางความจริงของวงจรแสดงดังรูปที่ 8.14 จากตารางความจริงเอาต์พุตที่ต้องการมีค่าลอจิก 1 และเอาต์พุตอื่นๆ มีค่าลอจิกเป็น 0 ดังนั้นสามารถเขียนสมการ SOP ของเอาต์พุตได้ดังนี้

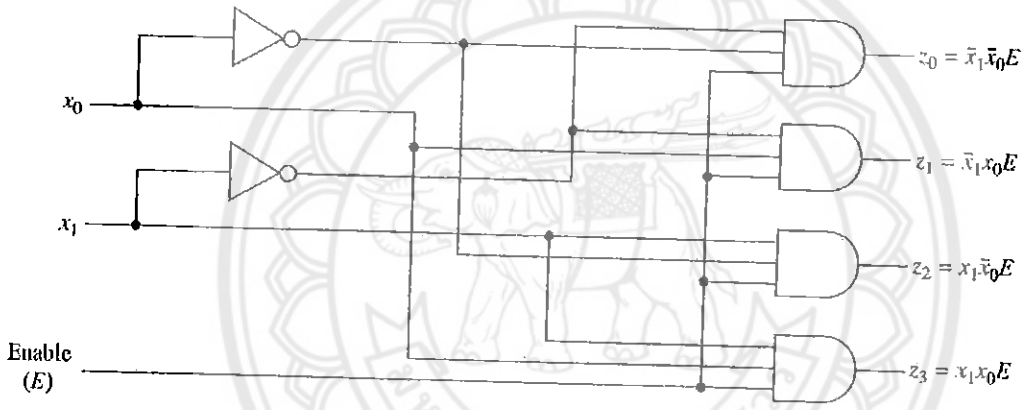
$$z_0 = E\bar{x}_1\bar{x}_0 \qquad z_1 = E\bar{x}_1 x_0$$
$$z_2 = Ex_1\bar{x}_0 \qquad z_3 = Ex_1 x_0$$



(ก) สัญลักษณ์

Inputs			Outputs			
E	x_1	x_0	z_0	z_1	z_2	z_3
0	×	×	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

(ข) ตารางความจริง



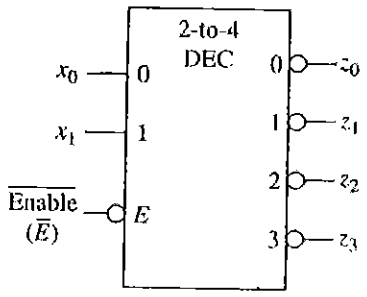
(ค) วงจรถอดรหัส

รูปที่ 8.14 วงจรถอดรหัส 2: 4 line ที่มีอินพุตอินาเบิล ACTIVE HIGH

วงจรถอดรหัส 2 : 4 line ที่มีอินพุตอินาเบิล ACTIVE LOW

เมื่อกำหนดให้อินพุตคือ x_1x_0 และ $\overline{E}$ (Enable) และเอาต์พุตคือ $z_3z_2z_1z_0$ สัญลักษณ์และตารางความจริงของวงจรแสดงดังรูปที่ 8.15 จากตารางความจริงเอาต์พุตที่ต้องการมีค่าลอจิก 0 และเอาต์พุตอื่นๆ มีค่าลอจิกเป็น 1 ดังนั้นสามารถเขียนสมการ POS ของเอาต์พุตได้ดังนี้

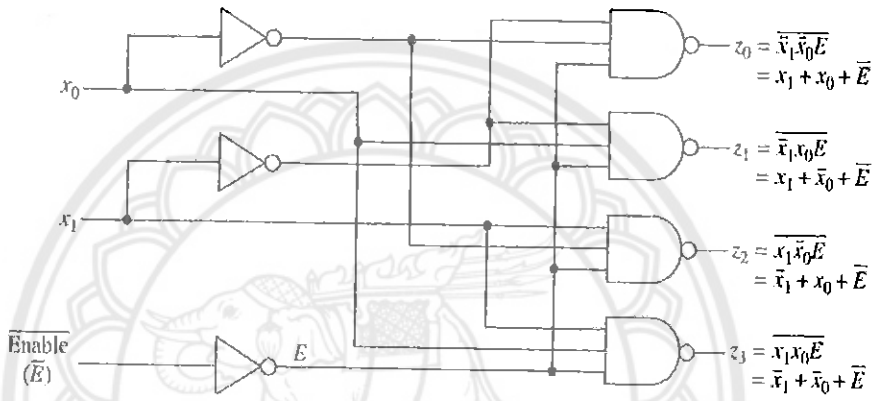
$$z_0 = \overline{E} + x_1 + x_0 \qquad z_1 = \overline{E} + x_1 + \bar{x}_0$$
$$z_2 = \overline{E} + \bar{x}_1 + x_0 \qquad z_3 = \overline{E} + \bar{x}_1 + \bar{x}_0$$



(ก) สัญลักษณ์

Inputs $\overline{E}$	x_1	x_0	Outputs z_0	z_1	z_2	z_3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	×	×	1	1	1	1

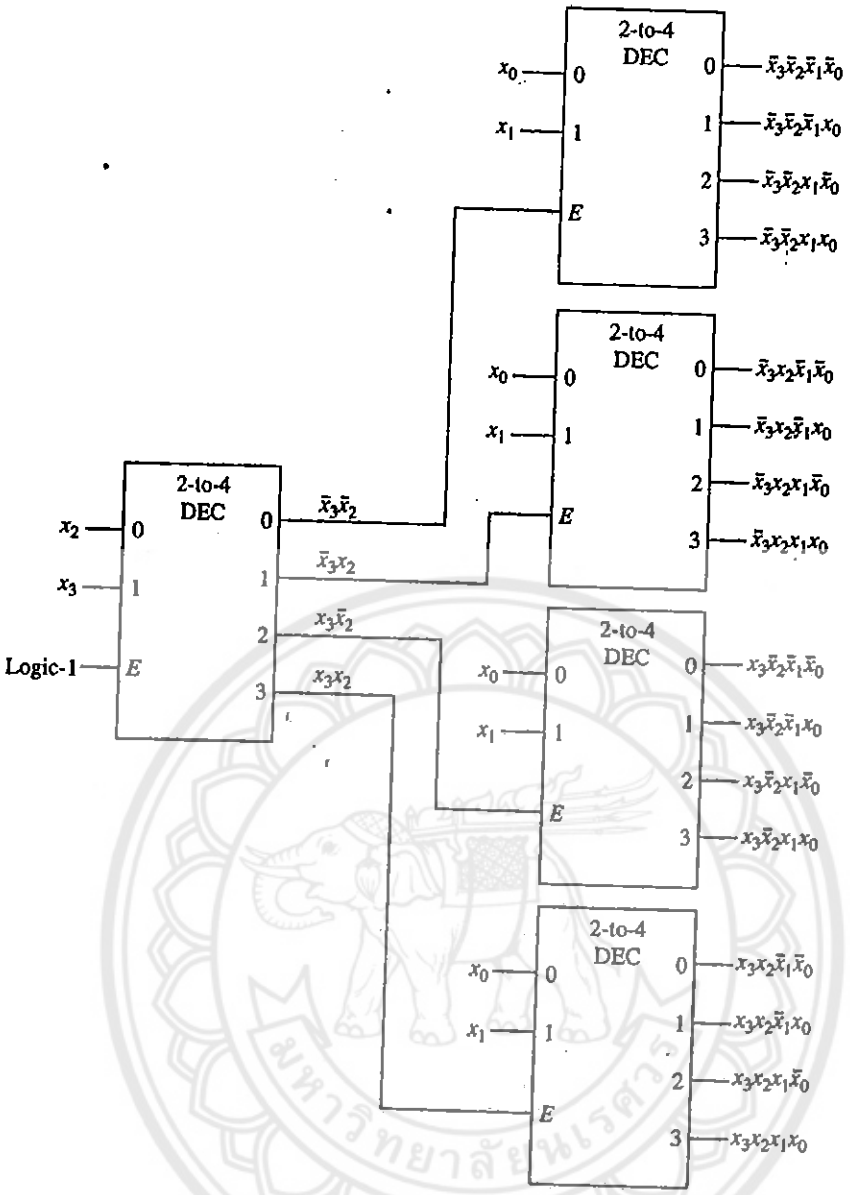
(ข) ตารางความจริง



(ค) วงจรถอดรหัส

รูปที่ 8.15 วงจรถอดรหัสแบบ 2: 4 line ที่มีอินพุตอินาเบิล ACTIVE LOW

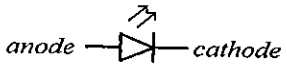
วงจรถอดรหัสขนาดเล็ก สามารถนำไปสร้างเป็นวงจรถอดรหัสที่มีขนาดใหญ่กว่าได้ ตัวอย่างเช่น วงจรถอดรหัส 2:4 line สามารถนำไปสร้างวงจรถอดรหัส 4:16 line ได้ดังรูปที่ 8.16 โดยใช้วงจรถอดรหัส 2:4 line จำนวน 5 วงจรมาต่อกัน เอาต์พุตของวงจรถอดรหัสทุกตัว สามารถเขียนอยู่ในรูปสมการบูลีนได้ จากรูปที่ 8.16 กำหนดให้วงจรถอดรหัสวงจรแรก มี $E = 1$ สามารถสร้างสมการเอาต์พุตในเทอมของตัวแปร x_2 และ x_3 ได้ทั้งหมด 4 สมการ สมการเอาต์พุตทั้ง 4 สมการ จะเป็นตัวกำหนดค่าอินาเบิลของวงจรถอดรหัสที่มีอินพุตเป็น x_1 และ x_0 จำนวน 4 วงจรต่อไป ดังนั้นจากวงจรทั้งหมดจะได้สมการเอาต์พุตในรูปมินเทอมของตัวแปร $x_3x_2x_1x_0$ จำนวน 16 สมการ



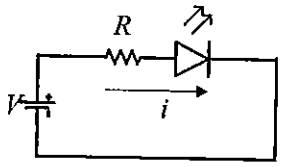
รูปที่ 8.16 วงจรถอดรหัส 4:16 line

(3) **วงจรถอดรหัส** 7 Segment

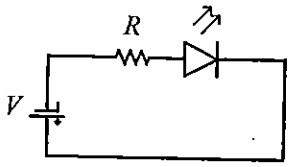
อุปกรณ์ที่ใช้ในการแสดงผลของวงจรดิจิตอลที่พบมากคือไดโอดชนิดเปล่งแสงได้ (Light Emitting Diode) หรือเรียกสั้นๆ ว่า LED หลักการทำงานของ LED จะเหมือนกับไดโอดชนิดทั่วไป คือควบคุมให้กระแสไหลไปในทิศทางเดียว แต่ที่แตกต่างจากไดโอดชนิดอื่นคือเมื่อ LED นำกระแส หรือมีกระแสไหลผ่าน LED จะเปล่งแสงออกมา โครงสร้างของ LED มี 2 ขาคืออาโนด (Anode) และคาโทด (Cathode) สัญลักษณ์และการทำงานของ LED แสดงดังรูปที่ 8.17



(ก) สัญลักษณ์



(ข) ไบอัสไปข้างหน้า (Forward Bias)



(ค) ไบอัสย้อนหลัง (Reverse bias)

รูปที่ 8.17 สัญลักษณ์และการทำงานของ LED

การไบอัสไปข้างหน้าจะมีกระแสไหลผ่าน LED ทำให้ LED เปล่งแสงออกมา โดยป้อนแรงดันไฟฟ้าที่ขาอาโนดมากกว่าขาคาโธด ถ้าแรงดันตกคร่อม LED มากกว่า 1.8 โวลต์จะทำให้กระแสไหลได้ ตัวต้านทานที่ต่ออนุกรมกับ LED จะทำหน้าที่จำกัดกระแสที่ไหลผ่าน LED ไม่ให้มีค่ามากเกินกว่าที่ LED ทนได้ ค่ากระแสสูงสุดที่ LED ทนได้ขึ้นอยู่กับ LED แต่ละเบอร์ ค่ากระแส i ที่ไหลผ่าน LED สามารถคำนวณได้จาก

$$i = \frac{V - 1.8}{R} \quad \text{แอมแปร์}$$

ตัวอย่างเช่น ถ้า $V = 5$ โวลต์และต้องการให้มีกระแสไหล 10 มิลลิแอมแปร์ ต้องใช้ตัวต้านทานมีค่าเท่ากับ

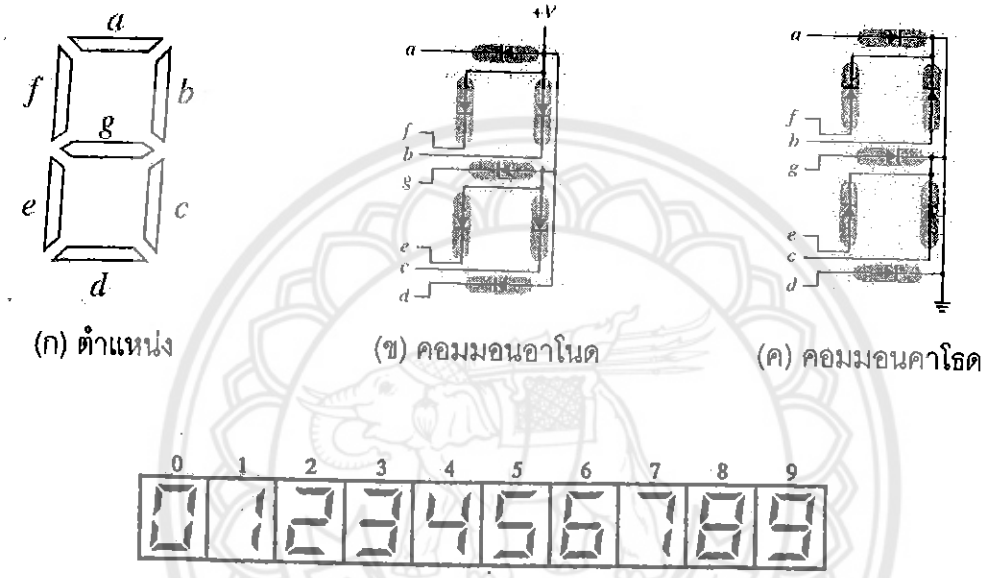
$$R = \frac{V - 1.8}{i} = \frac{5 - 1.8}{10 \times 10^{-3}} = 320 \text{ โอห์ม}$$

การไบอัสย้อนกลับจะไม่มีกระแสไหลผ่าน LED ทำให้ LED ไม่เปล่งแสงออกมา ในการใช้งานของ LED นอกจากจะใช้ตัวเดียวโดดๆ แล้ว ยังสามารถนำมาต่อเรียงกันเป็นเลข 8 ดังรูปที่ 8.18 เรียกว่า LED 7-Segment ซึ่งประกอบด้วย LED ทั้งหมด 7 ตัว แต่ละตัวเรียกว่าเซกเมนต์ (Segment) และมีชื่อกำกับเป็น a, b, c, d, e, f, g การกำหนดชื่อจะเรียงลำดับในทิศทางตามเข็มนาฬิกา เนื่องจากขา LED ทุกตัวจะต่อออกมาข้างนอก ทำให้มีขาจำนวนมาก จึงมีการรวมขาของ LED ทั้ง 7 ตัวเข้าด้วยกัน การรวมขามี 2 แบบ คือ

(1) แบบคอมมอนอาโนด (Common anode) จะนำขาอาโนดของ LED ทุกตัวมาต่อร่วมกัน การใช้งานจะต่อขานี้ไว้ที่แรงดันไฟเลี้ยง V_{cc} ส่วนขาคาโธดของแต่ละเซกเมนต์จะนำไปต่อกับเอาต์พุตของวงจรถอดรหัสที่มีเอาต์พุตแบบ ACTIVE LOW

(2) แบบคอมมอนคาโธด (Common cathode) จะนำขาคาโธดของ LED ทุกตัวมาต่อร่วมกัน การใช้งานจะต่อขานี้ไว้ที่กราวด์ (*GND*) ส่วนขาอาโนดของแต่ละเซกเมนต์จะนำไปต่อกับเอาต์พุตของวงจรถอดรหัสที่มีเอาต์พุตแบบ ACTIVE HIGH

จุดที่ต่อร่วมของขา LED ทุกตัวเรียกว่าจุดคอมมอนหรือจุดร่วม (Common) เมื่อป้อนแรงดันไฟฟ้าให้กับ LED ทุกตัว สามารถกำหนดให้ LED ดวงใดติดหรือดับได้ ดังนั้นสามารถแสดงเป็นค่าตัวเลขต่างๆ ได้ ดังรูปที่ 8.18 (ง)



(ก) ตำแหน่ง (ข) คอมมอนอาโนด (ค) คอมมอนคาโธด

(ง) การแสดงผลของ LED 7 Segment

รูปที่ 8.18 LED 7-Segment

ตารางที่ 8.1 การติดดับของ LED 7 segment

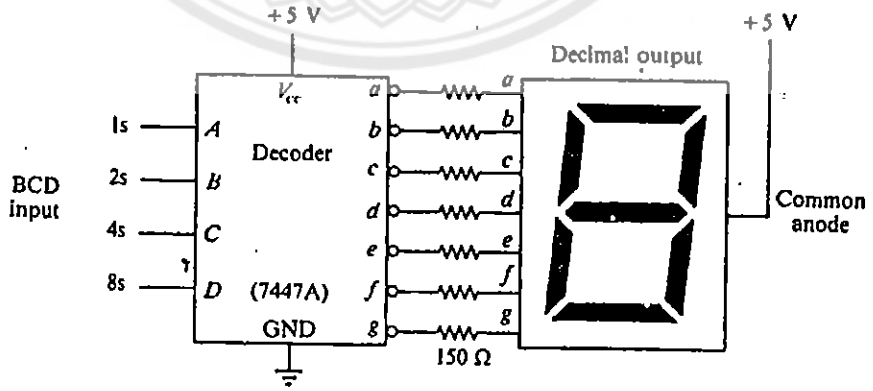
Decimal Input	Input Code	Output						
		a	b	c	d	e	f	g
0	0000	1	1	1	1	1	1	0
1	0001	0	1	1	0	0	0	0
2	0010	1	1	0	1	1	0	1
3	0011	1	1	1	1	0	0	1
4	0100	0	1	1	0	0	1	1
5	0101	1	0	1	1	0	1	1
6	0110	1	0	1	1	1	1	1
7	0111	1	1	1	0	0	0	0
8	1000	1	1	1	1	1	1	1
9	1001	1	1	1	1	0	1	1

การแสดงเลขฐานสิบ 0-9 ดังรูปที่ 8.18 (ง) ต้องมีวงจรถอดรหัสทำหน้าที่แปลงจากเลขฐานสองและทำให้ LED แต่ละส่วนมีการติดหรือดับ เพื่อแสดงค่าตัวเลข 0-9 จากตารางที่ 8.1 เมื่อแต่ละเซกเมนต์มีค่าลอจิกเป็น 1 แสดงว่า LED ติดและลอจิกเป็น 0 แสดงว่า LED ดับ จะเห็นว่า LED จะนำกระแสหรือติด เมื่อเอาต์พุตของวงจรถอดรหัสที่ต่อกับ LED นั้นมีค่าลอจิก 1 ดังนั้นการออกแบบวงจรถอดรหัสนี้จะใช้กับ LED 7 Segment แบบคอมมอนคาโธด

จากตารางที่ 8.1 เมื่อลดรูปสมการโดยใช้ KM จะได้สมการของแต่ละเซกเมนต์ดังนี้

$$a = A + C + BD + \overline{B}\,\overline{D} = A + C + (\overline{B \oplus D})$$
$$b = \overline{B} + \overline{C}\,\overline{D} + CD = \overline{B} + (\overline{C \oplus D})$$
$$c = B + \overline{C} + D$$
$$d = A + \overline{B}C + B\overline{C}D + C\overline{D} + \overline{B}\,\overline{D}$$
$$e = C\overline{D} + \overline{B}\,\overline{D}$$
$$f = A + B\overline{D} + B\overline{C} + \overline{C}\,\overline{D}$$
$$g = A + C\overline{D} + B\overline{C} + \overline{B}C = A + C\overline{D} + (B \oplus C)$$

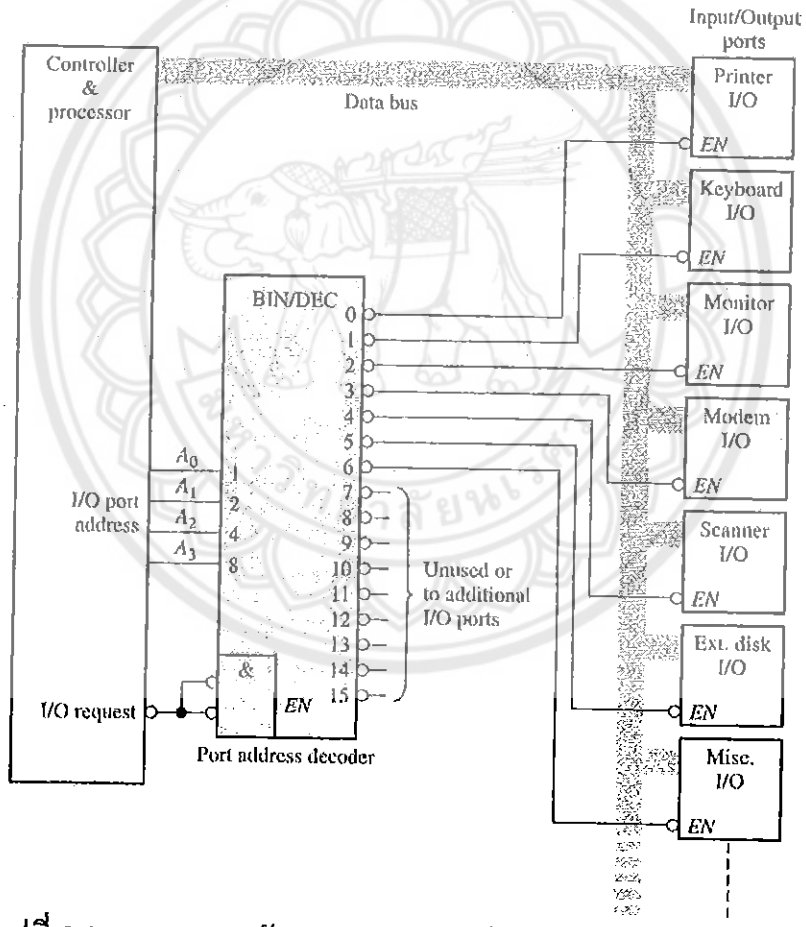
วงจรถอดรหัสที่ใช้ขับตัวแสดงผล LED 7 Segment ที่นิยมใช้กันได้แก่ ไอซีเบอร์ 7446, 7447, 7448, 7449 เป็นต้น ไอซีเบอร์ 7446 และ 7447 เป็นวงจรถอดรหัสแบบ ACTIVE LOW ใช้ขับตัวแสดงผล 7 Segment แบบคอมมอนอาโนด ไอซีเบอร์ 7448 และ 7449 เป็นวงจรถอดรหัสแบบ ACTIVE HIGH ใช้ขับตัวแสดงผล LED 7 Segment แบบคอมมอนคาโธด ตัวอย่างการใช้งานของไอซีเบอร์ 7447 เมื่อต่อกับตัวแสดงผล LED 7 Segment แบบคอมมอนอาโนด แสดงดังรูปที่ 8.19



รูปที่ 8.19 การต่อวงจรถอดรหัสกับ LED 7 Segment แบบคอมมอนอาโนด

(4) ตัวอย่างการประยุกต์ใช้งาน

ตัวอย่างวงจรการติดต่ออุปกรณ์อินพุต/เอาต์พุตของคอมพิวเตอร์ เนื่องจากคอมพิวเตอร์ต้องมีการติดต่อกับอุปกรณ์รอบนอก (External Device or Peripherals) โดยการติดต่อผ่านพอร์ต (I/O port) แต่ละพอร์ตจะมีค่าตำแหน่งไม่ซ้ำกันเรียกว่าแอดเดรส (address) เมื่อคอมพิวเตอร์ต้องการติดต่อสื่อสารกับอุปกรณ์ใด จะทำการส่งแอดเดรสพอร์ตของอุปกรณ์นั้นไปยังวงจรถอดรหัส ค่าแอดเดรสของพอร์ตจะอยู่ในรูปของเลขฐานสอง วงจรถอดรหัสจะแปลงเลขฐานสองและให้ค่าอินาเบิลมาขับอุปกรณ์ เมื่อติดต่ออุปกรณ์ได้แล้ว คอมพิวเตอร์จะทำการรับส่งข้อมูลผ่านบัสข้อมูล (Data bus) เช่น บัสข้อมูลขนาด 8 บิตประกอบด้วยสายสัญญาณจำนวน 8 เส้นขนานกัน ทำให้สามารถส่งข้อมูลทีละ 8 บิต ข้อมูลจะส่งไปยังอุปกรณ์ทุกตัว แต่ข้อมูลจะเข้าหรือออกเฉพาะพอร์ตที่วงจรได้ถอดรหัสเท่านั้น



รูปที่ 8.20 วงจรถอดรหัสแอดเดรสของพอร์ตอย่างง่ายในคอมพิวเตอร์

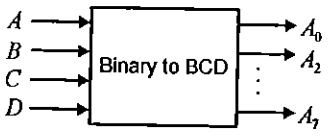
8.4 วงจรการแปลงรหัส (Code Converters)

การออกแบบวงจรที่ใช้ในการแปลงจากรหัสหนึ่งไปเป็นรหัสอื่นๆ เรียกว่าวงจรการแปลงรหัส (Code Converters)

ตัวอย่างที่ 8.1 การออกแบบวงจรแปลงรหัสเลขฐานสองขนาด 4 บิตเป็นรหัส BCD (Binary to BCD Converter)

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



(2) ตารางความจริง

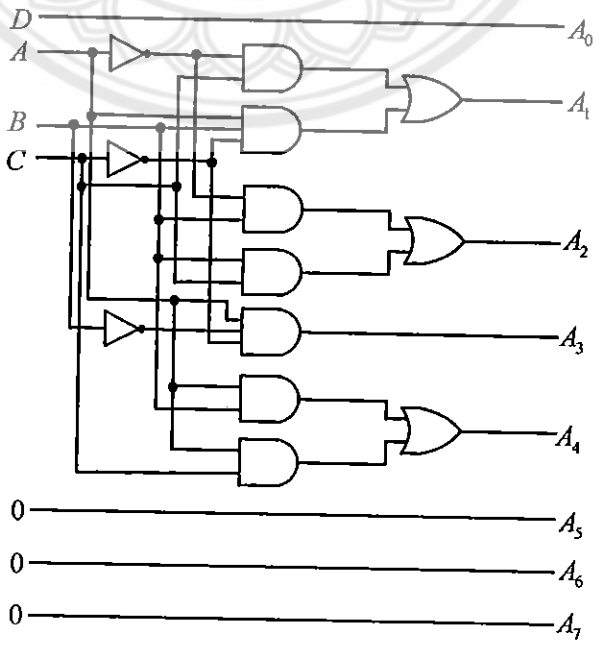
Decimal Digit	Input Binary ABCD	Output BCD Code $A_7A_6A_5A_4$ $A_3A_2A_1A_0$
0	0000	0000 0000
1	0001	0000 0001
2	0010	0000 0010
3	0011	0000 0011
4	0100	0000 0100
5	0101	0000 0101
6	0110	0000 0110
7	0111	0000 0111
8	1000	0000 1000
9	1001	0000 1001
10	1010	0001 0000
11	1011	0001 0001
12	1100	0001 0010
13	1101	0001 0011
14	1110	0001 0100
15	1111	0001 0101

(3) ลดรูปสมการโดยใช้ KM

$A_0 = D$ $\quad A_1 = \overline{A}C + AB\overline{C}$ $\quad A_2 = \overline{A}B + BC = B(\overline{A} + C)$ $\quad A_3 = A\overline{B}\overline{C}$

$A_4 = AB + AC$ $\quad A_5 = 0$ $\quad A_6 = 0$ $\quad A_7 = 0$

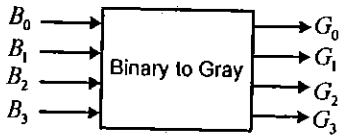
(4) เขียนวงจร



ตัวอย่างที่ 8.2 การออกแบบวงจรแปลงรหัสเลขฐานสองเป็นรหัสเกรย์ขนาด 4 บิต (Binary to Gray Code Converter)

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



(2) ตารางความจริง

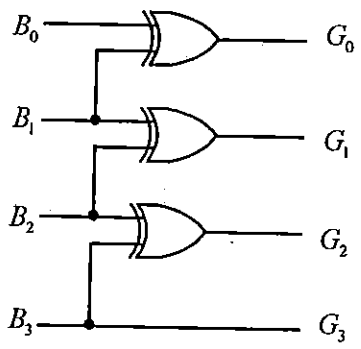
Decimal Digit	Input Binary $B_3B_2B_1B_0$	Output Gray code $G_3G_2G_1G_0$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0

(3) ลดรูปสมการโดยใช้ KM

$$G_3 = B_3 \qquad G_2 = B_3 \oplus B_2$$

$$G_1 = B_2 \oplus B_1 \qquad G_0 = B_1 \oplus B_0$$

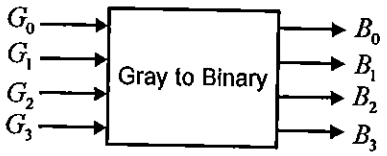
(4) เขียนวงจร



ตัวอย่างที่ 8.3 การออกแบบวงจรแปลงรหัสเกรย์เป็นเลขฐานสองขนาด 4 บิต (Gray to Binary Code Converter)

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



(2) ตารางความจริง

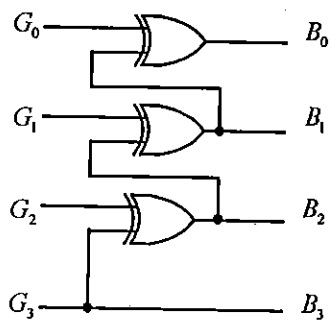
Decimal Digit	Input Gray $G_3G_2G_1G_0$	Output Binary $B_3B_2B_1B_0$
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	0 1 1 0
6	0 1 1 0	0 1 0 0
7	0 1 1 1	0 1 0 1
8	1 0 0 0	1 1 1 1
9	1 0 0 1	1 1 1 0
10	1 0 1 0	1 1 0 0
11	1 0 1 1	1 1 0 1
12	1 1 0 0	1 0 0 0
13	1 1 0 1	1 0 0 1
14	1 1 1 0	1 0 1 1
15	1 1 1 1	1 0 1 0

(3) ลดรูปสมการโดยใช้ KM

$$B_3 = G_3 \qquad B_2 = G_3 \oplus G_2$$

$$B_1 = G_3 \oplus G_2 \oplus G_1 \qquad B_0 = G_3 \oplus G_2 \oplus G_1 \oplus G_0$$

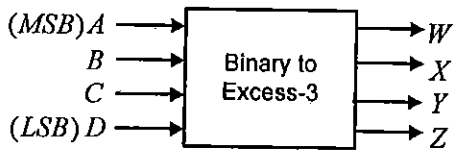
(4) เขียนวงจร



ตัวอย่างที่ 8.4 การออกแบบวงจรแปลงรหัส BCD เป็นรหัสเกิน 3 ขนาด 4 บิต (BCD to Excess-3 Code Converter)

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



(2) ตารางความจริง

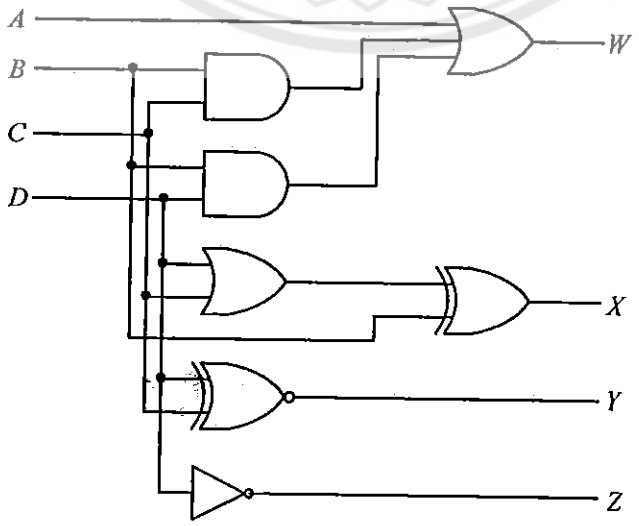
Decimal Digit	Input BCD (ABCD)	Output Excess-3 (WXYZ)
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

(3) ลดรูปสมการโดยใช้ KM

$$W = A + BD + BC \qquad X = B \oplus (D + C)$$

$$Y = \overline{C \oplus D} \qquad Z = \overline{D}$$

(4) เขียนวงจร



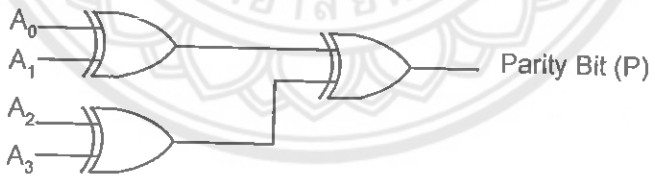
8.5 วงจรพาริตี้ (Parity)

ในบางระบบจำเป็นต้องมีการเพิ่มบิตเข้าไปในข้อมูล เพื่อใช้ในการตรวจสอบว่าข้อมูลที่มีการรับส่งระหว่างวงจรหนึ่งไปยังอีกวงจรหนึ่งว่าข้อมูลนั้นไม่มีการเปลี่ยนแปลง บิตที่ใส่เพิ่มเข้ามานี้เรียกว่าบิตพาริตี้ (Parity bit) ระบบสามารถเลือกใช้พาริตี้คู่หรือพาริตี้คี่ได้ ถ้าเลือกใช้พาริตี้คู่แสดงว่าบิตพาริตี้ที่เพิ่มเข้ามาในบิตข้อมูล ทำให้บิตทั้งหมด (บิตข้อมูล+บิตพารีตี้) มีลอจิก 1 เป็นจำนวนคู่ เช่นเดียวกันถ้าระบบเลือกใช้พาริตี้คี่แสดงว่าบิตพาริตี้ที่เพิ่มเข้ามาในบิตข้อมูล ทำให้บิตทั้งหมด (บิตข้อมูล+บิตพารีตี้) มีลอจิก 1 เป็นจำนวนคี่

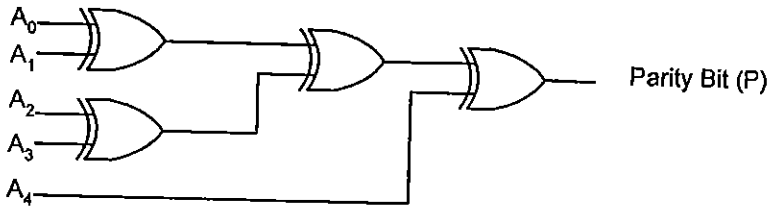
(1) **วงจรสร้างพาริตี้** (Parity Generator) พิจารณาคุณสมบัติของเอกซ์คลูซีฟออร์เกต ถ้าสมมุติให้เอาต์พุต Y คือบิตพาริตี้ ตัวแปรอินพุต A และ B มีค่าลอจิก 1 เป็นจำนวนคู่จะให้เอาต์พุต $Y = 0$ ตรงกันข้ามถ้า A และ B มีค่าลอจิก 1 เป็นจำนวนคี่จะให้เอาต์พุต $Y = 1$ ดังนั้นคุณสมบัติของเอ็กซ์คลูซีฟออร์เกตจึงเหมือนกับพาริตี้คู่ ดังนั้นเอ็กซ์คลูซีฟออร์เกตสามารถนำมาใช้ออกแบบวงจรสร้างพาริตี้คู่ได้

Input		Output
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

ตัวอย่างวงจรสร้างพาริตี้คู่ของข้อมูลขนาด 4 บิต ($A_3A_2A_1A_0$) และขนาด 5 บิต ($A_4A_3A_2A_1A_0$) แสดงดังรูปที่ 8.20 เมื่อ P คือบิตพาริตี้



(ก) วงจรสร้างพาริตี้คู่ของข้อมูลขนาด 4 บิต

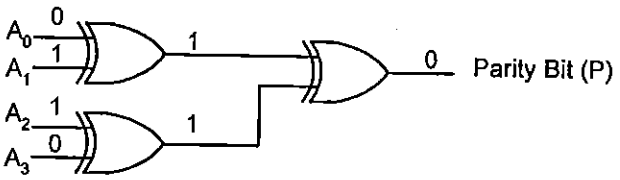


(ข) วงจรสร้างพาริตี้คู่ของข้อมูลขนาด 5 บิต

รูปที่ 8.21 วงจรสร้างพาริตี้คู่ของข้อมูล

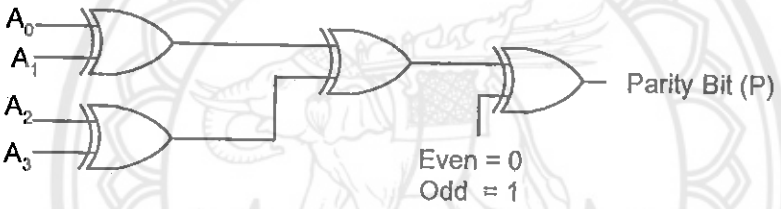
ตัวอย่างที่ 8.5 จงใช้วงจรสร้างพาริตี้คู่รูปที่ 8.21(ก) สร้างบิตพาริตี้ของข้อมูล $A_3A_2A_1A_0 = 0110$

<u>วิธีทำ</u>

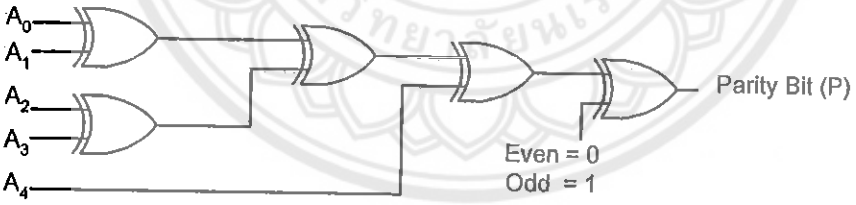


บิตข้อมูลที่กำหนดให้มีจำนวนบิตที่เป็น 1 เป็นจำนวนคู่ ดังนั้นบิตพาริตี้จึงมีค่าเป็น 0

การออกแบบวงจรสร้างพาริตี้คู่/คี่ที่อยู่ในวงจรเดียวกัน สามารถทำได้โดยเพิ่มเอกซ์คลูซีฟออร์เกตพิเศษในวงจร โดยกำหนดขาเอกซ์คลูซีฟออร์เกต 1 ขาเป็นขาควบคุม (Control input) ขาควบคุมนี้จะทำหน้าที่เป็นตัวกำหนดว่าจะให้วงจรสร้างพาริตี้คู่หรือคี่ ถ้าขาควบคุมเป็น 1 จะเป็นวงจรสร้างพาริตี้คี่และถ้าขาควบคุมเป็น 0 จะเป็นวงจรสร้างพาริตี้คู่ ดังรูปที่ 8.22



(ก) วงจรสร้างพาริตี้คู่/คี่ของข้อมูลขนาด 4 บิต

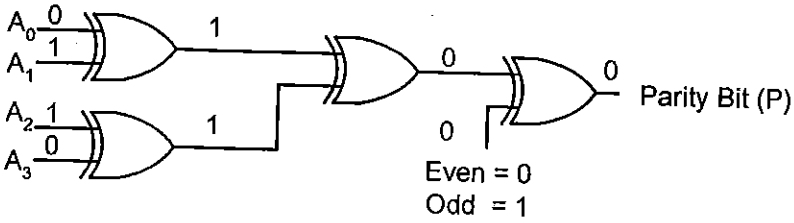


(ข) วงจรสร้างพาริตี้คู่/คี่ของข้อมูลขนาด 5 บิต

รูปที่ 8.22 วงจรสร้างพาริตี้คู่/คี่ของข้อมูล

ตัวอย่างที่ 8.6 จงใช้วงจรสร้างพาริตี้คู่/คี่รูปที่ 8.22(ก) สร้างบิตพาริตี้คู่ข้อมูล $A_3A_2A_1A_0 = 0110$

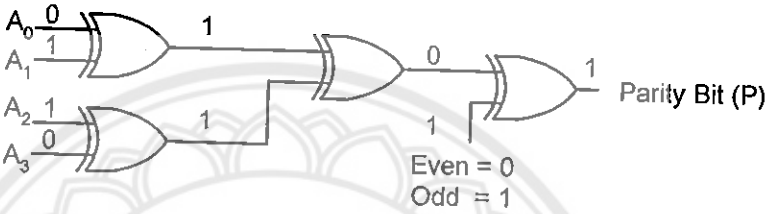
<u>วิธีทำ</u>



จากบิตข้อมูลที่กำหนดให้มีจำนวนบิตที่เป็น 1 จำนวนคู่ ดังนั้นบิตพาริตี้จึงมีค่าเป็น 0

ตัวอย่างที่ 8.7 จงใช้วงจรสร้างพาริตี้คู่/คี่รูปที่ 8.22(ก) สร้างบิตพาริตี้คี่ข้อมูล $A_3A_2A_1A_0 = 0110$

<u>วิธีทำ</u>

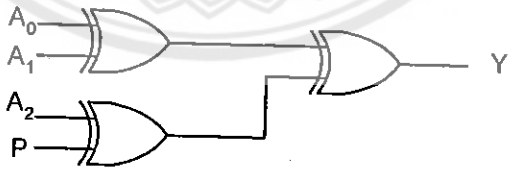


จากบิตข้อมูลที่กำหนดให้มีจำนวนบิตที่เป็น 1 จำนวนคู่ ดังนั้นบิตพาริตี้จึงมีค่าเป็น 1

(2) **วงจรตรวจสอบพาริตี้** (Parity Checker) วงจรที่สามารถแสดงได้ว่าจำนวนของบิตที่เป็น 1 มีจำนวนคู่หรือคี่เรียกว่าวงจรตรวจสอบพาริตี้ วงจรตรวจสอบพาริตี้กำหนดเงื่อนไขของเอาต์พุตดังนี้

$$Y = \begin{cases} 0 & even \quad parity \\ 1 & odd \quad parity \end{cases}$$

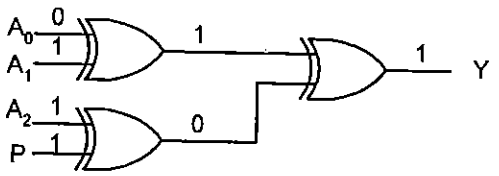
การกำหนดเงื่อนไขมาจากคุณสมบัติของเกตเอ็กซ์คลูซิฟออร์



รูปที่ 8.23 วงจรตรวจสอบพาริตี้ขนาด 4 บิต

ตัวอย่างที่ 8.8 สมมุติระบบเป็นพาริตี้คู่ เมื่อด้านรับได้รับข้อมูลจำนวน 4 บิต (บิตข้อมูล+บิตพาริตี้) เป็น $PA_2A_1A_0 = 1110$ จงใช้วงจรตรวจสอบพาริตี้รูปที่ 8.23 ว่าข้อมูลที่ได้มีความผิดพลาดเกิดขึ้นหรือไม่

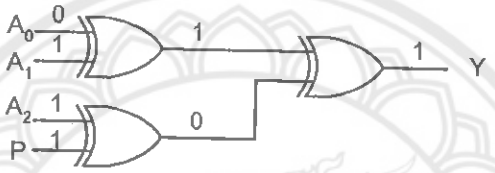
<u>วิธีทำ</u>



จากข้อมูลที่กำหนดให้ วงจรจะให้เอาต์พุตมีค่าเป็น 1 แสดงว่าเป็นพาริตี้คี่ แสดงว่าในระหว่างที่มีการรับส่งข้อมูลอาจมีบิตที่ผิดไปจำนวน 1 บิตได้

ตัวอย่างที่ 8.9 สมมุติระบบเป็นพาริตี้คี่ เมื่อด้านรับได้รับข้อมูลจำนวน 4 บิต (บิตข้อมูล+บิตพาริตี้) เป็น $PA_2A_1A_0 = 1110$ จงใช้วงจรตรวจสอบพาริตี้รูปที่ 8.23 ว่าข้อมูลที่ได้มีความผิดพลาดเกิดขึ้นหรือไม่

<u>วิธีทำ</u>



จากข้อมูลที่กำหนดให้ วงจรให้เอาต์พุตมีค่าเป็น 1 แสดงว่าเป็นพาริตี้คี่ แสดงว่าระบบไม่มีความผิดพลาดในการรับส่งข้อมูลเกิดขึ้น

(3) ตัวอย่างการออกแบบวงจรพาริตี้

สามารถออกแบบวงจรพาริตี้ต่างๆ ตามขั้นตอนการออกแบบวงจรทั่วไปได้

ตัวอย่างที่ 8.10 จงออกแบบวงจรสร้างพาริตี้คี่ของเลขฐานแปด 1 หลัก

<u>วิธีทำ</u> เลขฐานแปด 1 หลักเขียนแทนด้วยเลขฐานสองได้จำนวน 3 บิตและกำหนดให้เอาต์พุต P เป็นบิตพาริตี้

(1) ตารางความจริง

A	B	C	Parity bit (P)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

(2) จากตารางความจริงสามารถเขียนสมการ SOP และลดรูปสมการได้ดังนี้

$$
\begin{aligned}
P &= \sum m(0,3,5,6) \\
&= \overline{A}\,\overline{B}\,\overline{C} + \overline{A}BC + A\overline{B}C + AB\overline{C} \\
&= \overline{A}(\overline{B}\,\overline{C} + BC) + A(\overline{B}C + B\overline{C}) \\
&= \overline{A}(\overline{B \oplus C}) + A(B \oplus C) \\
&= \overline{A \oplus (B \oplus C)}
\end{aligned}
$$

(3) เขียนวงจร

ตัวอย่างที่ 8.11 จงออกแบบวงจรสร้างพาริตี้คู่ของเลขฐานแปด 1 หลัก

<u>วิธีทำ</u> เลขฐานแปด 1 หลักเขียนแทนด้วยเลขฐานสองจะได้เลขฐานสองจำนวน 3 บิตและกำหนดให้เอาต์พุต P เป็นบิตพาริตี้

(1) ตารางความจริง

A	B	C	Parity bit (P)
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(2) จากตารางความจริงสามารถเขียนสมการ SOP และลดรูปสมการได้ดังนี้

$$
\begin{aligned}
P &= \sum m(1,2,4,7) \\
&= \overline{\sum m(0,3,5,6)} \\
&= A \oplus (B \oplus C)
\end{aligned}
$$

(3) เขียนวงจร

ตัวอย่างที่ 8.12 จงออกแบบวงจรตรวจสอบพาริตี้ขนาด 3 บิต (บิตข้อมูล+บิตพาริตี้) เมื่อกำหนดเอาต์พุต Y ดังนี้

$$Y = \begin{cases} 0 & even \quad parity \\ 1 & odd \quad parity \end{cases}$$

<u>วิธีทำ</u>

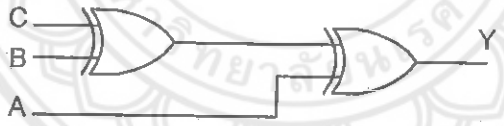
(1) กำหนดบิตข้อมูลและบิตพาริตี้คือ ABC ตามลำดับ และเอาต์พุต Y เป็นไปตามเงื่อนไขที่โจทย์กำหนด ดังนั้นสามารถเขียนตารางความจริงได้ดังนี้

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

(2) จากตารางความจริงเขียนสมการ SOP และลดรูปสมการได้ดังนี้

$$\begin{aligned} Y &= \sum m(1,2,4,7) \\ &= \overline{\sum m(0,3,5,6)} \\ &= A \oplus (B \oplus C) \end{aligned}$$

(3) เขียนวงจร



8.6 วงจรมัลติเพลกซ์และดีมัลติเพลกซ์ (Multiplexer and Demultiplexer)

(1) วงจรมัลติเพลกซ์

เรียกสั้นๆ ว่าวงจร MUX หรือวงจรเลือกข้อมูล (Data Selector) เป็นวงจรที่มีหลายอินพุต แต่มีเอาต์พุตเพียงเอาต์พุตเดียวเท่านั้น ทำหน้าที่เลือกข้อมูลอินพุตมาใช้เพียงครั้งละ 1 ข้อมูลเท่านั้น โดยมีตัวเลือกข้อมูล (Data Select) ทำหน้าที่ควบคุมการเลือกข้อมูลอินพุต ลักษณะของวงจร MUX แสดงดังรูปที่ 8.24 มีรายละเอียดดังนี้

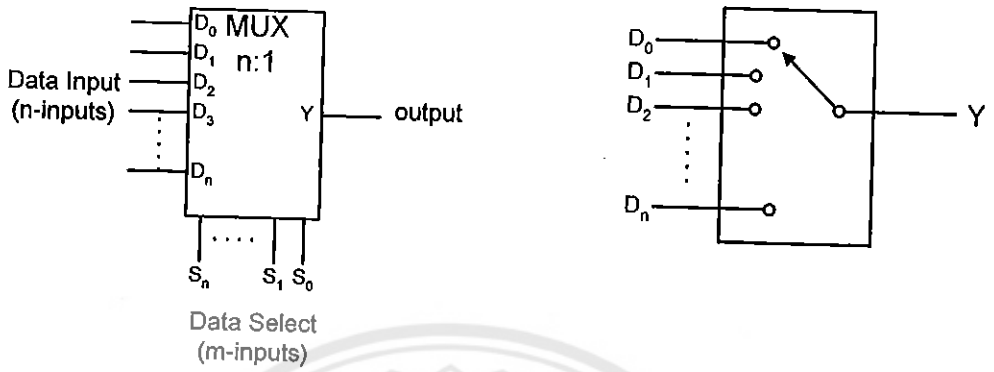
1.1 เอาต์พุตของ MUX มี 1 เอาต์พุต (Y) อาจเป็นแบบ ACTIVE LOW หรือ ACTIVE HIGH

1.2 ขา Data input ของ MUX มีหลายอินพุต เป็นตัวรับสัญญาณข้อมูล (Data) เข้ามา

1.3 ตัวลือกข้อมูล (Data select) ทำหน้าที่ในการเลือกข้อมูล จากขา Data input ไปใช้งาน

1.4 ขา Enable (E) อาจเป็นแบบ ACTIVE LOW หรือ ACTIVE HIGH

การกำหนดขนาดของวงจร MUX จะอยู่ในรูปของ $2^m:1$ *line MUX* หรือ $n:1$ *line MUX* เมื่อ $n=2^m$, $n = Data\ Input$ และ $m = Data\ Select$



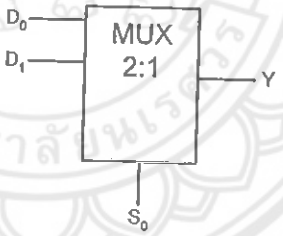
(ก) วงจร $2^m:1$ *line MUX* (ข) สวิตช์แบบหมุน (Rotary Switch)

รูปที่ 8.24 โครงสร้างและการทำงานของวงจร MUX

ตัวอย่างที่ 8.13 จงออกแบบวงจร $2:1$ *line MUX*

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



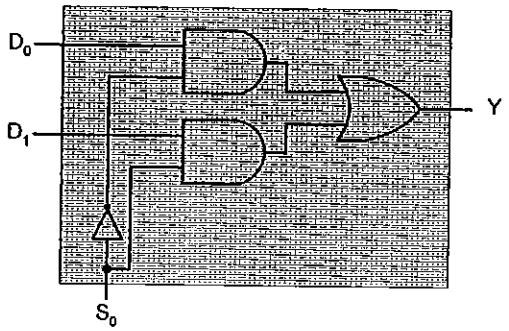
(2) ตารางความจริง

Data Select	Output
S_0	Y
0	D_0
1	D_1

(3) สมการ SOP จากตารางความจริง

$$Y = \overline{S}_0 D_0 + S_0 D_1$$

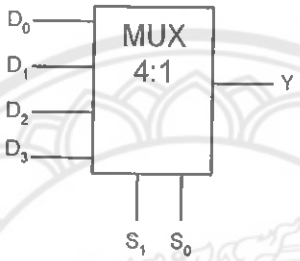
(4) เขียนวงจร



ตัวอย่างที่ 8.14 จงออกแบบวงจร 4:1 *line MUX*

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



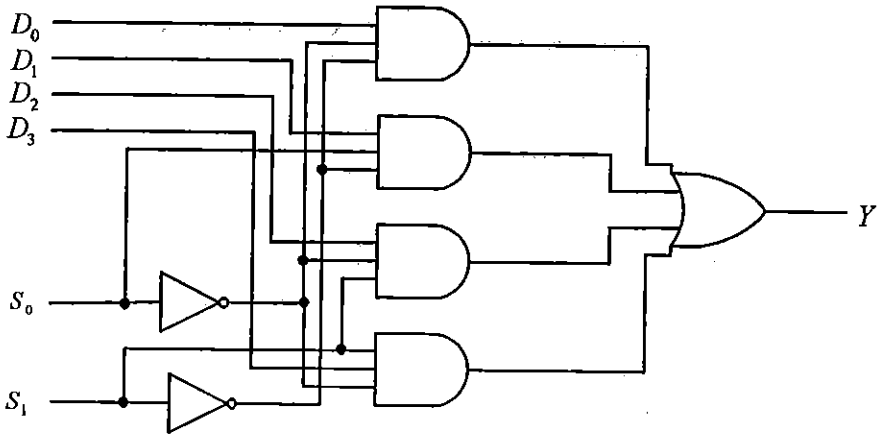
(2) ตารางความจริง

Data Select		Output
S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

(3) สมการ SOP จากตารางความจริง

$$Y = \overline{S}_1\overline{S}_0D_0 + \overline{S}_1S_0D_1 + S_1\overline{S}_0D_2 + S_1S_0D_3$$

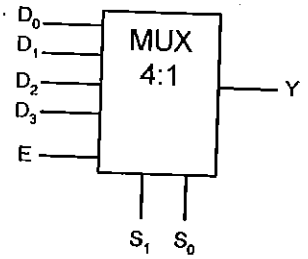
(4) เขียนวงจร



ตัวอย่างที่ 8.15 จงออกแบบวงจร 4:1 *line MUX* ที่มีอินพุตอินาเบิล ACTIVE HIGH

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



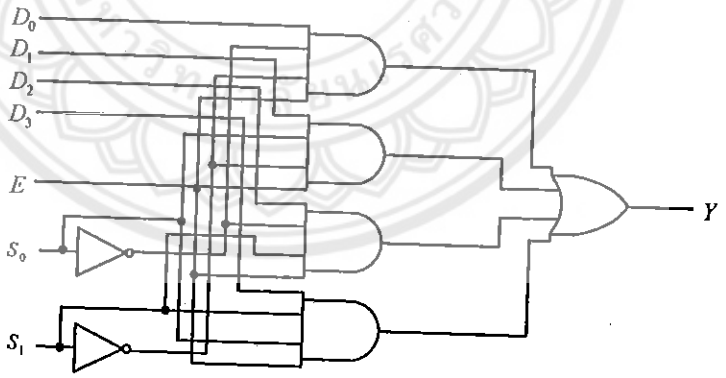
(2) ตารางความจริง

Data Select			output
E	S_1	S_0	Y
0	X	X	0
1	0	0	D_0
1	0	1	D_1
1	1	0	D_2
1	1	1	D_3

(3) สมการ SOP จากตารางความจริง

$$Y = E\bar{S}_1\bar{S}_0D_0 + E\bar{S}_1S_0D_1 + ES_1\bar{S}_0D_2 + ES_1S_0D_3$$
$$= (\bar{S}_1\bar{S}_0D_0 + \bar{S}_1S_0D_1 + S_1\bar{S}_0D_2 + S_1S_0D_3)E$$

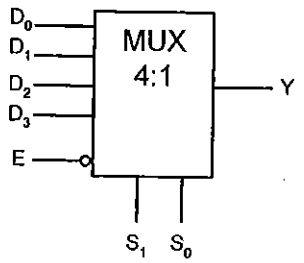
(4) เขียนวงจร



ตัวอย่างที่ 8.16 จงออกแบบวงจร 4:1 *line MUX* ที่มีอินพุตอินาเบิล ACTIVE LOW

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



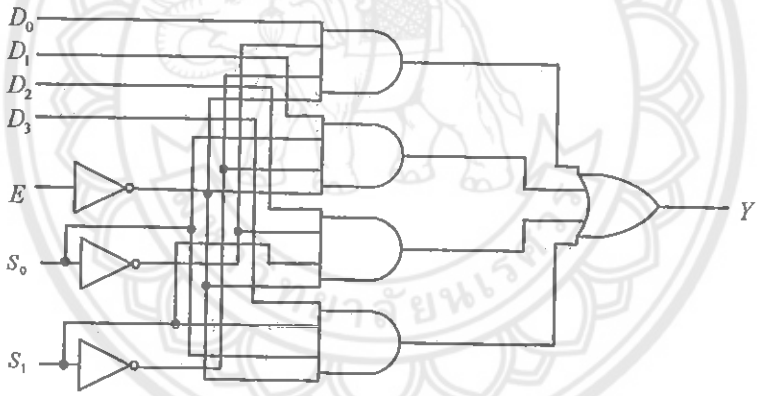
(2) ตารางความจริง

Data Select			Output
E	S1	S0	Y
0	0	0	D_0
0	0	1	D_1
0	1	0	D_2
0	1	1	D_3
1	X	X	0

(3) สมการ SOP จากตารางความจริง

$$Y = \overline{E}\overline{S}_1\overline{S}_0D_0 + \overline{E}\overline{S}_1S_0D_1 + \overline{E}S_1\overline{S}_0D_2 + \overline{E}S_1S_0D_3$$
$$= (\overline{S}_1\overline{S}_0D_0 + \overline{S}_1S_0D_1 + S_1\overline{S}_0D_2 + S_1S_0D_3)\overline{E}$$

(4) เขียนวงจร



(2) **วงจรดีมัลติเพลกซ์**

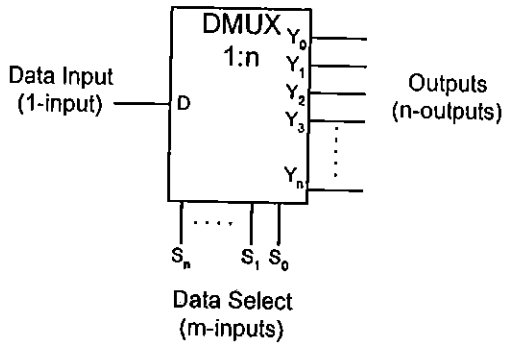
เรียกสั้นๆ ว่าวงจร DMUX หรือวงจรกระจายข้อมูล (Data Distributor) วงจรทำงานตรงกันข้ามกับวงจร MUX เป็นวงจรที่มีเพียงอินพุตเดียวแต่มีเอาต์พุตหลายเอาต์พุต ทำหน้าที่กระจายข้อมูลอินพุตออกไปยังเอาต์พุตใดเอาต์พุตหนึ่ง โดยมีตัวเลือก (Data Select) ควบคุมการเลือกข้อมูลอินพุตส่งไปยังเอาต์พุต ลักษณะของวงจร DMUX เป็นดังรูปที่ 8.25 รายละเอียดมีดังนี้

1.1 เอาต์พุตของ DMUX มีหลายเอาต์พุต (Y)

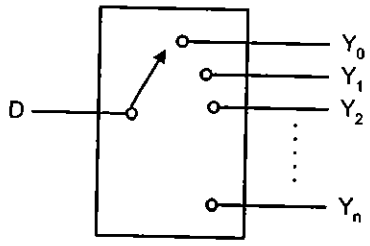
1.2 ขา Data input ของ DMUX มีเพียงอินพุตเดียว

1.3 ขา Data select เป็นขาทำหน้าที่ส่งข้อมูล Data input ไปยังเอาต์พุตใดเอาต์พุตหนึ่ง

1.4 ขา Enable (E) อาจเป็นแบบ ACTIVE LOW หรือ ACTIVE HIGH



(ก) วงจร $1:2^m$ *line DMUX*



(ข) สวิตซ์แบบหมุน (Rotary Switch)

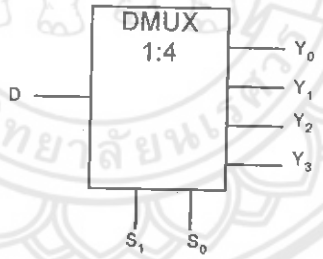
รูปที่ 8.25 โครงสร้างและการทำงานของวงจร DMUX

การกำหนดขนาดของวงจร DMUX จะอยู่ในรูปของ $1:2^m$ *line DMUX* หรือ $1:n$ *line DMUX* เมื่อ $n=2^m$, $n = Output$ และ $m = Data\ \ Select$

ตัวอย่างที่ 8.17 จงออกแบบวงจร $1:4$ *line DMUX*

<u>วิธีทำ</u>

(1) บล็อกไดอะแกรม



(2) ตารางความจริง

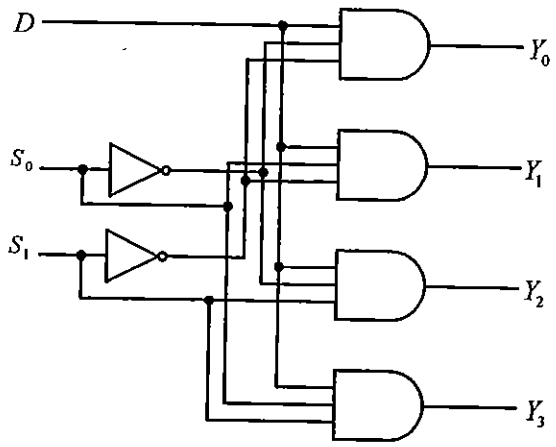
Data select		Output			
S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	**D**	0	0	0
0	1	0	**D**	0	0
1	0	0	0	**D**	0
1	1	0	0	0	**D**

(3) สมการ SOP จากตารางความจริง

$$Y_0 = \overline{S}_1\overline{S}_0 D \qquad Y_1 = \overline{S}_1 S_0 D$$

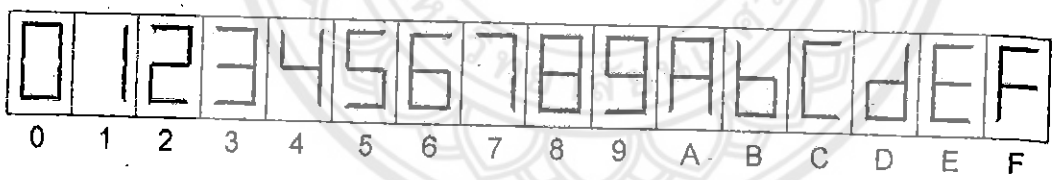
$$Y_2 = S_1\overline{S}_0 D \qquad Y_3 = S_1 S_0 D$$

(4) เขียนวงจร



แบบฝึกหัดท้ายบท

1. จงออกแบบวงจรถอดรหัสจาก BCD เป็นเลขฐานสิบโดยใช้แอนด์เกต
2. จงออกแบบวงจรถอดรหัส 3 : 8 line with Enable ACTIVE HIGH โดยใช้ NAND เกตเท่านั้น จากนั้นให้นำบล็อกไดอะแกรมวงจรถอดรหัส 3 : 8 line ที่ได้ ไปทำการสร้างวงจรถอดรหัส 4:16 line
3. จงออกแบบวงจรถอดรหัสเลขฐานสองขนาด 4 บิต ไปเป็นรหัส 7 ส่วน โดยให้แสดงผลดังรูปที่กำหนด โดยใช้ LED แบบ Common Anode กำหนดตัวแปรรหัสไบนารีขนาด 4 บิต เป็น ABCD



4. จงออกแบบวงจรมัลติเพลกซ์ขนาด 8:1 line
5. จงออกแบบวงจรดีมัลติเพลกซ์ 1:4 line ที่มีสัญญาณ Enable ควบคุมการทำงานแบบ Active LOW จากนั้นให้นำบล็อกไดอะแกรมของวงจรดีมัลติเพลกซ์ 1:4 line ไปสร้างเป็นวงจรดีมัลติเพลกซ์ 1:8 line

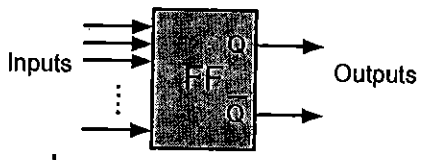
บทที่ 9
ฟลิปฟลอป (Flip-Flops)

วัตถุประสงค์

1. สามารถเข้าใจหลักการทำงานของฟลิปฟลอปแต่ละชนิด
2. สามารถเข้าใจหลักการทำงานของมาสเตอร์สลาฟฟลิปฟลอป
3. สามารถนำฟลิปฟลอปไปประยุกต์ใช้งานเบื้องต้นได้

9.1 บทนำ

อุปกรณ์มัลติไวเบรเตอร์ (Multivibrator) ได้แก่อุปกรณ์อะสเตเบิล (Astable) โมโนสเตเบิล (Monostable) และไบสเตเบิล (Bistable) อุปกรณ์อะสเตเบิลเป็นอุปกรณ์ที่ไม่มีสถานะเสถียรและถูกนำไปใช้ในวงจรออสซิลเลเตอร์ (Oscillator) เพื่อสร้างสัญญาณพัลส์รูปสี่เหลี่ยม (Square wave pulse) และนำมาใช้เป็นสัญญาณนาฬิกาในวงจรดิจิตอล อุปกรณ์โมโนสเตเบิลที่รู้จักกันดีคือวันช็อต (One-shot) จะมีเอาต์พุตที่เสถียรสถานะเดียวเท่านั้น ใช้สร้างสัญญาณพัลส์ที่ควบคุมความกว้างของสัญญาณเมื่อถูกกระตุ้น (Trigger) สำหรับอุปกรณ์ไบสเตเบิลสามารถแบ่งออกได้เป็น 2 ประเภท คือแลตช์ (Latch) และฟลิปฟลอป (Flip-Flop) แลตช์เป็นวงจรลอจิกที่ทำหน้าที่ในการเก็บข้อมูลชั่วคราว ประกอบด้วย 2 เอาต์พุต เอาต์พุตทั้งสองจะมีค่าตรงกันข้ามเสมอ ส่วนฟลิปฟลอปจะมีลักษณะเหมือนกับแลตช์ แต่แตกต่างกันที่สัญญาณควบคุม (Enable) หรือสัญญาณนาฬิกา (Clock) โดยฟลิปฟลอปจะถูกกระตุ้นด้วยขอบของสัญญาณนาฬิกา (Edge trigger) มากกว่าระดับของสัญญาณนาฬิกา (Level trigger) ในเนื้อหาบทนี้จะใช้เทอมของฟลิปฟลอปแทนแลตช์ด้วย อุปกรณ์ฟลิปฟลอปจะนำไปใช้ในการสร้างวงจรนับ (Counters) รีจิสเตอร์ (Register) วงจรลำดับและหน่วยความจำ เป็นต้น



รูปที่ 9.1 สัญลักษณ์ของฟลิปฟลอป

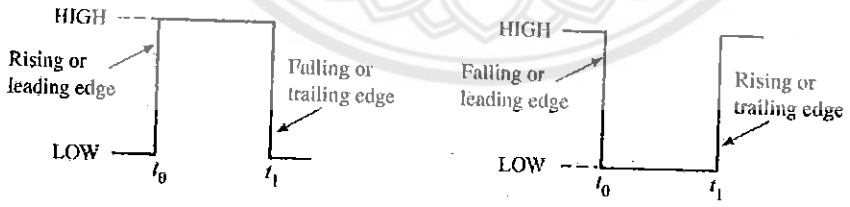
จากรูปที่ 9.1 แสดงสัญลักษณ์ทั่วไปของฟลิปฟลอป 1 ตัวจะเห็นได้ว่าฟลิปฟลอปประกอบด้วยอินพุตตั้งแต่ 1 อินพุตขึ้นไปและมีเอาต์พุต 2 เอาต์พุตคือ Q และ $\overline{Q}$ ซึ่งมีค่าตรงกันข้ามเสมอ สภาวะของเอาต์พุตที่สำคัญมีดังนี้

1. เมื่อ $Q = 1$ และ $\overline{Q} = 0$ เรียกว่าสภาวะ SET หรือ PRESET (PRE) หรือ ON
2. เมื่อ $Q = 0$ และ $\overline{Q} = 1$ เรียกว่าสภาวะ RESET หรือ CLEAR (CLR) หรือ OFF

9.2 สัญญาณนาฬิกาและพัลส์ (Clock and Pulse Signals)

สัญญาณนาฬิกาทำหน้าที่ควบคุมการทำงานของฟลิปฟลอป ซึ่งฟลิปฟลอปแต่ละตัวอาจจะมีหรือไม่มีก็ได้ โดยปกติฟลิปฟลอปจะมีการเปลี่ยนสภาวะของเอาต์พุตเมื่อมีการป้อนอินพุตโดยตรง ถ้าวงจรที่ออกแบบมีการใช้ฟลิปฟลอปต่อร่วมกันหลายๆ ตัวและต้องการให้ฟลิปฟลอปในวงจรมีการเปลี่ยนสภาวะพร้อมกัน จำเป็นต้องมีสัญญาณมาควบคุมการทำงานของฟลิปฟลอปเพื่อให้ ฟลิปฟลอปแต่ละตัวมีการเปลี่ยนสภาวะพร้อมกัน สัญญาณควบคุมดังกล่าวเรียกว่าสัญญาณนาฬิกา สัญลักษณ์ตัวย่อที่ใช้แทนสัญญาณนาฬิกาคือ CLK, CK, C หรือ CLP โดยทั่วไปสัญญาณนาฬิกาจะมีลักษณะเป็นรูปคลื่นสี่เหลี่ยม (Square Wave) ที่มีความต่อเนื่องและมีคาบเวลาแน่นอนหรือมีความถี่ค่าหนึ่ง

สัญญาณที่ส่งออกมาในช่วงเวลาสั้นๆ เพื่อกระตุ้นให้ระบบทำงานอย่างใดอย่างหนึ่งเรียกว่าสัญญาณพัลส์ (Pulse) ซึ่งแบ่งออกเป็น 2 แบบคือสัญญาณพัลส์แบบบวก (Positive Pulse) และสัญญาณพัลส์แบบลบ (Negative Pulse) จากรูปที่ 9.2 แสดงสัญญาณพัลส์ในอุดมคติ (Ideal Pulse) ที่ใช้เวลาในการเปลี่ยนสภาวะลอจิกจาก t_0 ถึง t_1 มีค่าเท่ากับศูนย์หรือการเปลี่ยนสภาวะลอจิกเป็นอย่างทันทีทันใด (Instantaneous) จุดที่มีการเปลี่ยนสัญญาณจากสภาวะลอจิก "0" เป็น "1" เรียกว่าขอบนำ (Leading Edge) หรือขอบขาขึ้น (Rising Edge) ส่วนจุดที่มีการเปลี่ยนสัญญาณจากสภาวะลอจิก "1" เป็น "0" เรียกว่าขอบตาม (Trailing Edge) หรือขอบขาลง (Falling Edge)



(ก) สัญญาณพัลส์แบบบวก (ข) สัญญาณพัลส์แบบลบ

รูปที่ 9.2 สัญญาณพัลส์ในอุดมคติ (Ideal Pulse)

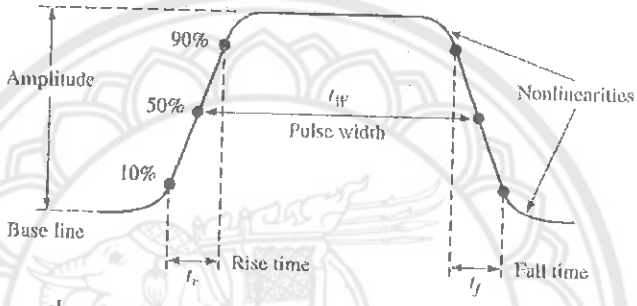
แม้ว่าในระบบดิจิตอลจะสมมุติเป็นสัญญาณพัลส์ในอุดมคติ แต่ในทางปฏิบัติสัญญาณพัลส์จะไม่ได้เกิดขึ้นในทันทีทันใดหรือไม่สามารถเปลี่ยนระดับของแรงดันไฟฟ้าได้อย่างทันทีทันใด แต่จะใช้เวลาช่วงหนึ่งในการเปลี่ยนสภาวะลอจิกดังรูปที่ 9.3 และกำหนดนิยามของสัญญาณใหม่ได้ดังนี้

เวลาขึ้น (Rise time, t_r) เป็นเวลาที่ใช้ในการเปลี่ยนจากสภาวะลอจิก "0" เป็น "1" โดยเริ่มจากแอมปลิจูด (Amplitude) ของพัลส์มีค่าเป็น 10% จนถึง 90%

เวลาลง (Fall time, t_f) เป็นเวลาที่ใช้ในการเปลี่ยนจากสภาวะลอจิก "1" เป็น "0" โดยเริ่มจากแอมปลิจูด (Amplitude) ของพัลส์มีค่าเป็น 90% จนถึง 10%

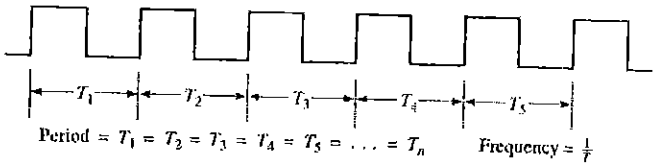
จากรูปที่ 9.3 แอมปลิจูดของสัญญาณพัลส์ที่เหลืออีก 10% ทั้งด้านบนและด้านล่างจะไม่รวมอยู่ในช่วงเวลาขึ้นหรือเวลาลงเนื่องจากช่วงนี้สัญญาณจะมีลักษณะไม่เป็นเชิงเส้น (Nonlinear)

ความกว้างของพัลส์ (Pulse width, t_W) ถ้าสัญญาณพัลส์ในอุดมคติจะวัดจากขอบขาขึ้นถึงขอบขาลง แต่ถ้าสัญญาณพัลส์ในทางปฏิบัติจะวัดเวลาจากขอบขาขึ้นถึงขอบขาลงที่แอมปลิจูดของสัญญาณเท่ากับ 50%



รูปที่ 9.3 สัญญาณพัลส์ในทางปฏิบัติ (Nonideal Pulse)

ถ้าสัญญาณพัลส์มีการเกิดขึ้นซ้ำกันเป็นคาบ (Period) อย่างต่อเนื่องเรียกว่าสัญญาณรูปสี่เหลี่ยม (Square signal) ถ้ากำหนดให้ T คือคาบเวลาของสัญญาณซึ่งเป็นเวลาตั้งแต่ขอบขาขึ้นถึงขอบขาลงของสัญญาณพัลส์ ถ้าคาบเวลาของสัญญาณพัลส์มีค่าคงที่เรียกว่าสัญญาณพัลส์แบบเป็นคาบ (Periodic) แต่ถ้าคาบเวลาของสัญญาณพัลส์แต่ละพัลส์มีค่าไม่คงที่เรียกว่าสัญญาณพัลส์แบบไม่เป็นคาบ (Nonperiodic) ดังรูปที่ 9.4



(ก) สัญญาณพัลส์แบบเป็นคาบ

(ข) สัญญาณพัลส์แบบไม่เป็นคาบ

รูปที่ 9.4 สัญญาณพัลส์แบบต่างๆ

เมื่อพิจารณาสัญญาณพัลส์แบบเป็นคาบ เวลาใน 1 คาบเรียกว่า 1 ไซเคิล (Cycle) ประกอบด้วยสภาวะลอจิก "1" และ "0" อย่างละ 1 ช่วง ถ้าสัญญาณพัลส์แบบเป็นคาบมีค่าคาบเวลา T วินาที ดังนั้นจากรูปที่ 9.4 (ก) สัญญาณมีความถี่ (Frequency, f) ในหน่วยเฮิรตซ์ (Hertz, Hz) หรือจำนวนไซเคิลต่อ 1 วินาที ดังนั้นสามารถหาความสัมพันธ์ระหว่างความถี่และคาบเวลาได้จาก

$$f = \frac{1}{T} \quad \text{หรือ} \quad T = \frac{1}{f}$$

อัตราส่วนระหว่างความกว้างของสัญญาณพัลส์กับคาบเวลาเรียกว่าค่าดิวตี้ไซเคิล (Duty Cycle) หรือรอบของการทำงาน หน่วยเป็นเปอร์เซ็นต์ หาได้จาก

$$Duty\ Cycle = \frac{T_w}{T} \times 100\%$$

ตัวอย่างที่ 9.1 กำหนดสัญญาณพัลส์ดังรูป จงหาค่าคาบเวลา ความถี่และค่าดิวตี้ไซเคิล

<u>วิธีทำ</u>

(ก) คาบเวลาหาได้จากเวลาเริ่มต้นของพัลส์ลูกหนึ่งไปจนถึงเวลาเริ่มต้นของพัลส์ลูกถัดไป จากรูป $T = 10 \quad ms$

(ข) ความถี่ของสัญญาณหาได้จาก

$$f = \frac{1}{T} = \frac{1}{10ms} = 100 \quad Hz$$

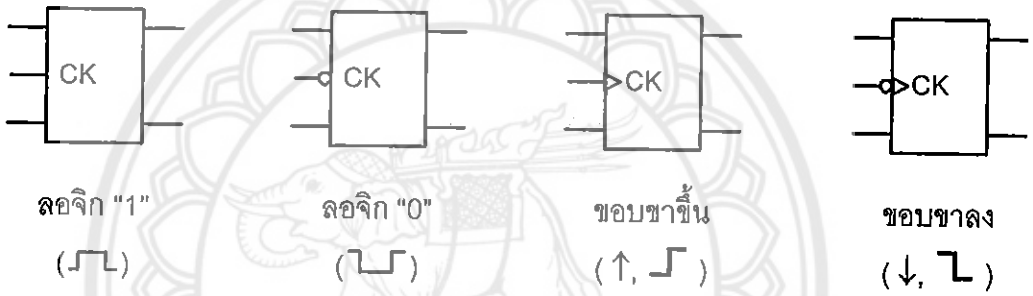
(ค) ค่าดิวตี้ไซเคิลหาได้จาก

$$Duty\ Cycle = \frac{T_w}{T} \times 100\% = \frac{1ms}{10ms} \times 100\% = 10\%$$

อุปกรณ์ฟลิปฟลอปที่มีขาควบคุม (Enable) จะมีการทำงานเมื่อขาควบคุมถูกกระตุ้นด้วยสัญญาณนาฬิกา ลักษณะการกระตุ้นการทำงานของฟลิปฟลอปสามารถแบ่งได้ 2 แบบคือ

1. การกระตุ้นด้วยระดับของสัญญาณ (Level Trigger) ลักษณะของสัญญาณเป็นลอจิก "1" หรือ "0" สำหรับตารางความจริงที่มีอินพุตของสัญญาณนาฬิกาจะใช้สัญลักษณ์ ⎍ และ ⎌ ตามลำดับ

2. การกระตุ้นด้วยขอบของสัญญาณ (Edge Trigger) เครื่องหมายสามเหลี่ยมที่ขาสัญญาณนาฬิกาหมายถึงฟลิปลอปทำงานเมื่อถูกกระตุ้นด้วยขอบขาขึ้น ถ้ามีเครื่องหมายสามเหลี่ยมและวงกลม (Bubbles) หมายถึงฟลิปลอปทำงานเมื่อถูกกระตุ้นด้วยขอบขาลง สำหรับตารางความจริงที่มีอินพุตของสัญญาณนาฬิกาจะใช้สัญลักษณ์ ↑, ⎍ และ ↓, ⎌ แทนสัญญาณนาฬิกาที่กระตุ้นที่ขอบขาขึ้นและขาลงตามลำดับ



รูปที่ 9.5 สัญลักษณ์ขาของสัญญาณนาฬิกาในฟลิปลอป

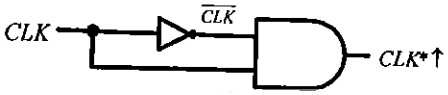
วงจรสร้างสัญญาณพัลส์แคบ ๆ (Edge Detector)

ฟลิปฟลอปที่มีสัญญาณควบคุม จะทำงานเมื่อสัญญาณควบคุมมีสภาวะลอจิกเป็น "0" หรือ "1" บางครั้งฟลิปฟลอปต้องการทำงานในช่วงเวลาสั้นๆ ดังนั้นเมื่อสัญญาณมีการเปลี่ยนจากสภาวะลอจิก "0" เป็น "1" (ขอบขาขึ้น) หรือสัญญาณมีการเปลี่ยนจากสภาวะลอจิก "1" เป็น "0" (ขอบขาลง) สามารถทำให้สัญญาณมีช่วงเวลาสั้นมากๆ เหมือนกับการทำงานนั้นเกิดขึ้นที่ขอบของสัญญาณควบคุม การทำสัญญาณให้มีช่วงเวลาสั้นๆ นั้นจะใช้คุณสมบัติของการหน่วงเวลาของอุปกรณ์ (Delay time) จากวงจรดังรูปที่ 9.6 (ก) และ (ข) สมมุติให้นอตเกตมีเวลาหน่วงสำหรับการทำงานภายในเป็นเวลา Δt โดยทั่วไปค่า $\Delta t = 20$ ns ส่วนแอนด์เกตและออร์เกตไม่มีเวลาหน่วง

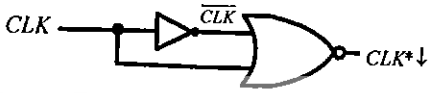
วงจรการสร้างสัญญาณที่ขอบขาขึ้นประกอบด้วยนอตเกตและแอนด์เกต เมื่อป้อนสัญญาณ CLK ที่อินพุตของนอตเกตจะให้เอาต์พุตเป็น $\overline{CLK}$ โดยการเปลี่ยนสภาวะลอจิกของนอตเกตต้องใช้เวลาเท่ากับ Δt และเอาต์พุตของแอนด์เกตเป็น $CLK^* \uparrow$ โดยสัญญาณ $CLK^* \uparrow$

จะมีลอจิก "1" เมื่อสัญญาณ CLK มีการเปลี่ยนสภาวะลอจิกจาก "0" เป็น "1" แสดงไดอะแกรมเวลาได้ดังรูปที่ 9.6 (ค)

วงจรการสร้างสัญญาณที่ขอบขาลงประกอบด้วยนอตเกตและนอร์เกต เมื่อป้อนสัญญาณ CLK ที่อินพุตของนอตเกตจะให้เอาต์พุตเป็น $\overline{CLK}$ โดยการเปลี่ยนสภาวะลอจิกของนอตเกตต้องใช้เวลาเท่ากับ Δt และเอาต์พุตของนอร์เกตเป็น $CLK^* \downarrow$ โดยสัญญาณ $CLK^* \downarrow$ จะมีลอจิก "1" เมื่อสัญญาณ CLK มีการเปลี่ยนสภาวะลอจิกจาก "1" เป็น "0" แสดงไดอะแกรมเวลาได้ดังรูปที่ 9.6 (ค)



(ก) สัญญาณที่ขอบขาขึ้น



(ข) สัญญาณที่ขอบขาลง

(ค) การสร้างสัญญาณที่ขอบขาขึ้นและขอบขาลง

รูปที่ 9.6 วงจรและการสร้างขอบของสัญญาณนาฬิกา

9.3 ชนิดของฟลิปฟลอป

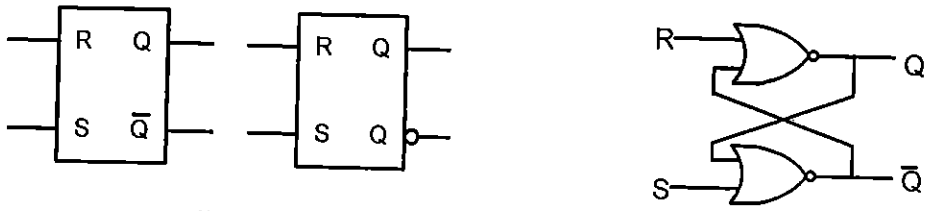
ฟลิปฟลอปเป็นอุปกรณ์ที่สร้างมาจากเกตและให้เอาต์พุต 2 เอาต์พุตเท่านั้นคือ Q และ $\overline{Q}$ ซึ่งเอาต์พุตทั้งสองจะต้องมีค่าตรงกันข้ามเสมอ ส่วนอินพุตของฟลิปฟลอปแต่ละชนิดจะแตกต่างกันขึ้นอยู่กับชนิดของฟลิปฟลอปนั้น สามารถแบ่งฟลิปฟลอปเป็น 4 ชนิดคือ RS, D, JK และ T ฟลิปฟลอป รายละเอียดมีดังนี้

9.3.1 RS ฟลิปฟลอป (RS Flip-Flop)

SR ฟลิปฟลอปเป็นฟลิปฟลอปพื้นฐานที่สุด อินพุตของ SR ฟลิปฟลอปมี 2 อินพุตคือ R และ S ที่มาจากคำว่า RESET และ SET ตามลำดับ วงจร RS ฟลิปฟลอปสามารถสร้างจากนอร์หรือแนนด์เกต ดังนี้

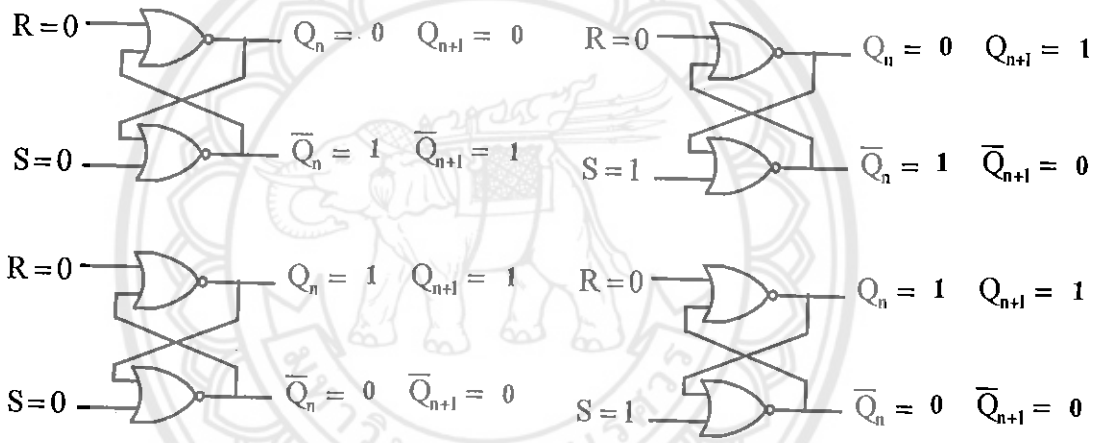
(1) RS ฟลิปฟลอปแบบนอร์เกต

วงจรของ RS ฟลิปฟลอปประกอบด้วยนอร์เกต 2 ตัว มาต่อไขว้กันเรียกว่าการต่อไขว้แบบป้อนกลับ (Cross-coupled Feedback) โดยเอาต์พุตของนอร์เกตตัวหนึ่งจะป้อนกลับเป็นอินพุตให้กับนอร์เกตอีกตัวหนึ่ง ลักษณะอินพุตเป็นแบบ ACTIVE HIGH



(ก) สัญลักษณ์ (ข) วงจร

รูปที่ 9.7 RS ฟลิปฟลอปแบบนอร์เกต



(ก) กรณี $R = 0, S = 0$ (ข) กรณี $R = 0, S = 1$

รูปที่ 9.8 ตัวอย่างการทำงาน RS ฟลิปฟลอปแบบนอร์เกต

ถ้ากำหนดให้ Q_n คือเอาต์พุตเดิมและ Q_{n+1} คือเอาต์พุตที่เกิดขึ้นหลังจากที่วงจรทำงานไปแล้ว จากวงจรดังรูปที่ 9.7 (ข) สามารถเขียนสมการบูลีนเอาต์พุตของนอร์เกตได้ดังนี้

$$Q_{n+1} = \overline{R + \overline{Q}_n} \quad \text{และ} \quad \overline{Q}_{n+1} = \overline{S + Q_n}$$

สามารถอธิบายการทำงานของวงจร RS ฟลิปฟลอปได้ดังนี้

จากรูปที่ 9.8 (ก) เมื่อกำหนดให้อินพุต $R = 0, S = 0$ ถ้าเอาต์พุตเดิมเป็น $Q = 0, \overline{Q} = 1$ หลังจากวงจรทำงานไปแล้วจะให้เอาต์พุตเหมือนเดิมเป็น $Q = 0, \overline{Q} = 1$ ถ้าเอาต์พุตเดิมเป็น

$Q=1,\overline{Q}=0$ หลังจากวงจรทำงานไปแล้วจะให้เอาต์พุตเหมือนเดิมเป็น $Q=1,\overline{Q}=0$ นั่นคือเมื่ออินพุต $R=0,S=0$ จะให้เอาต์พุต Q และ $\overline{Q}$ ยังคงสภาวะเดิมไม่มีการเปลี่ยนแปลง

จากรูปที่ 9.8 (ข) เมื่อกำหนดให้อินพุต $R=0,S=1$ ถ้าเอาต์พุตเดิมเป็น $Q=0,\overline{Q}=1$ หลังจากวงจรทำงานไปแล้วจะให้เอาต์พุตเปลี่ยนเป็น $Q=1,\overline{Q}=0$ ถ้าเอาต์พุตเดิมเป็น $Q=1,\overline{Q}=0$ หลังจากวงจรทำงานไปแล้วจะให้เอาต์พุตเหมือนเดิมเป็น $Q=1,\overline{Q}=0$ นั่นคือเมื่ออินพุต $R=0,S=1$ จะให้เอาต์พุตเป็น $Q=1$ และ $\overline{Q}=0$ ซึ่งเป็นสภาวะ SET

ทำนองเดียวกันเมื่อกำหนดให้อินพุต $R=1,S=0$ และ $R=1,S=1$ สามารถเขียนตารางการทำงานของวงจรได้ดังตารางที่ 9.1 และเขียนตารางความจริงได้ดังตารางที่ 9.2

ตารางที่ 9.1 ตารางการทำงานของวงจร

Input		Output				โหมดการทำงานของเอาต์พุต (Mode)
R	S	Q_n	$\overline{Q}_n$	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	0	1	0	1	ไม่เปลี่ยนแปลง (No change, Latch, Hold)
		1	0	1	0	
0	1	0	1	1	0	เซต (SET)
		1	0	1	0	
1	0	0	1	0	1	รีเซต (RESET)
		1	0	0	1	
1	1	0	1	0	0	ไม่ใช้งาน (Unused, Invalid : $Q=\overline{Q}=0$)
		1	0	0	0	

ตารางที่ 9.2 ตารางความจริงที่นำไปใช้งาน

Input		Output
R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	-

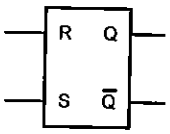
(2) RS ฟลิปฟลอปแบบแนนด์เกต

วงจรของ RS ฟลิปฟลอปประกอบด้วยแนนด์เกต 2 ตัว มาต่อไขว้กันเรียกว่าการต่อไขว้แบบป้อนกลับ อินพุตของ RS ฟลิปฟลอปแบบแนนด์เกตมี 2 แบบ คืออินพุตแบบ ACTIVE HIGH และ ACTIVE LOW

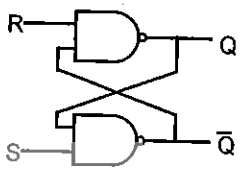
RS ฟลิปฟลอปอินพุตแบบ ACTIVE HIGH

สัญลักษณ์ โครงสร้างวงจรและตารางความจริงแสดงดังรูปที่ 9.9 โดยสามารถวิเคราะห์การทำงานของวงจรได้เช่นเดียวกันกับ RS ฟลิปฟลอปแบบนอร์เกต สามารถเขียนสมการบูลีนเอาต์พุตของแนนด์เกตได้ดังนี้

$$Q_{n+1} = \overline{R \cdot \overline{Q}_n} \quad \text{และ} \quad \overline{Q}_{n+1} = \overline{S \cdot Q_n}$$



(ก) สัญลักษณ์



(ข) วงจร

Input		Output		โหมดการทำงานของเอาต์พุต (Mode)
R	S	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	1	1	ไม่ใช้งาน (Unused, Invalid : $Q = \overline{Q} = 1$)
0	1	1	0	เซต (SET)
1	0	0	1	รีเซต (RESET)
1	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change, Latch, Hold)

(ค) ตารางความจริง

Input		Output
R	S	Q_{n+1}
0	0	-
0	1	1
1	0	0
1	1	Q_n

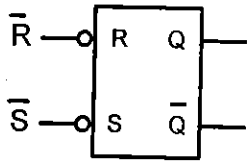
(ง) ตารางความจริงที่นำไปใช้งาน

รูปที่ 9.9 RS ฟลิปฟลอปที่มีอินพุตแบบ ACTIVE HIGH

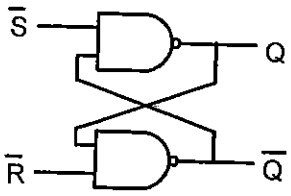
RS ฟลิปฟลอปอินพุตแบบ ACTIVE LOW

สัญลักษณ์ โครงสร้างวงจรและตารางความจริงแสดงดังรูปที่ 9.10 สามารถเขียนสมการบูลีนเอาต์พุตของแนนด์เกตได้ดังนี้

$$Q_{n+1} = \overline{\overline{S} \cdot \overline{Q}_n} \quad \text{และ} \quad \overline{Q}_{n+1} = \overline{\overline{R} \cdot Q_n}$$



(ก) สัญลักษณ์



(ข) วงจร

Input				Output		โหมดการทำงานของเอาต์พุต (Mode)
R	S	$\overline{R}$	$\overline{S}$	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	1	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change, Latch, Hold)
0	1	1	0	1	0	เซต (SET)
1	0	0	1	0	1	รีเซต (RESET)
1	1	0	0	1	1	ไม่ใช้งาน (Unused, Invalid, $Q = \overline{Q} = 1$)

(ค) ตารางความจริง

Input		Output
$\overline{R}$	$\overline{S}$	Q_{n+1}
0	0	-
0	1	0
1	0	1
1	1	Q_n

(ง) ตารางความจริงที่นำไปใช้งาน

รูปที่ 9.10 RS ฟลิปฟลอปที่มีอินพุตแบบ ACTIVE LOW

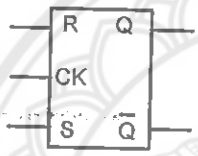
จาก RS ฟลิปฟลอปแบบนอร์เกตและแนนด์เกตสามารถสรุปความแตกต่างได้ดังนี้

1. RS ฟลิปฟลอปแบบนอร์เกตจะมีอินพุตแบบเดียวคือ ACTIVE HIGH ส่วนแบบแนนด์เกตจะมีอินพุต ACTIVE HIGH และ ACTIVE LOW

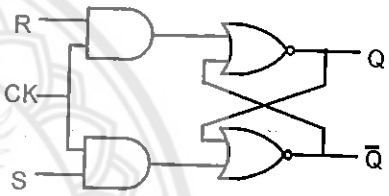
2. การทำงานของ RS ฟลิปฟลอปแบบนอร์เกตจะให้เอาต์พุตที่สภาวะไม่ใช้งานเป็น $Q = \overline{Q} = 0$ ส่วน RS ฟลิปฟลอปแบบแนนด์เกตจะให้เอาต์พุตที่สภาวะไม่ใช้งานเป็น $Q = \overline{Q} = 1$

(3) RS ฟลิปฟลอปแบบมีสัญญาณควบคุม

RS ฟลิปฟลอปแบบนอร์และแนนด์เกตที่ได้อธิบายในข้อ (1) และ (2) เมื่อป้อนอินพุตที่ขา R และ S จะมีการทำงานทันที แต่การใช้งานของวงจรดิจิตอลทั่วไป จำเป็นต้องมีการควบคุมการทำงานด้วยสัญญาณนาฬิกาเพื่อให้ฟลิปฟลอปหรืออุปกรณ์ดิจิตอลในวงจรทั้งหมดทำงานพร้อมกัน ดังนั้นวงจร RS ฟลิปฟลอปจะเพิ่มเกตเข้าไปในวงจร จากรูปที่ 9.11 (ข) เพิ่มแอนด์เกตในวงจร RS ฟลิปฟลอปแบบนอร์เกตและรูปที่ 9.12 (ข) เพิ่มแนนด์เกตในวงจร RS ฟลิปฟลอปแบบแนนด์เกตที่มีอินพุต ACTIVE LOW ฟลิปฟลอปจะทำงานเมื่อถูกกระตุ้นด้วยระดับของสัญญาณลอจิก "1" ถ้าต้องการให้ฟลิปฟลอปทำงานเมื่อถูกกระตุ้นด้วยขอบของสัญญาณ จะนำวงจรสร้างสัญญาณพัลส์แคบๆ (Edge detector) ที่กล่าวในข้อ 9.2 มาใส่ในวงจรฟลิปฟลอป สามารถแก้ไขวงจรให้ทำงานที่ขอบขาขึ้นและขอบขาลงได้



(ก) สัญลักษณ์

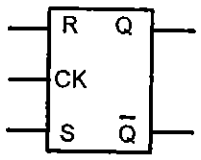


(ข) วงจร

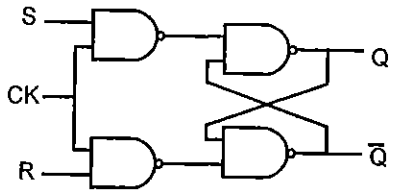
Input			Output		โหมดการทำงานของเอาต์พุต (Mode)
CK	R	S	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	0	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	1	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	1	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	1	1	0	เซต (SET)
1	1	0	0	1	รีเซต (RESET)
1	1	1	0	0	ไม่ใช้งาน (Unused, Invalid : $Q = \overline{Q} = 0$)

(ค) ตารางความจริง

รูปที่ 9.11 RS ฟลิปฟลอปที่มีสัญญาณควบคุม



(ก) สัญลักษณ์



(ข) วงจร

Input			Output		โหมดการทำงานของเอาต์พุต (Mode)
CK	R	S	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	0	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	1	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	1	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	1	1	0	เซต (SET)
1	1	0	0	1	รีเซต (RESET)
1	1	1	1	1	ไม่ใช้งาน (Unused, Invalid : $Q = \overline{Q} = 1$)

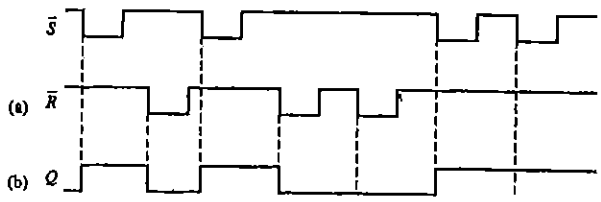
(ค) ตารางความจริง

รูปที่ 9.12 RS ฟลิปฟลอปที่มีสัญญาณควบคุม

ตารางที่ 9.3 ตารางความจริงที่นำไปใช้งาน

Input			Output
CK	R	S	Q_{n+1}
0	X	X	Q_n
1	0	0	Q_n
1	0	1	1
1	1	0	0
1	1	1	-

ตัวอย่างที่ 9.2 ถ้ากำหนดรูปคลื่นของสัญญาณอินพุต $\overline{R}$ และ $\overline{S}$ ของ RS ฟลิปฟลอปดังรูปที่ 9.13 (a) จงหาเอาต์พุต Q สมมุติให้เอาต์พุต Q มีค่าเริ่มต้นเป็น LOW

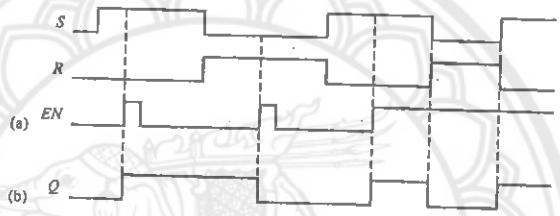


รูปที่ 9.13

<u>วิธีทำ</u>

เอาต์พุต Q ที่ได้แสดงดังรูปที่ 9.13 (b)

ตัวอย่างที่ 9.3 ถ้ากำหนดรูปคลื่นของสัญญาณอินพุต R , S และสัญญาณควบคุม (EN) ของ RS ฟลิปฟลอปดังรูปที่ 9.14 (a) จงหาเอาต์พุต Q สมมุติให้เอาต์พุต Q มีค่าเริ่มต้นเป็น RESET

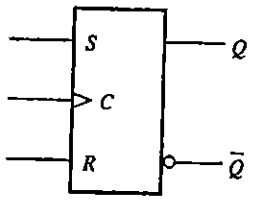


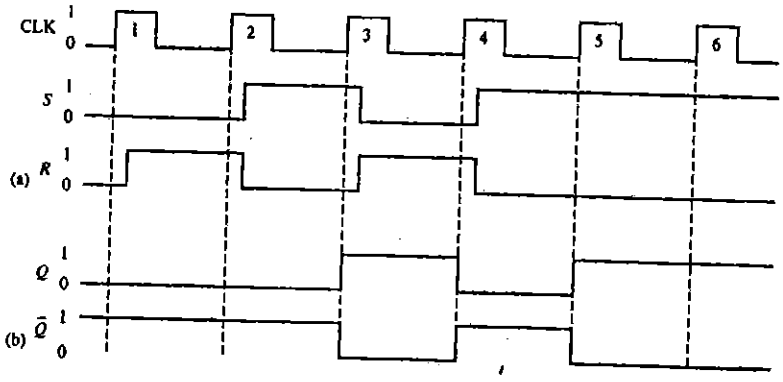
รูปที่ 9.14

<u>วิธีทำ</u>

เอาต์พุต Q ที่ได้แสดงดังรูปที่ 9.14 (b) จะเห็นได้ว่าที่เวลาใดๆ เมื่อ S เป็น HIGH และ R เป็น LOW และ EN เป็น HIGH จะให้เอาต์พุต Q เป็น SET ถ้า S เป็น LOW และ R เป็น HIGH และ EN เป็น HIGH จะให้เอาต์พุต Q เป็น RESET

ตัวอย่างที่ 9.4 ถ้ากำหนดรูปคลื่นของสัญญาณอินพุต R , S , CLK ของ RS ฟลิปฟลอปดังรูปที่ 9.15 (a) จงหาเอาต์พุต Q และ $\overline{Q}$ สมมุติให้เอาต์พุต Q มีค่าเริ่มต้นเป็น RESET





รูปที่ 9.15

<u>วิธีทำ</u>

เอาต์พุต Q และ $\overline{Q}$ ที่ได้แสดงดังรูปที่ 9.15 (b) สามารถอธิบายการทำงานได้ดังนี้

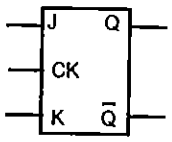
1. สัญญาณนาฬิกาลูกที่ 1 เมื่ออินพุต S เป็น LOW และ R เป็น LOW ดังนั้นเอาต์พุต Q ไม่เปลี่ยนแปลง
2. สัญญาณนาฬิกาลูกที่ 2 เมื่ออินพุต S เป็น LOW และ R เป็น HIGH ดังนั้นเอาต์พุต Q ยังคงเป็น LOW (RESET)
3. สัญญาณนาฬิกาลูกที่ 3 เมื่ออินพุต S เป็น HIGH และ R เป็น LOW ดังนั้นเอาต์พุต Q เปลี่ยนเป็น HIGH (SET)
4. สัญญาณนาฬิกาลูกที่ 4 เมื่ออินพุต S เป็น LOW และ R เป็น HIGH ดังนั้นเอาต์พุต Q เปลี่ยนเป็น LOW (RESET)
5. สัญญาณนาฬิกาลูกที่ 5 เมื่ออินพุต S เป็น HIGH และ R เป็น LOW ดังนั้นเอาต์พุต Q เปลี่ยนเป็น HIGH (SET)
6. สัญญาณนาฬิกาลูกที่ 6 เมื่ออินพุต S เป็น HIGH และ R เป็น LOW ดังนั้นเอาต์พุต Q ยังคงเป็น HIGH (SET)

9.3.2 JK ฟลิปฟลอป (JK Flip-Flop)

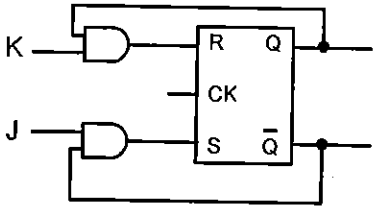
JK ฟลิปฟลอปเป็นฟลิปฟลอปที่นิยมใช้ในการออกแบบวงจรดิจิตอลมาก สามารถนำไปสร้างเป็นฟลิปฟลอปชนิดอื่นได้ด้วย แต่ข้อจำกัดของ RS ฟลิปฟลอปที่มีสภาวะไม่ใช้งานหรือให้เอาต์พุต $Q = \overline{Q}$ จึงมีการแก้ไขวงจร RS ฟลิปฟลอปใหม่โดยเพิ่มแอนด์เกตดังรูปที่ 9.16 (ข) เพื่อให้สามารถใช้งานเมื่อกำหนดอินพุตได้ทุกกรณี สำหรับ RS ฟลิปฟลอปที่ใช้ในวงจรนี้เป็น RS ฟลิปฟลอปแบบนอร์เกตและมีสัญญาณนาฬิกาควบคุมการทำงาน สามารถเขียนสมการบูลีนได้ดังนี้

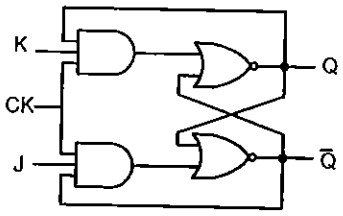
$$R = KQ_n \qquad S = J\overline{Q}_n$$

$$Q_{n+1} = \overline{R + \overline{Q}_n} \qquad \overline{Q}_{n+1} = \overline{S + Q_n}$$



(ก) สัญลักษณ์





(ข) วงจร

Input			Output		โหมดการทำงานของเอาต์พุต (Mode)
CK	J	K	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	0	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	1	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
0	1	1	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	0	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	1	0	1	รีเซต (RESET)
1	1	0	1	0	เซต (SET)
1	1	1	$\overline{Q}_n$	Q_n	เปลี่ยนสภาวะตรงกันข้าม (Toggle)

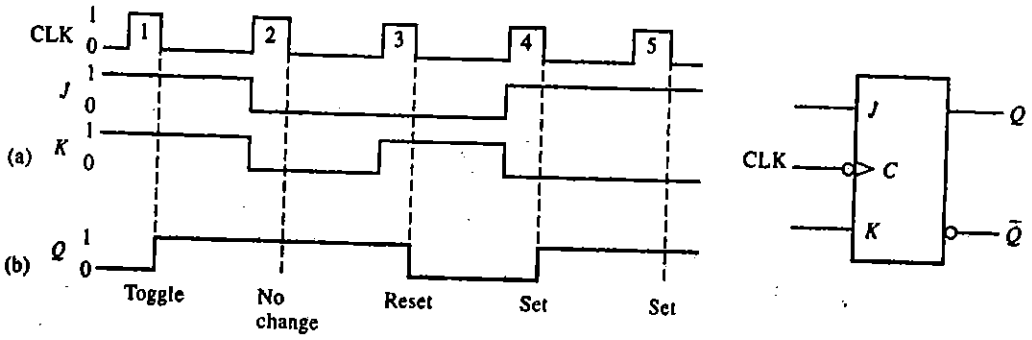
(ค) ตารางความจริง

รูปที่ 9.16 JK ฟลิปฟลอป

ตารางที่ 9.4 ตารางความจริงที่นำไปใช้งาน

Input			Output
CK	J	K	Q_{n+1}
0	X	X	Q_n
1	0	0	Q_n
1	0	1	0
1	1	0	1
1	1	1	$\overline{Q}_n$

ตัวอย่างที่ 9.5 ถ้ากำหนดรูปคลื่นของสัญญาณอินพุต J, K, CLK ของ JK ฟลิปฟลอปดังรูปที่ 9.17(a) จงหาเอาต์พุต Q สมมุติให้ฟลิปฟลอปเริ่มต้นเป็น RESET



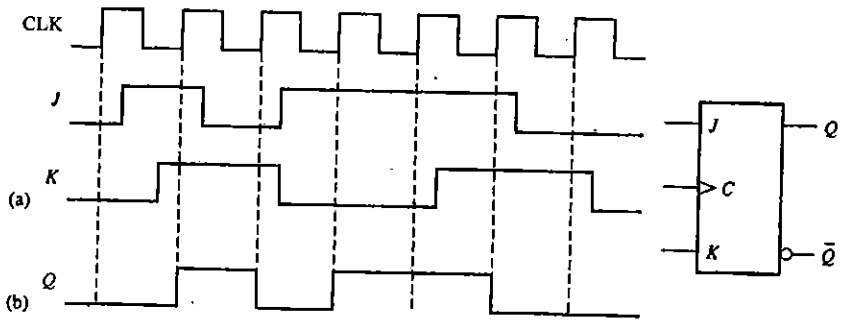
รูปที่ 9.17

<u>วิธีทำ</u>

จาก JK ฟลิปฟลอปที่กำหนดให้ จะทำงานเมื่อถูกกระตุ้นด้วยขอบขาลงของสัญญาณนาฬิกา ดังนั้นเอาต์พุต Q จะเปลี่ยนแปลงที่ขอบขาลงเท่านั้น จากอินพุต JK ที่กำหนดให้ จะได้เอาต์พุต Q ดังรูปที่ 9.17 (b) สามารถอธิบายการทำงานได้ดังนี้

1. ที่สัญญาณนาฬิกาลูกที่ 1 เมื่ออินพุต J และ K เป็น HIGH ดังนั้นเอาต์พุต Q จะ Toggle เปลี่ยนเป็น HIGH
2. ที่สัญญาณนาฬิกาลูกที่ 2 เมื่ออินพุต J และ K เป็น LOW ดังนั้นเอาต์พุต Q ไม่เปลี่ยนแปลง
3. ที่สัญญาณนาฬิกาลูกที่ 3 เมื่ออินพุต J เป็น LOW และ K เป็น HIGH ดังนั้นเอาต์พุต Q เปลี่ยนเป็น LOW (RESET)
4. ที่สัญญาณนาฬิกาลูกที่ 4 เมื่ออินพุต J เป็น HIGH และ K เป็น LOW ดังนั้นเอาต์พุต Q เปลี่ยนเป็น HIGH (SET)
5. ที่สัญญาณนาฬิกาลูกที่ 5 เมื่ออินพุต J เป็น HIGH และ K เป็น LOW ดังนั้นเอาต์พุต Q ยังคงเป็น HIGH (SET)

ตัวอย่างที่ 9.6 ถ้ากำหนดรูปคลื่นของสัญญาณอินพุต J, K, CLK ของ JK ฟลิปฟลอปดังรูปที่ 9.18(a) จงหาเอาต์พุต Q สมมุติให้ฟลิปฟลอปเริ่มต้นเป็น RESET



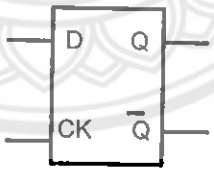
รูปที่ 9.18

<u>วิธีทำ</u>

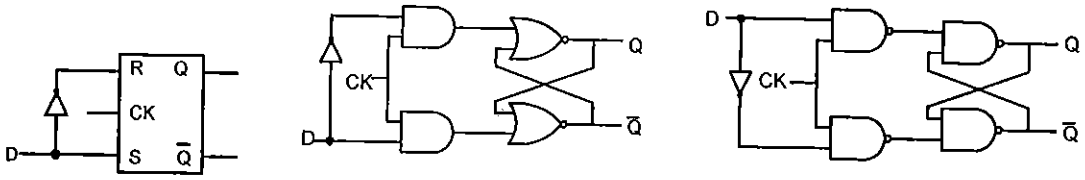
จาก JK ฟลิปฟลอปที่กำหนดให้ จะทำงานเมื่อถูกกระตุ้นด้วยขอบขาขึ้นของสัญญาณนาฬิกา ดังนั้นเอาต์พุต Q จะเปลี่ยนแปลงที่ขอบขาขึ้นเท่านั้น การเปลี่ยนแปลงของอินพุต JK หลังขอบขาขึ้นของสัญญาณนาฬิกาจะไม่มีผลต่อเอาต์พุต จากอินพุต JK ที่กำหนดให้ จะได้เอาต์พุต Q แสดงดังรูปที่ 9.18 (b)

9.3.3 D ฟลิปฟลอป (D Flip-Flop)

D ฟลิปฟลอปจะนำมาใช้ในการเก็บข้อมูล สามารถเก็บข้อมูลได้ 1 บิต อินพุตของ D ฟลิปฟลอปมีเพียงอินพุตเดียวเท่านั้นคือ D (Data, Delay) ซึ่งเป็นข้อมูลที่ต้องการเก็บไว้ในฟลิปฟลอป สัญญาณควบคุมการทำงานของ D ฟลิปฟลอปมีทั้งแบบกระตุ้นด้วยระดับของสัญญาณและกระตุ้นด้วยขอบของสัญญาณ ฟลิปฟลอปชนิดนี้สามารถสร้างจาก RS และ JK ฟลิปฟลอป ดังรูปที่ 9.20 (ก) และรูปที่ 9.21 (ก) ตามลำดับ



รูปที่ 9.19 สัญลักษณ์

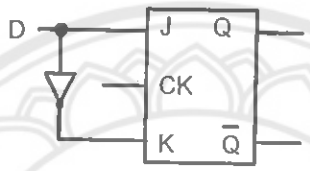


(ก)

Input		Output		โหมดการทำงานของเอาต์พุต (Mode)
CK	D	Q_{n+1}	$\overline{Q}_{n+1}$	
0	X	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	0	1	รีเซต (RESET)
1	1	1	0	เซต (SET)

(ข) ตารางความจริง

รูปที่ 9.20 D ฟลิปฟลอปที่สร้างจาก RS ฟลิปฟลอป



(ก) วงจร

Input		Output		โหมดการทำงานของเอาต์พุต (Mode)
CK	D	Q_{n+1}	$\overline{Q}_{n+1}$	
0	X	Q_n	$\overline{Q}_n$	ไม่เปลี่ยนแปลง (No change)
1	0	0	1	รีเซต (RESET)
1	1	1	0	เซต (SET)

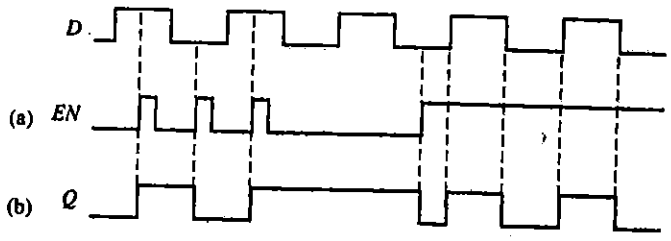
(ข) ตารางความจริง

รูปที่ 9.21 D ฟลิปฟลอปที่สร้างจาก JK ฟลิปฟลอป

ตารางที่ 9.5 ตารางความจริงที่นำไปใช้งาน

Input		Output
CK	D	Q_{n+1}
0	X	Q_n
1	0	0
1	1	1

ตัวอย่างที่ 9.7 ถ้ากำหนดรูปคลื่นของสัญญาณอินพุต D และ EN ของ D ฟลิปฟลอป ดังรูปที่ 9.22(a) จงหาเอาต์พุต Q สมมุติให้ฟลิปฟลอปเริ่มต้นเป็น RESET

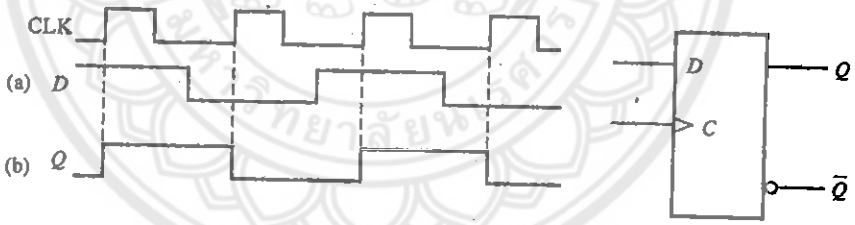


รูปที่ 9.22

<u>วิธีทำ</u>

จากอินพุต D ที่กำหนดให้ จะได้เอาต์พุต Q แสดงดังรูปที่ 9.22 (b) เมื่ออินพุต D และ EN เป็น HIGH จะให้เอาต์พุต Q เป็น HIGH ถ้าอินพุต D เป็น LOW และ EN เป็น HIGH จะให้เอาต์พุต Q เป็น LOW เมื่อ EN เป็น LOW จะไม่มีผลต่ออินพุต D

ตัวอย่างที่ 9.8 ถ้ากำหนดรูปคลื่นของสัญญาณอินพุต D และ CLK ของ D ฟลิปฟลอป ดังรูปที่ 9.23(a) จงหาเอาต์พุต Q สมมุติให้ฟลิปฟลอปเริ่มต้นเป็น RESET



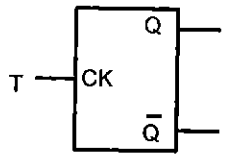
รูปที่ 9.23

<u>วิธีทำ</u>

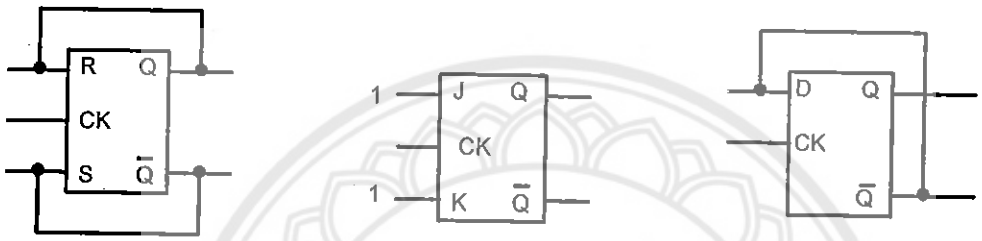
จาก D ฟลิปฟลอปที่กำหนดให้ เอาต์พุต Q จะเปลี่ยนแปลงที่ขอบขาขึ้นของสัญญาณนาฬิกาเท่านั้น ดังนั้นจะได้เอาต์พุต Q แสดงดังรูปที่ 9.23 (b)

9.3.4 T ฟลิปฟลอป (Toggle Flip-Flop)

T ฟลิปฟลอปเป็นการนำฟลิปฟลอปมาทำงานในสภาวะ Toggle เมื่อมีสัญญาณมากระตุ้นการทำงานของฟลิปฟลอป เอาต์พุตคือ Q และ $\overline{Q}$ ที่ได้หลังจากทำงานไปแล้วจะมีค่าตรงกันข้ามกับเอาต์พุตเดิม T ฟลิปฟลอปสามารถสร้างได้จาก RS, JK และ D ฟลิปฟลอป ดังรูปที่ 9.24 (ข)



(ก) สัญลักษณ์



(ข) วงจร

Input	Output		โหมดการทำงานของเอาต์พุต (Mode)
T	Q_{n+1}	$\overline{Q}_{n+1}$	
0	Q_n	$\overline{Q}_n$	ไม่มีการเปลี่ยนแปลง (No change)
1	$\overline{Q}_n$	Q_n	เปลี่ยนสภาวะตรงกันข้าม (Toggle)

(ค) ตารางความจริง

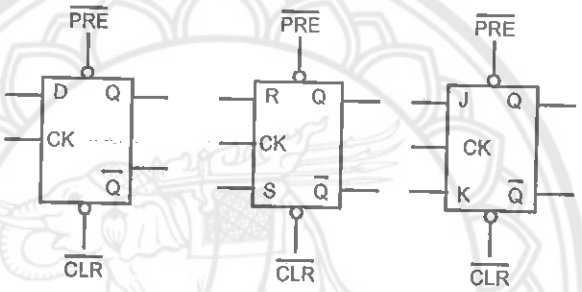
รูปที่ 9.24 T ฟลิปฟลอป

ตารางที่ 9.6 ตารางความจริงที่นำไปใช้งาน

Input	Output
T, CK	Q_{n+1}
0	Q_n
1	$\overline{Q}_n$

9.4 ประเภทอินพุตของฟลิปฟลอป

อินพุตของฟลิปฟลอปที่มีสัญญาณนาฬิกาควบคุมการทำงานที่กล่าวในข้อ 9.3 คืออินพุต R, S, J, K, D และ T เรียกอินพุตประเภทนี้ว่าอินพุตซิงโครนัส (Synchronous Input) การเพิ่มอินพุตอีกกลุ่มหนึ่งเข้าไปในฟลิปฟลอปที่มีสัญญาณนาฬิกาควบคุมได้แก่ PRESET (PRE) และ CLEAR (CLR) ทำให้ฟลิปฟลอปสามารถใช้งานได้กว้างขึ้น เรียกอินพุตประเภทนี้ว่าอินพุตอะซิงโครนัส (Asynchronous Input) หรืออินพุตไดเร็ก (Direct Input) โดยทั่วไปขา PRESET (PRE) และ CLEAR (CLR) จะเป็นแบบ ACTIVE LOW ชื่อที่ใช้เรียกอินพุตอะซิงโครนัสมีหลายแบบแสดงดังรูปที่ 9.25 (ข) การทำงานของอินพุตอะซิงโครนัสจะไม่ขึ้นกับอินพุตซิงโครนัส สามารถเขียนตารางการทำงานได้ดังตารางที่ 9.7 จะเห็นได้ว่าเมื่อขา $\overline{PRE}$ เป็น LOW เอาต์พุต Q = 1 และขา $\overline{CLR}$ เป็น LOW เอาต์พุต Q = 0 โดยไม่คำนึงว่าอินพุตซิงโครนัสจะมีหรือไม่มีก็ตาม แต่เมื่อขา $\overline{PRE}$ และ $\overline{CLR}$ เป็น HIGH เอาต์พุต Q จะขึ้นอยู่กับอินพุตซิงโครนัส



(ก) สัญลักษณ์

Asynchronous SET	Asynchronous RESET
$\overline{PRE}$	$\overline{CLR}$
$\overline{S}$	$\overline{R}$
$\overline{S}_D$	$\overline{R}_D$

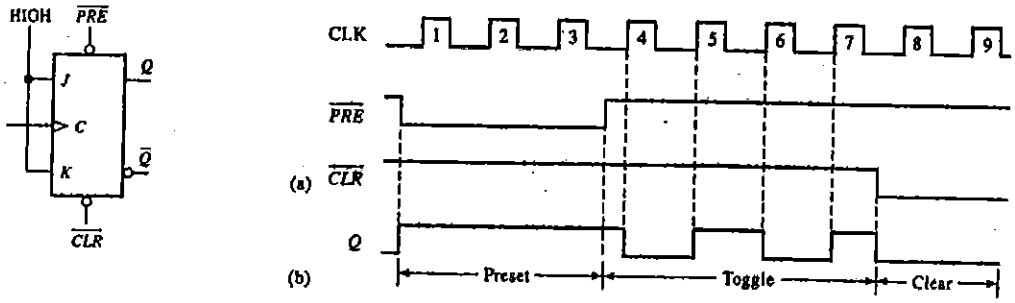
(ข) สัญลักษณ์ของอินพุต

รูปที่ 9.25 ฟลิปฟลอปที่มีอินพุตซิงโครนัสและอะซิงโครนัส

ตารางที่ 9.7 ตารางการทำงานของฟลิปฟลอปที่มีอินพุตซิงโครนัสและอะซิงโครนัส

Asynchronous Input		Output	โหมดการทำงานของเอาต์พุต (Mode)
$\overline{PRE}$	$\overline{CLR}$	Q_{n+1}	
0	0	-	ไม่ใช้งาน (Unused)
0	1	1	เซต (SET)
1	0	0	รีเซต (RESET)
1	1	ขึ้นอยู่กับ Synchronous Input (Clocked Operation)	

ตัวอย่างที่ 9.9 ถ้ากำหนดสัญญาณอินพุต $\overline{PRE}$ และ $\overline{CLR}$ ของ JK ฟลิปฟลอป ดังรูปที่ 9.26(a) จงหาเอาต์พุต Q สมมุติเริ่มต้นเอาต์พุต Q เป็น RESET



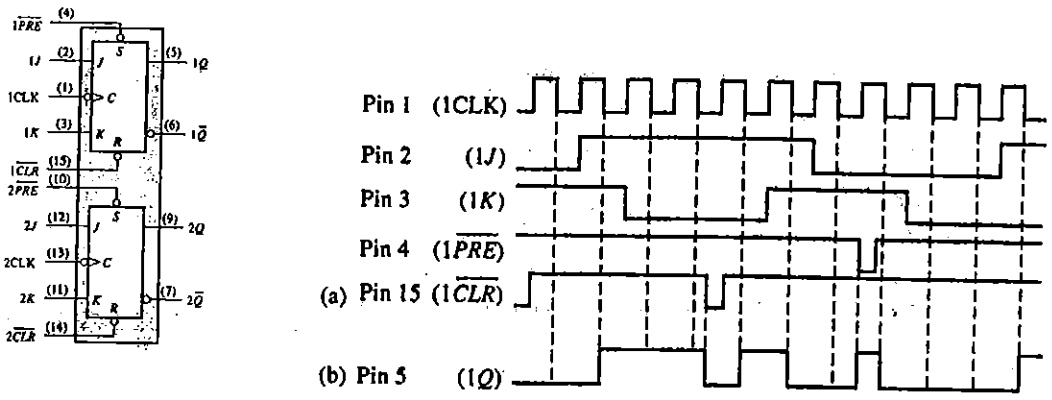
รูปที่ 9.26

วิธีทำ

จาก JK ฟลิปฟลอปที่กำหนดให้ จะให้เอาต์พุต Q แสดงดังรูปที่ 9.26 (b) สามารถอธิบายการทำงานได้ดังนี้

1. สำหรับสัญญาณนาฬิกาลูกที่ 1, 2 และ 3 ขา $\overline{PRE}$ เป็น LOW ดังนั้นเอาต์พุต Q = 1 หรือฟลิปฟลอปอยู่ในสภาวะ SET โดยไม่คำนึงถึงอินพุต J, K และ CLK
2. สำหรับสัญญาณนาฬิกาลูกที่ 4, 5, 6 และ 7 ขา $\overline{PRE}$ และ $\overline{CLR}$ เป็น HIGH ทั้งคู่ ดังนั้นเอาต์พุต Q จะขึ้นกับอินพุต J, K และ CLK และเมื่ออินพุต J และ K เป็น HIGH การทำงานของฟลิปฟลอปเป็นสภาวะ Toggle
3. สำหรับสัญญาณนาฬิกาลูกที่ 8 และ 9 ขา $\overline{CLR}$ เป็น LOW ดังนั้นเอาต์พุต Q = 0 หรือฟลิปฟลอปอยู่ในสภาวะ RESET โดยไม่คำนึงถึงอินพุต J, K และ CLK

ตัวอย่างที่ 9.10 ถ้ากำหนดสัญญาณอินพุต $1J$, $1K$, $1CLK$, $1\overline{PRE}$ และ $1\overline{CLR}$ ของ JK ฟลิปฟลอปที่กระตุ้นด้วยสัญญาณนาฬิกาขอบขาลงในไอซีเบอร์ 74HC112 เป็นดังรูปที่ 9.27(a) จงหาเอาต์พุต 1Q



IC 74HC112

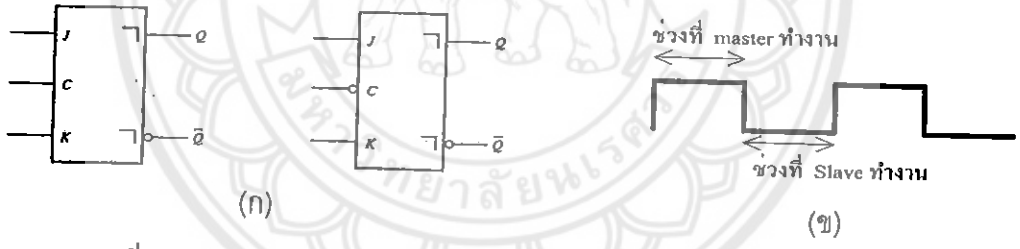
รูปที่ 9.27

<u>วิธีทำ</u>

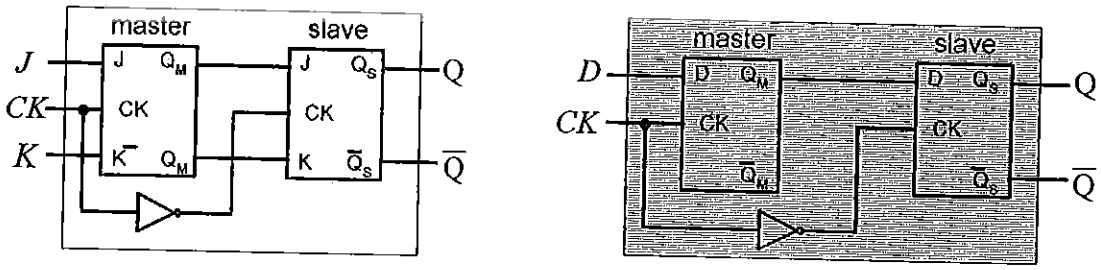
เอาต์พุต Q ที่ได้แสดงดังรูปที่ 9.27 (b) จะเห็นได้ว่าเมื่ออินพุต $1\overline{PRE}$ และ $1\overline{CLR}$ เป็น LOW จะให้เอาต์พุต Q เป็น SET หรือ RESET โดยไม่คำนึงถึงอินพุต $1J$, $1K$ และ $1CLK$

9.5 Master-Slave ฟลิปฟลอป

ฟลิปฟลอปที่กล่าวในข้อ 9.4 จะทำงานหรือเอาต์พุตเปลี่ยนสภาวะ เมื่อกระตุ้นด้วยระดับหรือขอบของสัญญาณนาฬิกา แต่มีฟลิปฟลอบอีกชนิดหนึ่งเรียกว่า Master-Slave ฟลิปฟลอปจะทำงานเมื่อกระตุ้นด้วยสัญญาณนาฬิกาครบ 1 ไซเคิลจึงมีการเปลี่ยนสภาวะ (Pulse trigger) จากรูปที่ 9.29 เป็น Master Slave JK ฟลิปฟลอปและ Master Slave D ฟลิปฟลอป โครงสร้างภายในประกอบด้วยฟลิปฟลอปแบบมีสัญญาณควบคุม 2 ตัวมาต่ออนุกรมกัน ตัวแรกสุดเรียกว่า Master และตัวที่สองเรียกว่า Slave โดยนำเอาต์พุต Q และ $\overline{Q}$ ของตัว Master ต่อกับอินพุตของตัว Slave การทำงานของ Master และ Slave จะทำงานเมื่อกระตุ้นด้วยระดับของสัญญาณต่างกัน โดยที่ Master จะทำงานช่วงที่สัญญาณนาฬิกาเป็น "1" ขณะที่ Slave จะอยู่ในสภาวะแลตส์หรือเอาต์พุตยังคงเหมือนเดิม แต่ Slave จะทำงานช่วงที่สัญญาณนาฬิกาเป็น "0" ขณะที่ Master จะอยู่ในสภาวะแลตส์หรือเอาต์พุตยังคงเหมือนเดิม ดังรูปที่ 9.28 (ข)

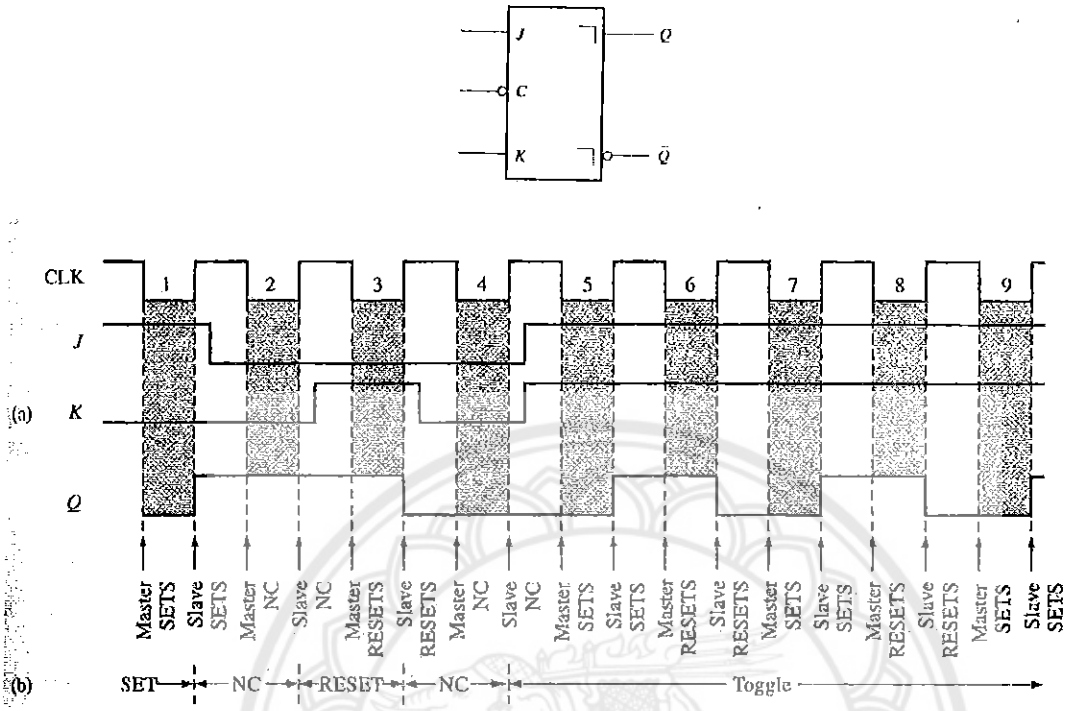


รูปที่ 9.28 สัญลักษณ์ Master Slave JK ฟลิปฟลอปและการทำงานใน 1 ไซเคิล



รูปที่ 9.29 Master Slave ฟลิปฟลอป

ตัวอย่างที่ 9.11 ถ้ากำหนดสัญญาณอินพุต Master-Slave JK ฟลิปฟลอป เป็นดังรูปที่ 9.30(a) จงหาเอาต์พุต 1Q



รูปที่ 9.30

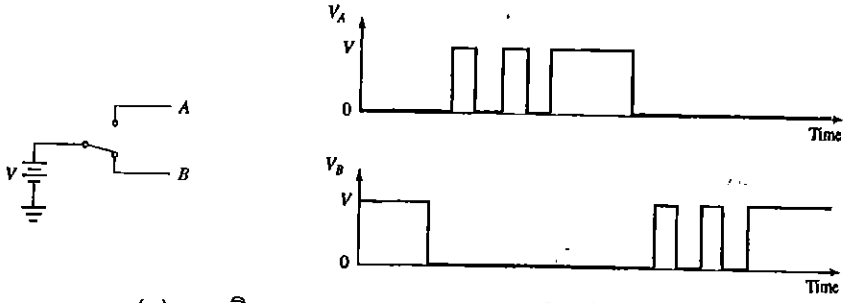
<u>วิธีทำ</u>

เอาต์พุต Q ที่ได้แสดงดังรูปที่ 9.30 (b) จะเห็นได้ว่าสถานะของอินพุและเอาต์พุตจะเริ่มต้นและสิ้นสุดในสัญญาณนาฬิกาแต่ละลูก ในที่นี้ NC หมายถึงไม่เปลี่ยนแปลง (NC)

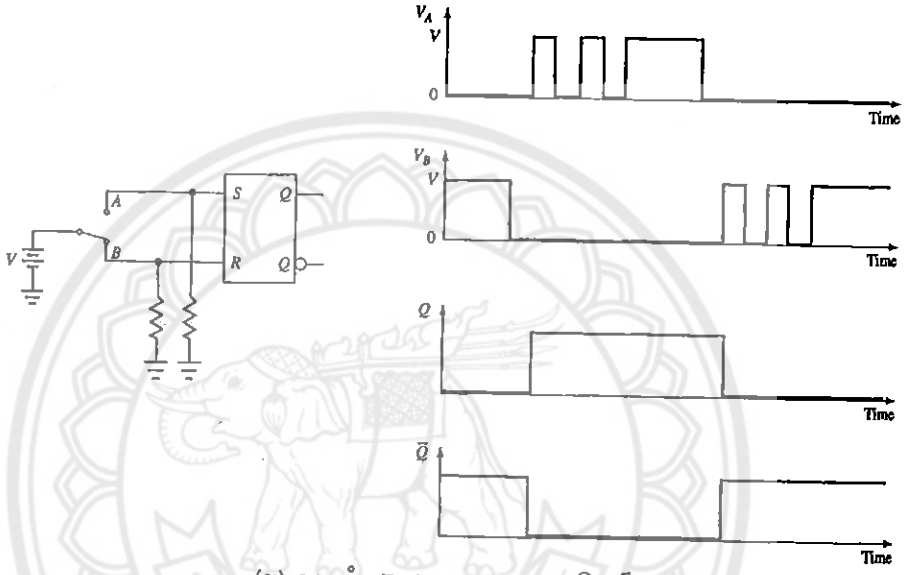
9.6 การประยุกต์ใช้งานฟลิฟลอป

จากคุณสมบัติของฟลิปฟลอปที่สามารถเก็บข้อมูลและการทำงานพร้อมกันตามสัญญาณนาฬิกา ทำให้นำไปประยุกต์ใช้งานได้มากมาย เช่น การเก็บข้อมูลแบบขนาน (Parallel data storage) วงจรหารความถี่ (Frequency Division) วงจรนับ (Counter) เป็นต้น ตัวอย่างที่น่าสนใจในการประยุกต์ใช้ฟลิปฟลอป ไปแก้ความผิดเพี้ยนของสัญญาณที่เกิดจากการสั่นของสวิตซ์ (Contact bounce) เรียกว่าการทำ Debounce โดยใช้ RS ฟลิปฟลอป

โดยปกติสวิตซ์ที่ใช้ในการเปิดปิดวงจรทางไฟฟ้า หน้าสัมผัสของวงจรสวิตซ์ทำจากโลหะ ทำให้การเปิดปิดสวิตซ์ไม่ราบเรียบ สวิตซ์จะมีการสั่นเปิดปิดหลายครั้ง ก่อนที่สวิตซ์จะเปิดหรือปิดอย่างสนิท ทำให้เกิดสัญญาณเป็นรูปคลื่นหลายลูกออกมา เรียกว่า Bounce ซึ่งเกิดขึ้นเร็วมากทำให้วงจรลอจิกทำงานผิดพลาดได้



(ก) การเกิด Contact bounce ของสวิตซ์



(ข) การทำ Debounce ของสวิตซ์

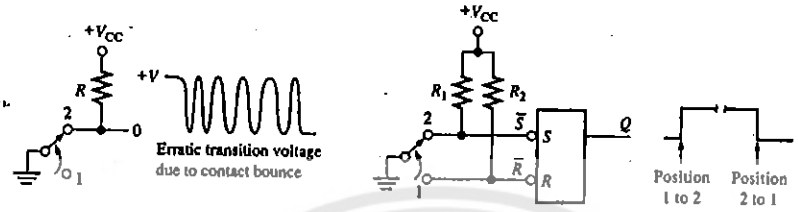
รูปที่ 9.30 การทำ Debounce โดยใช้ RS ฟลิปฟลอปแบบอินพุต ACTIVE HIGH

จากรูปที่ 9.30 (ก) อธิบายการทำงานได้ดังนี้ ถ้าสวิตซ์อยู่ที่ตำแหน่ง B ระดับแรงดัน V_B เท่ากับ +V โวลต์ ขณะที่แรงดัน V_A เท่ากับ 0 โวลต์ ถ้าเลื่อนสวิตซ์จากตำแหน่ง B ไปยัง A ขณะที่สวิตซ์อยู่ระหว่างตำแหน่ง A และ B นั้น แรงดัน V_A และ V_B จะเป็น 0 โวลต์ จนกระทั่งสวิตซ์สัมผัสที่ตำแหน่ง A ทำให้ค่าแรงดันเปลี่ยนไป โดยแรงดัน V_B เท่ากับ 0 โวลต์และแรงดัน V_A เท่ากับ +V โวลต์ แต่เนื่องจากการเกิด Contact bounce ของสวิตซ์ทำให้แรงดัน V_A มีค่าเป็น 0 และ +V โวลต์สลับไปมา จนกระทั่งสวิตซ์หยุดนิ่งที่ตำแหน่ง A แรงดัน V_A เท่ากับ +V โวลต์ ทำนองเดียวกันถ้าเลื่อนสวิตซ์จากตำแหน่ง A ไปยัง B ลักษณะของสัญญาณที่ได้จะเหมือนกัน

จากรูปที่ 9.30 (ข) เมื่อนำ RS ฟลิปฟลอปแบบนอร์เกตมาเพิ่มในวงจร สามารถอธิบายการทำงานได้ดังนี้ สมมุติให้แรงดัน +V โวลต์แทนลอจิก "1" และแรงดัน 0 โวลต์ (ต่อลงกราวนด์) แทนลอจิก "0" ถ้าสวิตซ์อยู่ที่ตำแหน่ง B จะทำให้ $V_B = 1$ ขณะที่ $V_A = 0$ ทำให้เอาต์พุต $Q = 0$ ($R = 1, S = 0$) ถ้าเลื่อนสวิตซ์จากตำแหน่ง B ไปยัง A ขณะที่สวิตซ์อยู่ระหว่างตำแหน่ง A และ

B นั้น แรงดัน V_A และ V_B จะเป็น 0 โวลต์ ทำให้เอาต์พุต Q = 0 (R = 0, S = 0) เหมือนเดิม เมื่อสวิตซ์สัมผัสที่ตำแหน่ง A ทำให้เอาต์พุต Q = 1 (R = 0, S = 1) แต่เนื่องจากเกิดการสั่นของสัญญาณทำให้แรงดัน V_A มีค่าเป็น 0 และ +V โวลต์สลับไปมาแต่เอาต์พุต Q = 1 เหมือนเดิม พิจารณาทำนองเดียวกันเมื่อเลื่อนสวิตซ์จากตำแหน่ง A ไปยัง B ทำให้เอาต์พุต Q = 0

สำหรับรูปที่ 9.31 เป็นการนำ RS ฟลิปฟลอปแบบแนนด์เกตที่มีอินพุตแบบ ACTIVE LOW มาเพิ่มในวงจร สามารถอธิบายการทำงานเช่นเดียวกับรูปที่ 9.30



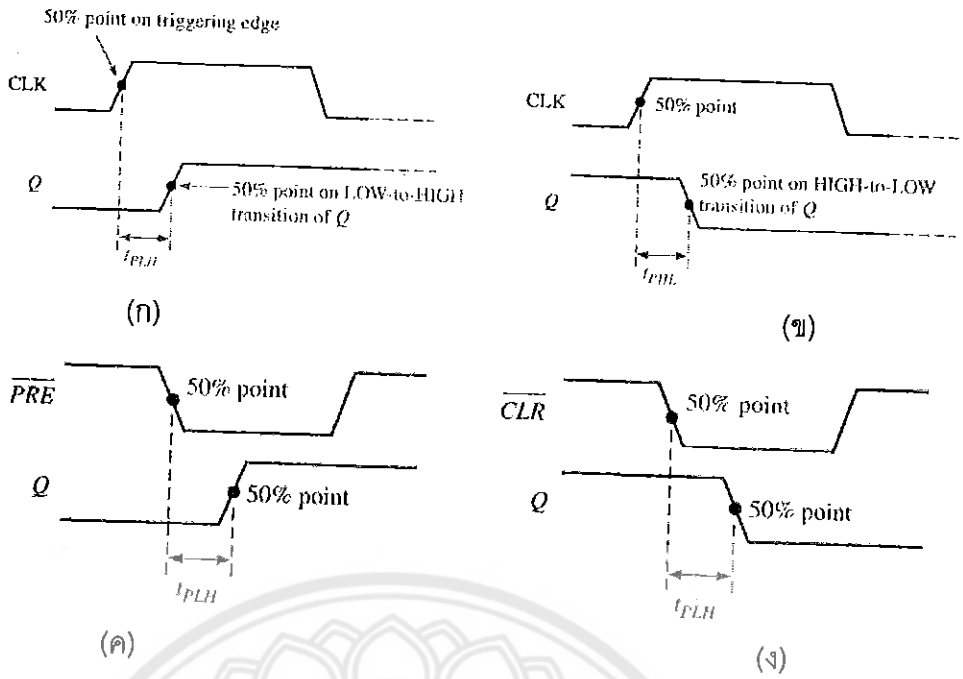
รูปที่ 9.31 การทำ Debounce โดยใช้ RS ฟลิปฟลอปแบบอินพุต ACTIVE LOW

9.7 เวลาที่ใช้ในการทำงานของฟลิปฟลอป

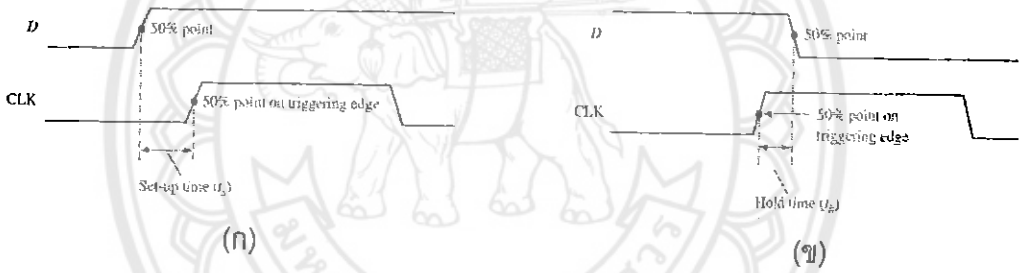
จากตารางความจริงของฟลิปฟลอปแต่ละชนิด เมื่อป้อนอินพุตเข้าไปจะให้เอาต์พุตที่สภาวะหนึ่ง เอาต์พุตที่ได้จะไม่เกิดขึ้นในทันทีทันใด แต่จะเกิดขึ้นหลังจากเวลาหน่วงผ่านไป เนื่องจากเกตที่นำมาสร้างฟลิปฟลอปมีค่าเวลาหน่วงค่าหนึ่งเสมอ ดังนั้นการพิจารณาเวลาที่ใช้ในการทำงานของฟลิปฟลอปที่เกี่ยวข้อง ได้แก่

Propagation Delay Time หมายถึงช่วงเวลาการทำงานของฟลิปฟลอป โดยเริ่มตั้งแต่สัญญาณนาฬิกาเริ่มเปลี่ยนแปลงจนถึงเวลาเอาต์พุตเกิดการเปลี่ยนแปลง ดังรูปที่ 9.32 สามารถแบ่งออกเป็น 4 ประเภทดังนี้

1. t_{PLH} หมายถึงช่วงเวลาที่สัญญาณนาฬิกาที่ขอบขาขึ้นจนถึงการเปลี่ยนแปลงของเอาต์พุตจาก LOW เป็น HIGH ดังรูปที่ 9.32 (ก)
2. t_{PHL} หมายถึงช่วงเวลาที่สัญญาณนาฬิกาที่ขอบขาขึ้นจนถึงการเปลี่ยนแปลงของเอาต์พุตจาก HIGH เป็น LOW ดังรูปที่ 9.32 (ข)
3. t_{PLH} หมายถึงช่วงเวลาที่สัญญาณนาฬิกาที่ขอบขาลงหรืออินพุต $\overline{PRE}$ จนถึงการเปลี่ยนแปลงของเอาต์พุตจาก LOW เป็น HIGH ดังรูปที่ 9.32 (ค)
4. t_{PLH} หมายถึงช่วงเวลาที่สัญญาณนาฬิกาที่ขอบขาลงหรืออินพุต $\overline{CLR}$ จนถึงการเปลี่ยนแปลงของเอาต์พุตจาก HIGH เป็น LOW ดังรูปที่ 9.32 (ง)



รูปที่ 9.32 Propagation Delay Time



รูปที่ 9.33 **ช่วงเวลา** t_S **และ** t_H **ของฟลิปฟลอป**

Setup Time (t_S) หมายถึงช่วงเวลาที่น้อยที่สุด (minimum interval) ที่**อินพุตซิงโครนัส** (J, K, R, S, D) มีค่าลอจิกคงที่ <u>ก่อน</u>ที่สัญญาณนาฬิกาจะเข้ามา ดังรูปที่ 9.33 (ก)

Hold Time (t_H) หมายถึงช่วงเวลาที่น้อยที่สุด (minimum interval) ที่อินพุตซิงโครนัส (J, K, R, S, D) มีค่าลอจิกคงที่ <u>หลัง</u>จากที่สัญญาณนาฬิกาเข้ามา ดังรูปที่ 9.33 (ข)

ความถี่สูงสุดของสัญญาณนาฬิกา (Maximum Clock Frequency, f_{max}) หมายถึงความถี่สูงสุดของสัญญาณนาฬิกาที่ฟลิปฟลอปยังคงใช้งานได้

ความกว้างของสัญญาณพัลส์ที่น้อยที่สุด (Minimum Pulse width, $t_{W(min)}$) หมายถึงช่วงเวลาที่น้อยที่สุดของสัญญาณนาฬิกาที่ทำให้ฟลิปฟลอปทำงานได้ แบ่งเป็น 2 ประเภทคือ $t_{WH(min)}$ หมายถึงช่วงเวลาที่น้อยที่สุดของสัญญาณนาฬิกาที่ต้องเป็น HIGH ที่ทำให้ฟลิปฟลอปทำงานได้และ $t_{WL(min)}$ หมายถึงช่วงเวลาที่น้อยที่สุดของสัญญาณนาฬิกาที่ต้องเป็น LOW ที่ทำให้ฟลิปฟลอปทำงานได้

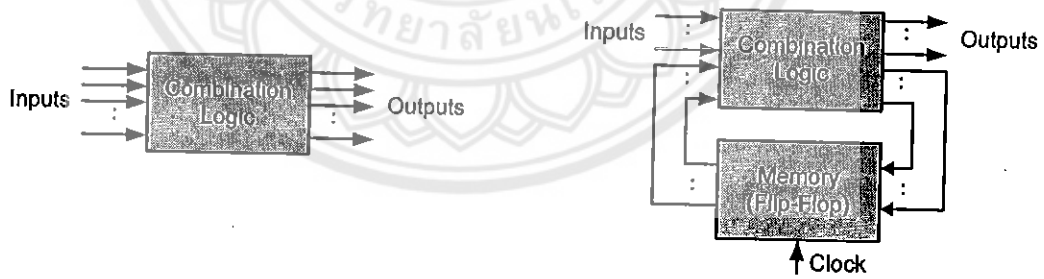
บทที่ 10
วงจรซีเควนเชียลแบบซิงโครนัส
(Synchronous Sequential Circuit)

วัตถุประสงค์

1. สามารถเข้าใจหลักการทำงานของวงจรซีเควนเชียลแบบซิงโครนัสได้
2. สามารถวิเคราะห์และออกแบบวงจรซีเควนเชียลแบบซิงโครนัส

10.1 บทนำ

วงจรลอจิกในระบบดิจิตอลสามารถแบ่งเป็น 2 ประเภท วงจรลอจิกเชิงจัดหมู่หรือวงจรคอมบิเนชัน (Combination Logic Circuit) และวงจรลอจิกเชิงลำดับหรือวงจรซีเควนเชียล (Sequential Logic Circuit) โดยทั่วไปลักษณะของวงจรคอมบิเนชันเอาต์พุตจะขึ้นอยู่กับอินพุตในขณะนั้น เมื่อเปลี่ยนอินพุตใหม่ ข้อมูลหรือเอาต์พุตต่างๆ ของอินพุตก่อนหน้านี้จะหายไป นั่นคือวงจรคอมบิเนชันไม่มีหน่วยความจำในการเก็บข้อมูลก่อนหน้านี้ แต่วงจรการใช้งานจริงจำเป็นต้องนำข้อมูลที่เกิดขึ้นก่อนหน้านี้มาใช้งานต่อไป จึงต้องมีหน่วยความจำมาเก็บข้อมูลเอาต์พุตเหล่านี้ไว้ ดังนั้นวงจรที่มีเอาต์พุตขึ้นอยู่กับอินพุตในขณะนั้นและเอาต์พุตที่ได้ก่อนหน้านี้เรียกว่าวงจรซีเควนเชียล สามารถเปรียบเทียบบล็อกไดอะแกรมของวงจรคอมบเนชันและวงจรซีเควนเชียลได้ดังรูปที่ 10.1



(ก) วงจรคอมบิเนชัน (ข) วงจรซีเควนเชียล

รูปที่ 10.1 ประเภทของวงจรลอจิก

จากรูปที่ 10.1 (ข) วงจรซีเควนเชียลประกอบด้วยวงจรคอมบิเนชันและหน่วยความจำมาต่อรวมกัน ส่วนของหน่วยความจำทำหน้าที่เป็นวงจรป้อนกลับ (Feed back) ในส่วนของวงจรคอมบิเนชันจะรับอินพุตเข้ามา 2 ชุดคือข้อมูลอินพุตจากภายนอกวงจรและอินพุตที่เก็บไว้ในหน่วยความจำ ส่วนเอาต์พุตที่ได้จะขึ้นกับอินพุต 2 ชุดนี้ ดังนั้นวงจรซีเควนเชียลเป็นการทำงานของ

อินพุต เอาต์พุต ทั้งภายในและภายนอกของวงจรที่มีลักษณะเป็นไปตามลำดับเวลา (Timing sequence)

การแบ่งประเภทของวงจรซีเควนเชียล

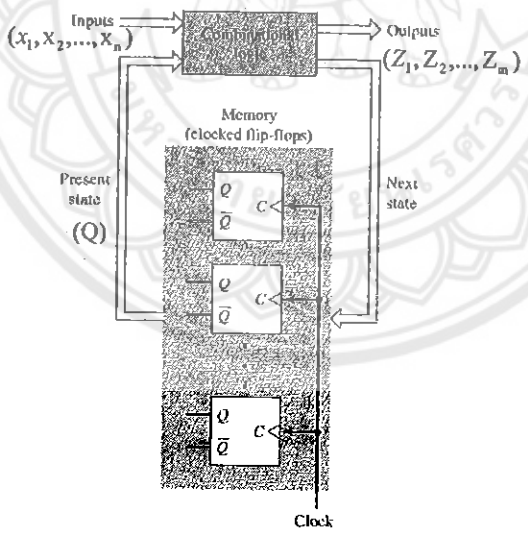
วงจรซีเควนเชียลสามารถแบ่งตามลักษณะสัญญาณนาฬิกาของฟลิปฟลอปใช้ในวงจร ได้ 2 ประเภท คือ

1. วงจรซีเควนเชียลแบบซิงโครนัส (Synchronous Sequential Circuit) วงจรจะทำงานภายใต้การควบคุมของสัญญาณนาฬิกาเสมอ ถ้าไม่มีสัญญาณนาฬิกาเข้ามา วงจรจะไม่ทำงาน ดังนั้น ฟลิปฟลอปทุกตัวต้องได้รับสัญญาณนาฬิกาจากภายนอกพร้อมกันและขาสัญญาณนาฬิกาของ ฟลิปฟลอปทุกตัวจะต่อร่วมกัน

2. วงจรซีเควนเชียลแบบอะซิงโครนัส (Asynchronous Sequential Circuit) วงจรภายในจะทำงานไม่พร้อมกัน วงจรอาจจะมีหรือไม่มีสัญญาณนาฬิกาก็ได้ ถ้ามีสัญญาณนาฬิกาจะมีฟลิปฟลอปบางตัวที่ไม่ได้รับสัญญาณนาฬิกาจากภายนอกโดยตรง

โดยทั่วไปการออกแบบวงจรแบบอะซิงโครนัสยุ่งยากกว่าแบบซิงโครนัส ทำให้วงจรซีเควนเชียลแบบซิงโครนัสเป็นที่นิยมมากกว่า

10.2 หลักการเบื้องต้น



รูปที่ 10.2 วงจรซีเควนเชียลแบบซิงโครนัส

โครงสร้างของวงจรซีเควนเชียลแบบซิงโครนัสทั่วไปแสดงดังรูปที่ 10.2 เรียกว่าโครงสร้างแบบ Mealy (Mealy Model) หน่วยความจำจะใช้ฟลิปฟลอปเป็นอุปกรณ์ที่ทำหน้าที่เก็บข้อมูลเลขฐานสอง โดยฟลิปฟลอป 1 ตัวสามารถเก็บข้อมูลได้ 1 บิต ถ้าต้องการเก็บข้อมูลจำนวนหลายบิต จะต้องใช้ฟลิปฟลอปจำนวนหลายตัว ข้อมูลที่เก็บในหน่วยความจำนี้เรียกว่าสถานะ (State)

ของวงจร วงจรซีเควนเชียลรับข้อมูลอินพุตจากภายนอกเขียนแทนด้วย $x_1, x_2, ..., x_n$ ค่าอินพุตจากภายนอกนี้ทำให้สถานะเอาต์พุตของหน่วยความจำเปลี่ยนแปลงไป เอาต์พุตของหน่วยความจำจะป้อนกลับไปเป็นอินพุตให้กับวงจรคอมบิเนชันเรียกเอาต์พุตนี้ว่าสถานะปัจจุบัน (Present state, PS) อินพุตที่ป้อนให้กับหน่วยความจำเรียกว่าสภานะถัดไป (Next state, NS) จะใช้เฉพาะกรณีที่ใช้ D ฟลิปฟลอปเป็นหน่วยความจำเท่านั้น ถ้าเป็นฟลิปฟลอปชนิดอื่นค่าของ NS จะเกิดขึ้นจากการทำงานของฟลิฟลอปนั้น ส่วนเอาต์พุตของวงจรเขียนแทนด้วย $Z_1, Z_2, ..., Z_m$ สัญญาณนาฬิกาเป็นตัวควบคุมการเปลี่ยนสถานะของฟลิปฟลอป จะทำให้ฟลิปฟลอปทุกตัวในวงจรซีเควนเชียลทำงานพร้อมกันหรือเปลี่ยนสถานะพร้อมกัน

การอธิบายการทำงานของวงจร

การอธิบายการทำงานของวงจรเพื่อแสดงความสัมพันธ์ระหว่างอินพุต (x) เอาต์พุต (Z) รวมทั้งสถานะของฟลิปฟลอปที่ PS และ NS (Q) สามารถใช้ตารางสถานะและแผนภาพสถานะของวงจร ในที่นี้กำหนดให้ Q_n คือเอาต์พุตของฟลิปฟลอปที่ PS ก่อนกระตุ้นด้วยสัญญาณนาฬิกาให้กับวงจรและ Q_{n+1} คือเอาต์พุตของฟลิปฟลอปที่ NS หลังกระตุ้นด้วยสัญญาณนาฬิกาให้กับวงจร

1. **ตารางสถานะของวงจร** (State Table)

ลำดับของอินพุต เอาต์พุตและสถานะของฟลิปฟลอปสามารถแสดงด้วยตารางสถานะ ตารางนี้ประกอบด้วย 4 ส่วนคือ PS, NS, Input และ Output บางครั้งตารางอาจจะไม่มี Input หรือ Output ก็ได้ ดังตารางที่ 10.1 ในส่วน PS แสดงข้อมูลเอาต์พุตของฟลิฟลอปก่อนกระตุ้นด้วยสัญญาณนาฬิกาให้กับวงจรหรือฟลิปฟลอป ส่วน Input คือค่าของ x ที่เป็นไปได้ใน PS ส่วน NS เป็นเอาต์พุตของฟลิปฟลอปที่ได้หลังกระตุ้นด้วยสัญญาณนาฬิกา ส่วน Output เป็นข้อมูลที่ได้ขึ้นอยู่กับข้อมูล PS และ Input ก่อนกระตุ้นด้วยสัญญาณนาฬิกา นอกจากนี้การอธิบายการทำงานของวงจรอาจมีการเพิ่มในส่วนอินพุตของฟลิปฟลอปเข้าไปด้วยดังตารางที่ 10.2 เพื่อใช้ในการวิเคราะห์การทำงานของวงจร

ตารางสถานะจะแสดงค่าของข้อมูลของ PS และ Input ที่เป็นไปได้ทั้งหมด จากตัวอย่างตารางที่ 10.1 ข้อมูลเลขฐานสองที่เป็นไปได้ทั้งหมดมีค่าตั้งแต่ 000 ถึง 111(8 กรณี) หรือถ้าฟลิปฟลอปจำนวน m ตัวและข้อมูลใน Input จำนวน n ค่า จะได้ข้อมูลที่เป็นไปได้ทั้งหมด 2^{m+n} กรณี และเลขฐานสองจะมีค่าตั้งแต่ 0 ถึง $2^{m+n}-1$ ข้อมูลของ NS หาได้จากวงจรหรือสมการสถานะของฟลิฟลอป ส่วนของ Output สามารถมีได้หลายค่าหาได้จากสมการบูลีน

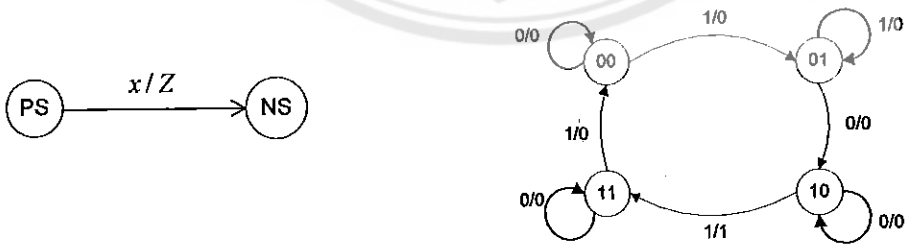
Present State		Input	Next State		Output
A	*B*	*x*	*A*	*B*	*y*
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	0	0	1
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	0
1	1	0	0	0	1
1	1	1	1	0	0

ตารางที่ 10.2 ตัวอย่างของตารางสถานะของวงจรที่มีอินพุตฟลิฟลอป

PRESENT STATE	INPUT	NEXT STATE	FLIP-FLOP INPUTS				OUTPUT
$Q_A Q_B$	X	$Q_A Q_B$	J_A	K_A	J_B	K_B	Z
00	0	00	0	X	0	X	0
00	1	01	0	X	1	X	0
01	0	10	1	X	X	1	0
01	1	01	0	X	X	0	0
10	0	10	X	0	0	X	1
10	1	11	X	0	1	X	1
11	0	11	X	0	X	0	0
11	1	00	X	1	X	1	0

2. **แผนภาพสถานะของวงจร** (State Diagram)

แผนภาพสถานะจะให้ข้อมูลอธิบายการทำงานของวงจรเหมือนกับตารางสถานะ แต่มีรูปแบบต่างกันเท่านั้น แผนภาพสถานะมีลักษณะเป็นกราฟ โดยเขียนข้อมูลของ PS และ NS ลงในวงกลม ข้อมูลนี้จะเขียนเป็นตัวเลขหรือตัวอักษร การเปลี่ยนสถานะจาก PS ไปยัง NS จะใช้ลูกศรชี้จาก PS ไป NS และที่ลูกศรจะเขียนอินพุตเอาต์พุตกำกับไว้ด้วยดังรูปที่ 10.3 (ก) ตัวอย่างการเขียนแผนภาพสถานะของตารางที่ 10.2 แสดงดังรูปที่ 10.3 (ข)



(ก) (ข)

รูปที่ 10.3 แผนภาพสถานะ

ตารางการเปลี่ยนสถานะของฟลิปฟลอป (Excitation Table)

ตารางการเปลี่ยนสถานะของฟลิปฟลอป (Excitation Table)

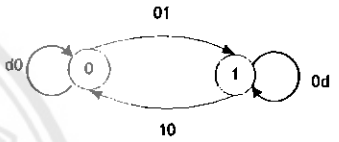
จากตารางความจริงของฟลิปฟลอปจะใช้พิจารณาจากอินพุตไปหาเอาต์พุตของฟลิปฟลอปหรือเมื่อรู้ข้อมูลอินพุตสามารถหาเอาต์พุตได้ ซึ่งนำไปใช้ในการวิเคราะห์วงจรซีเควนเชียล แต่ในการออกแบบวงจรซีเควนเชียลจะพิจารณาในทิศทางตรงกันข้ามคือจะรู้ข้อมูลเอาต์พุตก่อน แล้วจึงไปหาข้อมูลอินพุตที่ทำให้เกิดเอาต์พุตนั้น ดังนั้นตารางที่แสดงการเปลี่ยนแปลงจากเอาต์พุตไปยังอินพุตของฟลิปฟลอปเรียกว่าตารางการเปลี่ยนสถานะ สามารถเขียนตารางการเปลี่ยนสถานะของฟลิปฟลอปทุกชนิด ดังตารางที่ 10.3

ตารางที่ 10.3 ตารางการเปลี่ยนสถานะของฟลิปฟลอป

ตารางความจริง | ตารางการเปลี่ยนสถานะ | แผนภาพสถานะ

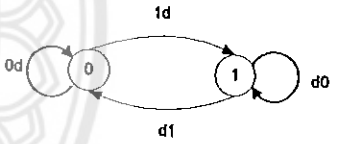
R	S	Q_{n+1}
0	0	Q_n
0	1	1
1	0	0
1	1	-

Q_n	Q_{n+1}	R	S
0	0	d	0
0	1	0	1
1	0	1	0
1	1	0	d



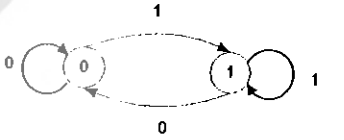
J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

Q_n	Q_{n+1}	J	K
0	0	0	d
0	1	1	d
1	0	d	1
1	1	d	0



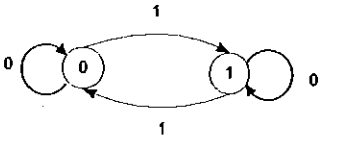
D	Q_{n+1}
0	0
1	1

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1



T	Q_{n+1}
0	Q_n
1	$\overline{Q}_n$

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

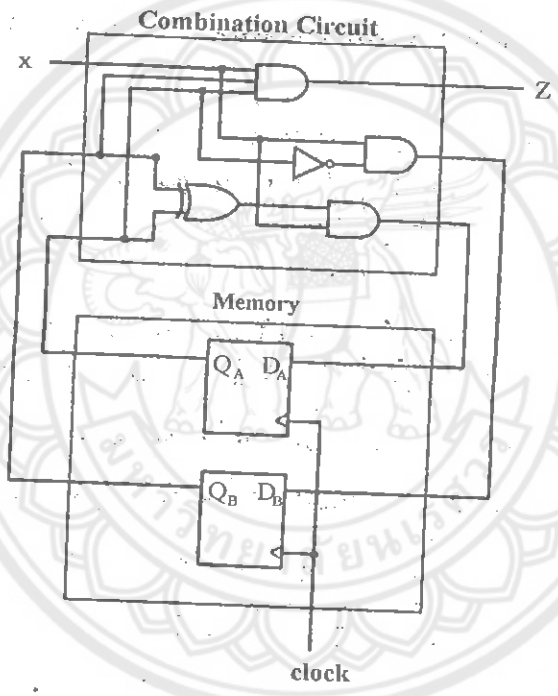


10.3 การวิเคราะห์วงจรซีเควนเชียล (Analysis of Sequential Circuits)

มีขั้นตอนดังนี้

1. เขียนสมการอินพุตของฟลิปฟลอป (Flip Flop Input Equations) และสมการเอาต์พุตของวงจร (Z)
2. เขียนตารางความจริงของฟลิปฟลอปที่ใช้ (Truth Table)
3. เขียนตารางสถานะของวงจร (State Table)
4. เขียนแผนภาพสถานะ (State Diagram)

ตัวอย่างที่ 10.1 จงวิเคราะห์หา State Table และ State Diagram ของวงจรซีเควนเชียลแบบซิงโครนัส ที่กำหนดให้ดังรูป



<u>วิธีทำ</u>

1. D FF Input Equation and Output Equation

$$D_A = x(Q_A \oplus Q_B)$$
$$D_B = x\overline{Q}_A$$
$$Z = xQ_AQ_B$$

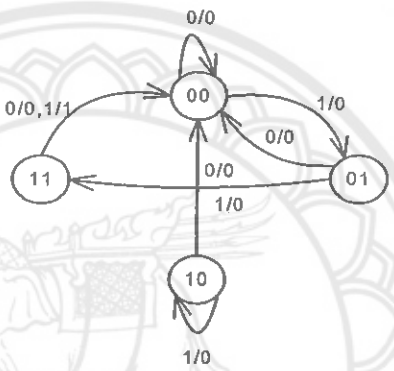
2. D FF Truth Table

D	Q_{n+1}
0	0
1	1

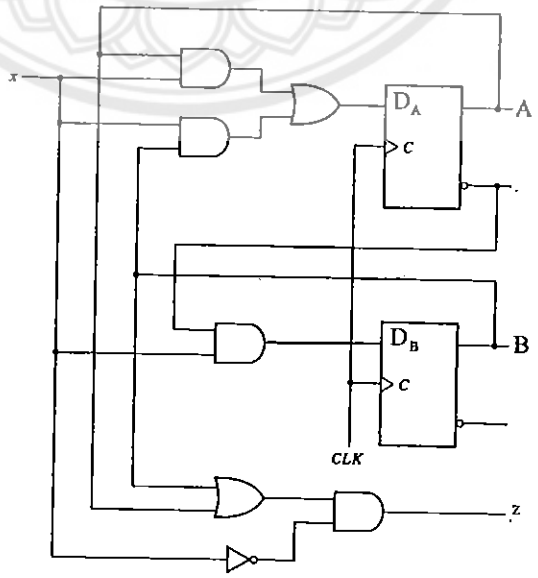
3. State Table

Present State		Input	Next State		Output	D FF Input	
Q_A	Q_B	x	Q_A	Q_B	Z_L	D_A	D_B
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1
0	1	0	0	0	0	0	0
0	1	1	1	1	0	1	1
1	0	0	0	0	0	0	0
1	0	1	1	0	0	1	0
1	1	0	0	0	0	0	0
1	1	1	0	0	1	0	0

4. State Diagram



ตัวอย่างที่ 10.2 จงวิเคราะห์หา State Table และ State Diagram ของวงจรซีเควนเชียลแบบซิงโครนัส ที่กำหนดให้ดังรูป



<u>วิธีทำ</u>

$$D_A = Ax + Bx = x(A+B) \quad D_B = \overline{A}x \qquad Z = \overline{x}(A+B)$$

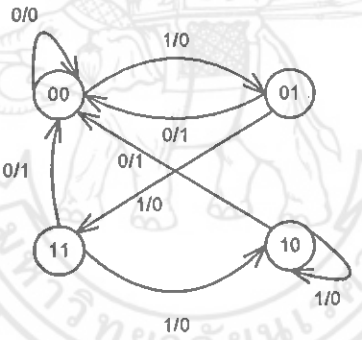
2. D FF Truth Table

D	Q_{n+1}
0	0
1	1

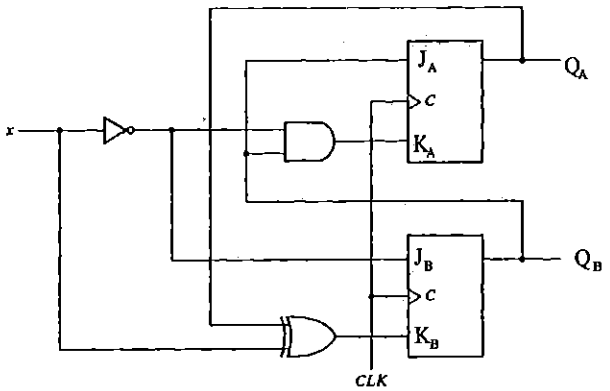
3. State Table

Present State		Input	Next State		Output	D FF Input	
A	B	x	A	B	z	D_A	D_B
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1
0	1	0	0	0	1	0	0
0	1	1	1	1	0	1	1
1	0	0	0	0	1	0	0
1	0	1	1	0	0	1	0
1	1	0	0	0	1	0	0
1	1	1	1	0	0	1	0

4. State Diagram



ตัวอย่างที่ 10.3 จงวิเคราะห์หา State Table และ State Diagram ของวงจรซีเควนเชียลแบบซิงโครนัส ที่กำหนดให้ดังรูป



<u>วิธีทำ</u>

<u>วิธีทำ</u>

1. JK FF Input Equation and Output Equation

$$J_A = Q_B \qquad K_A = Q_B\bar{x}$$
$$J_B = \bar{x} \qquad K_B = Q_A \oplus x$$

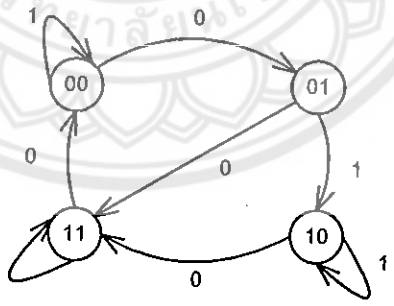
2. JK FF Truth Table

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\bar{Q}_n$

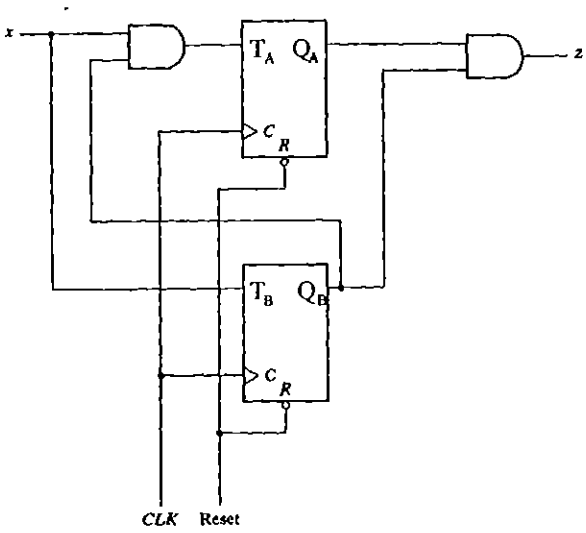
3. State Table

Present State		Input	Next State		JK FF Input	
Q_A	Q_B	x	Q_A	Q_B	J_A K_A	J_B K_B
0	0	0	0	1	0 0	1 0
0	0	1	0	0	0 0	0 1
0	1	0	1	1	1 1	1 0
0	1	1	1	0	1 0	0 1
1	0	0	1	1	0 0	1 1
1	0	1	1	0	0 0	0 0
1	1	0	0	0	1 1	1 1
1	1	1	1	1	1 0	0 0

4. State Diagram



ตัวอย่างที่ 10.4 จงวิเคราะห์หา State Table และ State Diagram ของวงจรซีเควนเชียลแบบซิงโครนัส ที่กำหนดให้ดังรูป



<u>วิธีทำ</u>

1. T FF Input Equation and Output Equation

$$T_A = Q_B x \qquad T_B = x \qquad z = Q_A Q_B$$

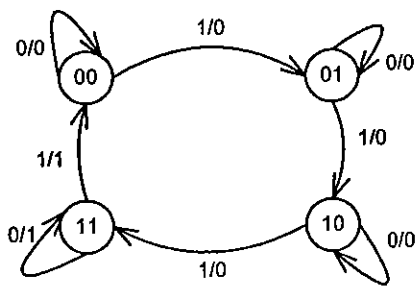
2. T FF Truth Table

T	Q_{n+1}
0	Q_n
1	$\overline{Q}_n$

3. State Table

Present State		Input	Next State		Output	T FF Input	
Q_A	Q_B	x	Q_A	Q_B	Z	T_A	T_B
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1
0	1	0	0	1	0	0	0
0	1	1	1	0	0	1	1
1	0	0	1	0	0	0	0
1	0	1	1	1	0	0	1
1	1	0	1	1	1	0	0
1	1	1	0	0	1	1	1

4. State Diagram

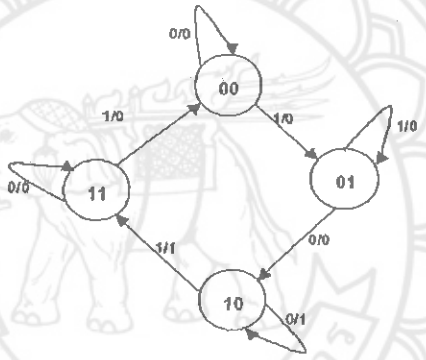


10.4 การออกแบบวงจรซีเควนเชียล (Design of Sequential Circuits)

มีขั้นตอนดังนี้

1. เขียนแผนภาพสถานะ (State Diagram) โดยกำหนดค่าสถานะต่างๆ ในรูปของเลขไบนารี
2. เขียนตารางการเปลี่ยนแปลงสถานะของฟลิปฟลอปที่เลือกใช้ (Excitation Table)
3. เขียนตารางสถานะของวงจร (State Table)
4. หาสมการอินพุตของฟลิปฟลอป (Flip Flop Input Equations) และสมการเอาต์พุตของวงจรในรูปอย่างง่าย
5. เขียนวงจรซีเควนเชียล (Sequential Circuit)

ตัวอย่างที่ 10.5 จงออกแบบวงจรซีเควนเชียลแบบซิงโครนัสตาม State Diagram ที่กำหนด โดยใช้ JK FF



วิธีทำ

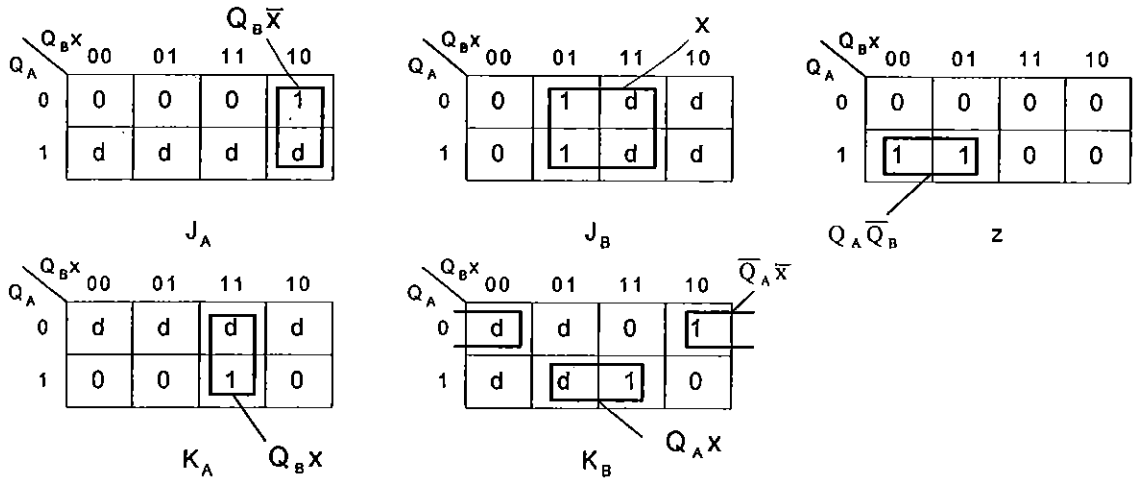
1. Excitation Table

Q_n	Q_{n+1}	J	K
0	0	0	d
0	1	1	d
1	0	d	1
1	1	d	0

2. State Table

Present State		Input	Next State		JK FF Input		Output
Q_A	Q_B	x	Q_A	Q_B	J_A K_A	J_B K_B	Z
0	0	0	0	0	0 d	0 d	0
0	0	1	0	1	0 d	1 d	0
0	1	0	1	0	1 d	d 1	0
0	1	1	0	1	0 d	d 0	0
1	0	0	1	0	d 0	0 d	1
1	0	1	1	1	d 0	1 d	1
1	1	0	1	1	d 0	d 0	0
1	1	1	0	0	d 1	d 1	0

3. JK Input Equation and Output Equation

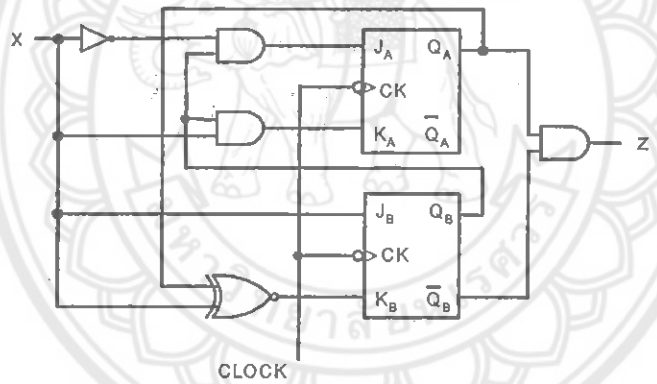


$$J_A = Q_B\bar{x} \qquad K_A = Q_Bx$$

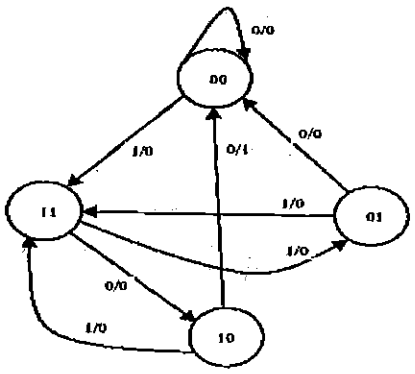
$$J_B = x \qquad K_B = Q_Ax + \bar{Q}_A\bar{x} = \overline{Q_A \oplus x}$$

$$z = Q_A\bar{Q}_B$$

4. Sequential Circuit



ตัวอย่างที่ 10.6 จงออกแบบวงจรซิงโครนัสตาม State Diagram ที่กำหนด โดยใช้ RS FF



<u>วิธีทำ</u>

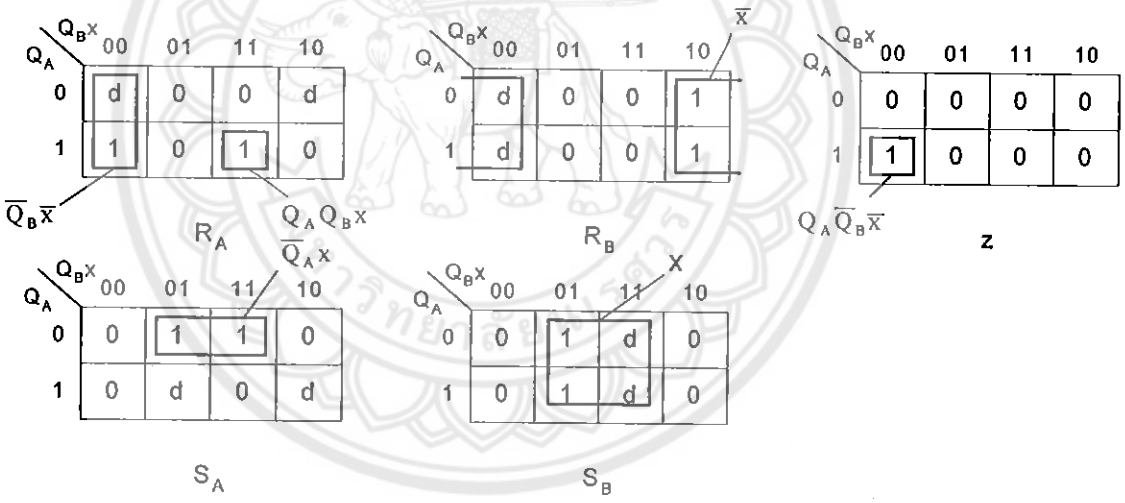
1. Excitation Table

Q_n	Q_{n+1}	R	S
0	0	d	0
0	1	0	1
1	0	1	0
1	1	0	d

2. State Table

Present State		Input	Next State		RS FF Input		Output
Q_A	Q_B	x	Q_A	Q_B	R_A S_A	R_B S_B	z
0	0	0	0	0	d 0	d 0	0
0	0	1	1	1	0 1	0 1	0
0	1	0	0	0	d 0	1 0	0
0	1	1	1	1	0 1	0 d	0
1	0	0	0	0	1 0	d 0	1
1	0	1	1	1	0 d	0 1	0
1	1	0	1	0	0 d	1 0	0
1	1	1	0	1	1 0	0 d	0

3. RS Input Equation and Output Equation

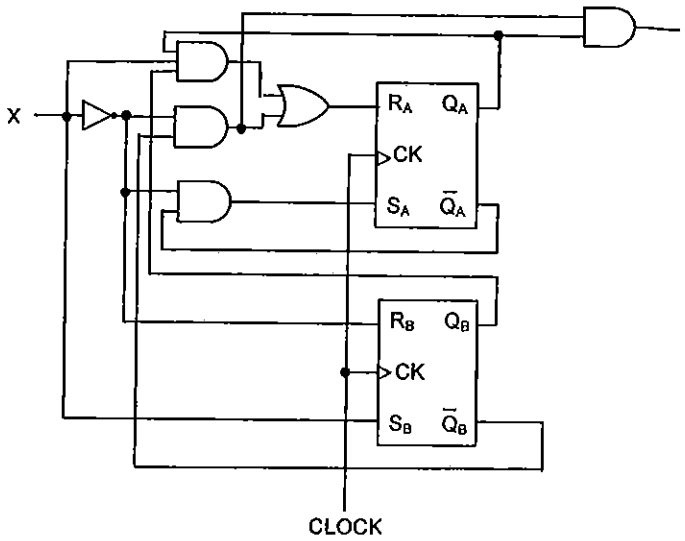


$$R_A = \overline{Q}_B\overline{x} + Q_AQ_Bx \ , \ S_A = \overline{Q}_Ax$$

$$R_B = \overline{x} \qquad , \ S_B = x$$

$$z = Q_A\overline{Q}_B\overline{x}$$

4. Sequential Circuit


X
R_A
Q_A
CK
S_A
$\bar{Q}_A$
R_B
Q_B
CK
S_B
$\bar{Q}_B$
CLOCK

บทที่ 11
วงจรนับ(Counters)

วัตถุประสงค์

1. สามารถเข้าใจหลักการทำงานของวงจรนับแบบซิงโครนัสและอะซิงโครนัส
2. สามารถออกแบบวงจรนับแบบซิงโครนัสและอะซิงโครนัสได้

วงจรนับเป็นวงจรที่สำคัญวงจรหนึ่ง นิยมใช้งานกันมากในระบบดิจิตอล ไม่ว่าจะใช้เป็นวงจรหลักหรือใช่ร่วมกับวงจรอื่น เช่น การวัดความถี่ของสัญญาณจะใช้วงจรนับทำการนับสัญญาณพัลส์ใน 1 วินาที ค่าที่นับได้เป็นจำนวนพัลส์ต่อวินาทีหรือเรียกว่าความถี่ของสัญญาณ การประยุกต์ใช้ในวงจรหารความถี่ เครื่องวัดแรงดันและกระแสที่เป็นตัวเลข ระบบคอมพิวเตอร์ เป็นต้น ปัจจุบันมีการผลิตไอซีวงจรนับมากมาย ซึ่งจำเป็นต้องเข้าใจหลักการทำงานและการออกแบบวงจรนับเพื่อให้สามารถออกแบบวงจรนับที่มีการทำงานตามต้องการรวมทั้งสามารถนำไอซีวงจรนับมาประยุกต์ใช้งานได้อย่างมีประสิทธิภาพ

11.1 ประเภทของวงจรนับ

วงจรนับจะสร้างจากฟลิปฟลอปตั้งแต่หนึ่งตัวขึ้นไป บางวงจรอาจประกอบด้วยเกตชนิดต่างๆ สามารถแบ่งวงจรนับได้หลายประเภทขึ้นอยู่กับสิ่งที่นำมาพิจารณาได้แก่ การกระตุ้นการทำงานของวงจรด้วยสัญญาณนาฬิกา สถานะหรือลำดับของการนับ จำนวนฟลิปฟลอปที่ใช้ ลักษณะการแสดงผลการนับ เป็นต้น

1) **ลักษณะของสัญญาณนาฬิกา** สามารถแบ่งเป็น 2 ประเภทดังนี้

วงจรนับอะซิงโครนัส (Asynchronous counter) หรือวงจรนับแบบระลอก (Ripple counter) ลักษณะการทำงานของวงจรคือ ฟลิปฟลอปตัวแรกเท่านั้นจะได้รับสัญญาณนาฬิกาจากภายนอก ดังนั้นฟลิปฟลอปแต่ละตัวจะทำงานไม่เป็นจังหวะพร้อมกัน วงจรนับซิงโครนัส (Synchronous counter) หรือวงจรนับแบบขนาน (Parallel counter) ลักษณะการทำงานของวงจรคือ ฟลิปฟลอปทุกตัวได้รับสัญญาณนาฬิกาจากภายนอกและทำงานเป็นจังหวะพร้อมกัน

2) **จำนวนสถานะของการนับ (N)** หรือลำดับของการนับ ในที่นี้เรียกว่ามอดูลัสที่ N (Modulus-N) หรือเรียกสั้นๆ ว่า MOD-N เช่น การออกแบบวงจรนับเลขฐานสองจาก 0000 ถึง 1111 (0 ถึง 15) เรียกว่า วงจรนับ MOD-16 เป็นต้น โดยทั่วไปวงจรนับประกอบด้วยฟลิปฟลอปจำนวน n ตัวหรือมีเลขฐานสองจำนวน n บิต ดังนั้นลำดับของการนับมีจำนวน 2^n ค่าและมีค่าตั้งแต่ 0 ถึง (2^n-1) หรือ $N \leq 2^n$ ถ้าวงจรมีสัญญาณนาฬิกาหรือสัญญาณพัลส์ป้อนเข้ามาจำนวน N ลูก (cycle) วงจรจะเริ่มการนับที่ค่าเริ่มต้นใหม่อีกครั้ง

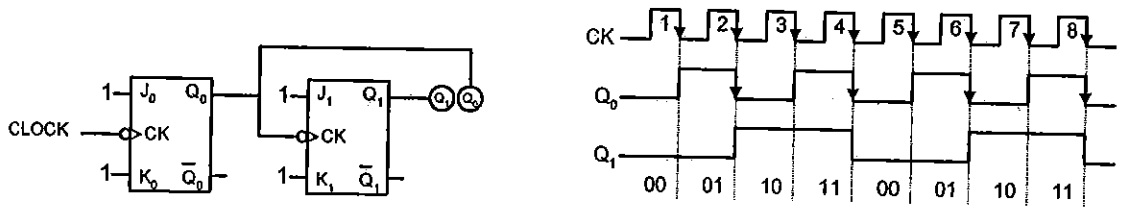
3) **จำนวนบิตของการนับ** เอาต์พุตของวงจรนับคือเอาต์พุตของฟลิปฟลอปแต่ละตัว ดังนั้นจำนวนบิตของวงจรนับเท่ากับจำนวนฟลิปฟลอปในวงจร

4) **ลักษณะของการแสดงผล** โดยทั่วไปวงจรนับส่วนใหญ่จะแสดงผลในรูปของเลขฐานสอง แต่ในการออกแบบวงจรนับอาจแสดงผลในรูปแบบอื่นได้ เช่น รหัสเกรย์ รหัสเกิน 3 เป็นต้น

11.2 วงจรนับอะซิงโครนัส (Asynchronous counter)

วงจรนับชนิดนี้ประกอบด้วยฟลิปฟลอปที่ทำงานในสภาวะ Toggle นำมาต่อเรียงกันแบบอนุกรม ฟลิปฟลอปจะเป็นชนิดใดก็ได้ เช่น JK FF หรือ D FF ที่มีการทำงานในสภาวะ Toggle แต่ในการออกแบบนิยมใช้ JK FF เนื่องจาก JK FF มีสภาวะการทำงานแบบ Toggle อยู่แล้ว ถ้ากำหนดให้ n คือจำนวนบิตของการนับ สามารถแบ่งลักษณะของการนับ MOD-N ได้เป็น 2 กรณีคือ กรณีแรก $N=2^n$ เป็นการนับเลขเรียงลำดับแบบลงตัวตามจำนวนบิตที่ใช้ เช่น n=2 บิต จะได้ $N=2^2=4$ ลำดับของการนับที่ได้เป็น 0-1-2-3 หรือ 3-2-1-0 หรือเรียกว่า MOD-4 เป็นต้น กรณีที่สอง $N < 2^n$ เป็นการนับเรียงลำดับแบบไม่ลงตัวตามจำนวนบิตที่ใช้ เช่น n=2 บิต จะได้ $N=2^2=4$ แต่ลำดับของการนับที่ได้อาจเป็น 0-1-2 หรือ 2-1-0 หรือเรียกว่า MOD-3 เป็นต้น ในที่นี้สามารถออกแบบวงจรนับแบบนับขึ้น (Count-up counter) หรือนับลง (Count-down counter) ได้ การต่อวงจรนับอะซิงโครนัสมีลักษณะดังนี้ ฟลิปฟลอปตัวที่ 1 จะรับสัญญาณนาฬิกาจากภายนอกเพียงตัวเดียวเท่านั้น เอาต์พุตของฟลิปฟลอปตัวที่ 1 จะต่อกับขา clock ของฟลิปฟลอปตัวที่ 2 เอาต์พุตของฟลิปฟลอปตัวที่ 2 จะต่อกับขา clock ของฟลิปฟลอปตัวที่ 3 เป็นเช่นนี้ไปเรื่อยๆ ทำให้ฟลิปฟลอปแต่ละตัวทำงานไม่เป็นจังหวะพร้อมกัน

วงจรนับอะซิงโครนัส MOD-N กรณีที่ $N=2^n$ ลำดับของการนับจะพิจารณาจากเอาต์พุตฟลิปฟลอปแต่ละตัว โดยเอาต์พุตของฟลิปฟลอปตัวที่ 1 จะเป็นบิตนัยสำคัญที่น้อยที่สุด (LSB) ส่วนเอาต์พุตของฟลิปฟลอปตัวสุดท้ายจะเป็นบิตนัยสำคัญที่มากที่สุด (MSB) ตัวอย่างเช่น การออกแบบวงจรนับขึ้นอะซิงโครนัส MOD-4 โดยใช้ clock ที่ขอบขาลง ดังรูปที่ 11.1



(ก) วงจร (ข) timing diagram ของลำดับการนับ

รูปที่ 11.1 วงจรนับขึ้นอะซิงโครนัส MOD-4

วงจรประกอบด้วย JK FF 2 ตัว ฟลิปฟลอปแต่ละตัวมีการทำงานในสภาวะ Toggle เมื่อขา J=1 และ K=1 ขาเอาต์พุต Q ของฟลิปฟลอปตัวที่ 1 จะต่อกับขา clock ของฟลิปฟลอปตัวที่ 2 ทำให้เอาต์พุต Q ของฟลิปฟลอปแต่ละตัวมีการเปลี่ยนสถานะเป็นตรงข้ามกับสถานะเดิมทุกครั้งที่ขา clock มีการเปลี่ยนจาก 1 เป็น 0 (ขอบขาลง) เมื่อเขียนลำดับการนับของวงจรในรูปของเลขฐานสองจะได้เป็น Q_1Q_0 โดยที่ Q_0 เป็นบิตนัยสำคัญที่น้อยที่สุด (LSB) และ Q_1 เป็นบิตนัยสำคัญที่มากที่สุด (MSB) ดังรูปที่ 11.1 (ข) การทำงานของวงจรนับเป็นดังนี้ สมมุติกำหนดเริ่มแรกให้เอาต์พุต Q_1Q_0= 00

เมื่อป้อน clock ลูกที่ 1 พิจารณาการเปลี่ยนแปลงในช่วงขอบขาลง ทำให้ Q_0 เปลี่ยนจาก 0 เป็น 1 ส่วน Q_1 มีค่าเป็น 0 เหมือนเดิม ทำให้สถานะการนับเป็น 01

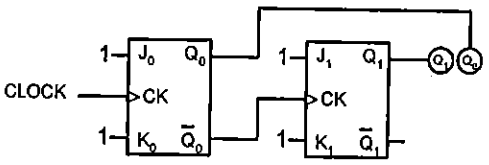
เมื่อป้อน clock ลูกที่ 2 พิจารณาการเปลี่ยนแปลงในช่วงขอบขาลง ทำให้ Q_0 เปลี่ยนจาก 1 เป็น 0 ส่วน Q_1 เปลี่ยนจาก 0 เป็น 1 ทำให้สถานะการนับเป็น 10

เมื่อป้อน clock ลูกที่ 3 พิจารณาการเปลี่ยนแปลงในช่วงขอบขาลง ทำให้ Q_0 เปลี่ยนจาก 0 เป็น 1 ส่วน Q_1 มีค่าเป็น 1 เหมือนเดิม ทำให้สถานะการนับเป็น 11

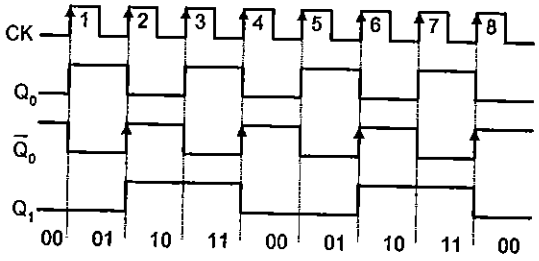
เมื่อป้อน clock ลูกที่ 4 พิจารณาการเปลี่ยนแปลงในช่วงขอบขาลง ทำให้ Q_0 เปลี่ยนจาก 1 เป็น 0 ส่วน Q_1 เปลี่ยนจาก 1 เป็น 0 ทำให้สถานะการนับเป็น 00

เมื่อพิจารณาเอาต์พุต Q_1Q_0 จะเห็นว่าเมื่อ clock เข้ามาแต่ละครั้ง สถานะการนับของวงจรจะเพิ่มทีละ 1 จนกระทั่ง clock เข้ามา 4 ลูกจะได้ลำดับการนับเป็น 00 01 10 11 ตามลำดับ เมื่อได้ค่าการนับสูงสุดแล้วจะวนกลับมาเริ่มนับที่ 00 ใหม่

ถ้าต้องการออกแบบวงจรนับขึ้นอะซิงโครนัส MOD-4 โดยใช้ clock ที่ขอบขาขึ้น จะต้องนำขาเอาต์พุต $\overline{Q}$ ของฟลิปฟลอปตัวที่ 1 ต่อกับขา clock ของฟลิปฟลอปตัวที่ 2 ดังรูปที่ 11.2 (ก) ทำให้เอาต์พุต Q ของฟลิปฟลอปแต่ละตัวมีการเปลี่ยนสถานะเป็นตรงข้ามกับสถานะเดิมทุกครั้งที่ขา clock มีการเปลี่ยนจาก 0 เป็น 1 (ขอบขาขึ้น)



(ก) วงจร

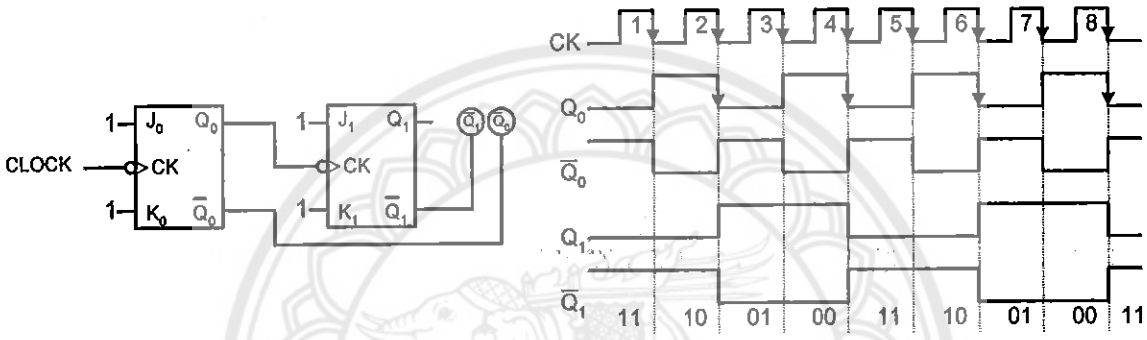


(ข) timing diagram ของลำดับการนับ

รูปที่ 11.2 วงจรนับขึ้นอะซิงโครนัส MOD-4

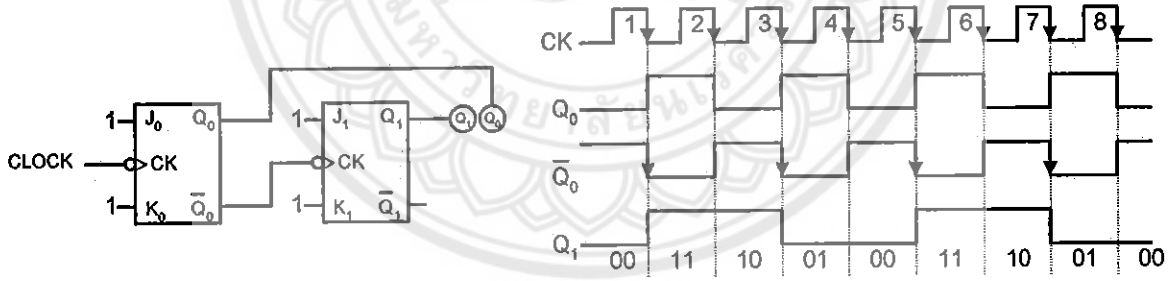
ถ้าต้องการออกแบบวงจรนับให้ทำการนับค่าได้มากขึ้น สามารถเพิ่มจำนวนฟลิปฟลอปเข้าไป ทำให้จำนวนบิตของการนับเท่ากับจำนวนของฟลิปฟลอปที่ต่ออนุกรมกัน นอกจากนี้จากตัวอย่างวงจรนับขึ้นดังรูปที่ 11.1 สามารถดัดแปลงเป็นวงจรนับลงอะซิงโครนัส MOD-4 ได้โดยการเปลี่ยนอย่างใดอย่างหนึ่งของวงจรดังต่อไปนี้

1. เปลี่ยนเอาต์พุตที่แสดงสถานะของการนับจาก Q เป็น $\overline{Q}$ ดังรูปที่ 11.3
2. เปลี่ยนเอาต์พุต Q เป็น $\overline{Q}$ เพื่อต่อกับขา clock ของฟลิปฟลอปตัวถัดไป ดังรูปที่ 11.4
3. เปลี่ยนสัญญาณที่กระตุ้นการทำงานของฟลิปฟลอปจากใช้ clock ที่ขอบขาลงเป็น clock ที่ขอบขาขึ้น ดังรูปที่ 11.5



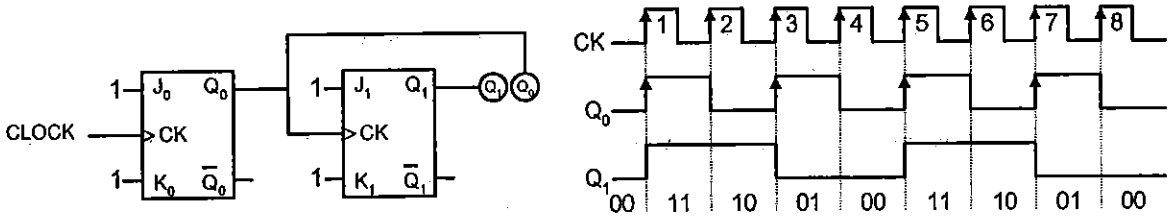
(ก) วงจร (ข) timing diagram ของลำดับการนับ

รูปที่ 11.3 วงจรนับลงอะซิงโครนัส MOD-4



(ก) วงจร (ข) timing diagram ของลำดับการนับ

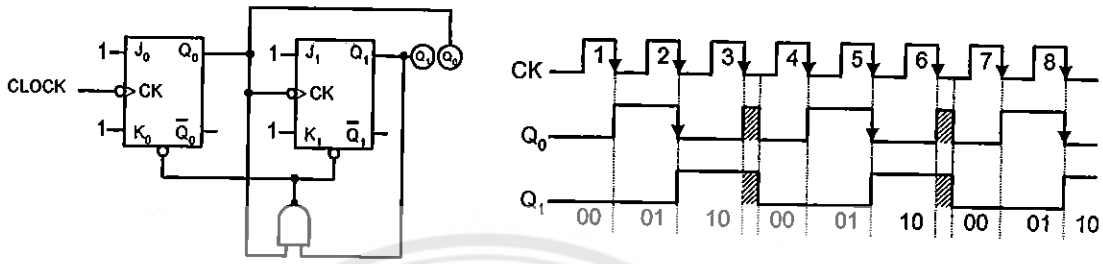
รูปที่ 11.4 วงจรนับลงอะซิงโครนัส MOD-4



(ก) วงจร (ข) timing diagram ของลำดับการนับ

รูปที่ 11.5 วงจรนับลงอะซิงโครนัส MOD-4

วงจรนับอะซิงโครนัส MOD-N กรณีที่ N<2^n สามารถออกแบบวงจรนับได้ทั้งแบบนับขึ้น (Count-up counter) **หรือนับลง** (Count-down counter) **ที่มีลำดับการนับตามต้องการ โดยการเพิ่มแนนด์เกตเข้าไปในวงจร เอาต์พุตที่ได้จากแนนด์เกตจะนำไปต่อเข้ากับขา clear ของฟลิปฟลอป เพื่อควบคุมให้วงจรมีการเริ่มนับใหม่ ตัวอย่างเช่น การออกแบบวงจรนับขึ้นอะซิงโครนัส MOD-3 ที่มีลำดับการนับเป็น 0-1-2 ดังรูปที่ 11.6**



(ก) วงจร (ข) timing diagram ของลำดับการนับ

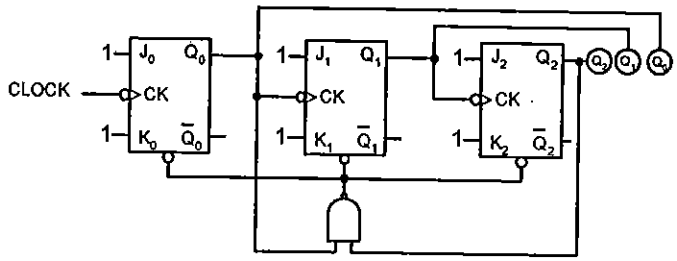
รูปที่ 11.6 วงจรนับขึ้นอะซิงโครนัส MOD-3

จากวงจรรูปที่ 11.1 เป็นวงจรนับขึ้น MOD-4 ถ้าต้องการออกแบบเป็นวงจรนับขึ้น MOD-3 ให้นำค่า 3 หรือ Q_1Q_0 =11 ไปต่อกับอินพุตของแนนด์เกต และนำเอาต์พุตของแนนด์เกตไปต่อกับขา clear ของฟลิปฟลอปทุกตัว เพื่อทำการรีเซตให้วงจรเริ่มลำดับการนับใหม่

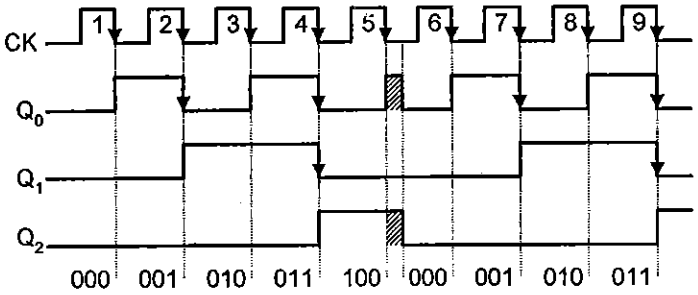
ตัวอย่างที่ 11.1 จงออกแบบวงจรนับขึ้น MOD-5 พร้อมเขียนตารางความจริง Timing diagram และ State diagram ของวงจร

<u>วิธีทำ</u>

การออกแบบวงจรจะใช้ JK FF 3 ตัว มีลำดับการนับเป็น $Q_2Q_1Q_0$ เมื่อวงจรมีลำดับการนับเป็น 5 ($Q_2Q_1Q_0$= 101) ให้นำ $Q_2\overline{Q}_1Q_0$ หรือ Q_2Q_0 ไปต่อกับอินพุตของแนนด์เกตและนำเอาต์พุตของแนนด์เกตต่อกับขา clear ของฟลิปฟลอปทุกตัว เพื่อให้วงจรเริ่มลำดับการนับใหม่



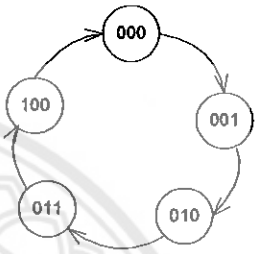
(ก) วงจร



(ข) Timing diagram ของลำดับการนับ

Q_2	Q_1	Q_0	จำนวนของสัญญาณ CLK
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	เริ่มลำดับการนับใหม่

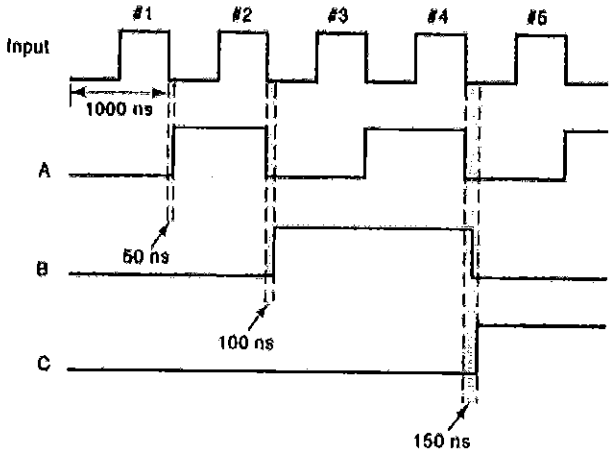
(ค) ตารางความจริง



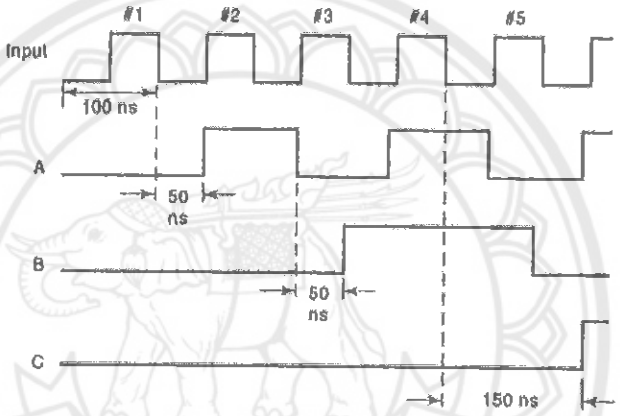
(ง) State diagram

รูปที่ 11.7 วงจรนับขึ้นอะซิงโครนัส MOD-5

จากตัวอย่างวงจรนับอะซิงโครนัสจะเห็นได้ว่าการออกแบบวงจรสามารถออกแบบและเข้าใจหลักการทำงานของวงจรได้ง่าย วงจรไม่ซับซ้อนมากเนื่องจากใช้เกตจำนวนน้อย แต่วงจรนับอะซิงโครนัสมีข้อจำกัดในการใช้งานคือ วงจรมีการทำงานช้าเนื่องจากวงจรจะต้องนำเอาต์พุตของฟลิปฟลอปก่อนหน้านี้ไปเป็นสัญญาณ clock ให้กับของฟลิปฟลอปตัวถัดไป ถ้าวงจรประกอบด้วยฟลิปฟลอปจำนวนมากและพิจารณาค่าเวลาหน่วงของฟลิปฟลอปแต่ละตัวแล้ว จะทำให้วงจรทำงานช้าและลำดับการนับผิดพลาดได้ ดังรูปที่ 11.8 (ก) จะเห็นว่าเอาต์พุตของฟลิปฟลอป A จะหน่วงเวลาไป 50 ns เอาต์พุตของฟลิปฟลอป B จะหน่วงเวลาไป 100 ns เอาต์พุตของฟลิปฟลอป C จะหน่วงเวลาไป 150 ns เมื่อเทียบกับสัญญาณ clock ถ้าวงจรใช้สัญญาณ clock ความถี่สูงดังรูปที่ 11.8 (ข) ทำให้ลำดับการนับผิดพลาด นอกจากนี้การเกิด spike ในวงจรเพื่อรีเซตควบคุมการนับใหม่ ถ้านำไปประยุกต์ใช้ในวงจรที่ใช้สัญญาณ clock ความถี่สูง จะทำให้วงจรนับทำงานผิดพลาดได้เช่นกัน



(ก) สัญญาณนาฬิกาที่ความถี่เท่ากับ 1 MHz



(ข) สัญญาณนาฬิกาที่ความถี่เท่ากับ 10 MHz

รูปที่ 11.8 เวลาหน่วงในวงจรนับอะซิงโครนัส

11.3 วงจรนับซิงโครนัส (Synchronous counter)

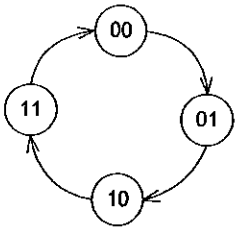
การออกแบบวงจรนับซิงโครนัส MOD-N เมื่อ $N \leq 2^n$ มีขั้นตอนการออกแบบเช่นเดียวกับวงจรซีเควนเชียลแบบซิงโครนัส ขั้นตอนมีดังนี้

1. เขียนแผนภาพสถานะ (State diagram) โดยกำหนดค่าสถานะต่างๆ ในรูปของเลขฐานสอง
2. เขียนตารางการเปลี่ยนแปลงสถานะของฟลิปฟลอปที่เลือกใช้ (Excitation table)
3. เขียนตารางสถานะของวงจร (State table)
4. หาสมการอินพุตของฟลิปฟลอป (FF Input equations) และเอาต์พุต (Output equations) ของวงจรในรูปสมการอย่างง่าย
5. เขียนวงจรนับ (Counter circuit)

ตัวอย่างที่ 11.2 จงออกแบบวงจรนับขึ้นแบบซิงโครนัส MOD-4 โดยมีลำดับการนับดังนี้ 0-1-2-3 กำหนดให้ใช้ JK FF

<u>วิธีทำ</u>

1. State diagram



2. Excitation table

Q_n	Q_{n+1}	J	K
0	0	0	d
0	1	1	d
1	0	d	1
1	1	d	0

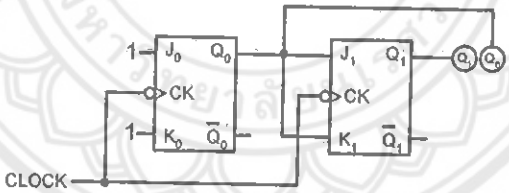
3. State table

Present State		Next State		JK FF Input	
Q_1	Q_0	Q_1	Q_0	J_1 K_1	J_0 K_0
0	0	0	1	0 d	1 d
0	1	1	0	1 d	d 1
1	0	1	1	d 0	1 d
1	1	0	0	d 1	d 1

4. JK FF Input equations

$$J_0 = 1 \quad K_0 = 1 \quad J_1 = Q_0 \quad K_1 = Q_0$$

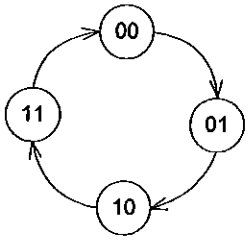
5. Counter circuit



ตัวอย่างที่ 11.3 จงออกแบบวงจรนับขึ้นแบบซิงโครนัส MOD-4 (0-1-2-3) โดยมีลำดับการนับดังนี้ 0-1-2-3 กำหนดให้ใช้ D FF

<u>วิธีทำ</u>

1. State diagram



2. Excitation table

Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

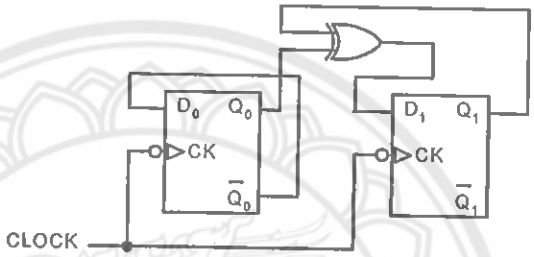
3. State table

Present State		Next State		D FF Input
Q_1	Q_0	Q_1	Q_0	$D_1 D_0$
0	0	0	1	0 1
0	1	1	0	1 0
1	0	1	1	1 1
1	1	0	0	0 0

4. D FF Input equation

$$D_0 = \overline{Q}_0 \qquad D_1 = \overline{Q}_1 Q_0 + Q_1 \overline{Q}_0 = Q_1 \oplus Q_0$$

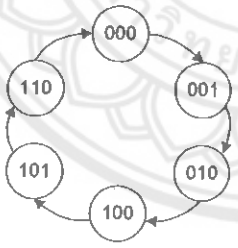
5. Counter circuit



ตัวอย่างที่ 11.4 จงออกแบบวงจรนับซิงโครนัส MOD-6 โดยมีลำดับการนับดังนี้ 0, 1, 2, 4, 5, 6, 0,... กำหนดให้ใช้ JK FF

<u>วิธีทำ</u>

1. State diagram



2. Excitation table

Q_n	Q_{n+1}	J_n	K_n
0	0	0	d
0	1	1	d
1	0	d	1
1	1	d	0

3. State table

Present State	Next State	JK FF Input		
$Q_2 Q_1 Q_0$	$Q_2 Q_1 Q_0$	$J_2 K_2$	$J_1 K_1$	$J_0 K_0$
0 0 0	0 0 1	0 d	0 d	1 d
0 0 1	0 1 0	0 d	1 d	d 1
0 1 0	1 0 0	1 d	d 1	0 d
1 0 0	1 0 1	d 0	0 d	1 d
1 0 1	1 1 0	d 0	1 d	d 1
1 1 0	0 0 0	d 1	d 1	0 d

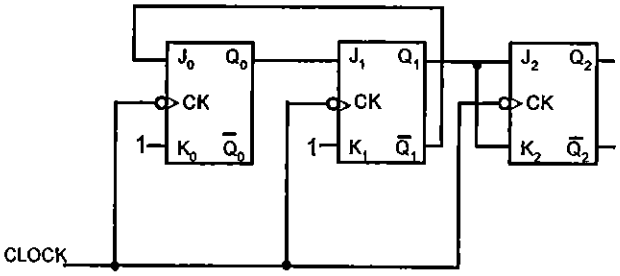
4.JK FF Input equations

$J_0 = \overline{Q}_1 \quad K_0 = 1$

$J_1 = Q_0 \quad K_1 = 1$

$J_2 = Q_1 \quad K_2 = Q_1$

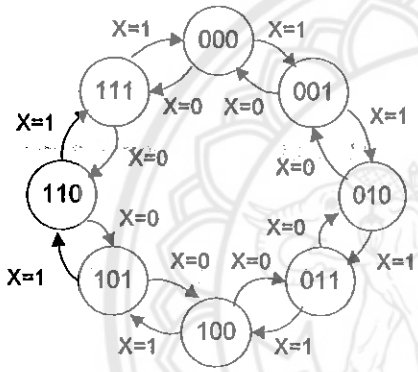
5. Counter circuit



ตัวอย่างที่ 11.5 จงออกแบบวงจรนับขึ้น/ลงซิงโครนัส MOD-8 โดยมีลำดับการนับดังนี้ 0, 1, 2, 4, 5, 6, 7, 6, 5, 4, 3, 2, 1, 0, 1, ... กำหนดให้ใช้ JK FF

<u>วิธีทำ</u>

1. State diagram



2. Excitation table

Q_0	Q_{n+1}	J	K
0	0	0	d
0	1	1	d
1	0	d	1
1	1	d	0

3. State table

Input	Present State	Next State	JK Input FF		
X	$Q_2Q_1Q_0$	$Q_2Q_1Q_0$	J_2K_2	J_1K_1	J_0K_0
0	0 0 0	1 1 1	1 d	1 d	1 d
0	0 0 1	0 0 0	0 d	0 d	d 1
0	0 1 0	0 0 1	0 d	d 1	1 d
0	0 1 1	0 1 0	0 d	d 0	d 1
0	1 0 0	0 1 1	d 1	1 d	1 d
0	1 0 1	1 0 0	d 0	0 d	d 1
0	1 1 0	1 0 1	d 0	d 1	1 d
0	1 1 1	1 1 0	d 0	d 0	d 1
1	0 0 0	0 0 1	0 d	0 d	1 d
1	0 0 1	0 1 0	0 d	1 d	d 1
1	0 1 0	0 1 1	0 d	d 0	1 d
1	0 1 1	1 0 0	1 d	d 1	d 1
1	1 0 0	1 0 1	d 0	0 d	1 d
1	1 0 1	1 1 0	d 0	1 d	d 1
1	1 1 0	1 1 1	d 0	d 0	1 d
1	1 1 1	0 0 0	d 1	d 1	d 1

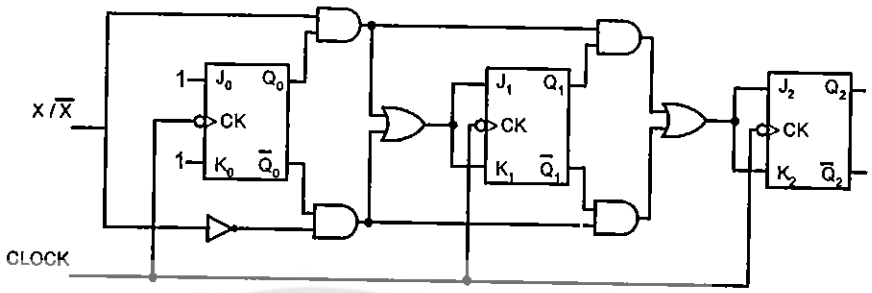
4. JK FF Input equations

$$J_0 = 1 \qquad K_0 = 1$$

$$J_1 = \overline{Q}_0\overline{x} + Q_0 x = Q_0 \oplus x \qquad K_1 = \overline{Q}_0\overline{x} + Q_0 x = Q_0 \oplus x$$

$$J_2 = \overline{Q}_1\overline{Q}_0\overline{x} + Q_1 Q_0 x \qquad K_2 = \overline{Q}_1\overline{Q}_0\overline{x} + Q_1 Q_0 x$$

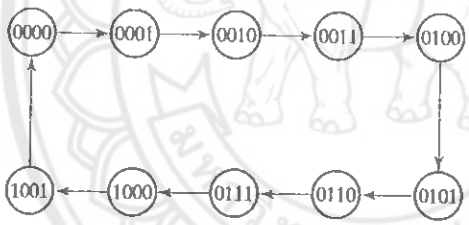
4. Counter circuit



ตัวอย่างที่ 11.6 จงออกแบบวงจรนับขึ้นซิงโครนัสของเลข BCD โดยใช้ T FF **กำหนดให้เอาต์พุต** z **ของลำดับการนับจาก** 9 ไป 0 มีค่าเป็น 1

<u>วิธีทำ</u>

1. State diagram



2.Excitation table

Q_n	Q_{n+1}	T
0	0	0
0	1	1
1	0	1
1	1	0

3. State table

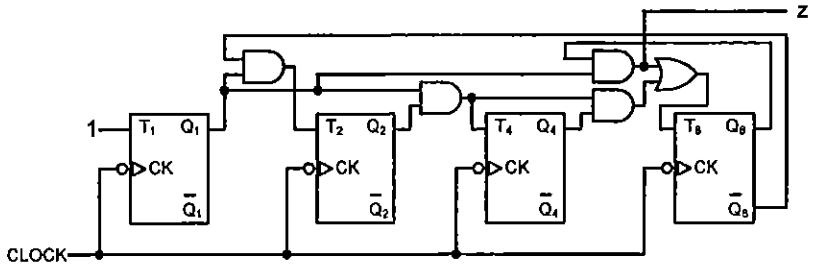
Present State	Next State	Output	T FF Input
$Q_8\ Q_4\ Q_2\ Q_1$	$Q_8\ Q_4\ Q_2\ Q_1$	z	$T_8\ T_4\ T_2\ T_1$
0 0 0 0	0 0 0 1	0	0 0 0 1
0 0 0 1	0 0 1 0	0	0 0 1 1
0 0 1 0	0 0 1 1	0	0 0 0 1
0 0 1 1	0 1 0 0	0	0 1 1 1
0 1 0 0	0 1 0 1	0	0 0 0 1
0 1 0 1	0 1 1 0	0	0 0 1 1
0 1 1 0	0 1 1 1	0	0 0 0 1
0 1 1 1	1 0 0 0	0	1 1 1 1
1 0 0 0	1 0 0 1	0	0 0 0 1
1 0 0 1	0 0 0 0	1	1 0 0 1

4. T FF Input equations

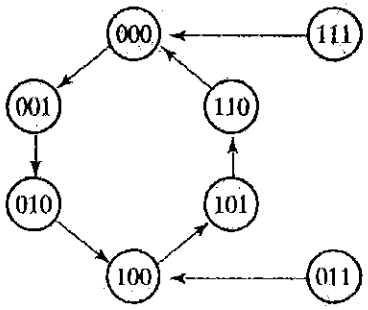
$$T_8 = Q_8Q_1 + Q_4Q_2Q_1 \qquad T_4 = Q_2Q_1$$

$$T_2 = \overline{Q}_8Q_1 \qquad T_1 = 1 \qquad z = Q_8Q_1$$

5. Counter circuit



โดยปกติวงจรนับที่มีฟลิปฟลอปจำนวน n ตัว จะมีสถานะการนับเท่ากับ 2^n ค่า แต่การออกแบบวงจรนับซิงโครนัสสามารถกำหนดรูปแบบการนับได้หลายรูปแบบ โดยไม่จำเป็นต้องนับแบบเลขฐานสองเท่านั้น ถ้าต้องการออกแบบวงจรนับกับสถานะของการนับที่ไม่ใช้ (Counter with unused states) จากตัวอย่างที่ 11.4 วงจรนับมีลำดับของการนับดังนี้ 0, 1, 2, 4, 5, 6, 0, ... ซึ่งการนับเลข 3 และ 7 ไม่ได้อยู่ในอยู่ลำดับของการนับด้วย ดังนั้นใน state table จะกำหนดการนับตรงส่วนนี้เป็น don't care term แต่ถ้าหากพิจารณาในการใช้งานจริง วงจรนับอาจทำการนับ 3 หรือ 7 ได้เนื่องจากความผิดพลาดในการทำงานของวงจรเอง ดังนั้นจึงจำเป็นต้องนำการนับ 3 และ 7 มาใช้การออกแบบวงจรด้วย เพื่อให้วงจรสามารถทำการนับต่อไปได้ การหาลำดับของการนับต่อไปของ 3 และ 7 สามารถวิเคราะห์จากวงจรเดิมที่ได้ทำการออกแบบไปแล้ว ดังนั้นถ้าวงจรนับ 3 ลำดับการนับต่อไปจะเป็น 4 และถ้าวงจรนับ 7 ลำดับการนับต่อไปจะเป็น 0 ดังนั้นจะได้ state diagram ของวงจรการนับใหม่เป็นดังรูป

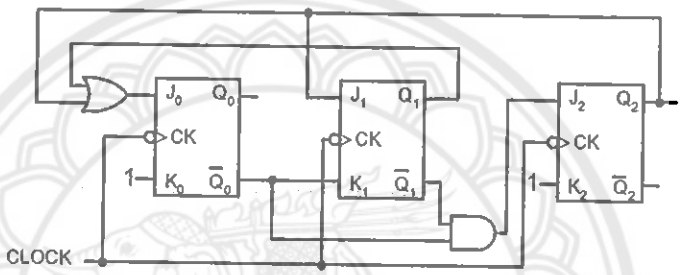


11.4 การหาลำดับของการนับของวงจร

การหาลำดับการนับของวงจรจะเหมือนกับการวิเคราะห์วงจรซีเควนเชียลแบบซิงโครนัส ขั้นตอนมีดังนี้

1. เขียนสมการอินพุตของฟลิปฟลอป (Flip Flop Input equations)
2. เขียนตารางความจริงของฟลิปฟลอปที่ใช้ (Truth table)
3. เขียนตารางสถานะของวงจร (State table) โดยสมมุติกำหนดค่าเริ่มต้นของการนับเป็น 0
4. เขียนแผนภาพสถานะ (State diagram)

ตัวอย่างที่ 11.7 จงหาลำดับการนับของวงจรต่อไปนี้



<u>วิธีทำ</u>

1. FF Input equations

$J_2 = \overline{Q}_1\overline{Q}_0 \qquad K_2 = 1$

$J_1 = Q_2 \qquad K_1 = \overline{Q}_0$

$J_0 = Q_2 + Q_1 \qquad K_0 = 1$

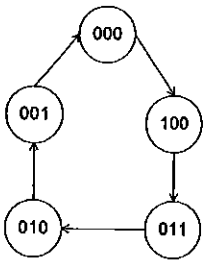
2.Truth table

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

3. State table

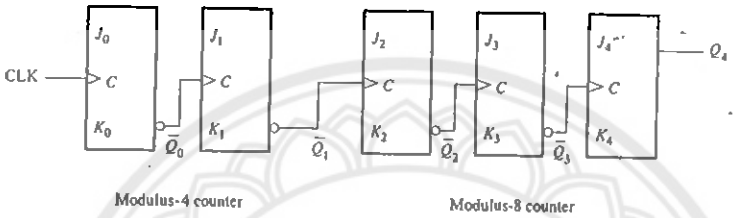
Present State	Next State	JK FF Input		
$Q_2\ Q_1\ Q_0$	$Q_2\ Q_1\ Q_0$	$J_2\ K_2$	$J_1\ K_1$	$J_0\ K_0$
0 0 0	1 0 0	1 1	0 1	0 1
1 0 0	0 1 1	1 1	1 1	1 1
0 1 1	0 1 0	0 1	0 0	1 1
0 1 0	0 0 1	0 1	0 0	1 1
0 0 1	0 0 0	0 1	0 0	0 1

4. State diagram

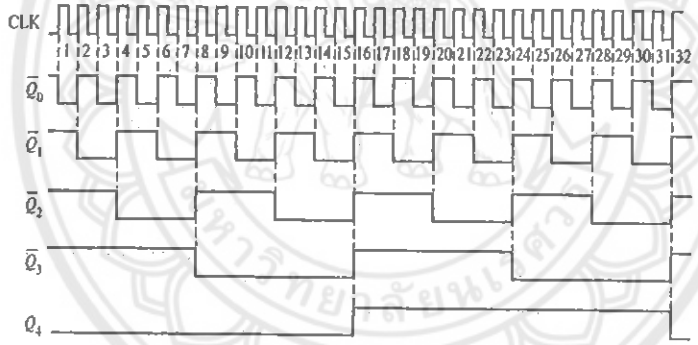


11.5 การนำวงจรนับมาต่อแบบเรียงลำดับ (Cascaded counters)

การออกแบบวงจรนับสามารถนำวงจรนับ MOD-N ใดๆ มาต่อเรียงกัน เพื่อให้เป็นวงจรนับมีลำดับของการนับเพิ่มขึ้นได้ การหาสถานะการนับทั้งหมดของวงจรรวมสามารถหาได้จากการนำค่า N ของวงจรนับแต่ละวงจรที่ต่อกันทั้งหมดมาคูณกัน ตัวอย่างเช่น วงจรดังรูปที่ 11.9 กำหนดให้ J และ K มีค่าเท่ากับ 1 เมื่อนำวงจรนับอะซิงโครนัส MOD-4 และ MOD-8 มาต่อกัน จะได้วงจรนับ MOD-32 (4x8) เมื่อพิจารณาไดอะแกรมเวลาของวงจรดังรูปที่ 11.10 จะเห็นได้ว่าเมื่อสัญญาณนาฬิกาเข้ามา 32 ลูก เอาต์พุต Q_4 ของวงจรนับนับอะซิงโครนัส MOD-8 จะเกิดสัญญาณเพียง 1 ลูกเท่านั้น



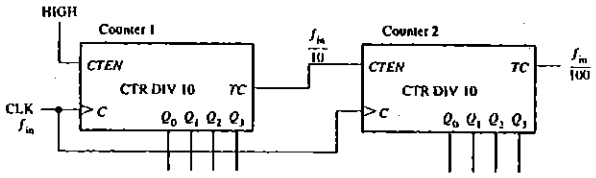
รูปที่ 11.9 การต่อวงจรนับ 2 วงจรแบบเรียงลำดับ



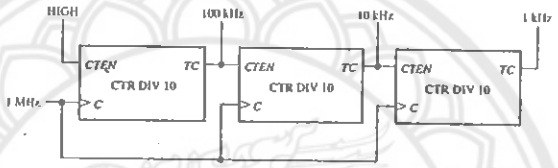
รูปที่ 11.10 Timing diagram ของวงจรรูปที่ 11.9

จากรูปที่ 11.11 เป็นการนำวงจรนับซิงโครนัส MOD-10 จำนวน 2 ชุดมาต่อเรียงกัน โดยภายในวงจรนับจะมีขาควบคุมการทำงานเพิ่มขึ้นมา 2 ขาคือขา CTEN (count enable) เป็นขาที่ใช้ควบคุมการรับและขา TC(Terminal count) เป็นขาเอาต์พุตที่ได้จากหารของวงจรนับ ในไอซีของวงจรนับจะเรียกขานี้ว่า RCO (Ripple clock output) จากวงจรขา TC ของวงจรที่ 1 จะต่อกับขา CTEN ของวงจรที่ 2 ถ้าขา TC ของวงจรที่ 1 เป็น 0 แสดงว่าวงจรที่ 1 ทำการนับเพียงวงจรเดียว แต่ถ้าขา TC ของวงจรที่ 1 เป็น 1 แสดงว่าวงจรที่ 2 ทำการนับได้ ดังนั้นเมื่อวงจรที่ 1 นับสัญญาณ clock ได้ 10 ลูก วงจรที่ 2 จะนับสัญญาณ clock ได้ 1 ลูก ทำให้วงจรนี้เหมือนกับวงจร MOD-100 หรือวงจร DIV-100 ซึ่งวงจร MOD-100 สามารถนำมาใช้เป็นวงจรหารความถี่ของสัญญาณนาฬิกา

ได้ ตัวอย่างเช่น เมื่อสัญญาณนาฬิกามีความถี่ 1 MHz ถ้าต้องการใช้ความถี่ 100 kHz, 10 kHz และ 1 kHz สามารถนำวงจรนับ MOD-10 มาต่อเรียงกัน 3 ชุดดังรูปที่ 11.12 เมื่อป้อนความถี่ 1 MHz เข้าวงจร MOD-10 จะได้เอาต์พุตเป็นความถี่ 100 kHz นำความถี่ที่ได้นี้ป้อนเข้าวงจร MOD-10 ตัวถัดไปจะได้เอาต์พุตความถี่ 10 kHz ความถี่ค่านี้ป้อนเข้าวงจร MOD-10 ตัวถัดสุดท้ายจะได้เอาต์พุตความถี่ 1 kHz



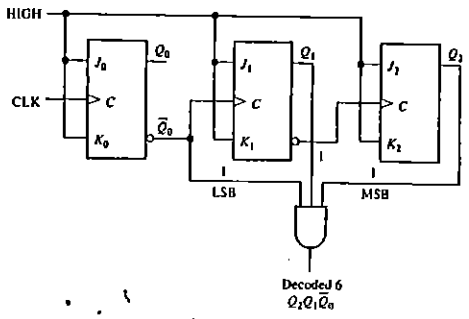
รูปที่ 11.11 การต่อวงจรนับ DIV-10 จำนวน 2 ชุดแบบเรียงลำดับ



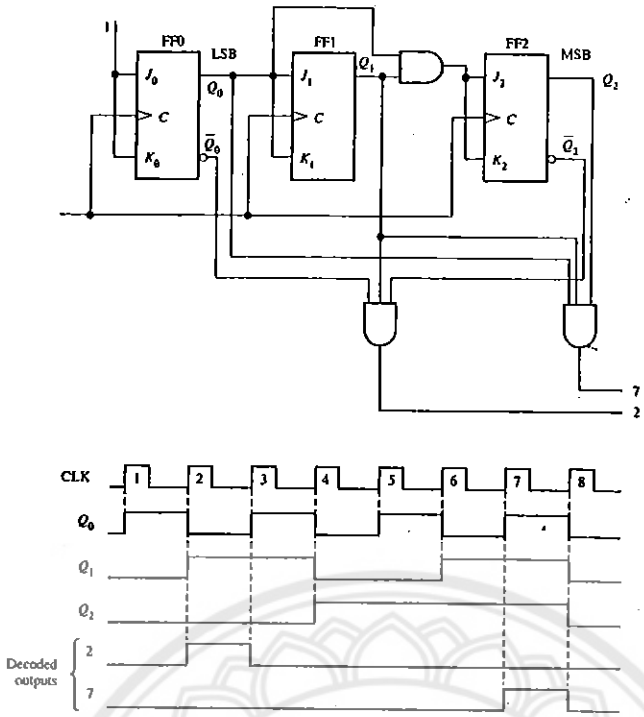
รูปที่ 11.12 การต่อวงจรนับ DIV-10 จำนวน 3 ชุดแบบเรียงลำดับ

11.6 การถอดรหัสวงจรนับ (Counter decoding)

เมื่อวงจรนับทำการนับสามารถดูข้อมูลเอาต์พุตของฟลิปฟลอปที่ได้แต่ละบิต หากต้องการทราบสถานะของการนับที่ค่าใดค่าหนึ่ง เช่น ต้องการทราบว่าวงจรนับแบบ 3 บิตมีสถานะของการนับเป็น 6 (110) หรือยัง สามารถทำการถอดรหัสวงจรนับโดยการนำแอนด์เกตมาต่อในวงจรดังรูปที่ 11.13 ถ้าเอาต์พุตของวงจรที่ได้มีค่าเป็น 6 เมื่อ Q_2=1 Q_1=1 และ $\overline{Q}_0$ = 1 ป้อนเข้าที่แอนด์เกต จะได้เอาต์พุตเป็น HIGH การต่อแบบนี้เรียกว่าการถอดรหัสแบบ Active HIGH แต่ถ้าต้องการเอาต์พุตของการถอดรหัสเป็น LOW สามารถใช้เกตแนนด์ การต่อแบบนี้เรียกว่าการถอดรหัสแบบ Active LOW



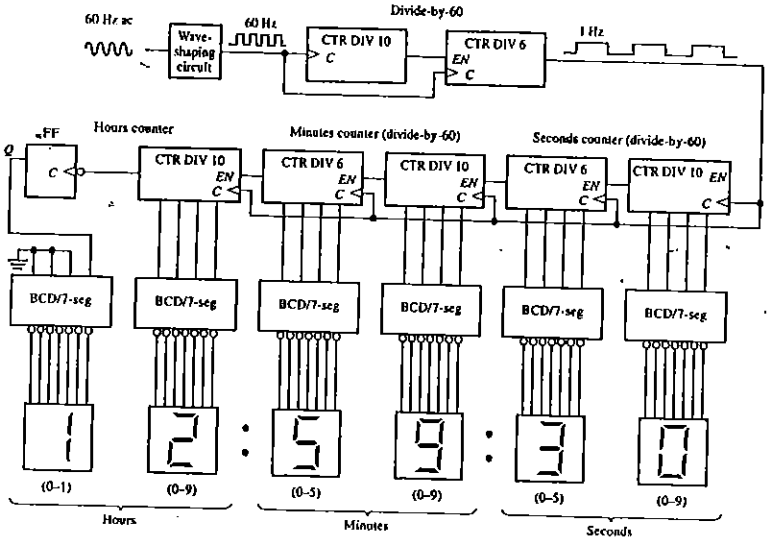
รูปที่ 11.13 การถอดรหัสที่สถานะของการนับเท่ากับ 6 (110)



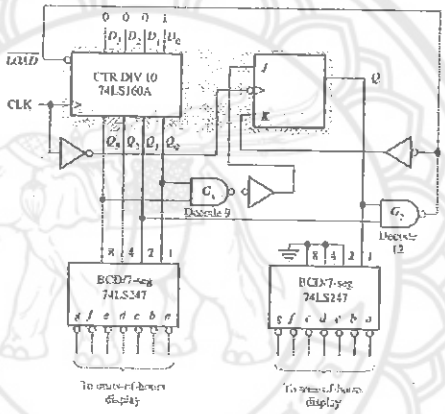
รูปที่ 11.15 การถอดรหัสที่สถานะของการนับเท่ากับ 2 (110) และ 7 (111)

11.7 การประยุกต์ใช้งานวงจรนับ

วงจรนับสามารถนำมาประยุกต์ใช้งานต่างๆ ได้มากมาย เช่น นาฬิกาดิจิตอล การแปลงข้อมูลแบบขนานเป็นอนุกรม การควบคุมการจอดรถในที่จอดรถ เป็นต้น **นาฬิกาดิจิตอล ดังรูปที่** 11.16 เป็นนาฬิกาบอกเวลา 12 ชั่วโมงในรูปแบบ AM และ PM แสดงค่าเป็นชั่วโมง นาทีและวินาที สามารถสร้างจากวงจร MOD ต่างๆ ในชุดของวินาทีและนาทีประกอบด้วย MOD-10 และ MOD-6 ซึ่งจะนับตั้งแต่ 0-59 ส่วนชุดชั่วโมงประกอบด้วย MOD-10 และ MOD-2 ซึ่งจะนับตั้งแต่ 0-12 วงจรทำงานเริ่มต้นจากการรับสัญญาณความถี่ 60 Hz เข้ามาในรูปคลื่นไซน์ จากนั้นทำการแปลงเป็นสัญญาณพัลส์รูปสี่เหลี่ยม สัญญาณที่ได้จะป้อนเข้าวงจร MOD-60 (DIV) จะได้เอาต์พุตเป็นสัญญาณนาฬิกาความถี่ 1 Hz หรือได้สัญญาณพัลส์ออกมาทุกๆ 1 วินาที ซึ่งสัญญาณพัลส์นี้จะป้อนเข้าวงจรนับในหน่วยวินาที นาที และชั่วโมง ตามลำดับ รายละเอียดการออกแบบวงจรในหน่วยชั่วโมงแสดงดังรูปที่ 11.17 เมื่อวงจรนับ MOD-20 ทำการนับถึง 13 ให้รีเซตเอาต์พุตเพื่อเริ่มการนับใหม่

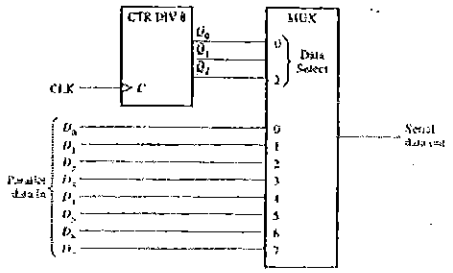


รูปที่ 11.16 วงจรนาฬิกาดิจิตอลอย่างง่าย

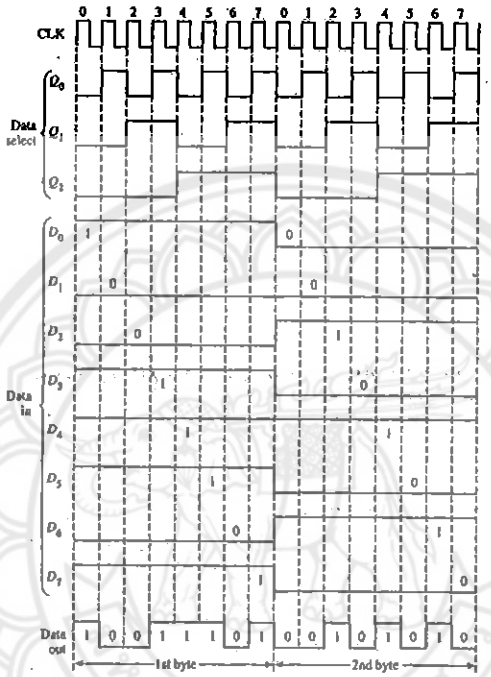


รูปที่ 11.17 วงจรนับ MOD-20 ในหน่วยชั่วโมง

การแปลงข้อมูลแบบขนานเป็นอนุกรม สามารถออกแบบวงจรได้ดังรูปที่ 11.18 โดยวงจรนับ MOD-8 จะให้ค่าลำดับการนับมีค่าตั้งแต่ 0-7 ค่าการนับนี้จะป้อนเข้าวงจรมัลติเพล็กซ์ (MUX) ที่ data select เพื่อใช้ในการเลือกข้อมูลบิตใดๆ ส่งออกไปยังเอาต์พุต ข้อมูลอินพุตของวงจรมัลติเพล็กซ์ที่เข้ามาพร้อมกันทุกบิตเรียกว่าข้อมูลแบบขนานและข้อมูลเอาต์พุตของวงจรมัลติเพล็กซ์ที่ออกมาแต่ละบิตเรียกว่าข้อมูลแบบอนุกรม ตัวอย่างการทำงานของวงจรแสดงดังรูปที่ 11.19 สมมุติว่าป้อนข้อมูลอินพุต 8 บิต (D_0-D_7) เข้าที่ MUX เมื่อสัญญาณ clock ลูกที่ 1 เข้ามา ทำให้วงจรนับเป็น 000 ดังนั้นข้อมูลบิต D_0 จะถูกเลือกออกมาทางเอาต์พุตของวงจร MUX เมื่อสัญญาณ clock ลูกที่ 2 เข้ามา ทำให้วงจรนับเป็น 001 ดังนั้นข้อมูลบิต D_1 จะถูกเลือกออกมาทางเอาต์พุตของวงจร MUX เป็นเช่นนี้ไปเรื่อยๆ จนกระทั่งสัญญาณ clock เข้ามาครบ 8 ลูก ข้อมูลจะถูกส่งออกมา 8 บิต การแปลงข้อมูลแบบขนานเป็นอนุกรมนี้สามารถควบคุมความเร็วได้ขึ้นอยู่กับวงจรนับ

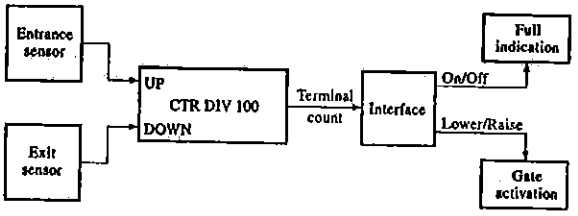


รูปที่ 11.18 บล็อกไดอะแกรมแสดงการแปลงข้อมูลแบบขนานเป็นอนุกรม



รูปที่ 11.19 การแปลงข้อมูลแบบขนานเป็นอนุกรม

การควบคุมการจอดรถในที่จอดรถ เนื่องจากปัญหาที่จอดรถไม่เพียงพอ สามารถออกแบบวงจรอย่างง่ายดังรูปที่ 11.20 สมมุติให้ที่จอดรถนี้จำกัดจำนวนรถที่เข้ามาจอดได้เพียง 100 คันเท่านั้น โดยใช้วงจรนับขึ้น/ลง MOD-100 เมื่อมีรถเข้าออกจะมี sensor ตรวจจับ สัญญาณจาก sensor จะส่งไปยังวงจรนับเพื่อทำการนับจำนวนรถที่เข้าออก เอาต์พุตของวงจรนับที่ได้จะส่งไปยังแผงกั้นเพื่อควบคุมการปิดเปิดให้รถสามารถเข้ามาจอดได้และส่งไปยังสัญญาณไฟแจ้งเตือนว่าที่จอดรถเต็มหรือไม่



รูปที่ 11.20 บล็อกไดอะแกรมการควบคุมการจอดรถในที่จอดรถ

บทที่ 12
วงจรเลื่อนข้อมูล (Shift Register)

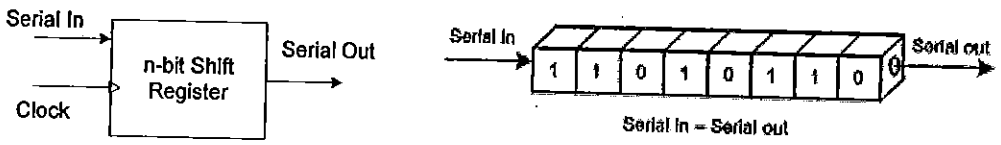
วัตถุประสงค์

1. สามารถเข้าใจหลักการทำงานของวงจรเลื่อนข้อมูลแบบ SISO SIPO PISO และ PIPO
2. สามารถออกแบบและนำวงจรเลื่อนข้อมูลไปประยุกต์ใช้งานต่างๆ ได้

วงจรเลื่อนข้อมูลหรือชิฟรีจิสเตอร์ (Shift register) เป็นวงจรซีเควนเชียลที่สร้างจากฟลิปฟลอปเพื่อทำหน้าที่เก็บและเลื่อนข้อมูล 0 หรือ 1 ในระบบดิจิตอล ซึ่งนำไปประยุกต์ใช้งานอย่างแพร่หลายทั้งในเครื่องคอมพิวเตอร์ เครื่องคำนวณ เครื่องพิมพ์ การหน่วงเวลาข้อมูล วงจรที่ต้องการส่งข้อมูลเข้าและออกแบบขนานหรือแบบอนุกรม เป็นต้น จากลักษณะการทำงานของฟลิปฟลอปชนิดต่างๆ เช่น D FF, JK FF, RS FF และ T FF จะมีสภาวะการทำงานแบบ latch นั่นคือเอาต์พุตของฟลิปฟลอปจะคงค่าเดิมไว้ แม้ว่าอินพุตจะมีการเปลี่ยนแปลงก็ตามหรือกล่าวได้ว่าฟลิปฟลอปจะมีการเก็บข้อมูลไว้นั่นเอง โดยทั่วไปฟลิปฟลอป 1 ตัว สามารถเก็บข้อมูลได้ 1 บิตหรือรีจิสเตอร์ขนาด 1 บิต ถ้านำฟลิปฟลอปชนิดเดียวกันจำนวน n ตัวมาต่อเรียงกันเพื่อเก็บข้อมูลจำนวน n บิต หรือเรียกว่ารีจิสเตอร์ขนาด n บิต ดังนั้นชิฟรีจิสเตอร์จึงเป็นกลุ่มของฟลิปฟลอปที่สามารถเก็บข้อมูลและมีการเลื่อนข้อมูลเข้ามาเก็บทางซ้าย ทางขวาหรือมีการเลื่อนข้อมูลเข้าออกเป็นชุด รูปแบบการเก็บหรือการเลื่อนข้อมูลดิจิตอลมี 2 รูปแบบคือแบบอนุกรม (Serial data) หมายถึงข้อมูลเกิดขึ้น 1 บิตเรียงตามลำดับทีละบิตเมื่อเวลาผ่านไป และแบบขนาน (Parallel data) หมายถึงข้อมูลเกิดขึ้นตั้งแต่ 1 บิตขึ้นไป สามารถพิจารณาการทำงานทีละหลายบิตพร้อมกันได้เมื่อเวลาผ่านไป ลักษณะโครงสร้างและการทำงานของชิฟรีจิสเตอร์สามารถแบ่งเป็น 4 ประเภท คือ

1. ชิฟรีจิสเตอร์แบบเข้าอนุกรม-ออกอนุกรม (Serial In-Serial Out, SISO)
2. ชิฟรีจิสเตอร์แบบเข้าอนุกรม-ออกขนาน (Serial In-Parallel Out, SIPO)
3. ชิฟรีจิสเตอร์เข้าขนาน-ออกอนุกรม (Parallel In-Serial Out, PISO)
4. ชิฟรีจิสเตอร์เข้าขนาน-ออกขนาน (Parallel In- Parallel Out, PIPO)

12.1 ชิฟรีจิสเตอร์แบบเข้าอนุกรม-ออกอนุกรม (SISO)

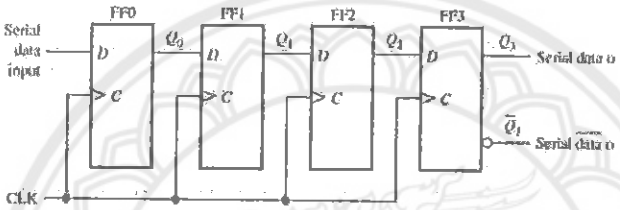


(ก) บล็อกไดอะแกรม (ข) ชิฟรีจิสเตอร์ขนาด 8 บิต

รูปที่ 12.1 ชิฟรีจิสเตอร์แบบ SISO

โครงสร้างทั่วไปและตัวอย่างของชิฟรีจิสเตอร์แบบ SISO ขนาด 8 บิต แสดงดังรูปที่ 12.1 ชิฟรีจิสเตอร์แบบนี้จะมีข้อมูลอินพุตเข้าแบบอนุกรมและให้ข้อมูลเอาต์พุตเป็นแบบอนุกรมด้วย นั่นคือข้อมูลมีการเลื่อนเข้ามาเก็บที่รีจิสเตอร์ทีละ 1 บิต และมีข้อมูลออกจากรีจิสเตอร์ทีละ 1 บิต โดยเรียงตามลำดับเวลาเช่นเดียวกัน จากรูป12.1 ข้อมูลมีการเลื่อนจากซ้ายไปขวา การเลื่อนข้อมูลเข้าหรือออกจะสัมพันธ์กับสัญญาณนาฬิกาที่ใช้ควบคุมการทำงาน

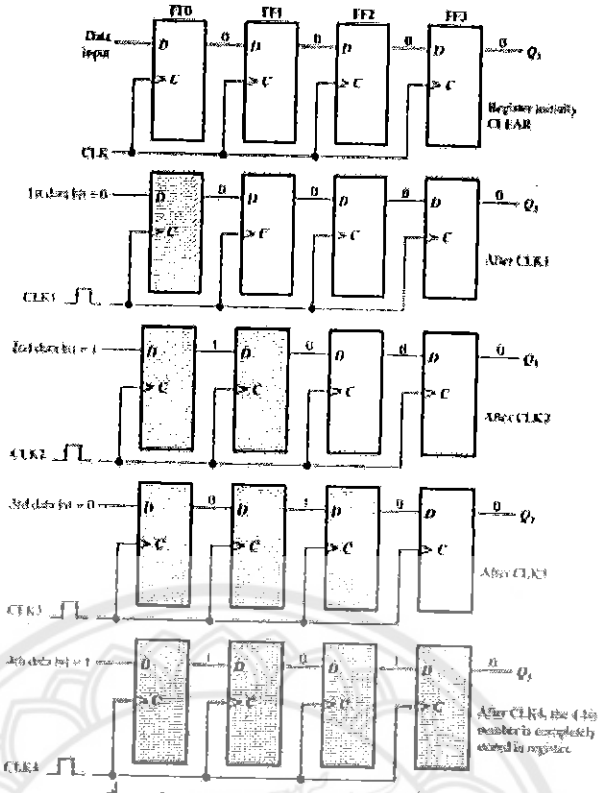
ตัวอย่างวงจรชิฟรีจิสเตอร์แบบ SISO ขนาด 4 บิตอย่างง่าย แสดงดังรูปที่ 12.2 วงจรประกอบด้วย D FF จำนวน 4 ตัวต่ออนุกรมกัน เมื่อสัญญาณ clock เข้ามากระตุ้นและมีการเปลี่ยนจาก 0 เป็น 1 (ขอบขาขึ้น) ทำให้ D FF ทำงานในสภาวะแบบ SET หรือ RESET ขึ้นอยู่กับอินพุตที่ขา D ของ FF แต่ละตัว



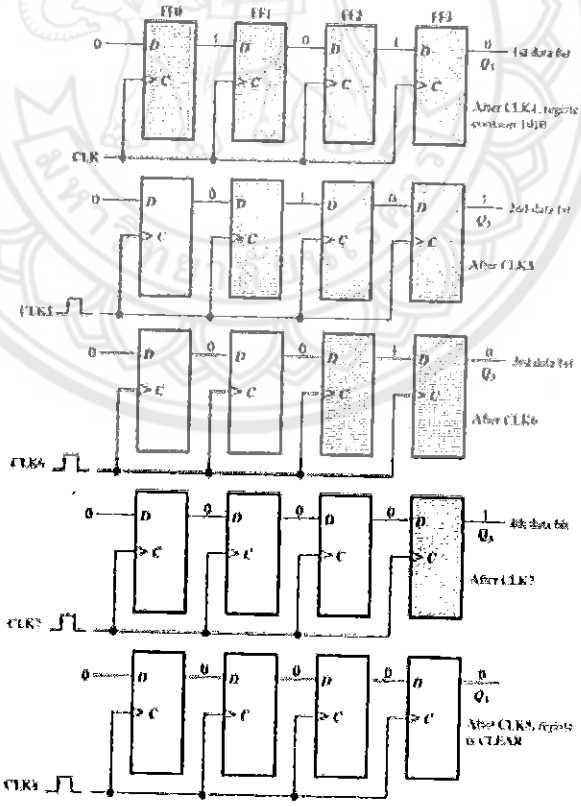
รูปที่ 12.2 ชิฟรีจิสเตอร์แบบ SISO ขนาด 4 บิต

จากรูปที่ 12.3 (ก) แสดงตัวอย่างการทำงานของวงจรชิฟรีจิสเตอร์แบบ SISO ขนาด 4 บิต เพื่อเลื่อนข้อมูล 1010 มาเก็บไว้ในรีจิสเตอร์ โดยเริ่มแรกวงจรทำการเคลียร์ข้อมูล ทำให้เอาต์พุตของรีจิสเตอร์เป็น 0 ทั้งหมด จากนั้นทำการป้อนข้อมูลบิตต่ำสุดเข้าไปก่อน(LSB) ข้อมูลบิตแรกคือ 0 ที่ขา D ของ FF0 เมื่อสัญญาณ clock ลูกแรกเข้ามา ข้อมูล 0 จะปรากฏที่เอาต์พุตของ FF0 ทำให้รีจิสเตอร์ FF0 เก็บค่า 0 ไว้ ต่อมาทำการป้อนข้อมูลบิตที่สองคือ 1 ที่ขา D ของ FF0 และขณะนี้ขา D ของ FF1 มีค่าเป็น 0 เนื่องจากขา D ของ FF1 ต่อกับเอาต์พุตของ FF0 เมื่อสัญญาณ clock ลูกที่สองเข้ามา ทำให้ FF0 เก็บค่า 1 และข้อมูลบิต 0 จะเลื่อนมาเก็บที่ FF1 จากนั้นป้อนข้อมูลบิตที่สามคือ 0 ที่ขา D ของ FF0 ขณะนี้ขา D ของ FF1 มีค่าเป็น 1 และขา D ของ FF2 มีค่าเป็น 0 เมื่อสัญญาณ clock ลูกที่สามเข้ามา ทำให้ FF0 เก็บค่า 0 FF1 เก็บค่า 1 และ FF2 เก็บค่า 0 ตามลำดับ ต่อมาทำการป้อนข้อมูลบิตที่สี่คือ 1 ที่ขา D ของ FF0 ขณะนี้ขา D ของ FF1 มีค่าเป็น 0 ขา D ของ FF2 มีค่าเป็น 1 และขา D ของ FF3 มีค่าเป็น 0 เมื่อสัญญาณ clock ลูกที่สี่เข้ามา ทำให้ FF0 เก็บค่า 1 FF1 เก็บค่า 0 FF2 เก็บค่า 1 และ FF3 เก็บค่า 0 ตามลำดับ จะเห็นได้ว่าเมื่อสัญญาณ clock เข้ามาครบสี่ลูก ทำให้รีจิสเตอร์เก็บข้อมูล 1010 ไว้

การเลื่อนข้อมูล 1010 ออกจากรีจิสเตอร์ แสดงดังรูปที่ 12.3 (ข) เมื่อสัญญาณ clock เข้ามาอีก 4 ลูกจะได้ข้อมูลออกจากรีจิสเตอร์ที่เอาต์พุต Q_3 ครบ 4 บิตและทำการเคลียร์รีจิสเตอร์ให้เป็น 0 ทั้งหมด



(ก) การเลื่อนข้อมูล 1010 เข้ามาเก็บในชิฟรีจิสเตอร์



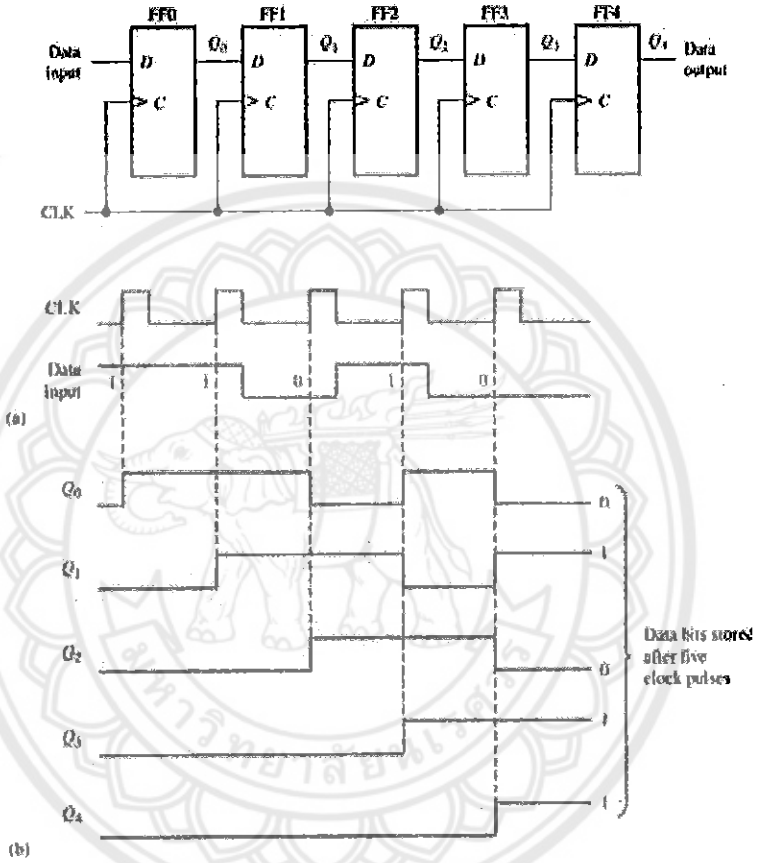
(ข) การเลื่อนข้อมูล 1010 ออกจากชิฟรีจิสเตอร์

รูปที่ 12.4 การเลื่อนข้อมูลชิฟรีจิสเตอร์

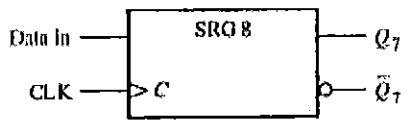
ตัวอย่างที่ 12.1 จากวงจรชิฟรีจิสเตอร์แบบ SISO ขนาด 5 บิต ถ้ากำหนดให้อินพุตและสัญญาณ clock เป็นดังรูป (a) จงหาเอาต์พุตของรีจิสเตอร์ (b) ถ้ากำหนดให้เริ่มต้นรีจิสเตอร์ถูกเคลียร์ทั้งหมด

<u>วิธีทำ</u>

เมื่อสัญญาณ clock ลูกแรกเข้ามาทำให้ข้อมูลบิตแรกคือ 1 จะเลื่อนเข้ามาเก็บที่รีจิสเตอร์ FF0 ทำให้ Q_0 มีค่าเท่ากับ 1 จากนั้นข้อมูลจะเลื่อนจากซ้ายไปขวา เมื่อสัญญาณ clock เข้ามาครบ 5 ลูก จะทำให้รีจิสเตอร์เก็บข้อมูลดังนี้ $Q_4\ Q_3\ Q_2\ Q_1\ Q_0 = 11010$



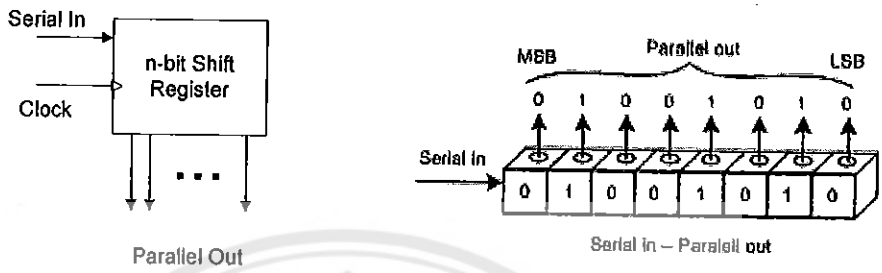
ตัวอย่างสัญลักษณ์ของชิฟรีจิสเตอร์แบบ SISO ขนาด 8 บิตแสดงดังรูปที่ 12.4 จะเขียนว่า SRG 8 ซึ่งย่อมาจาก Shift register



รูปที่ 12.4 สัญลักษณ์ของชิฟรีจิสเตอร์แบบ SISO ขนาด 8 บิต

12.2 ชิฟรีจิสเตอร์แบบเข้าอนุกรม-ออกขนาน (SIPO)

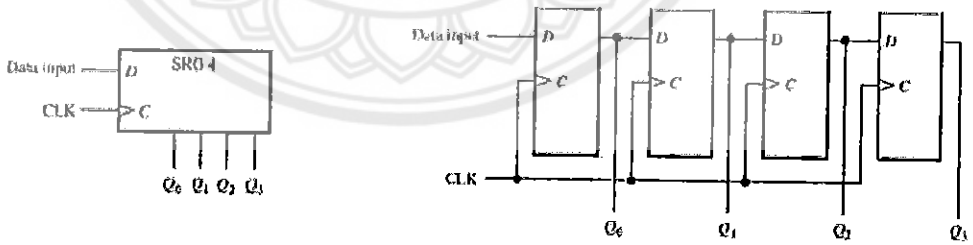
โครงสร้างทั่วไปและตัวอย่างของชิฟรีจิสเตอร์แบบ SIPO ขนาด 8 บิต แสดงดังรูปที่ 12.5 ชิฟรีจิสเตอร์แบบนี้มีข้อมูลอินพุตเข้าแบบอนุกรมและข้อมูลออกที่เอาต์พุตเป็นแบบขนาน นั่นคือจะมีข้อมูลเข้ามาที่รีจิสเตอร์ทีละ 1 บิตและมีข้อมูลออกจากรีจิสเตอร์ทีละหลายบิตตามจำนวนขาเอาต์พุตของฟลิปฟลอป



(ก) บล็อกไดอะแกรม (ข) ชิฟรีจิสเตอร์ขนาด 8 บิต

รูปที่ 12.5 ชิฟรีจิสเตอร์แบบ SIPO

ตัวอย่างของวงจรชิฟรีจิสเตอร์แบบ SIPO ขนาด 4 บิตแสดงดังรูปที่ 12.6 วงจรประกอบด้วย D FF จำนวน 4 ตัวต่ออนุกรมกัน เมื่อสัญญาณ clock เข้ามาและมีการเปลี่ยนจาก 0 เป็น 1 (ขอบขาขึ้น) ทำให้ D FF ทำงานในสภาวะแบบ SET หรือ RESET ขึ้นอยู่กับอินพุตที่ขา D ของ FF แต่ละตัว การทำงานของวงจรพิจารณาได้ดังนี้ ข้อมูลอินพุตจะป้อนเข้าที่ขา D ของฟลิปฟลอปตัวแรก ส่วนข้อมูลเอาต์พุตจะออกที่ขาเอาต์พุตของฟลิปฟลอปทุกตัว (Q_0-Q_3)

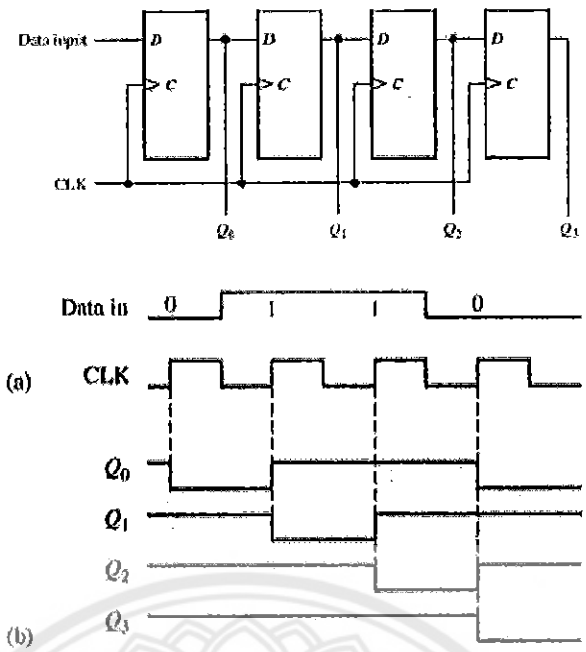


(ก) สัญลักษณ์ (ข) วงจร

รูปที่ 12.6 ชิฟรีจิสเตอร์แบบ SIPO ขนาด 4 บิต

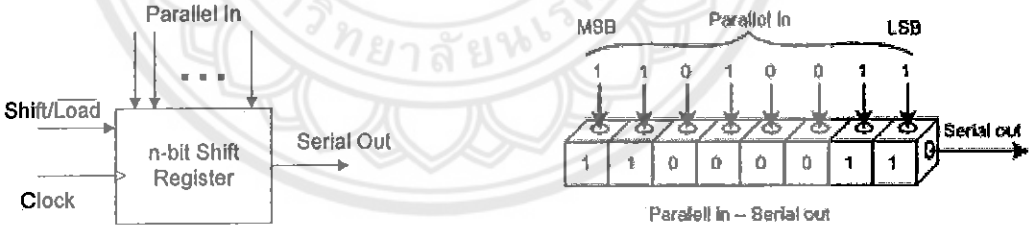
ตัวอย่างที่ 12.2 จากวงจรชิฟรีจิสเตอร์แบบ SIPO ขนาด 4 บิต กำหนดให้อินพุตและสัญญาณ clock เป็นดังรูป (a) จงหาเอาต์พุตของรีจิสเตอร์ (Q_0-Q_3)

<u>วิธีทำ</u> เมื่อสัญญาณ clock เข้ามาครบ 4 ลูกทำให้รีจิสเตอร์เก็บข้อมูลดังนี้ $Q_3\ Q_2\ Q_1\ Q_0 = 0110$



12.3 ชิฟรีจิสเตอร์แบบเข้าขนาน-ออกอนุกรม (PISO)

โครงสร้างทั่วไปและตัวอย่างของชิฟรีจิสเตอร์แบบ PISO ขนาด 8 บิต แสดงดังรูปที่ 12.7 ชิฟรีจิสเตอร์แบบนี้มีข้อมูลอินพุตเข้าไปเป็นแบบขนานและให้ข้อมูลทางเอาต์พุตเป็นแบบอนุกรม นั่นคือจะมีข้อมูลเข้าที่รีจิสเตอร์พร้อมกันทีละหลายบิตและใช้สัญญาณ clock ควบคุมการเลื่อนข้อมูลออกจากรีจิสเตอร์ทีละ 1 บิต



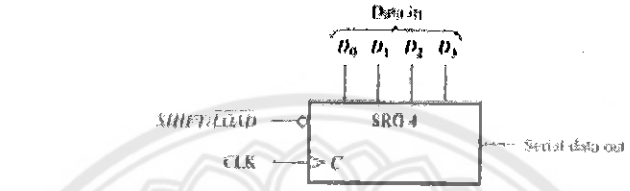
(ก) บล็อกไดอะแกรม (ข) ชิฟรีจิสเตอร์ขนาด 8 บิต

รูปที่ 12.7 ชิฟรีจิสเตอร์แบบ PISO

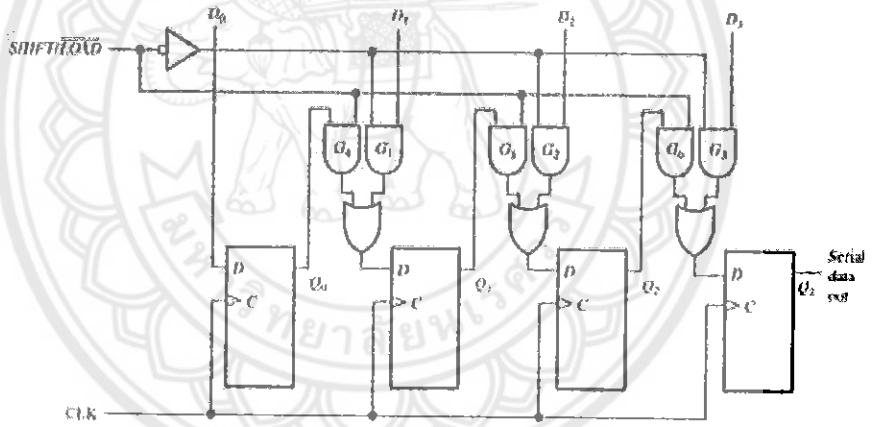
จากรูปที่ 12.8 แสดงสัญลักษณ์และวงจรชิฟรีจิสเตอร์แบบ PISO ขนาด 4 บิต โดยใช้ D FF จำนวน 4 ตัว ข้อมูลอินพุตจะถูกโหลดเข้ามาที่ขา D_0-D_3 เพื่อเก็บข้อมูลไว้ที่รีจิสเตอร์และข้อมูลเอาต์พุตจะส่งออกมาที่ขา Q_3 ทีละบิต ส่วนขา SHIFT/$\overline{\text{LOAD}}$ ใช้ในการควบคุมการทำงานในการเลื่อนและการโหลดข้อมูลดังนี้

ถ้าขา SHIFT/$\overline{\text{LOAD}}$ เป็น 0 วงจรจะทำการโหลดข้อมูลเข้ามาเก็บที่รีจิสเตอร์พร้อมกัน โดยแอนด์เกต G_1 G_2 G_3 จะมีขาหนึ่งมีค่าลอจิกเป็น 1 ส่วนอีกขาหนึ่งของแอนด์เกต G_1 G_2 G_3 จะขึ้นอยู่กับข้อมูลอินพุต D_1 D_2 D_3 ตามลำดับ เมื่อมีสัญญาณ clock เข้ามากระตุ้น ถ้าอินพุต D=1 จะให้เอาต์พุตเป็นเซตและ D=0 จะให้เอาต์พุตเป็นรีเซต

ถ้าขา SHIFT/$\overline{\text{LOAD}}$ เป็น 1 วงจรจะทำการเลื่อนข้อมูล โดยเกตแอนด์ G_4 G_5 G_6 จะมีขาหนึ่งมีค่าลอจิกเป็น 1 ส่วนอีกขาหนึ่งของเกตแอนด์ G_4 G_5 G_6 จะขึ้นอยู่กับข้อมูลเอาต์พุตของ D FF แต่ละตัว เมื่อมีสัญญาณ clock เข้ามากระตุ้นจะมีการเลื่อนข้อมูลไปทางขวาและออกทางเอาต์พุต Q_3



(ก) สัญลักษณ์

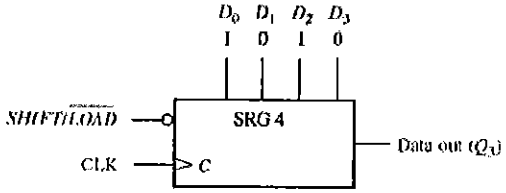


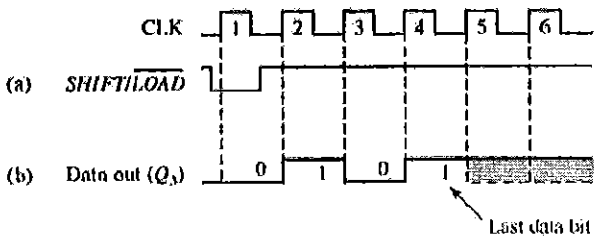
(ข) วงจร

รูปที่ 12.8 ชิฟรีจิสเตอร์แบบ PISO ขนาด 4 บิต

ตัวอย่างที่ 12.3 จากวงจรชิฟรีจิสเตอร์แบบ PISO ขนาด 4 บิต ถ้ากำหนดให้ข้อมูลอินพุต D_0 D_1 D_2D_3 = 1010 สัญญาณ clock และ SHIFT/$\overline{\text{LOAD}}$ เป็นดังรูป (a) จงหาเอาต์พุตของรีจิสเตอร์ Q_3

<u>**วิธีทำ**</u>

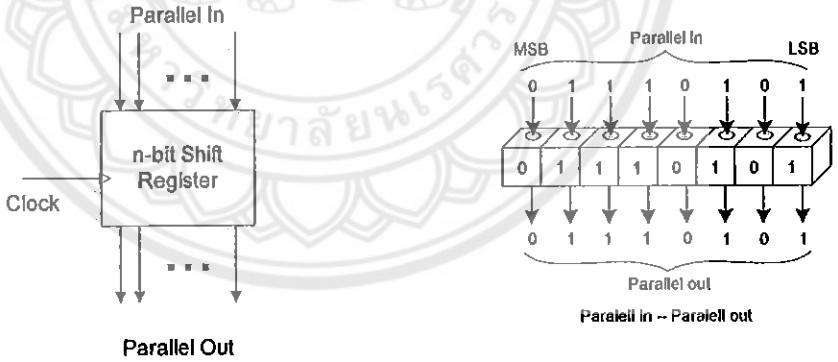




เมื่อสัญญาณ clock ลูกแรกเข้ามาและ SHIFT/$\overline{\text{LOAD}}$ เป็น 0 วงจรจะโหลดข้อมูล 1010 เข้ามาเก็บไว้ที่รีจิสเตอร์ ทำให้เอาต์พุต Q_3=0 ต่อมาสัญญาณ SHIFT/$\overline{\text{LOAD}}$ เป็น 1 วงจรจะทำการเลื่อนบิต ออกที่เอาต์พุต Q_3 ทุกครั้งที่มีสัญญาณ clock เข้ามา ดังนั้นเมื่อสัญญาณ clock ลูกที่สองเข้ามาทำให้ข้อมูลเอาต์พุต Q_0 เลื่อนไปอยู่ที่ Q_1 ข้อมูลเอาต์พุต Q_1 เลื่อนไปอยู่ที่ Q_2 และข้อมูลเอาต์พุต Q_2 เลื่อนไปอยู่ที่ Q_3 ตามลำดับ ทำให้เอาต์พุต Q_3=1 เมื่อสัญญาณ clock เข้ามาครบ 4 ลูก ข้อมูลจะเลื่อนออกที่เอาต์พุต Q_3 ครบ 4 บิต

12.4 ชิฟรีจิสเตอร์แบบเข้าขนาน-ออกขนาน (PIPO)

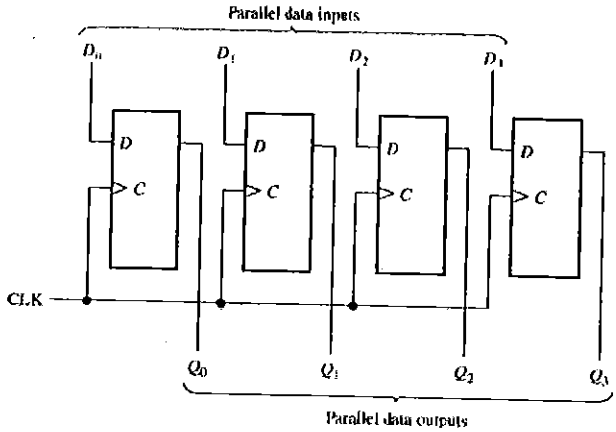
โครงสร้างทั่วไปและตัวอย่างของชิฟรีจิสเตอร์แบบ PIPO ขนาด 8 บิต แสดงดังรูปที่ 12.9 ชิฟรีจิสเตอร์แบบนี้มีข้อมูลอินพุตเข้าแบบขนานและให้ข้อมูลเอาต์พุตออกเป็นแบบขนานด้วย นั่นคือจะมีข้อมูลเข้าและออกที่รีจิสเตอร์พร้อมกันทุกบิต



(ก) บล็อกไดอะแกรม (ข) ชิฟรีจิสเตอร์ขนาด 8 บิต

รูปที่ 12.9 ชิฟรีจิสเตอร์แบบ PIPO

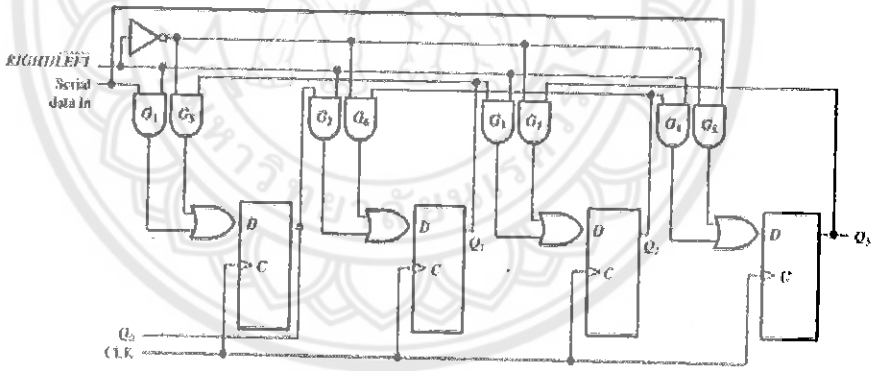
จากรูปที่ 12.10 แสดงวงจรชิฟรีจิสเตอร์แบบ PIPO ขนาด 4 บิต โดยใช้ D FF จำนวน 4 ตัว ข้อมูลอินพุต D_0-D_3 จะถูกโหลดเข้ามาเก็บที่รีจิสเตอร์พร้อมกันทุกบิต และข้อมูลเอาต์พุต Q_0-Q_3 จะถูกส่งออกจากรีจิสเตอร์พร้อมกันทุกบิตเช่นเดียวกัน



รูปที่ 12.10 ชิฟรีจิสเตอร์แบบ PIPO ขนาด 4 บิต

12.5 ชิฟรีจิสเตอร์แบบสองทิศทาง (Bidirectional Shift Register)

ชิฟรีจิสเตอร์แบบสองทิศทางสามารถกำหนดให้ข้อมูลมีการเลื่อนไปทางขวาหรือทางซ้ายได้ ขึ้นอยู่กับสัญญาณควบคุม จากรูปที่ 12.11 ถ้าให้ขาควบคุม RIGHT/$\overline{\text{LEFT}}$ เป็น 1 จะทำให้บิตข้อมูลที่เก็บอยู่ในรีจิสเตอร์เลื่อนไปทางขวา แต่ถ้าขาควบคุม RIGHT/$\overline{\text{LEFT}}$ เป็น 0 จะทำให้บิตข้อมูลที่เก็บอยู่ในรีจิสเตอร์เลื่อนไปทางซ้าย การเลื่อนข้อมูลแต่ละครั้งขึ้นอยู่กับสัญญาณ clock ที่เข้ามา

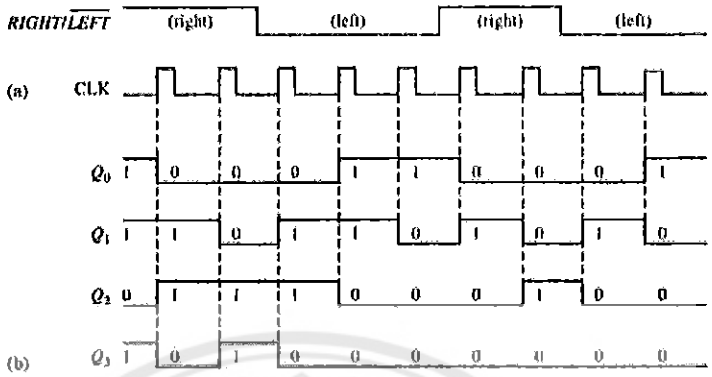


รูปที่ 12.11 ชิฟรีจิสเตอร์แบบสองทิศทางขนาด 4 บิต

พิจารณาการทำงานของวงจร เมื่อขาควบคุม RIGHT/$\overline{\text{LEFT}}$ เป็น 1 ทำให้เกต G_1 ถึง G_4 ถูกอินาเบิลและทำให้เอาต์พุต Q ของ D FFแต่ละตัวส่งค่าไปยัง D FF ที่อยู่ถัดไป เมื่อสัญญาณ clock เข้ามา บิตข้อมูลทั้งหมดจะเลื่อนไปทางขวา เมื่อขาควบคุม RIGHT/$\overline{\text{LEFT}}$ เป็น 0 ทำให้เกต G_5 ถึง G_8 ถูกอินาเบิลและทำให้เอาต์พุต Q ของ D FF แต่ละตัวส่งค่าไปยัง D FF ที่อยู่ก่อนหน้านี้ เมื่อสัญญาณ clock เข้ามา บิตข้อมูลทั้งหมดจะเลื่อนไปทางซ้าย

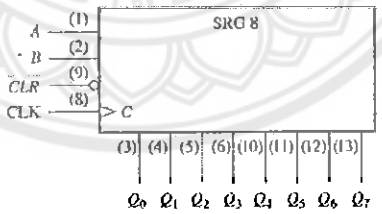
ตัวอย่างที่ 12.4 จากวงจรชิฟรีจิสเตอร์แบบสองทิศทางขนาด 4 บิตดังรูปที่ 12.11 ถ้ากำหนดให้ข้อมูลอินพุต D=0 เอาต์พุตเริ่มต้นของ Q_0 Q_1 Q_2 Q_3 = 1101 สัญญาณ clock และสัญญาณควบคุม RIGHT/$\overline{\text{LEFT}}$ เป็นดังรูป (a) จงหาเอาต์พุตของรีจิสเตอร์ทุกบิต

<u>วิธีทำ</u>

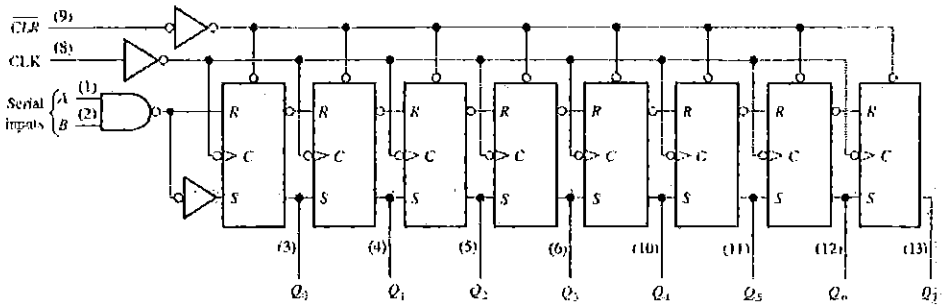


12.6 ไอซีชิฟรีจิสเตอร์

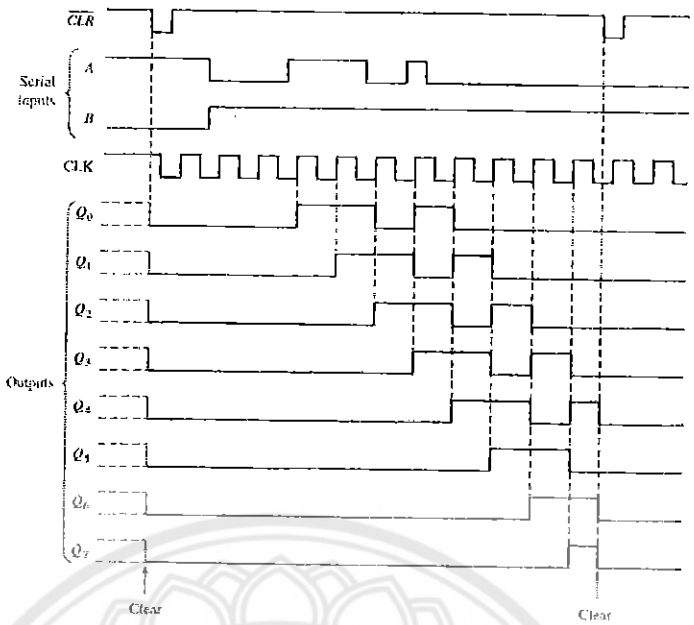
ไอซีเบอร์ 74HC164 เป็นไอซีชิฟรีจิสเตอร์แบบ SIPO ขนาด 8 บิต จากรูปที่ 12.12 (ข) แสดงโครงสร้างของวงจรสร้างจาก RS FF จำนวน 8 ตัวมาต่อเรียงกัน โดยข้อมูลอินพุตจะเข้าที่ขา A และ B ของแนนด์เกตและข้อมูลเอาต์พุตคือ Q_0 ถึง Q_8 สำหรับขา $\overline{CLR}$ จะ ACTIVE LOW เพื่อทำการเคลียร์ RS FF ทุกตัว ตัวอย่างไดอะแกรมเวลาของสัญญาณที่ได้จากไอซีเบอร์ 74HC164 แสดงได้ดังรูปที่12.12 (ค) จะเห็นได้ว่าเมื่อข้อมูลอินพุตที่ขา B=1 ทำให้ข้อมูลอินพุตที่ขา A ทำการเลื่อนเข้าและออกจากรีจิสเตอร์



(ก) สัญลักษณ์



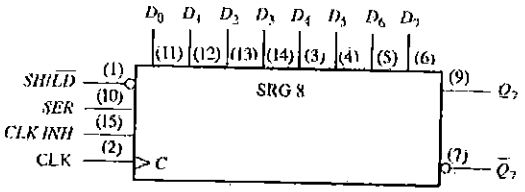
(ข) วงจร



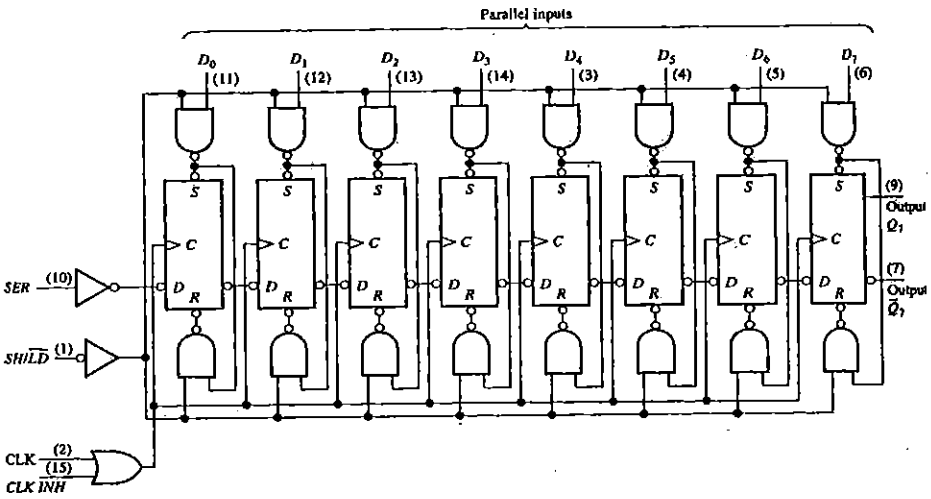
(ค) ตัวอย่างไดอะแกรมแสดงการการทำงาน

รูปที่ 12.12 ไอซีเบอร์ 74HC164 ชิฟรีจิสเตอร์แบบ SIPO ขนาด 8 บิต

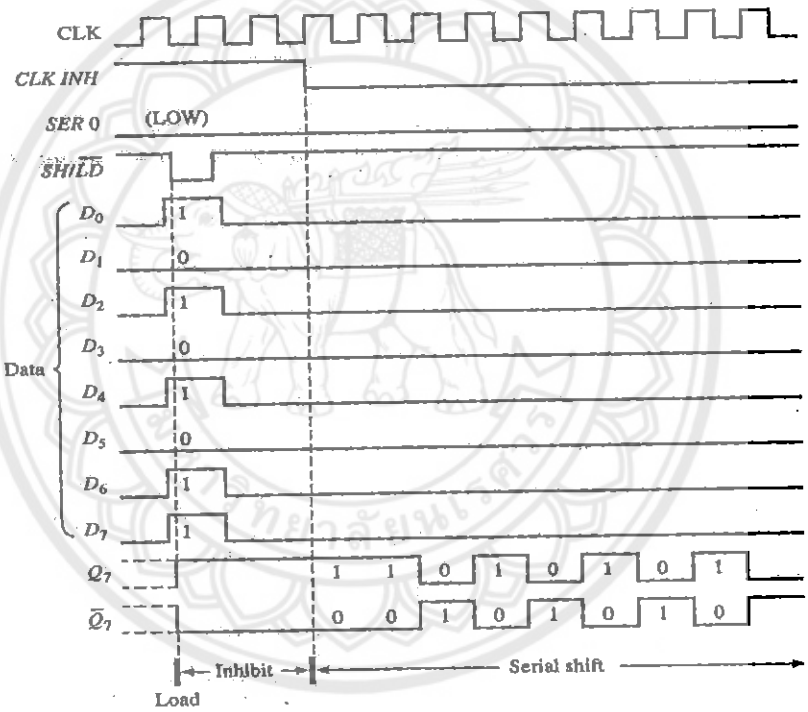
ไอซีเบอร์ 74HC165 เป็นไอซีชิฟรีจิสเตอร์แบบ PISO ขนาด 8 บิต ไอซีนี้สามารถทำงานแบบ SISO ได้เช่นเดียวกัน จากรูปที่ 12.13 (ข) แสดงโครงสร้างของวงจรที่สร้างจาก D FF จำนวน 8 ตัว เมื่อขา *SHIFT* / $\overline{LOAD}$ เป็น 0 จะทำให้ข้อมูลอินพุต D_0 ถึง D_7 ถูกโหลดเข้ามาเก็บไว้ที่รีจิสเตอร์ เมื่อข้อมูลอินพุต D_0 ถึง D_7 เป็น 1 ทำให้เอาต์พุตของแนนด์เกตแถวบนทุกตัวเป็น 0 ทำให้ฟลิปฟลอปทุกตัวทำการเซต (Asynchronous set) แต่ถ้าข้อมูลอินพุต D_0 ถึง D_7 เป็น 0 ทำให้เอาต์พุตของแนนด์เกตแถวล่างทุกตัวเป็น 0 ทำให้ฟลิปฟลอปทุกตัวทำการรีเซต (Asynchronous reset) นอกจากนี้ข้อมูลอินพุตจะป้อนเข้าแบบอนุกรมที่ขา *SER* ข้อมูลเอาต์พุตจะออกจากรีจิสเตอร์ที่ Q_7 ส่วนขา *CLK INH* ทำการควบคุมสัญญาณ clock ตลอดเวลา



(ก) สัญลักษณ์



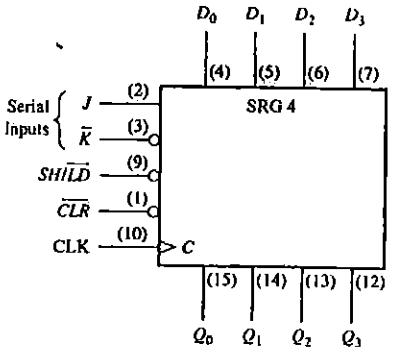
(ข) วงจร



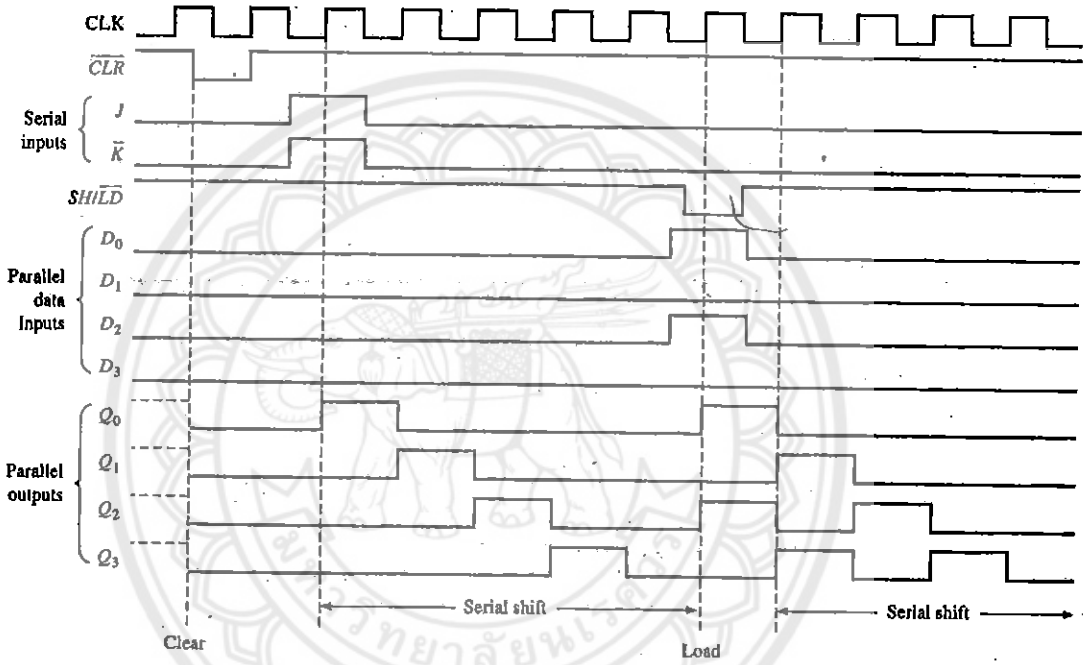
(ค) ตัวอย่างไดอะแกรมแสดงการการทำงาน

รูปที่ 12.13 ไอซีเบอร์ 74HC165 ชิฟรีจิสเตอร์แบบ PISO ขนาด 8 บิต

ไอซีเบอร์ 74HC195 เป็นไอซีชิฟรีจิสเตอร์แบบ PIPO ขนาด 4 บิต จากรูปที่ 12.14 แสดงสัญลักษณ์และตัวอย่างไดอะแกรมการทำงานของวงจร



(ก) สัญลักษณ์



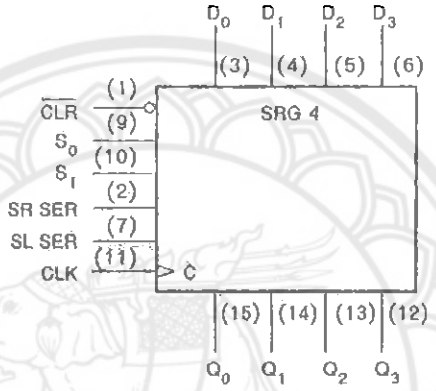
(ข) ตัวอย่างไดอะแกรมแสดงการการทำงาน

รูปที่ 12.14 ไอซีเบอร์ 74HC195 ชิฟรีจิสเตอร์แบบ PIPO ขนาด 4 บิต

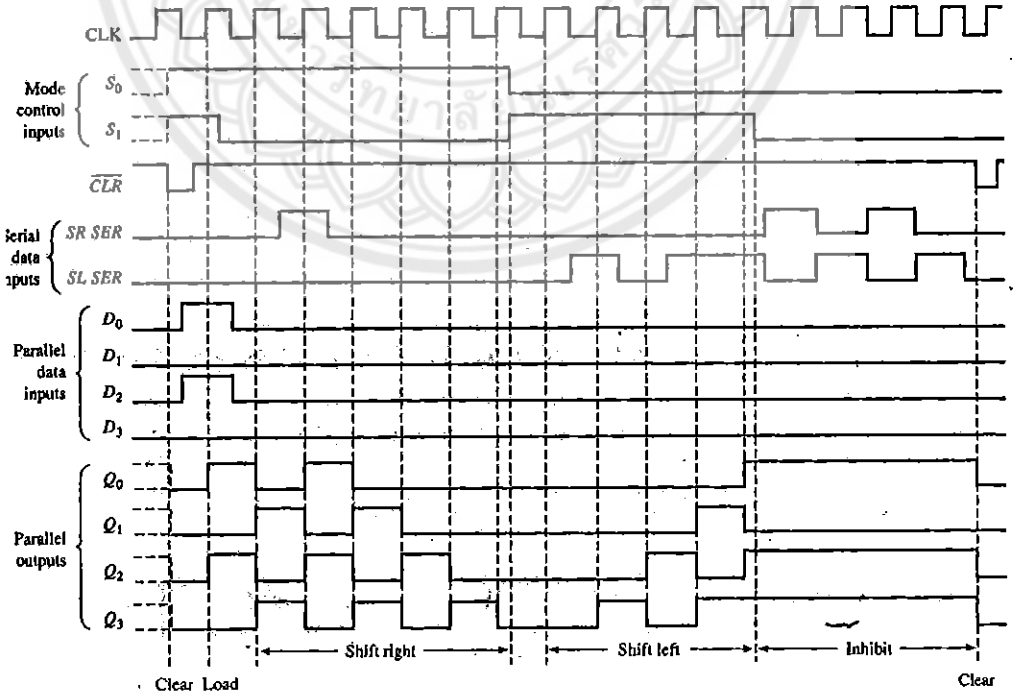
ชิฟรีจิสเตอร์นี้จะมีข้อมูลอินพุตเข้าแบบอนุกรมที่ขา J และ $\overline{K}$ ดังนั้นชิฟรีจิสเตอร์สามารถทำงานได้ทุกรูปแบบ โดยพิจารณารูปแบบของข้อมูลอินพุตที่เข้าและข้อมูลเอาต์พุตที่ออกจากรีจิสเตอร์ดังนี้การทำงานแบบ SISO จะพิจารณาข้อมูลอินพุตที่ละบิตเข้าที่ ขา J และ $\overline{K}$ และข้อมูลเอาต์พุตออกที่ละบิตที่ขา Q_3 การทำงานแบบ SIPO จะพิจารณาข้อมูลอินพุตที่ละบิตเข้าที่ ขา J และ $\overline{K}$ และข้อมูลเอาต์พุตออกพร้อมกันทุกบิตที่ขา Q_0 Q_1 Q_2 Q_3 การทำงานแบบ PISO จะพิจารณาข้อมูลอินพุตที่เข้ามาพร้อมกันทุกบิตขา D_0 D_1 D_2 D_3 และข้อมูลเอาต์พุตออกที่ละบิตที่ขา Q_3 การทำงานแบบ PIPO จะพิจารณาข้อมูลอินพุตที่เข้ามาพร้อมกันทุกบิตขา D_0 D_1 D_2 D_3 และข้อมูลเอาต์พุตออกพร้อมกันทุกบิตที่ขา Q_0 Q_1 Q_2 Q_3

การทำงานของวงจรเป็นดังนี้ ถ้าขา $SHIFT/\overline{LOAD}$ ($SH/\overline{LD}$) เป็น 0 จะทำให้ข้อมูลอินพุตโหลดเข้ามาเก็บที่รีจิสเตอร์พร้อมกัน เมื่อมีสัญญาณ clock กระตุ้นเข้ามา แต่ถ้า ขา $SH/\overline{LD}$ เป็น 1 จะทำให้ข้อมูลที่เก็บในรีจิสเตอร์ทำการเลื่อนไปทางขวาหรือเลื่อนจาก Q_0 ไปยัง Q_3 เมื่อมีสัญญาณ clock เข้ามา และข้อมูลอินพุตที่ขา J และ $\overline{K}$ จะเลื่อนเข้าเก็บที่รีจิสเตอร์ Q_0 แทนที่ข้อมูลเดิม ส่วนขา $\overline{CLR}$ จะทำการเคลียร์ข้อมูลทั้งหมดเมื่อขานี้เป็น 0

ไอซีเบอร์ 74HC194 เป็นไอซีชิฟรีจิสเตอร์ขนาด 4 บิตที่สามารถทำงานได้ทุกรูปแบบ (4 bit Bidirectional Universal Shift Register) จากรูปที่ 12.15 แสดงสัญลักษณ์และไดอะแกรมตัวอย่างการทำงานของวงจร รูปแบบการทำงานของวงจรขึ้นอยู่กับอินพุตที่ขา S_0 และ S_1 ดังนี้



(ก) สัญลักษณ์



(ข) ตัวอย่างไดอะแกรมแสดงการการทำงาน

รูปที่ 12.15 ไอซีเบอร์ 74HC194

เมื่อขา S_0=1 และ S_1=1 ข้อมูลอินพุต D_0 D_1 D_2 D_3 ทั้งสี่บิตจะถูกโหลดเข้ามาเก็บที่รีจิสเตอร์ เมื่อมีขอบขาขึ้นของสัญญาณ clock เข้ามา

เมื่อขา S_0=1 และ S_1=0 ข้อมูลที่เก็บในรีจิสเตอร์จะมีการเลื่อนไปทางขวาและข้อมูลอินพุตที่ขา SR SER จะเลื่อนเข้ามาเก็บที่รีจิสเตอร์ Q_0 เมื่อมีขอบขาขึ้นของสัญญาณ clock เข้ามา

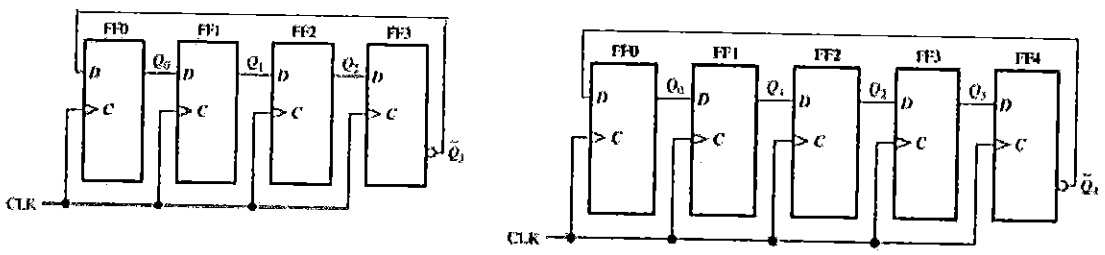
เมื่อขา S_0=0 และ S_1=1 ข้อมูลที่เก็บในรีจิสเตอร์จะมีการเลื่อนไปทางซ้ายและข้อมูลอินพุตที่ขา SL SER จะเลื่อนเข้ามาเก็บที่รีจิสเตอร์ Q_3 เมื่อมีขอบขาขึ้นของสัญญาณ clock เข้ามา

เมื่อขา S_0=0 และ S_1=0 ข้อมูลที่เก็บในรีจิสเตอร์ไม่มีการเปลี่ยนแปลง

12.7 วงจรนับด้วยชิฟรีจิสเตอร์

ชิฟรีจิสเตอร์สามารถนำมาสร้างเป็นวงจรนับได้ โดยนำเอาต์พุตบิตสูงสุดของวงจรมาป้อนกลับเข้าที่อินพุตบิตต่ำสุดและพิจารณาลำดับการนับที่เอาต์พุตของรีจิสเตอร์ทุกตัว วงจรนับที่ใช้ชิฟรีจิสเตอร์ที่สำคัญคือวงจรนับแบบจอห์นสัน (Johnson counter) และวงจรนับแบบวงแหวน (Ring counter)

วงจรนับแบบจอห์นสัน เป็นวงจรที่สร้างจากฟลิปฟลอปชนิดต่างๆ มาต่อเรียงกันและนำอินเวิร์ตเอาต์พุตของฟลิปฟลอปบิตสูงสุดมาป้อนกลับเข้าที่ขาอินพุตของฟลิปฟลอปบิตต่ำสุด จากรูปที่ 12.16 เป็นวงจรนับจอห์นสันที่สร้างจาก D FF จะเห็นว่าเอาต์พุต Q ของฟลิปฟลอปจะต่อกับขาอินพุตของฟลิปฟลอปบิตถัดไปและเอาต์พุต $\overline{Q}$ ของฟลิปฟลอปบิตสุดท้ายจะนำมาต่อเข้าที่ขาอินพุตของฟลิปฟลอปบิตแรก เมื่อพิจารณาลำดับการนับ สถานะของการนับ N มีค่าเท่ากับ 2n เมื่อ n คือจำนวนบิตหรือจำนวนฟลิปฟลอปที่ใช้ ดังนั้นถ้าเป็นวงจรนับแบบจอห์นสันขนาด 4 บิต สถานะของการนับ N มีค่าเท่ากับ 8 และวงจรนับแบบจอห์นสันขนาด 5 บิต สถานะของการนับ N มีค่าเท่ากับ 10 ลำดับของการนับทั้งสองวงจรแสดงดังตารางที่ 12.1 จะเห็นว่าวงจรจะเลื่อนบิต 1 จากซ้ายไปขวาจนเต็ม จากนั้นจะเลื่อนบิต 0 เข้ามาแทนที่จากซ้ายไปขวาจนเต็ม การนับจะสลับกันเมื่อมีสัญญาณ clock เข้ามา สำหรับไดอะแกรมลำดับการนับของวงจร แสดงดังรูปที่ 12.17



(ก) วงจรนับแบบจอห์นสันขนาด 4 บิต (ข) วงจรนับแบบจอห์นสันขนาด 5 บิต

รูปที่ 12.16 วงจรนับแบบจอห์นสัน

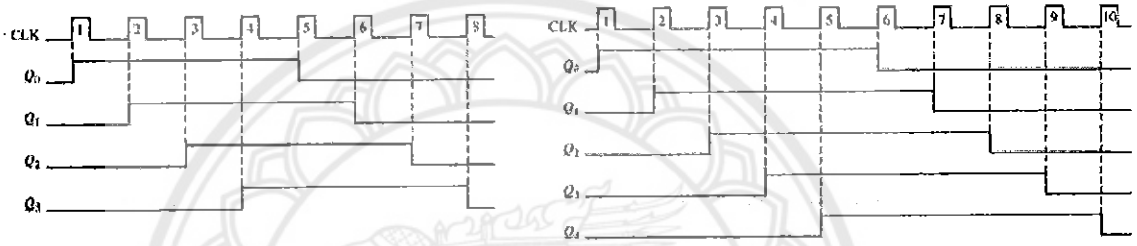
ตารางที่ 12.1 ลำดับการนับของวงจรนับแบบจอห์นสัน

CLOCK	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

(ก) การนับแบบจอห์นสันขนาด 4 บิต

CLOCK	Q_0	Q_1	Q_2	Q_3	Q_4
0	0	0	0	0	0
1	1	0	0	0	0
2	1	1	0	0	0
3	1	1	1	0	0
4	1	1	1	1	0
5	1	1	1	1	1
6	0	1	1	1	1
7	0	0	1	1	1
8	0	0	0	1	1
9	0	0	0	0	1

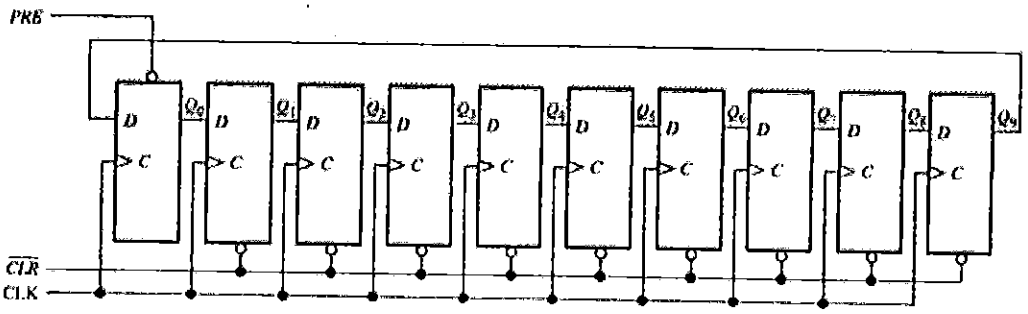
(ข) การนับแบบจอห์นสันขนาด 5 บิต



(ก) วงจรนับจอห์นสันขนาด 4 บิต (ก) วงจรนับจอห์นสันขนาด 5 บิต

รูปที่ 12.17 ไดอะแกรมเวลาของการนับแบบจอห์นสัน

วงจรนับแบบวงแหวน เป็นวงจรที่สร้างจากฟลิปฟลอปชนิดต่างๆ มาต่อเรียงกันและนำเอาต์พุต Q ของฟลิปฟลอปบิตสูงสุดมาป้อนกลับเข้าที่ขาอินพุตของฟลิปฟลอปบิตต่ำสุด จากรูปที่ 12.18 เป็นวงจรนับวงแหวนขนาด 10 บิตที่สร้างจาก D FF จำนวน 10 ตัว จะเห็นว่าโครงสร้างวงจรเหมือนกับวงจรนับจอห์นสัน แต่ต่างกันที่วงจรนับวงแหวนจะนำเอาต์พุต Q ของฟลิปฟลอปบิตสุดท้ายจะนำมาต่อเข้าที่ขาอินพุตของฟลิปฟลอปบิตแรก พิจารณาสถานะของการนับ N มีค่าเท่ากับ n เมื่อ n คือจำนวนบิต ดังนั้นถ้าเป็นวงจรนับวงแหวนขนาด 10 บิต สถานะของการนับ N มีค่าเท่ากับ 10 แสดงลำดับการนับได้ดังตารางที่ 12.2 การทำงานของวงจรเริ่มจากทำการเคลียร์ฟลิปฟลอปทุกตัว ลำดับการนับพิจารณาได้จากเอาต์พุต Q ของฟลิปฟลอปทุกตัว ถ้าให้ขา $\overline{PRE}$ ของ D FF ตัวแรกเป็น 0 ฟลิปฟลอปจะทำการเซต ทำให้เอาต์พุตบิตแรก Q_0 = 1 จากนั้นเมื่อสัญญาณ clock กระตุ้นเข้ามาแต่ละครั้ง ทำให้ข้อมูลบิต 1 เลื่อนจาก Q_0 ไปยัง Q_1 และข้อมูลจาก Q_1 จะเลื่อนไปยัง Q_2 เป็นเช่นนี้ตลอดทั้งวงจร ดังนั้นจะเห็นว่าข้อมูลบิต 1 มีการเลื่อนภายในวงจรจากซ้ายไปขวาและวนกลับมาที่เดิม การวนแบบนี้จะเหมือนกับวงแหวนนั่นเอง

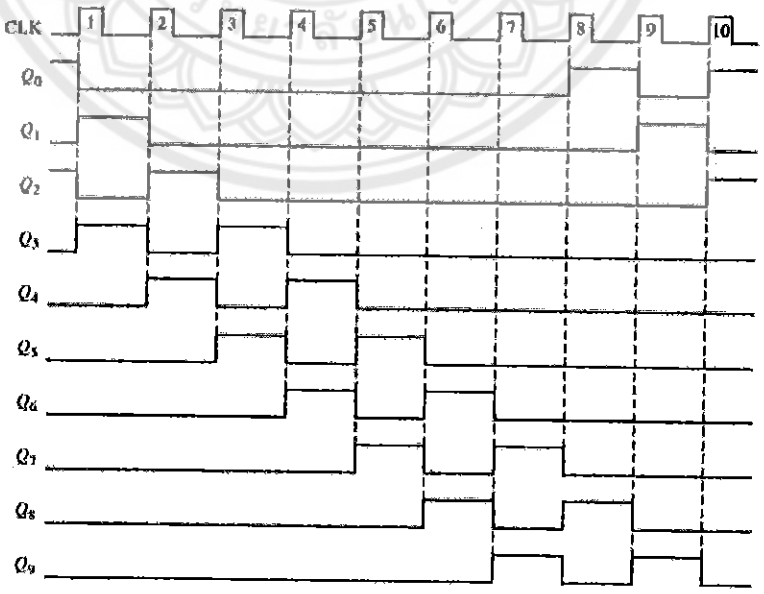


รูปที่ 12.18 วงจรนับแบบวงแหวนขนาด 10 บิต

ตารางที่ 12.2 ลำดับการนับของวงจรนับแบบวงแหวน

CLOCK PULSE	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7	Q_8	Q_9
0	1	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	0	1

ตัวอย่างไดอะแกรมเวลาแสดงลำดับการนับของวงจรแสดงดังรูปที่ 12.19 เมื่อกำหนดให้ข้อมูลเริ่มต้นเป็น 1010000000 สามารถแสดงเอาต์พุต Q ของฟลิปฟลอปทุกตัวได้ดังรูป



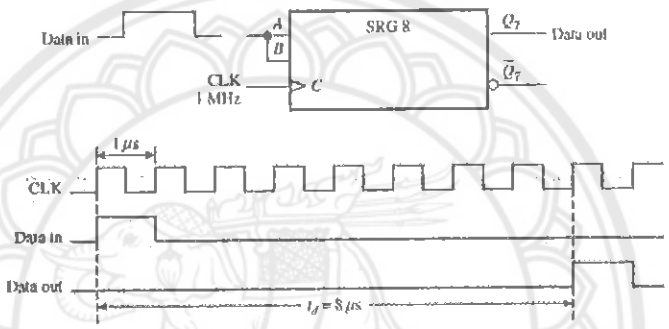
รูปที่ 12.19 ตัวอย่างไดอะแกรมเวลาของการนับแบบวงแหวน

12.8 การประยุกต์ใช้งานชิฟรีจิสเตอร์

ชิฟรีจิสเตอร์สามารถนำไปประยุกต์ใช้งานได้มากมาย เช่น การหน่วงเวลาข้อมูล (Time delay) การแปลงข้อมูลจากอนุกรมเป็นขนาน (Serial to Parallel data converter) การบวกเลขฐานสองแบบอนุกรม (Serial adder) การรับส่งข้อมูลระหว่างรีจิสเตอร์แบบอนุกรมหรือแบบขนาน เป็นต้น

การหน่วงเวลาข้อมูล

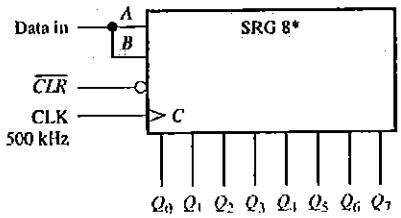
ชิฟรีจิสเตอร์แบบ SISO สามารถนำมาใช้ในการหน่วงเวลาของข้อมูลจากอินพุตไปยังเอาต์พุตได้ เวลาที่ใช้ในการหน่วงข้อมูลขึ้นอยู่กับจำนวนบิตของข้อมูลและความถี่ของสัญญาณ clock ที่เข้ามา



รูปที่ 12.20 การหน่วงเวลาข้อมูลด้วยชิฟรีจิสเตอร์

จากรูปที่ 12.20 เป็นชิฟรีจิสเตอร์แบบ SISO ขนาด 8 บิต ข้อมูลอินพุตแบบอนุกรมจะส่งเข้ามาในวงจรที่ขา A และ B เมื่อสัญญาณ clock เข้ามา 1 ลูกทำให้ข้อมูลเลื่อน 1 ครั้ง ถ้าสัญญาณ clock เข้ามาครบจำนวน 8 ลูก จะได้ข้อมูลออกทางเอาต์พุต Q_7 ถ้าสมมุติสัญญาณ clock มีความถี่เท่ากับ 1 MHz ดังนั้นสัญญาณ clock มีคาบเวลาเท่ากับ 1 μs ดังนั้นจากวงจรทำให้มีเวลาหน่วง (t_d) เท่ากับ 8 μs

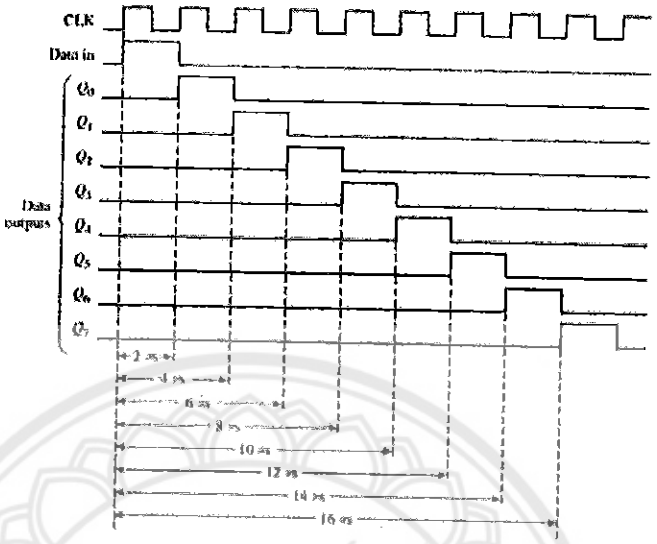
ตัวอย่างที่ 12.5 จากชิฟรีจิสเตอร์แบบ SIPO ขนาด 8 บิตดังรูป ถ้ากำหนดให้สัญญาณ clock มีความถี่เท่ากับ 500 kHz จงหาค่าเวลาหน่วงของข้อมูลเอาต์พุตทุกบิต



* Data shifts from Q_0 toward Q_7.

<u>วิธีทำ</u>

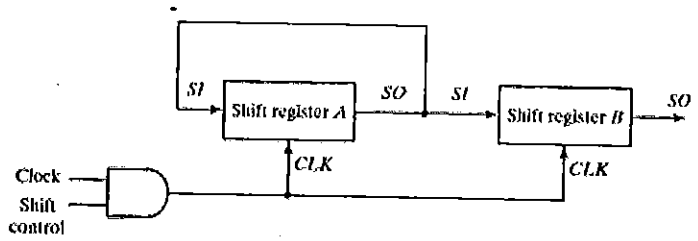
สัญญาณ clock มีคาบเวลาเท่ากับ 2 μs ดังนั้นเวลาที่ใช้ในการหน่วงข้อมูลแต่ละบิตจะมีค่าตั้งแต่ 2 μs จนถึง 16 μs



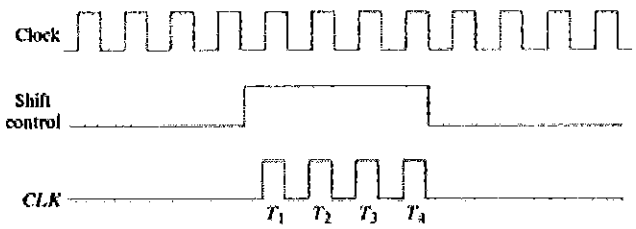
การส่งข้อมูลระหว่างรีจิสเตอร์แบบอนุกรม (Serial transfer)

การส่งข้อมูลแบบอนุกรมระหว่างรีจิสเตอร์ A และ B แสดงดังรูปที่ 12.21 (ก) จะเห็นว่าเอาต์พุตของรีจิสเตอร์ A (SO) จะต่อกับขาอินพุตของรีจิสเตอร์ B และรีจิสเตอร์ B จะส่งข้อมูลออกที่เอาต์พุตแบบอนุกรม ข้อมูลนี้อาจส่งไปยังรีจิสเตอร์ตัวอื่นได้ นอกจากนี้รีจิสเตอร์ A ยังนำเอาต์พุตต่อกับขาอินพุตของตัวเองด้วย เพื่อป้องกันไม่ให้ข้อมูลในรีจิสเตอร์หายไป สำหรับขา shift control ต่อกับแอนด์เกต เพื่อทำหน้าที่ควบคุมการเลื่อนข้อมูลของรีจิสเตอร์และสัญญาณ clock

ถ้ากำหนดให้ชิฟรีจิสเตอร์ A และ B มีข้อมูล 4 บิต จากรูปที่ 12.21 (ข) ขา shift control=1 จะกำหนดให้สัญญาณ clock เข้ามา 4 ลูกคือ T_1 ถึง T_4 ที่ขอบขาขึ้นของสัญญาณ clock เหล่านี้เป็นตัวกำหนดให้รีจิสเตอร์ A และ B ทำการเลื่อนข้อมูล (enable) แต่ถ้าขา shift control=0 รีจิสเตอร์ A และ B ไม่มีการเลื่อนข้อมูล (disabled)



(ก) บล็อกไดอะแกรม



(ข) ไดอะแกรมเวลา

Timing Pulse	Shift Register *A*	Shift Register *B*
Initial value	1 0 1 1	0 0 1 0
After T_1	1 1 0 1	1 0 0 1
After T_2	1 1 1 0	1 1 0 0
After T_3	0 1 1 1	0 1 1 0
After T_4	1 0 1 1	1 0 1 1

(ค) ตัวอย่างการส่งข้อมูลแบบอนุกรม

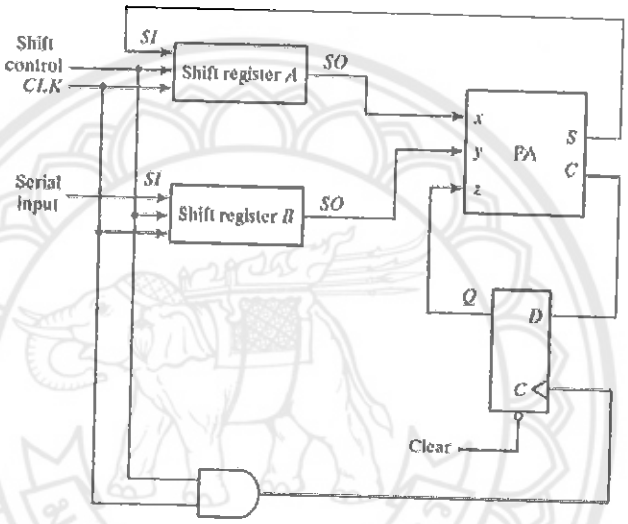
รูปที่ 12.21 การส่งข้อมูลแบบอนุกรมระหว่างรีจิสเตอร์ A และ B

การทำงานของวงจรเป็นดังนี้ สมมุติเริ่มต้นให้รีจิสเตอร์ A เก็บข้อมูล 1011 และรีจิสเตอร์ B เก็บข้อมูล 0010 ตามลำดับ การส่งข้อมูลแบบอนุกรมจากรีจิสเตอร์ A ไปยัง B จะเกิดขึ้นตามลำดับที่สัญญาณ clock ตั้งแต่ T_1 ถึง T_4 เข้ามาดังตารางที่ 12.21 (ค) เมื่อสัญญาณ T_1 เข้ามา ข้อมูลบิตทางขวาสุดของรีจิสเตอร์ A จะเลื่อนไปยังด้านซ้ายสุดของรีจิสเตอร์ B และเลื่อนไปยังด้านซ้ายสุดของตัวมันเองด้วย ในขณะเดียวกันข้อมูลทุกบิตของรีจิสเตอร์ A และ B จะเลื่อนไปทางขวา 1 ตำแหน่ง ดังนั้นบิตข้อมูลที่เอาต์พุตของรีจิสเตอร์ B จะเปลี่ยนจาก 0 เป็น 1 จนกระทั่งหลังสัญญาณ T_4 เข้ามาทำให้ข้อมูลในรีจิสเตอร์ A และ B เป็น 1011 จะเห็นว่าข้อมูลรีจิสเตอร์ A ถูกส่งไปยังรีจิสเตอร์ B แต่ข้อมูลที่เก็บใน รีจิสเตอร์ A ยังคงเหมือนเดิมไม่เปลี่ยนแปลง

การบวกเลขฐานสองแบบอนุกรม

ในการบวกเลขฐานสองที่มีจำนวนมากกว่า 1 บิต ถ้าใช้วงจรบวกแบบขนาน ข้อมูลทุกบิตจะป้อนเข้าที่วงจร FA (Full adder) และทำการบวกพร้อมกัน ดังนั้นเมื่อต้องการบวกข้อมูลจำนวน n บิตต้องมีวงจร FA จำนวน n ชุด แต่ถ้าเป็นวงจรบวกแบบอนุกรม ข้อมูลจะป้อนเข้าที่วงจร FA ทีละ 1 บิต ดังนั้นจะใช้วงจร FA จำนวน 1 ชุดเท่านั้น จะเห็นว่าวงจรบวกแบบขนานทำงานได้เร็วกว่าแบบอนุกรม แต่วงจรจะมีขนาดใหญ่กว่า ดังนั้นในกรณีที่มีการบวกเลขจำนวนหลายบิตและไม่คำนึงถึงเวลาที่ใช้ในการทำงานมากนักและต้องการให้วงจรมีขนาดเล็ก สามารถใช้วงจรบวกแบบอนุกรมได้

วงจรบวกแบบอนุกรม ประกอบด้วยวงจร FA จำนวน 1 ชุดและชิฟรีจิสเตอร์แบบ PISO จำนวน 2 ชุด แต่ละชุดมีขนาดเท่ากับข้อมูลที่นำมาบวกกัน จากรูปที่ 12.22 รีจิสเตอร์ A จะเก็บข้อมูลของตัวตั้ง (augends) และรีจิสเตอร์ B จะเก็บข้อมูลของตัวบวก (addend) ส่วน D FF ทำหน้าที่เก็บตัวทด (carry out) ของวงจร FA เพื่อนำไปใช้บวกกับบิตที่สูงกว่าของข้อมูลคู่ถัดไป ผลบวกที่ได้ (sum) ของวงจร FA สามารถนำไปเก็บไว้ที่ชิฟรีจิสเตอร์อีกชุดหนึ่งได้ แต่จากวงจรดังรูปที่ 12.22 จะเห็นว่าผลบวกที่ได้ (sum) ของวงจร FA ถูกนำไปเก็บไว้ที่ชิฟรีจิสเตอร์ A ทำให้ชิฟรีจิสเตอร์ A เก็บข้อมูลตัวตั้งและผลบวกไว้ ส่วนรีจิสเตอร์ B สามารถรับข้อมูลชุดใหม่เข้ามาเก็บไว้เพื่อทำการบวกต่อไปได้



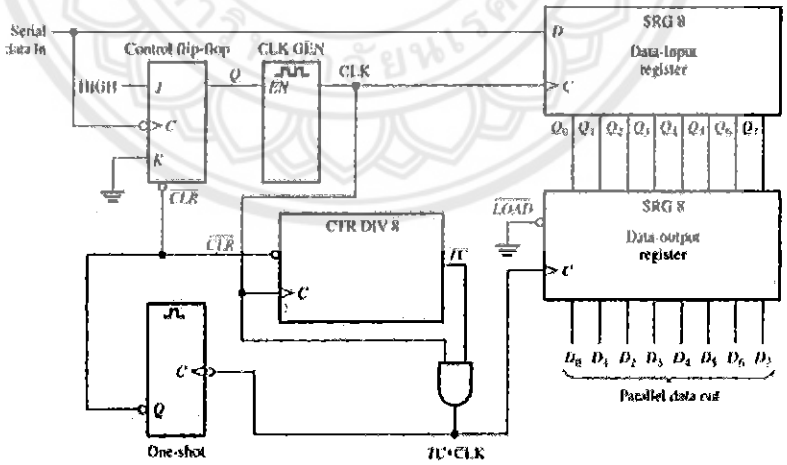
รูปที่ 12.22 วงจรบวกแบบอนุกรม

การทำงานของวงจรบวกแบบอนุกรมมีดังนี้ เริ่มต้นทำการโหลดข้อมูลเก็บในรีจิสเตอร์ โดยรีจิสเตอร์ A จะเก็บข้อมูลของตัวตั้ง (augends) และรีจิสเตอร์ B จะเก็บข้อมูลของตัวบวก (addend) ส่วน D FF ทำการเคลียร์ข้อมูลให้เอาต์พุต Q เป็น 0 ในตอนนี้เอาต์พุต SO ของรีจิสเตอร์ A และ B จะเก็บข้อมูลบิตต่ำสุด ข้อมูลนี้จะถูกส่งไปยังขา x และ y ของวงจร FA เพื่อทำการบวก ผลบวก (S) ที่ได้จะนำไปเก็บไว้ที่รีจิสเตอร์ A และตัวทด (C) จะนำไปเก็บไว้ที่ D FF เมื่อขา shift control ทำงานจะทำให้มีการเลื่อนข้อมูลในรีจิสเตอร์และ D FF ดังนั้นเมื่อมีขอบขาขึ้นของสัญญาณ clock เข้ามา ทำให้ข้อมูลในรีจิสเตอร์ A และ B เลื่อนไปทางขวา 1 บิต ที่รีจิสเตอร์ A จะนำผลบวกที่ได้ของบิตก่อนหน้านี้มาเก็บไว้ทางซ้ายสุดของรีจิสเตอร์ A ที่ D FF นำตัวทด (C) ของบิตก่อนหน้านี้ไปป้อนที่ขา z ของวงจร FA เมื่อสัญญาณ clock เข้ามาเท่ากับจำนวนบิตของรีจิสเตอร์ A หรือ B แล้ว การบวกเลขชุดแรกจะเสร็จสมบูรณ์ ผลบวกที่ได้จะเก็บไว้ที่รีจิสเตอร์ A

และบิตตัวทดสุดท้ายของการบวกจะเก็บไว้ที่ D FF ถ้าขา shift control ยังทำงานอยู่ การบวกยังคงดำเนินต่อไป จนกว่าขานี้จะไม่ทำงาน วงจรจะหยุดทำการบวก

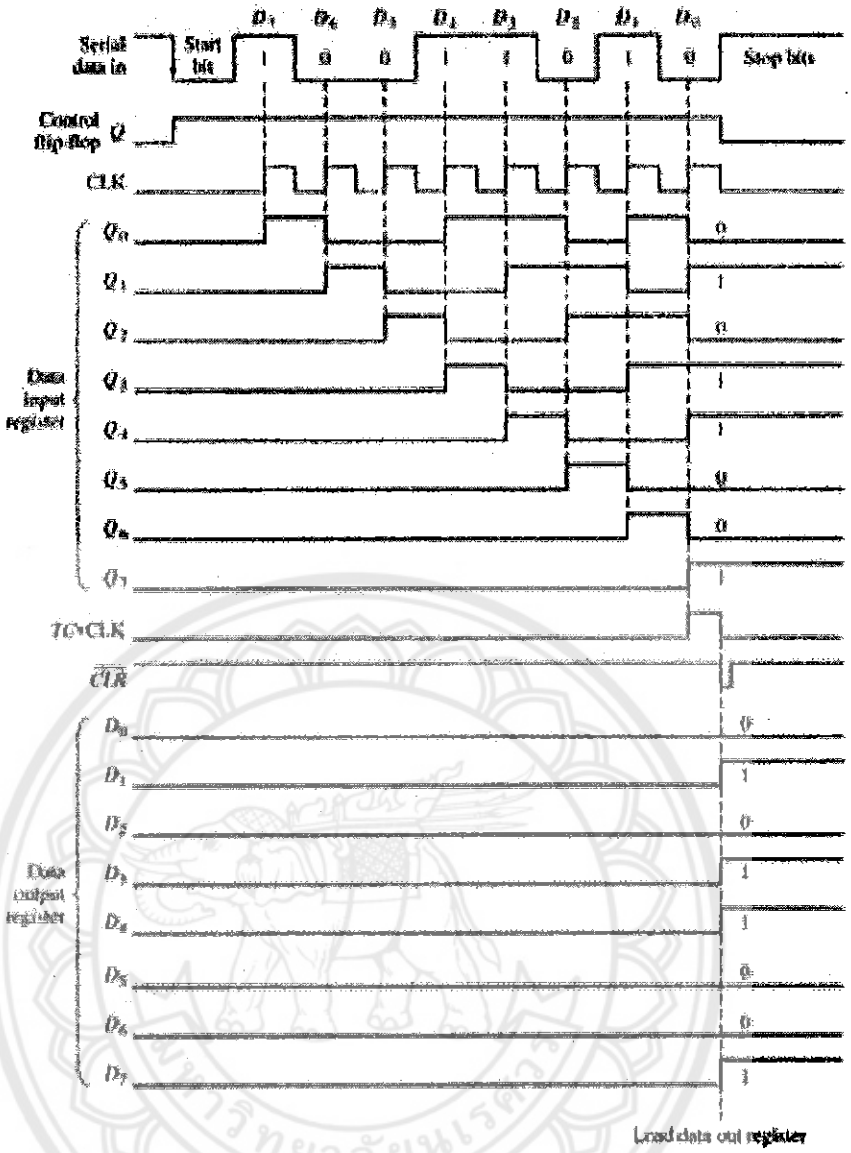
การแปลงข้อมูลจากอนุกรมเป็นขนาน

ในระบบดิจิตอล มีการส่งข้อมูลแบบอนุกรมระหว่างระบบด้วยกันเพื่อประหยัดสายส่งสัญญาณ โดยเฉพาะสัญญาณไกล ๆ แม้แต่ในคอมพิวเตอร์หรือระบบไมโครโปรเซสเซอร์ที่ต้องการข้อมูลแบบขนาน เพื่อนำข้อมูลที่หลายบิตมาประมวลผลพร้อมกัน จึงจำเป็นต้องมีการแปลงรูปแบบข้อมูล จากรูปที่ 12.23 แสดงวงจรการแปลงข้อมูลจากอนุกรมเป็นขนานอย่างง่าย โดยใช้ชิฟรีจิสเตอร์แบบ SIPO และ PIPO ขนาด 8 บิต การทำงานของวงจรจะเริ่มจากการนำข้อมูลอินพุตดังรูปที่ 12.24 ข้อมูลอินพุตจะมีบิตที่บอกการเริ่มต้นของข้อมูล (start bit) และบิตที่บอกการสิ้นสุดของข้อมูล (stop bit) ดังนั้นข้อมูลอินพุตมีจำนวนบิตทั้งหมดเท่ากับ 11 บิต ถ้าบิตข้อมูลมีการเปลี่ยนแปลงจาก 1 เป็น 0 แสดงว่ามีบิตเริ่มต้นเข้ามาในวงจร ทำให้ JK FF ถูกเซต สัญญาณเซตที่ได้จะไปควบคุมวงจรกำเนิดสัญญาณ clock (Clock generator) ทำงาน สัญญาณ clock จะทำหน้าที่ควบคุมการทำงานของชิฟรีจิสเตอร์แบบ SIPO และวงจรหาร 8 เมื่อสัญญาณ clock เข้ามาครบ 8 ลูก ข้อมูลอินพุตทั้งหมดจะถูกเก็บไว้ที่ชิฟรีจิสเตอร์และที่ขา TC ของวงจรหาร 8 จะมีค่าเป็น 1 ทำให้ชิฟรีจิสเตอร์แบบ PIPO ส่งข้อมูลแบบขนานจำนวน 8 บิตออกมา และสัญญาณ TC ของวงจรหาร 8 นี้จะกระตุ้นการทำงานของฟลิปฟลอปเพื่อให้วงจรกำเนิดสัญญาณ clock หยุดทำงานด้วย ตัวอย่างไดอะแกรมแสดงการส่งข้อมูลอินพุตแบบอนุกรม (D_7-D_0) โดยส่งบิต D_7 เข้าไปก่อน



รูปที่ 12.23 วงจรการแปลงข้อมูลจากอนุกรมเป็นขนาน

รูปที่ 12.24 รูปแบบของข้อมูลอินพุตแบบอนุกรม



รูปที่ 12.25 ไดอะแกรมเวลาแสดงการแปลงข้อมูลจากอนุกรมเป็นขนาน

บรรณานุกรม

1. Thomas L.Floyd, "*Digital Fundamentals*", 7th Edition, Prentice Hall International, 2000.
2. Ronals J.Tocci, Neal S. Widmer, and Gregory L. Moss, "*Digital Systems Principle and Applications*", 9th Edition, Pearson Prentice Hall, 2004.
3. James W. Stewart and Chao-Ying Wanf, "*Digital Electronics laboratory*", 2nd Edition, Pearson Prentice Hall, 2004.
4. Donald D. Givone, "*Digital Principles and Design*", McGraw-HILL Higher Education, 2003.
5. M.Morris Mano, "*Digital Design*", 3rd Edition, Prentice Hall, Pearson Education International, 2002.
6. James Bignell and Robert Donovan, "*Digital Electronics*", 4th Edition, Delmar, Thomson Learning, 2000.
7. Richard Monk, "*Digital Electronics: A Practical Approach with EASY PC and PULSAR*", Newnes, An imprint of Butterworth-Heinemann, 1998.
8. Roger L.Tokheim, M.S., Digital, "*Schaum's Outline of Theory and Problems of Digital Principles*", 3rd Edition, McGraw-HILL, 1994.
9. Daniel D. Gajski, "*Principle of Digital Design*", Prentice Hall, 1997.
10. Parag K. Lala, "*Practical Digital Logic Design and Testing*", Prentice Hall, Engwood Cliffs, New Jercy, 1996.
11. Miles J. Murdocca and Vincent P. Heuring, "*Principles of Computer Architecture*", Prentice Hall International, 2000.