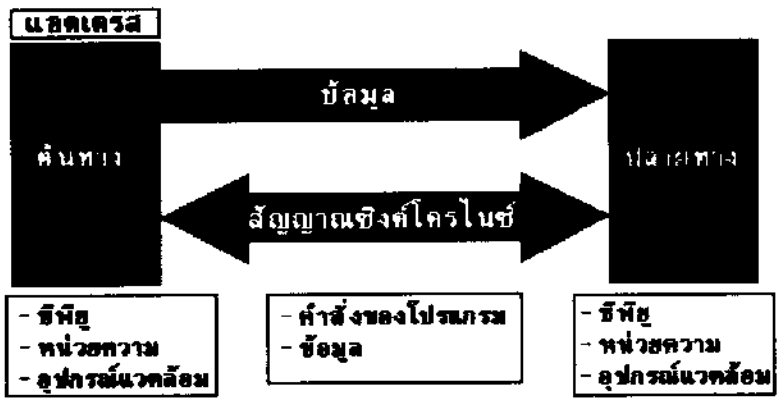
บทที่ 2
หลักการและทฤษฎี

2.1 การติดต่อกับไมโครคอมพิวเตอร์

การอินเตอร์เฟสกับคอมพิวเตอร์หรือไมโครโปรเซสเซอร์

การอินเตอร์เฟสกับคอมพิวเตอร์หรือไมโครโปรเซสเซอร์ คือการทำงานติดต่อร่วมกันระหว่างซีพียูกับอุปกรณ์อื่น ๆ กับการโอนถ่ายข้อมูลระหว่างอุปกรณ์ต่าง ๆ นอกเหนือจากจะต้องทำงานติดต่อกับ RAM, ROM แล้วยังต้องมีการติดต่อกับอุปกรณ์ภายนอกที่มีการส่งข้อมูลอินพุท, เอาท์พุทอีกทางหนึ่ง ซึ่งเป็นการเพิ่มประสิทธิภาพให้ระบบสมบูรณ์ ในระบบต่างของอุปกรณ์อิเล็กทรอนิกส์ จะทำงานต่อเนื่องเป็นลูกโซ่ ดังเช่น การส่งรับข้อมูลจากซีพียูไปยังส่วนอื่น ๆ เป็นต้น



รูปที่ 2.1 การติดต่อกับไมโครคอมพิวเตอร์

การที่จะโอนย้ายข้อมูลทุกตัวนั้นจะต้องมีแหล่งที่ส่งข้อมูล และแหล่งที่รับข้อมูลสำหรับขบวนการเหล่านั้น จะมีส่วนที่สำคัญว่า ข้อมูลนั้นเป็น แอดเดรสหรือว่าเป็นดาต้า จะส่งไปยังจุดไหน ตัวอย่างเช่น ส่งไปยังหน่วยความจำ หรืออุปกรณ์อินพุต/เอาท์พุต และจะส่งเมื่อไร การทำงานเหล่านี้โดยทั่ว ๆ ไป จะต้องมีสัญญาณ ในการตรวจสอบอุปกรณ์ว่าพร้อมที่จะส่ง/รับข้อมูล หรือยังก่อนเสมอ เนื่องจากจุดที่ส่งและรับ ข้อมูล จะต้องมีสัญญาณตรวจสอบความพร้อมเสมอเพื่อที่จะให้ข้อมูลที่เราใช้งานนั้น ๆ เป็นระเบียบ ตัวอย่างเช่น ส่งข้อมูลจากซีพียูไปที่อุปกรณ์รอบข้าง เป็นต้น ซึ่งจุดรับส่งคู่หนึ่ง ๆ อาจจะเป็นระหว่างซีพียูด้วยกัน หรือ ซีพียูกับหน่วยความจำ หรือ ซีพียูกับอุปกรณ์รอบข้าง หรือ ระหว่างอุปกรณ์รอบข้างด้วยกัน หรือ ระหว่างหน่วยความจำกับอุปกรณ์รอบข้าง สำหรับข้อมูลที่โอนย้ายไปมานั้นจะอยู่ในลักษณะของเลขฐานสอง ตัวอย่างเช่น --> 01101100

ซึ่งเลขแต่ละตัวจะแทนด้วย 1 bit อาจเป็น 8 bit หรือ 16 bit ก็ขึ้นอยู่กับของระบบนั้นๆ ถ้าหากเป็นการต่อจากพอร์ตพีซีไม่ว่าจะเป็น Serial หรือ Parallel ในสัญญาณที่ส่งมาจะมีระบบแรงดันไฟฟ้า

-Serail port (RS-232) --> ~+-3 ถึง +-25 V_{dc}

-Parallel port (Printer port) -->~5 V_{dc}(TTL) ต่อ 1 bit

จะเห็นได้ว่าระดับสัญญาณแรงดันไฟฟ้าที่ยกมาให้ดูนี้ เราสามารถที่จะควบคุมและนำมาใช้กับอุปกรณ์รอบข้างหรืออุปกรณ์ภายนอกได้ ดังจะยกตัวอย่าง เช่น Parallel (Printer port) ระดับแรงดันไฟฟ้า ~5 Vdc

สามารถนำ มาใช้ในการขับรีเลย์, ทรานซิเตอร์, หลอดไฟ~5 V_{dc} หรือ LED ให้ทำงานได้ โดยการเขียนโปรแกรมคอมพิวเตอร์ไปควบคุมที่พอร์ตเครื่องพิมพ์ (Printer) เป็นต้น

ดังนั้นการที่จะนำพีซีมาประยุกต์ใช้งานให้เกิดประสิทธิผลกับชีวิตประจำวันนั้น เป็นไปได้หลายวิธี อีกทั้งฮาร์ดแวร์ต่าง ๆ ของพีซีที่มีอยู่กับเครื่องสามารถใช้ ให้เกิดประโยชน์ได้ในหลาย ๆ ด้าน เพราะฉะนั้นผมจะกล่าวสิ่งที่ควรจะต้องรู้ต่อไปของการใช้พีซีติดต่ออุปกรณ์ (PC Interface Hardware) ก็คือการเชื่อมต่ออุปกรณ์กับพอร์ตต่อพ่วงของพีซีชนิดต่าง ๆ และระบบของการติดต่อสื่อสารของพีซี ดังมีรายละเอียดต่อไปนี้

การเชื่อมติดต่ออุปกรณ์เข้ากับคอมพิวเตอร์

2.1.1 คอนเน็คเตอร์ (Connecter) ที่อยู่ภายนอก ส่วนใหญ่จะอยู่ข้างหลังเครื่องคอมพิวเตอร์

- พอร์ตต่อคีย์บอร์ด หรือ อาจเรียกกันว่า PS/2, mini-DIN
- พอร์ตต่อเมาส์ หรือ อาจเรียกกันว่า PS/2, mini-DIN
- พอร์ตต่อจอภาพ
- พอร์ตต่ออนุกรม อาจเรียก Serial Port, Com port (Com1, Com2) ใช้ในระบบติดต่อสื่อสาร RS-232
- พอร์ตต่อขนาน อาจเรียก Parallel Port, Printer port (LPT1, LPT2) ส่วนใหญ่จะใช้พ่วงต่อกับเครื่องพิมพ์ (Printer)
- พอร์ตต่อจอยสติก ส่วนมากที่เห็นจะอยู่ที่เซาว์การ์ดเป็นส่วนใหญ่
- พอร์ตต่อโมเด็ม ตัวคอนเน็กเตอร์จะเป็นประเภทเดียวกับสายสัญญาณโทรศัพท์
- พอร์ตUSB (Universal Serial Bus) เป็นพอร์ตรุ่นใหม่ที่สามารถพ่วงอุปกรณ์ได้มากเริ่มว่างตลาดเช่น เม้าส์, คีย์บอร์ด, โมเด็ม, กล้องดิจิตอล เป็นต้น
- พอร์ตเชื่อมต่อระบบเครือข่าย จะมากับการ์ดแลนด์ เรียก พอร์ต RJ-45
- พอร์ต SCSI (Small Computer System Interface) มักใช้เชื่อมต่อกับอุปกรณ์ที่ต้องการความเร็วสูง อาจจะได้ยินมาจากชนิดของฮาร์ดดิส

2.1.2 คอนเน็คเตอร์ (Connecter) ที่อยู่ภายนอก ส่วนใหญ่จะอยู่ภายในเครื่องคอมพิวเตอร์

- EIDE (Enhanced Intergrated Drive Electronics) สายเชื่อมต่อกับฮาร์ดดิส
- SCSI (Small Computer System Interface) โดยมากจะมากับการ์ดที่เป็นแบบสกัซซี
- ฟลอปปี้ไดรฟ์ คอมพิวเตอร์ทุกเครื่องจะมีไว้ต่อฟลอปปี้ไดรฟ์
- คอนเน็กเตอร์อนุกรม มี 10 เข็มอยู่ที่แผงวงจรเมนบอร์ด
- คอนเน็กเตอร์ขนาน มี 26 เข็มอยู่ที่แผงวงจรเมนบอร์ด ถ้าต้องการขยาย พอร์ตขนาน ก็สามารถนำ Card I/O 8255 มาเสียบเข้าไปใน Slot ซึ่งเป็นประเภท Card PCI สำหรับ IC I/O 8255 จะมีพอร์ตเพิ่มขึ้นมาได้ 3 พอร์ตต่อ 1 ชุด IC 8255

ระบบที่ใช้ติดต่อสื่อสารข้อมูลของคอมพิวเตอร์

-USB (Universal Serial Port) รวมถึง Firewire (IEEE-1348) เป็นระบบใช้ติดต่อสื่อสารข้อมูลแบบใหม่ที่มีความเร็วสูง อีกทั้งเฉลียวฉลาดมากขึ้น ระหว่างเครื่องคอมพิวเตอร์ หรือคอมพิวเตอร์กับอุปกรณ์ภายนอก (Hardware) ซึ่ง USB ได้ถูกนำเข้ามาแทน การติดต่อแบบ RS-232 และ Centronics Printer Ports ดังจะเห็นได้จากอุปกรณ์โมเด็มหรืออุปกรณ์ตัวอื่น ๆ เป็นต้น
-->Firewire มันได้ถูกออกแบบเพื่อรองรับการสื่อสารสำหรับข้อมูลที่เป็น สัญญาณภาพ, เสียง, วีดีโอ รวมถึง ขนาดบล็อคของที่มีขนาดใหญ่

-Microwire, SPI, I^2C Interface การติดต่อสื่อสารเป็นแบบ Synchronous Serial เหมาะสำหรับใช้ในระยะสั้น ๆ ซึ่ง Microcontroller ส่วนใหญ่แล้วจะติดต่อแบบนี้

-Ethernet ใช้ติดต่อสื่อสารในระบบเครือข่ายหรือที่เรียก ระบบแลนด์ ที่มีเครื่องคอมพิวเตอร์ต่อกันหลายเครื่อง เรียกว่าระบบที่มีความเร็วสูงและความจุแต่ละอุปกรณ์ (Hardware) และโปรแกรม (Software) ซึ่งจะมีความซับซ้อน อีกทั้งราคาสูงกว่าระบบการติดต่อสื่อสารแบบอื่น ๆ ในที่นำมากล่าวทั้งหมดนี้

-Centronics Parallel Printer Port Interface สามารถส่งข้อมูลได้หลายบิตสำหรับการส่งหนึ่งครั้ง ซึ่งมีความเร็วสูงสุด มักจะนิยมใช้สำหรับการติดต่อสื่อสาร ระหว่างพีซีกับเครื่องพิมพ์ (Printer), เครื่องสแกนเนอร์, เครื่องเก็บข้อมูลแบบภายนอก (Data Acquisition Devices) เป็นต้น

-IrDA (Interface Data Association) เป็นการสื่อสารข้อมูลแบบไร้สาย โดยใช้แสงอินฟราเรด ซึ่งใช้ได้ในระยะทางสั้น ๆ ในที่สายเคเบิลไม่สามารถติดตั้งได้/เข้าไปไม่ถึงที่เห็นในชีวิตประจำวัน เช่น รีโมททีวี หรือวีดีโอ, เมาส์หรือคีย์บอร์ดอินฟราเรด เป็นต้น

-MIDI (Musical Instrument Digital Interface) ใช้สำหรับการสื่อสารแบบอนาล็อกในเครื่องมือด้านเครื่องเสียง, เครื่องมือด้านดนตรี (ซินติไซเซอร์/เปอร์คัดชั่น/กีตาร์แอฟเฟ็ก), เครื่องมือควบคุมเสียงในโรง ภาพยนตร์ (มิกเซอร์/อีเควอไรเซอร์/แอฟเฟ็กต่าง ๆ) ซึ่งมันจะใช้กระแสไฟประมาณ 5 mA ที่ความเร็ว 31.5 kbps

ตารางที่ 2.1 รายละเอียดของระบบ PC Interface

ระบบ การติดต่อสื่อสาร	รูปแบบ	จำนวนเครื่องสูงสุด	ระยะทางสูงสุด(เมตร)	ความเร็วสูงสุด (บิตต่อวินาที)
Parallel Printer Port	Parallel	2,8	3-9	1M
RS-232(EIA/TIA-232)	Asynchronous serial	2	15-30	20k(ถ้ามีไดเวอร์จะUp 115k)
RS-422(EIA/TIA-422)	Asynchronous serial	2	1220	10M
RS-485(EIA/TIA-485)	Asynchronous serial	32หน่วยของโหลด	1220	10M
USB	Asynchronous serial	127	4.8	12M
Firewire	Serial	64	4.5	400M
Microwire	Synchronous Serial	8	3	2M
Ethernet	Serial	1024	487	10M
MIDI	Serial current loop	2	4.5	31.5k
IrDA	Asynchronous Serial Infrared	2	1.8	115k
I^2C	Synchronous Serial	40	5.5	400k
SPI	Synchronous Serial	8	3	2.1M
IEEE-488(GPIB)	Parallel	15	18.3	1M

2.2 ทรัพยากรระบบคอมพิวเตอร์

IRQ อุปกรณ์แต่ละชนิดจะมี IRQ เป็นของตัวเอง

ตารางที่ 2.2 IRQ Setting Table

No. IRQ	Hardware
0	Computer Timer
1	Keyboard
3	Serial Port 2 (COM 2)
4	Serial Port 1 (COM 1)
5	Sound Card
6	Floppy Drive
7	Parallel Port
8	Clock
12	Mouse Port
14	Primary Hard Drive
15	Secondary Hard Drive

DMA(Direct Memory Access) การย้ายข้อมูลระหว่างอุปกรณ์กับคอมพิวเตอร์

ตารางที่ 2.3 DMA Channel Setting Tabel

No. DMA Channel	Hardware
1	Sound Card
2	Floppy Drive
3	Parallel Port or Voice Modem
5	Sound Card or SCSI Card
6	Sound Card or Network Interface Card

I/O Address อุปกรณ์จะต้องติดต่อกับ CPU ของคอมพิวเตอร์โดยมีตำแหน่งเป็นของตนเอง

ตารางที่ 2.4 I/O Address Setting Table

No. I/O Address	Hardware
220	Sound Card
2F8	Floppy Drive
300	Network Interface Card
378	Parallel Port
3F8	Serial Port

Memory Address การเคลื่อนย้ายข้อมูลต้องการใช้หน่วยความจำส่วนหนึ่งเก็บชั่วคราว

ตารางที่ 2.5 Memory Address Setting Table

No. Memory Address	Hardware
C0000	Video Card
C8000	Hard Drive
D0000	Network Interface Card
F0000	BIOS

สำหรับไว้ตั้งค่าของอินเทอร์รัพเวกเตอร์ของ com port

ตารางที่ 2.6 Interrupt Vector Table

No. Com Port	Interrupt Vector
Com1	0*0C
Com2	0*0B
Com3	0*0C
Com4	0*0B

โปรแกรมเมเบิลอินเทอร์รัพคอนโทรลเลอร์ เขียนโปรแกรมติดต่อ Com port

ตารางที่ 2.7 Programmable Interrupt Controller Table

Com Port	Programmable Interrupt Controller
Com1(IRQ4)	0*EF
Com2(IRQ3)	0*F7
Com1(IRQ4)	0*EF
Com2(IRQ3)	0*F7

ตาราง ตั้งค่า ยกเลิกการใช้อินเทอร์รัพรีเครส ในการเขียนโปรแกรมติดต่อ com port

ตารางที่ 2.8 Mask IRQ Table

Com Port	Mask IRQ Value
Com1(IRQ4)	0*10
Com2(IRQ3)	0*08
Com1(IRQ4)	0*10
Com2(IRQ3)	0*08

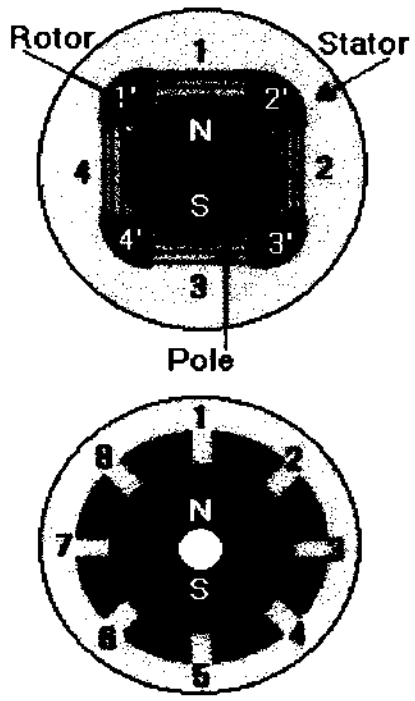
2.3 Stepper motor

ความรู้เบื้องต้น และ หลักการทำงาน Step Motor

รูปที่ 2.2 Stepper Motor

Stepper Motor เป็นมอเตอร์ที่มีลักษณะเมื่อเราป้อนไฟฟ้าให้กับมอเตอร์ทำให้หมุนเพียงเล็กน้อยตามเส้นรอบวงและหยุด ซึ่งต่าง จากมอเตอร์ ทั่วไปที่จะหมุนทันทีและตลอดเวลาเมื่อป้อนแรงดันไฟฟ้าข้อดีของ สเต็ปมอเตอร์ สามารถกำหนด ตำแหน่งของการหมุนด้วยตัวเลข(องศาหรือระยะทาง) ได้อย่างละเอียดโดย ใช้คอมพิวเตอร์หรือ ไมโครคอนโทรลเลอร์เป็น เครื่องกำหนดและจัดเก็บตัวเลข

โครงสร้างของขั้วแม่เหล็กบนสเตเตอร์ ทำมาจากแผ่นเหล็กวงแหวนที่มีขั้วๆ ยื่นออกมาประกอบกันเป็นชั้นๆ โดยที่แต่ละขั้วนั้นจะมีคอยล์ (ขดลวด) พันสวมอยู่ เมื่อมีการป้อนกระแสผ่านคอยล์ทำให้เกิดสนามแม่เหล็กไฟฟ้า (Electromagnetic) ดูดังรูปด้านล่างนี้ จะแสดงถึงองค์ประกอบที่กล่าวมา

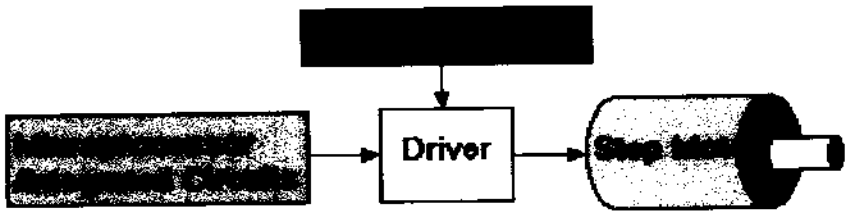


รูปที่ 2.3 โครงสร้างของขั้วแม่เหล็กบนสเตเตอร์

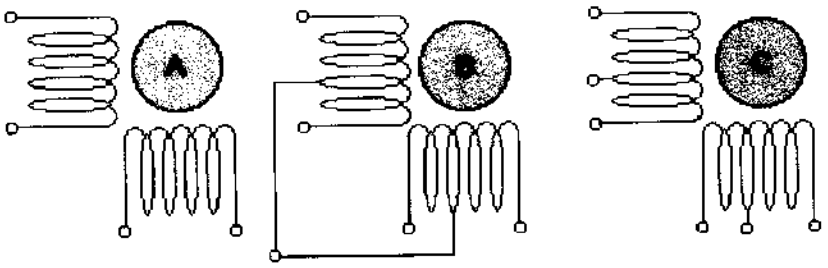
ในที่นี้ซึ่งถ้าเพิ่มจำนวนของขั้วแม่เหล็กมากขึ้นจะเพิ่มจำนวนของสเต็ปต่อวงจรรอบมากขึ้นตามด้วย ลองดูตามรูปด้านบน

ลักษณะการนำไปใช้งาน สเต็ปเปอร์มอเตอร์ ใช้งานลักษณะ Open Loop System แปลเป็นภาษาไทย ระบบเปิด คือ สเต็ปเปอร์มอเตอร์สามาทำงานได้โดยไม่ต้องมีการ ป้อนค่าพารามิเตอร์กลับมา (Feed back) แต่ทุกวิธีที่ต้องการกำหนดตำแหน่งที่แน่นอนนั้น จะต้องการป้อนกลับไปยังระบบและตัวบอก ตำแหน่งว่าถูกต้องหรือผิดพลาดให้รับทราบ

ดังเช่นวิธีที่ใช้กับสเต็ปเปอร์มอเตอร์ คือนำลิมิทสวิตช์ติดตามตำแหน่งที่จะตรวจจับ เมื่อสเต็ปเปอร์มอเตอร์ เริ่มหมุนแล้วหมุนไปจนถึงตำแหน่งของสวิตช์ตรวจจับสัญญาณ สวิตช์ทำงานก็จะป้อนกลับไปสู่ระบบทำให้รู้การทำงานของสเต็ปเปอร์มอเตอร์ตลอด วงจรไมโครคอนโทรลเลอร์เองจะมีจุดอ้างอิง ไว้ให้เริ่มต้นการทำงานและอ้างอิงตำแหน่งได้ถูกต้อง

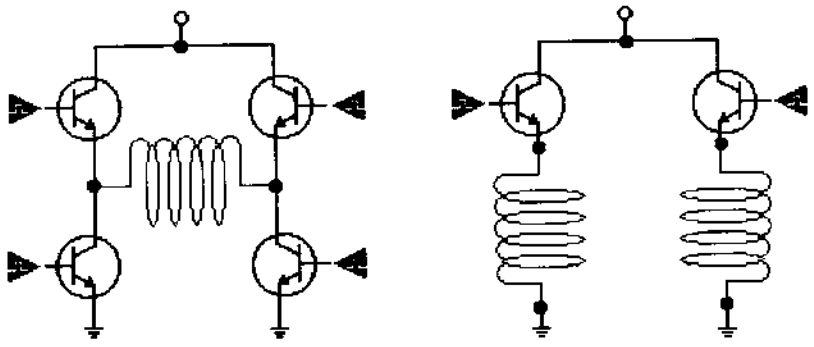


การควบคุมระบบสเต็ปมอเตอร์



A) แบบไบโพล่าร์ B)แบบยูนิโพลาร์ 5 สาย C)แบบยูนิโพลาร์แบบ 6 สาย

การพันขดลวดบนสเตเตอร์ของสเต็ปมอเตอร์



A) แบบไบโพลาร์ B) แบบยูนิโพลาร์

▶ คือ ต่อเข้ากับแหล่งจ่ายไฟหรือจากพอร์ตพีซีเพื่อทริกการนิสเตอร์ให้ทำงาน

วงจรการจ่ายไฟให้กับสเต็ปมอเตอร์

รูปที่ 2.4 การควบคุมระบบสเต็ปเปอร์มอเตอร์, การพันขดลวดบนสเตเตอร์, วงจรการจ่ายไฟ

โดยแนวทางสเต็ปเปอร์มอเตอร์เป็นอุปกรณ์จำพวกเชิงกลทางไฟฟ้า โดยมีกรุ๊ปของไบนารี่ โวลท์เดทเป็นอินพุตและการเคลื่อนที่ แบบเชิงมุมเป็นเอาท์พุต หรือว่าหมุนทีละสเต็ปซึ่งอยู่ระหว่าง 0.1 - 30 องศา อยู่ที่โครงสร้างของสเต็ปเปอร์มอเตอร์ โดยตามสัญญาณ พัลซ์ที่จ่ายให้กับขดสเตเตอร์ทำให้เกิดแรงผลักแก่โรเตอร์หมุนไป สเต็ปเปอร์มอเตอร์มีขดลวดหลายชุดในที่นี้เราเรียกว่า เฟส (Phase) ดังนั้นสัญญาณที่ต่อเนื่องเป็น ซีเควน (Sequence) ลักษณะของ ไบนารี่ (Binary) ซึ่งจะต้องไปผ่านวงจร ไดรเวอร์ (Driver) ก็จะทำให้โรเตอร์หมุนไปอย่างต่อเนื่อง ที่กล่าวมาสามารถดูได้จากรูปด้านบนชื่อ การควบคุมสเต็ปเปอร์มอเตอร์

ให้ดูที่รูปชื่อการพันขดลวดบนสเตเตอร์ของสเต็ปเปอร์มอเตอร์ จะเห็นว่าการพันมีด้วยกัน 2 วิธี คือ แบบไบโพล่าร์ (Bipolar) กับแบบยูนิโพล่าร์ (Unipolar)

แบบไบโพล่าร์ (Bipolar)

จะมีการพันขดลวดหนึ่งขด (จำนวนรอบของขดลวดขึ้นอยู่กับลักษณะการใช้งาน) ในแต่ขั้วแม่เหล็กของสเตเตอร์ โดยขั้วแม่เหล็กที่เกิดขึ้นที่สเตเตอร์จะถูกกำหนดโดยทิศทางของการไหลของกระแสไฟฟ้า ซึ่งสามารถทำให้เกิดขั้วแม่เหล็กในทิศทางตรงกันข้ามได้เพียงการกลับทิศทางของการไหลในกระแสไฟฟ้า โดยมาจากการควบคุมของวงจรสวิทชิ่งให้กลับขั้วไฟฟ้า

แบบยูนิโพล่าร์ (Unipolar)

แบบนี้มี 2 ขด บนแต่ละขั้วแม่เหล็กของสเตเตอร์ ทำให้แต่ละขดลวดเกิดขั้วแม่เหล็กในทิศทางตรงกันข้าม เช่นกันการกลับทิศทางขั้วแม่เหล็กทำได้โดยใช้วงจรสวิทชิ่งให้สลับขั้วหนึ่งไปยังอีกขั้วหนึ่งแทนกัน

พื้นฐานการสวิทชิ่งดูรูปด้านบนที่ชื่อวงจรการจ่ายไฟให้กับสเต็ปมอเตอร์ การพันขดลวดทั้ง 2 แบบที่กล่าวมาต่างกัน คือ แบบยูนิโพล่าร์จะทำให้เกิดแรงบิดน้อยกว่าแบบไบโพล่าร์ โดยสังเกตุจากสายไฟที่ต่อมาจากตัวสเต็ปมอเตอร์ซึ่งแบบไบโพลาร์จะมี 4 สาย ส่วนเป็นแบบยูนิโพล่าร์จะมี 5 สายหรือ 6 สาย

การสั่งงานควบคุมการหมุนของสเต็ปมอเตอร์

การควบคุมและสั่งงานให้สเต็ปเปอร์มอเตอร์ทำงานไปทีละสเต็ป สามารถทำได้โดยการจ่ายกำลังไฟไปยังขดลวดในแต่ละขดบนสเตเตอร์ โดยการป้อนจะทำในลักษณะเป็นลำดับหรือเรียกว่า ซีเควนเชียลในลูปที่ถูกต้อง ซึ่งจะแบบได้เป็น 3 รูปแบบ คือ แบบเวฟ (wave) แบบ 2 เฟส (2 phase) และแบบครึ่งสเต็ป (Half step) ซึ่งทั้ง 3 แบบนี้มีข้อดีและข้อเสียต่างกันออกไป

แบบเวฟ (wave)

จะเป็นการกระตุ้นแบบที่ง่ายที่สุด ซึ่งจะทำการกระตุ้นขดลวดทีละขดในเวลาหนึ่ง ๆ เรียงกันไป ตัวอย่างเช่น ขดที่ 1, 2, 3, 4, 1, 2, 3, 4 เป็นลำดับเช่นนี้ หรือ ขด 1, 4, 3, 2, 1, 4, 3, 2 เป็นลำดับกันไป ทั้งนี้ขึ้นอยู่กับทิศทางที่ต้องการให้มอเตอร์หมุนไป วงจรที่นำมากระตุ้นนั้นจะมีราคาค่อนข้างถูกกว่าและง่ายกว่า ดังในรูปของวงจรการจ่ายไฟ ในรูปที่ 2.4
เราสามารถเขียนขั้นตอนการทำงานเป็นตารางออกมาได้ดังนี้

ตารางที่ 2.9 การหมุนมอเตอร์แบบเวฟ

Step No.	Phase I	Phase 2	Phase 3	Phase 4
1	ON			
2		ON		
3			ON	
4				ON
5	ON			
6		ON		

<u>แบบ 2 เฟส (2 Phase)</u>

แบบนี้คล้ายกับการกระตุ้นในแบบเวฟแต่สิ่งที่ต่างกันคือ แบบ 2 เฟส จะกระตุ้นทีละ 2 ขด ที่อยู่ใกล้กันในเวลาเดียวกัน และจะเรียงลำดับกันไปเช่นเดียวกับแบบเวฟ ตัวอย่างการกระตุ้นขดลวดในลักษณะซีเควนให้เป็นดังนี้ 12, 23, 34, 41, 12, 23, 34, 41 เรียงลำดับกันไปเรื่อย ๆ หรือจะเป็น 14, 43, 32, 21, 14, 43, 32, 21 เรียงกันไปเรื่อย ๆ เช่นกัน ข้อดีข้อเสียของแบบ 2 เฟส แล้วมีดังนี้

ข้อดี การเพิ่มจำนวนขดลวดที่ถูกกระตุ้น ทำให้แรงบิดเกิดได้มากกว่าในแบบเวฟ ซึ่งทำให้โรเตอร์หมุนด้วยแรง ดึงแบบเต็มแรงจาก ทั้ง 2 ขดลวดที่กระตุ้นพร้อมกัน

ข้อเสีย สำหรับแบบ2เฟส การกระตุ้นขดลวดต้องใช้กำลังไฟมากขึ้นเป็น 2 เท่าของแบบเวฟ

เราสามารถเขียนลำดับการกระตุ้นของขดลวดแบบ 2 เฟส ได้ดังในตารางต่อไปนี้

ตารางที่ 2.10 การหมุนมอเตอร์แบบ 2 เฟส

Step No.	Phase 1	Phase 2	Phase 3	Phase 4
1	ON	ON		
2		ON	ON	
3			ON	ON
4	ON			ON
5	ON	ON		
6		ON	ON	

แบบครึ่งสเต็ป

แบบนี้แบบรูปแบบผสมผสานของการกระตุ้นระหว่างแบบเวฟ และแบบ 2 เฟส เพื่อให้จำนวนรอบของสเต็ปให้มากขึ้นเป็น 2 เท่า ซึ่งในระบบนี้จะทำการกระตุ้นขดลวดเรียงกันไปเรื่อย ๆ เป็นลำดับ ดังต่อไปนี้ 1, 12, 2, 23, 3, 34, 4, 41, 1, 12, 2, 23, 3, 34, 4, 41, 1 เป็นลำดับอยู่อย่างนี้ ถ้ากลับทิศทางการหมุนจะได้ดังนี้ 1, 41, 4, 43, 3, 32, 2, 21, 1, 41, 4, 43, 3, 32, 2, 21, 1 เป็นลำดับกันไป ข้อดีและข้อเสียของการกระตุ้นแบบครึ่งสเต็ป มีดังต่อไปนี้

ข้อดี การกระตุ้นแบบนี้จะให้แรงบิดที่เพิ่มมากขึ้น เนื่องจากช่วงสเต็ปที่มีระยะสั้นลงอีกประการหนึ่ง แต่ละสเต็ปเกิดแรงดึงจากขดลวด 2 ขดที่ถูกกระตุ้นพร้อมกัน เป็นผลให้ค่าตำแหน่งมีความถูกต้องสูงขึ้น

ข้อเสีย เช่นเดียวกับแบบ 2 เฟส คือต้องจ่ายกำลังไฟเป็น 2 เท่าของแบบเวฟ หรือจะใช้เท่ากับแบบ 2 เฟส

ดังนั้นเราสามารถนำลำดับการทำงานของ แบบครึ่งเฟส มาแสดงในรูปของตารางได้ดังนี้

ตารางที่ 2.11 การหมุนมอเตอร์แบบครึ่งสเต็ป

Step No.	Phase 1	Phase 2	Phase 3	Phase 4
1	ON			
2	ON	ON		
3		ON		
4		ON	ON	
5			ON	
6			ON	ON
7				ON
8	ON			ON
9	ON			
10	ON	ON		

ที่กล่าวมาข้างต้นเป็นพื้นฐานความรู้เกี่ยวกับสเต็ปเปอร์มอเตอร์ เพื่อที่จะนำให้สามารถนำไปประกอบกับการใช้ทำโครงงานต่าง ๆ ในงานอินเตอร์เฟส โดยการเขียนโปรแกรมไปควบคุมสเต็ปเปอร์มอเตอร์ได้ต่อไป

หลักการคำนวณหาค่าพารามิเตอร์ของสเต็ปเปอร์มอเตอร์

1. Tooth Pitch

P_s = Lenght of stator tooth pitch

P_r = Lenght of rotor tooth pitch

$P_s = 360/ N_s$; N_s = number of stator tooth.

$P_r = 360/ N_r$; N_r = number of rotor tooth.

2. Step Angle

"s = Pr/NP =360/NrNp ; Number of Phase.

= (1*Pr-Ps*1)x2 degree/step

3. Stepping Rate

Rs = 360/"s =NrNp step/round

4. Speed of step motor(W)

W = 60f/Rs = 60f/NpNr ="sf/6 (rpm)

5. Number of stator poles per phase

X =Ns/Np = Rs/Np(Np+-1) =Nr/Np+-1

จากลอจิกที่แสดงมาแล้วนั้น เราสามารถจัดหาอุปกรณ์จำพวก ไอซี (IC) มาประกอบกันจัดทำชุดไดรเวอร์ได้โดยไม่ยาก เช่น IC 74HC04 เป็นตัว inverter, IC 74HC32 เป็นตัว OR Gate, IC 74HC00 เป็นตัว NAND Gate เป็นต้น

2.4 การ Interface กับ Serial Port

Hardware

Hardware Properties

อุปกรณ์สื่อสารจะจำแนกออกเป็น 2 กลุ่มคือ

-อุปกรณ์ Data Communication Equipment (DCE)

-อุปกรณ์ Data Terminal Equipment (DTE)

DEC คืออุปกรณ์ที่ทำหน้าที่สื่อสารเช่น โมเด็ม (Modem) หรือ โทรศัพท์ หรือ แฟ็กซ์ (Fax) เป็นต้น ส่วน DTE คืออุปกรณ์ ที่ต่อกับ DCE เพื่อใช้รับและ/หรือส่งสาร ซึ่งก็คือเครื่อง Computer หรือ Terminal โดยใช้ UART เป็นตัว Interface นั่นเอง

คุณสมบัติทางไฟฟ้าของ Serial Port ตามมาตรฐาน RS-232C ของ Electronic Industry Association (EIA) พอสรุปได้ดังนี้

Logic "0" หรือ "Space" มีค่า +3Volt ถึง +25Volt

Logic "1" หรือ "Mark" มีค่า -3Volt ถึง -25Volt

ช่วง +3Volt ถึง –3Volt เป็นช่วง Undefined

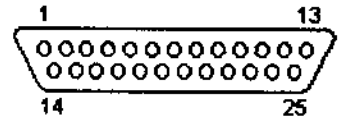
Open circuit voltage เมื่อเทียบกับ GND ต้องไม่เกิน 25Volt

Short circuit current ต้องไม่เกิน 500mA ซึ่ง Driver ต้องสามารถรองรับได้

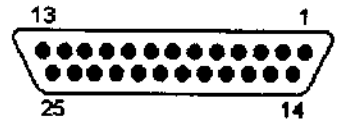
RS-232C กำหนด Baud rate ไว้ไม่เกิน 20K Baud ปัจจุบันได้แก้ไขให้รองรับกับ Technology ใหม่ได้ จึงมีการปรับปรุงถึง RS-232E

Serial Pinouts (DB25 & DB9)

Serial Port มีขั้วต่อ 2 แบบคือ แบบ D-Type 25 Pin และแบบ D-Type 9 Pin ซึ่งทั้ง 2 แบบ จะเป็นชนิดตัวผู้ทางด้านของ Computer ดังนั้นอุปกรณ์ที่จะนำมาต่อกับ Computer จึงต้องใช้ขั้วต่อชนิดตัวเมีย



(Male at the computer side)



(Female at the cable side)

รูปที่ 2.5 Serial Port แบบ D-Type 25 Pin

ตารางที่ 2.12 แสดงชื่อสัญญาณของขาต่างๆ

25 PIN D-SUB MALE at the computer.

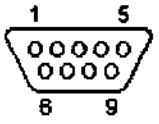
Pin	Name	Dir	Description
1	SHIELD	-	Shield Ground
2	TXD	→	Transmit Data
3	RXD	←	Receive Data
4	RTS	→	Request to Send
5	CTS	←	Clear to Send
6	DSR	←	Data Set Ready
7	GND	-	System Ground
8	CD	←	Carrier Detect
9	n/c	-	

ตารางที่ 2.12 (ต่อ) แสดงชื่อสัญญาณของขาต่างๆ

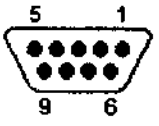
Pin	Name	Dir	Description
10	n/c	-	
11	n/c	-	
12	n/c	-	
13	n/c	-	
14	n/c	-	
15	n/c	-	
16	n/c	-	
17	n/c	-	
18	n/c	-	
19	n/c	-	
20	DTR	→	Data Terminal Ready
21	n/c	-	
22	RI	←	Ring Indicator
23	n/c	-	
24	n/c	-	
25	n/c	-	

Note: Direction is DTE (Computer) relative DCE (Modem).

Note: Do not connect SHIELD(1) to GND(7).



(Male at the computer side)



(Female at the cable side)

รูปที่ 2.6 Serial Port แบบ D-Type 9 Pin

ตารางที่ 2.13 9 PIN D-SUB MALE at the Computer.

Pin	Name	RS232	V.24	Dir	Description
1	CD	CF	109	←	Carrier Detect
2	RXD	BB	104	←	Receive Data
3	TXD	BA	103	→	Transmit Data
4	DTR	CD	108.2	→	Data Terminal Ready
5	GND	AB	102	-	System Ground
6	DSR	CC	107	←	Data Set Ready
7	RTS	CA	105	→	Request to Send
8	CTS	CB	106	←	Clear to Send
9	RI	CE	125	←	Ring Indicator

Note: Direction is DTE (Computer) relative DCE (Modem).

Note: RS232 column is RS232 circuit name.

Note: V.24 column is ITU-TSS V.24 circuit name.

ตารางที่ 2.14 D-Type 25 and D-Type 9 Connector

Pin No.(D-Type 25)	Pin No. (D-Type 9)	Abbreviation	Full Name
2	3	TD	Transmit Data
3	2	RD	Receive Data
4	7	RTS	Ready To Send
5	8	CTS	Clear To Send
6	6	DSR	Data Set Ready
7	5	SG	Signal Ground
8	1	(D)CD	(Data) Carrier Detect
20	4	DTR	Data Terminal Ready
22	9	RI	Ring Indicator

Pin Functions

ตาราง 2.15 แสดงถึงหน้าที่ของขาต่าง ๆ

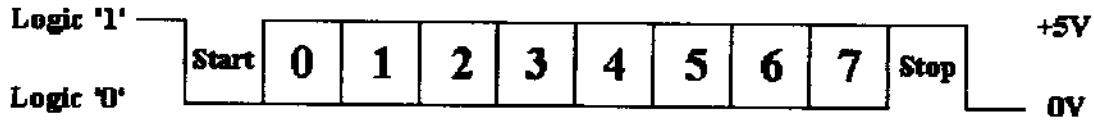
Abbreviation	Full Name	Originator	Function
TD	Transmit Data	DTE	Serial data output (TXD) from DTE.
RD	Receive Data	DCE	Serial data input (RXD) to DTE.
CTS	Clear To Send	DCE	Tell DTE that DCE is ready to exchange data.
(D)CD	(Data) Carrier Detect	DCE	Carrier from remote DCE is detected.
DSR	Data Set Ready	DCE	Tell DTE that DCE is ready to establish a link.
DTR	Data Terminal Ready	DTE	Tell DCE that DTE is ready to establish a link.
RTS	Ready To Send	DTE	Tell DCE that DTE is ready to exchange data.
RI	Ring Indicator	DCE	Ringing signal from the phone line is detected.

External Hardware – Interfacing Methods

RS-232 Waveforms

การสื่อสารโดย RS-232 เป็นการสื่อสารแบบ asynchronous หมายความว่าสัญญาณ clock ที่ใช้ควบคุมจังหวะไม่ได้ส่งไปพร้อมกับ Data แต่จะใช้ start bit เป็นตัว sync.ในแต่ละ word ของการสื่อสารและใช้สัญญาณ clock ภายในของแต่ละด้านเป็นตัวให้จังหวะเอง

Waveform.



รูปที่ 2.7 TTL/CMOS Serial Logic

แสดงลักษณะของสัญญาณจาก UART เมื่อใช้ format แบบ 8N1 คือ 8 data bits ไม่มี parity bit และมี 1 stop bit ขณะที่ idle จะอยู่ในสถานะ "Mark" หรือ logic "1" การส่งจะเริ่มจากการส่ง start bit คือ logic "0" และตามด้วย LSB bit จนหมด data bits และถ้ามี parity bit ก็จะส่งที่จุดนี้แล้วลงท้ายด้วย stop bit ซึ่งมีค่าเป็น logic "1" ในรูปได้แสดง bit ที่ต่อถัดจาก stop bit ซึ่งมีค่าเป็น logic "0" หมายความว่าเป็น start bit ของ การส่ง word ถัดไป แต่ถ้ายังไม่มีการส่ง word ถัดไป ก็ต้องอยู่ในสถานะของ logic "1" ซึ่งเป็นสถานะของ idle และถ้าสายอยู่ในสถานะของ logic "0" นานกว่า

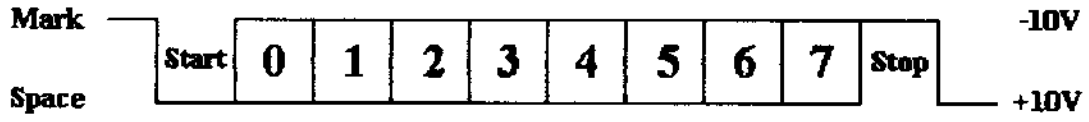
เวลาของการส่ง 1 full word ระบบจะถือว่าเป็นสัญญาณ "Break" เพื่อหยุดการสื่อสาร ดังนั้นต้องไม่ลืมที่จะสั่งในสายกลับสู่สถานะ idle เมื่อสิ้นสุดการส่ง

การรับ-ส่งข้อมูลในลักษณะนี้เรียกว่าแบบ frame คือมีกรอบปิดล้อมข้อมูลไว้ด้วย start bit และ stop bit

RS-232 Level Converters

สัญญาณ RS-232 มีค่าแรงไฟต่างจากที่ใช้ใน UART ดังแสดงในรูปที่ 2.8 ดังนั้นจึงต้องมี converter เพื่อแปลงระดับสัญญาณให้เหมาะสมก่อนที่จะเชื่อมต่อกับ serial port หรือ RS-232 port ของ computer

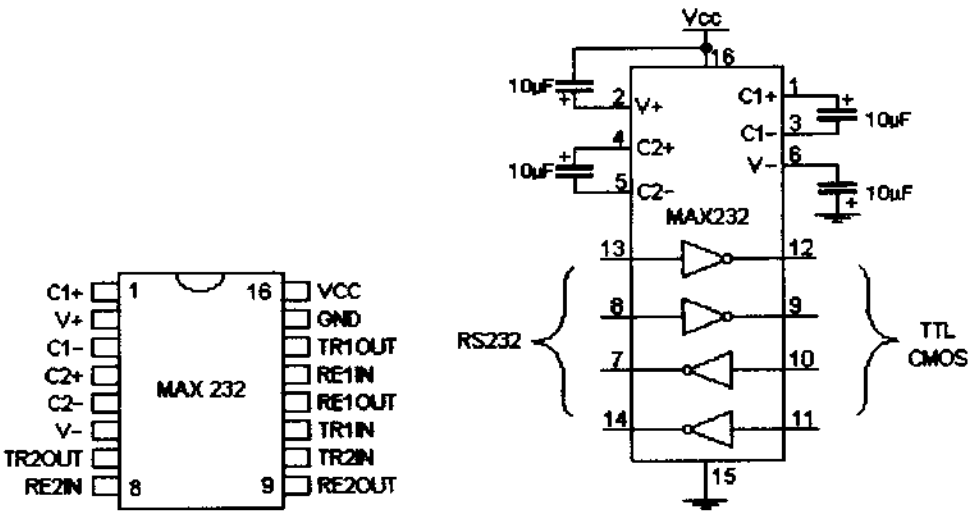
Waveform.



รูปที่ 2.8 RS-232 Logic

สำหรับสัญญาณ RS-232 นั้น logic "0" จะมีค่า +3 V ถึง +25 V และ logic "1" จะมีค่า –3 V ถึง –25 V ส่วนค่าระหว่าง –3 V ถึง +3 V เป็นค่า undefined ระดับสัญญาณนี้ใช้กับทุกสัญญาณไม่ใช่เฉพาะสัญญาณรับ-ส่งข้อมูลเท่านั้นแต่ยังรวมถึงสัญญาณควบคุมต่าง ๆ เช่น DTR, RTS, CTS, DCD, DSR เป็นต้น

ไอซี (IC) ที่ใช้มักจะเป็นหมายเลข 1488 (RS-232 Driver) และ 1489 (RS-232 Receiver) โดยภายในแต่ละตัวจะประกอบด้วย inverter 4 ตัวและต้องการไฟเลี้ยง 2 ชุดคือ +7.5 V ถึง +15 V และ –7.5 V ถึง –15 V ซึ่งอาจจะมีปัญหาในเครื่องที่มีไฟเลี้ยง +5 V เพียงชุดเดียว แต่ก็ยังมี ไอซี (IC) อีกตัวหนึ่งคือเบอร์ MAX-232 ซึ่งมีวงจร charge pump สามารถสร้างไฟ +10 V และ –10 V จากไฟ +5 V ได้ พร้อมทั้งมี 2 Tx และ 2 Rx อยู่ใน package เดียวกัน และรองรับ baud rate ได้ถึง 120 Kbps จึงสะดวกมากเพราะใช้ ไอซี (IC) เพียงตัวเดียว



รูปที่ 2.9 MAX-232

ส่วนการที่เราจะนำข้อมูลมาใช้งานก็ต้องแปลงเป็น parallel ก่อนซึ่งเป็นหน้าที่ของ UART ซึ่งปัจจุบัน microprocessor และ microcontroller มักจะมี serial communication interface (SCI) อยู่ในตัว แต่อาจจะมีงานบางอย่างที่ไม่ได้ใช้ microcontroller และต้องการ process ข้อมูลกับ serial communication interface เช่น ต่อ ADC เข้ากับ UART หรือต่อ LCD display เข้ากับ serial communication interface ก็ต้องใช้ UART ช่วย เช่นหมายเลข 8250 หรือ 16550A หรือหมายเลขอื่น ๆ ที่ได้กล่าวมาแล้ว แต่มี UART อีกพวกหนึ่งที่แยก Tx bus กับ Rx bus ออกจากกัน ทำให้มีความคล่องตัวมากขึ้น

ตารางที่ 2.16 การสรุปคุณสมบัติของมาตรฐานต่าง ๆ ที่นิยมใช้กันทั่วไป

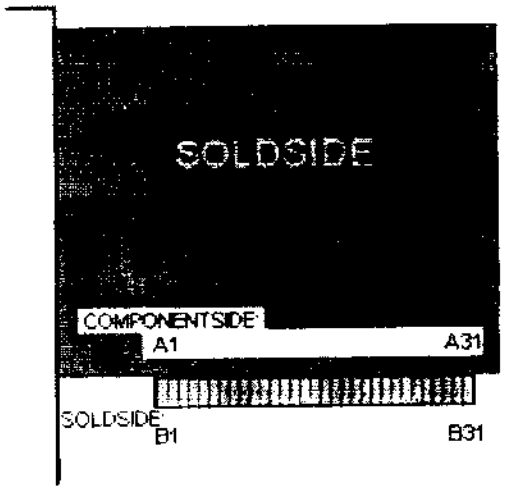
RS-232, RS-423, RS-422 and RS-485

Specification		RS-232	RS-423	RS-422	RS-485
Mode of operation		Single-ended	Single-ended	Differential	Differential
Total number of drivers and receivers on one line		1Tx 1Rx	1Tx 10Rx	1Tx 10Rx	1Tx 32Rx
Maximum cable length		50Ft	4,000Ft	4,000Ft	4,000Ft
Maximum data rate		20Kbps	100Kbps	10Mbps	10Mbps
Maximum driver output voltage		+/-25V	+/-6V	-0.25V to +6V	-7V to +12V
Driver output signal level (loaded min.)	Loaded	+/-5V to +/-15V	+/-3.6V	+/-2V	+/-1.5V
Driver output signal level (loaded max.)	Unloaded	+/-25V	+/-6V	+/-6V	+/-6V
Driver load impedance (ohms)		3K to 7K	>=450	100	54
Max. Driver current in high Z state	Power on	N/A	N/A	N/A	+/-100uA
Max. Driver current in high Z state	Power off	+/-6mA @ +/-2V	+/-100uA	+/-100uA	+/-100uA
Slew rate (max.)		30V/uS	Adjustable	N/A	N/A
Receiver input voltage range		+/-15V	+/-12V	-10V to +10V	-7V to +12V
Receiver input sensitivity		+/-3V	+/-200mV	+/-200mV	+/-200mV
Receiver input resistance (ohms)		3K to 7K	4K min.	4K min.	>=12K

2.5 Connector

ISA(technical)

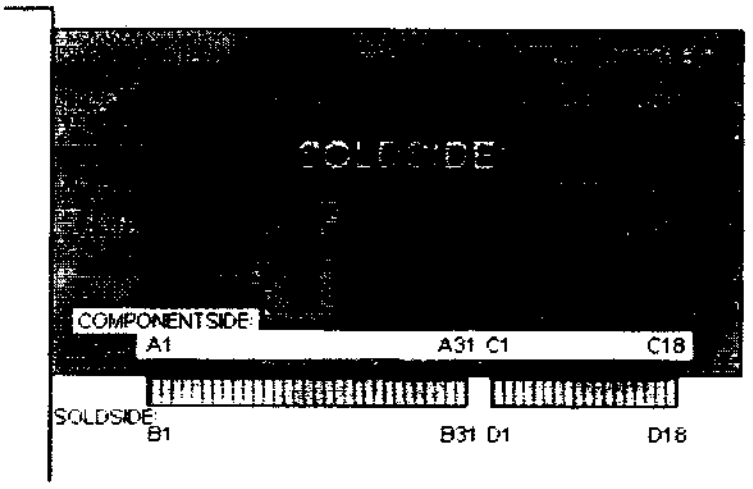
Physical Design



(at the card)

(at the computer)

รูปที่ 2.10 ISA Card 8-bit มี 62 pins

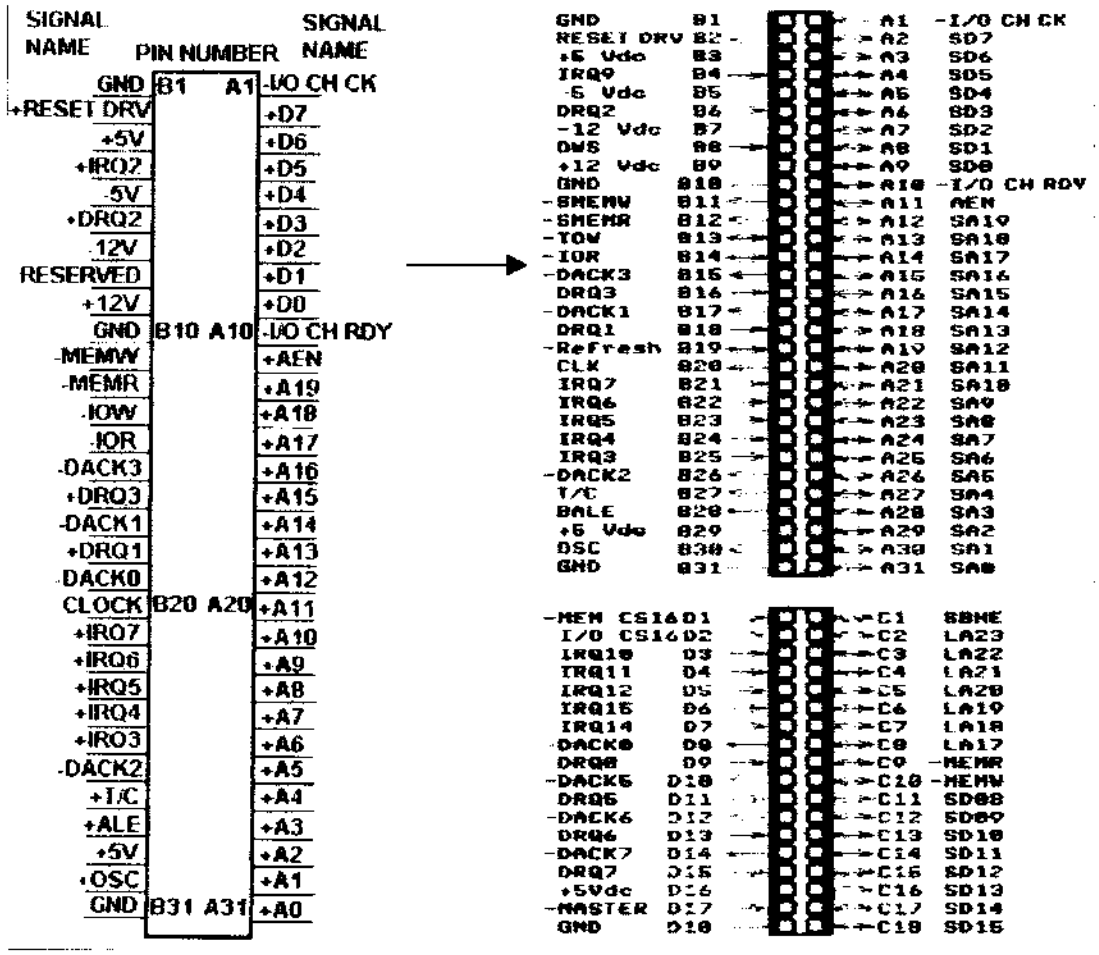


(at the card)

(at the computer)

รูปที่ 2.11 ISA Card 16-bit มี 98 pins

Socket ISA



รูปที่ 2.12 Socket ISA

หน้าที่ของขาสัญญาณที่สำคัญ

RESET DRV (B20) **คือ** สัญญาณเอาต์พุต "1" เมื่อจ่ายไฟให้ระบบจะเป็น "0"

IOR (B14) **คือ** สัญญาณเอาต์พุต "0" เพื่อแสดงว่าบัสไซเคิลของ I/O นี้ เป็นการอ่านข้อมูลที่แอสเดสตรงกับบัสส่งข้อมูลมายังบัส

IOW (B13) **คือ** สัญญาณเอาต์พุต "0" เพื่อแสดงว่าบัสไซเคิลของ I/O นี้ เป็นการเขียนข้อมูลที่แอสเดสตรงกับบัสส่งข้อมูลมายังบัส

MEMW (Memory Write B11) **คือ** สัญญาณเอาต์พุต "0" เพื่อให้หน่วยความจำที่แอสเดสตรงกับบัสนี้ทำการรับข้อมูล

MEMW (Memory Write B11) **คือ** สัญญาณเอาต์พุต "0" เพื่อให้หน่วยความจำที่แอสเดสตรงกับบัสนี้ทำการส่งข้อมูล

ALE (Address Latch Enable B28) คือ สัญญาณเอาต์พุต "0" เปลี่ยนเป็น "1" เมื่อแอสเดสถูกส่งออกบนบัสข้อมูลเรียบร้อย ใช้แยกค่าแอสเดส (SA0-SA19) และข้อมูล (SD0-SD7) ออกจากกัน

SA0-SA19 (Address Bus A31-A12) คือ สัญญาณเอาต์พุต 20 ขานี้ใช้สำหรับกำหนดแอสเดสของหน่วยความจำ หรือ I/O Hardware ที่ CPU ใช้

SD0-SD7 (Data Bus A9-A2) คือ สัญญาณเอาต์พุตแบบ Bi-Directional ทำหน้าที่ส่งผ่านข้อมูล I/O พอร์ตกับพีซี

I/O CH CK (Channel Check A1) คือ ใช้สัญญาณอินพุตที่ผิดพลาดโดยพาริตี้ ที่เกิดในการทำงานของ I/O Hardware เมื่อ "0" CPU จะถูกอินเทอร์รัพ

I/O CH RDY (Channel Ready A10) คือ ใช้สัญญาณอินพุตที่ใช้เพิ่มช่วงเวลาในบัสไซเคิล เมื่อ I/O Hardware/Memory ของขบวนการในบัสไซเคิล ทำงานไม่ทันตามเวลาปกติ

IRQ3-IRQ7 (Interrupt Request B25-B21) คือ ใช้ขออินเทอร์รัพจาก CPU เมื่อสัญญาณ "0"--> "1" ก็จะส่งสัญญาณ INT ไปที่ CPU

OSC (Oscillator B30) คือ สัญญาณเอาต์พุตต่อกับสัญญาณ Clock ที่มีค่าความถี่สูงสุดของบอร์ด(ไม่ควรนำสัญญาณไปใช้นี้ต่อกับส่วนอื่นๆ)

CLK (Clock B20) คือ สัญญาณเอาต์พุตต่อกับสัญญาณนาฬิกาที่สร้างจากสัญญาณ OSC (คาบเวลา=1/ความถี่ของ OSC ของบอร์ด)

2.6 Barcode

บาร์โค้ดมีประโยชน์มากในการใช้เก็บหรือบันทึกข้อมูลเกี่ยวกับปริมาณการขายสินค้านั้นๆเพื่อเป็นประโยชน์ในด้านการบริหารจัดการความเคลื่อนไหวของสินค้าประเภทต่าง ๆ ด้วยความ-รวดเร็วและให้ความถูกต้องและความปลอดภัยของข้อมูลสูง

ในเรื่องของความเร็วนั้น เราสามารถที่จะอ่านรายละเอียดของรหัสของอักขระจำนวน 12 ตัว ได้ภายในเวลาเพียง 2 วินาที ขณะเดียวกันในเรื่องความถูกต้องปลอดภัยนั้น บาร์โค้ดจะถูกพิมพ์ด้วยการใช้หมึกพิมพ์ที่มีผลตอบสนองต่อแสงอินฟราเรด ซึ่งเป็นการป้องกันการก๊อปปี้ จากการถ่ายแบบธรรมดา (photocopiers)

การอ่านรหัสของบาร์โค้ดสามารถอ่านผ่านข้ามช่องอากาศได้ โดยมีความยาวโฟกัส 2-4 มิลลิเมตร และเมื่อบาร์โค้ดถูกอ่านและถอดรหัสมาเป็นตัวอักขระแล้ว จะไม่เกิดการสับสนหรือความผิดพลาดขึ้นเหมือนกับแบบเก่าที่พิมพ์ตัวอักขระลงบนผลิตภัณฑ์หรือป้ายบอกราคา เช่น ØS กับ OS และ 1S กับ IS เป็นต้น

ในทางปฏิบัติ แถบบาร์โค้ดอาจจะเปรอะเปื้อนไปด้วยคราบน้ำมันหรือสิ่งสกปรก แต่อย่างไรก็ตามเรายังสามารถอ่านบาร์โค้ดนั้นได้ด้วยการใช้แสงอินฟราเรดส่องไปยังบาร์โค้ดนั้น โดยสิ่งที่ปกคลุมอยู่จะไม่มีผลประการใดต่อแสงอินฟราเรด

ประเภทของบาร์โค้ด

มีหลายวิธีที่จะเข้ารหัสข้อมูลด้วยการใช้แถบเส้น (bars) และช่องว่าง (spaces) แต่มีเพียง 4 ระบบหลัก ๆ เท่านั้นที่ใช้กันอยู่ในปัจจุบันคือ Code 39, EAN, Interleaved 2 of 5 และ Codabar

Code 39

Code 39 บางครั้งเราเรียกกันว่า 3 ใน 9 (3 of 9) ซึ่งพบเห็นได้ทุกๆ แห่ง โดยเฉพาะสินค้าขายปลีก สามารถแทนอักขระ A ถึง Z , ตัวเลข 0 ถึง 9 และอักขระพิเศษอีก 8 ตัวด้วยรหัสส่วนใหญ่เราจะพบเห็นรหัสบาร์โค้ดชนิดนี้โดยทั่วไป เช่น สินค้าที่มีอยู่ในร้านค้า หรือหมายเลขที่มีอยู่บนข้อความของเครื่องใช้ไฟฟ้า และส่วนประกอบของเครื่องยนต์ต่างๆ

รหัส Code 39 นี้มีอยู่ 2 ระดับ (กว้างและแคบ) นั่นคือแถบเส้นและช่องว่างจะเป็นได้ 2 ระดับ ไม่กว้างก็แคบ โดยมีอัตราส่วนของช่วงแคบต่อช่วงกว้างเท่ากับ 1:2.5

อักขระแต่ละตัวจะถูกแทนด้วยส่วนประกอบ 9 ส่วน โดยจะเป็นแถบเส้น 5 ส่วน และช่องว่างอีก 4 ส่วน สำหรับ 3 ใน 9 ส่วนนั้น เป็นส่วนกว้างซึ่งก็คือแถบเส้นหรือช่องว่างกว้าง (wide) แทนด้วยไบนารี "1" และส่วนที่แคบ (narrow) ก็แทนด้วยไบนารี "0" และจะมีช่องว่างแคบ ๆ กั้นระหว่างอักขระแต่ละตัว

ถึงแม้ว่ามีทั้งหมด 512 คอมบิเนชั่นของไบนารี 9 บิต แต่ที่เราใช้นั่นมีเพียง 44 คอมบิเนชั่น ดังนั้นรหัสนี้จะมีการตรวจสอบรหัสประจำตัวของมันเองอยู่ตลอดเวลา เพื่อความถูกต้องและปลอดภัยเป็นพิเศษ เราสามารถเพิ่มอักขระตรวจสอบเพิ่มไปในข้อความแต่ละข้อความ ด้วยการคำนวณหาค่า check-sum โดยการบวกค่าตรวจสอบ (check-sum) ประจำตัวของอักขระนั้นๆ ในหนึ่งข้อความ และนำผลรวมที่ได้มาหารด้วย 43 ซึ่งจะให้ค่าเศษที่เหลือหนึ่งค่า แล้วนำค่าเศษที่เหลือนี้มาเทียบกับค่า check-sum ในตารางที่ 2.17 ก็จะได้อักขระตรวจสอบ (check character) ที่จะนำมาเพิ่มต่อท้ายข้อความนั้นๆ

เราสามารถประยุกต์ใช้งานรหัส Code 39 ให้เป็นรหัสที่ซับซ้อนยิ่งขึ้นได้ โดยการใช้อักขระที่แน่นอนหนึ่งอักขระเป็นตัวเริ่มต้นของอักขระโดยใช้ 2 อักขระหรือกลุ่มอักขระหรือกลุ่มอักขระ เช่น การแทนด้วยรหัสแอสกี้ (ASCII code) รวมถึงรหัสควบคุม (control code) เช่น carriage return จะถูกแทนด้วยรหัส $M

จากตารางที่ 2.17 จะแสดงถึงรหัสเลขฐานสอง (binary code) อักขระ

หมายเหตุ อักขระ * นั้นไม่มีค่าตรวจสอบ (check value) ประจำตัวของมันเอง เพราะมันถูกใช้เป็นตัวแสดงถึงการเริ่มต้นและการสิ้นสุดของรหัสข้อความใน Code 39 ทุกๆข้อความ

ตารางที่ 2.17 อักขระของรหัส Code 39

ตัวอักษร	เลขฐานสอง	Check-sum
0	000110100	0
1	100100001	1
2	001100001	2
3	101100000	3
4	000110001	4
5	100110000	5
6	001110000	6
7	000100101	7
8	100100100	8
9	001100100	9
A	100001001	10
B	001001001	11
C	101001000	12
D	000011001	13

ตารางที่ 2.17 (ต่อ) อักขระของรหัส Code 39

ตัวอักษร	เลขฐานสอง	Check-sum
E	100011000	14
F	001011000	15
G	000001101	16
H	100001100	17
I	001001100	18
J	000011100	19
K	100000011	20
L	001000011	21
M	101000010	22
N	000010011	23
O	100010010	24
P	001010010	25
Q	000000111	26
R	100000110	27
S	001000110	28
T	000010110	29
U	110000001	30
V	011000001	31
W	111000000	32
X	010010001	33
Y	110010000	34
Z	011010000	35
-	010000101	36
.	110000100	37
SPACE	011000100	38
*	010010100	-
$	010101000	39
/	010100010	40
+	010001010	41

ตารางที่ 2.17 (ต่อ) อักขระของรหัส Code 39

ตัวอักษร	เลขฐานสอง	Check-sum
%	000101010	42

แต่ละตัว และค่าตัวอักขระตรวจสอบของมันเอง

ตัวอย่างการเข้าบาร์โค้ด Code 39

กำหนดให้รหัสของข้อความเป็น 98PQ

-ขั้นแรกหาผลรวมของค่า check-sum ของอักขระทุก ๆ ตัวในข้อความนั้น คือ 98PQ (9+8+25+26 = 68)

-หารผลรวมที่ได้ด้วย 43 (68/43 = 1 เศษ 25)

-ต่อมาให้ไปดูตารางที่ 1 ว่าอักขระใดที่มีค่า check-sum เท่ากับ 25 ซึ่งอักขระที่ได้คือ ตัว P

-ดังนั้นข้อความก็จะถูกแปลงเป็นรหัส Code 39 รวมทั้ง check-sum และอักขระเริ่มต้นและสิ้นสุดด้วย ดังนี้ *98PQP*

-จากนั้นนำข้อความที่ได้ มาแปลงเป็นรหัสไบนารีจะได้ดังนี้

98PQP ==> 010010100 / 0 / 001100100 / 0 / 100100100 / 0 / 001010010 / 0 / 000000111 / 0 / 010010100

-แล้วนำเลขไบนารีที่ได้มาแทนด้วยแถบเส้นหรือช่องว่าง โดยให้ไบนารี "0" แทนด้วยแถบเส้นหรือช่องว่างที่แคบ (narrow) และไบนารี "1" แทนด้วยส่วนที่กว้าง (wide) จะได้บาร์โค้ดตามความต้องการ

หมายเหตุ อักขระแต่ละตัวในข้อความหนึ่ง ๆ จะถูกแยกจากกันด้วยช่องว่างแคบ ๆ (narrow space) ซึ่งมีค่าไบนารีเป็น "0"

รูปที่ 2.13 Code 39 เป็นบาร์โค้ดที่ใช้กันอย่างกว้างขวาง โดยเฉพาะสินค้าขายปลีกทั่วไป

กว้าง) คือ ในแต่ละแถบเส้น หรือช่องว่างจะมีระดับความกว้าง 1, 2, 3 หรือ 4 โดยให้แถบเส้นแทนด้วยไบนารี "1" ในขณะที่ช่องว่างแทนด้วยไบนารี "0" ตัวอย่างเช่น รหัส 00011 ก็จะถูกแทนด้วยช่องว่างที่มีความกว้าง 3 ส่วน และตามด้วยแถบเส้นที่มีความกว้าง 2 ส่วน

อักขระแต่ละตัวถูกสร้างขึ้นด้วยเลขไบนารี 7 บิต โดยรหัสบาร์โค้ดหนึ่งๆ จะประกอบด้วยรหัสกั้นหน้า รหัสกั้นกลาง และรหัสกั้นหลัง (start, center and end guard bars) รหัสที่อยู่ทางด้านซ้ายของรหัสกั้นกลาง ถูกเข้ารหัสโดยใช้คอลัมน์ด้านขวามือ ความแตกต่างระหว่าง 2 คอลัมน์ทางด้านซ้ายมือนั่นคือ A จะใช้เข้ารหัสข้อมูลกับข้อความที่มีจำนวนเป็นคี่ (odd parity) และจำนวนเป็นคู่ (even parity) ใช้คอลัมน์ B

ตารางที่ 2.18 อักขระของรหัส EAN

เลขที่	ซ้ายมือ	ซ้ายมือ	ขวามือ
0	0001101	0100111	1110010
1	0011001	0110011	1100110
2	0010011	0011011	1101100
3	0111101	0100001	1000010
4	0100011	0011101	1011100
5	0110001	0111001	1001110
6	0101111	0000101	1010000
7	0111011	0010001	1000100
8	0110111	0001001	1001000
9	0001011	0010111	1110100

EAN 8 ใช้กับรหัสที่มีความยาว 5 หลัก ส่วนแบบ EAN 13 นั้นใช้กับรหัสที่มีความยาว 10 หลัก รหัสทั้งสองชนิดนี้ประกอบไปด้วยรหัสกั้นหน้า รหัสกั้นกลาง และรหัสกั้นหลัง อักขระแฟล็ก (flag character) 2 ตัว และอาจมีรหัส 2-5 หลักเพิ่มขึ้นมาอีก ซึ่งมีรายละเอียดดังนี้

รหัส EAN 8

บาร์โค้ดแบบ EAN 8 นั้นประกอบไปด้วยส่วนต่าง ๆ ตามลำดับดังนี้

-แถบรหัสกั้นหน้า ซึ่งเข้ารหัสด้วย 101

-อักขระแฟล็ก 2 ตัว เข้ารหัสด้วยคอลัมน์ซ้ายมือ A (ดูตารางที่ 2.18)

-ข้อมูลอักขระ 2 ตัวแรกเข้ารหัสด้วยคอลัมน์ซ้ายมือ A เช่นกัน

-แถบรหัสกั้นกลาง ซึ่งเข้ารหัสด้วย 01010

-ข้อมูลอักขระ 3 ตัวหลัง เข้ารหัสด้วยคอลัมน์ขวามือ

-อักขระตรวจสอบ ซึ่งเข้ารหัสด้วยคอลัมน์ขวามือเช่นกัน

-แถบรหัสกั้นหลัง เข้ารหัสด้วย 101

รูปที่ 2.14 EAN 8 ใช้สำหรับรหัสที่มีความยาว 5 หลัก

รหัส EAN 13

ส่วนบาร์โค้ดแบบ EAN 13 นั้นมีส่วนประกอบคล้ายคลึงกันดังนี้

-แถบรหัสกั้นหน้า ซึ่งเข้ารหัสด้วย 101

-อักขระแฟล็กที่สอง (second flag character) 1 ตัว เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

-ข้อมูลอักขระ 5 ตัวแรก เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

-แถบรหัสกั้นกลาง เข้ารหัสด้วย 01010

-ข้อมูลอักขระ 5 ตัวหลัง เข้ารหัสด้วยคอลัมน์ขวามือ

-อักขระตรวจสอบเข้ารหัสด้วยคอลัมน์ขวามือเช่นกัน

-และแถบรหัสกั้นหลัง เข้ารหัสด้วย 101

อักขระแฟล็กที่หนึ่งของรหัส EAN 13 ถูกเข้ารหัสด้วยการใช้พาริตี้ แพ็ทเทอน (parity pattern) ของอักขระแฟล็กที่สอง ข้อมูลอักขระ 5 ตัวแรก (first five data character) ดังตารางที่ 2.19

0	A	A	B	B	A	B
3	A	A	B	B	B	A
4	A	B	A	A	B	B
5	A	B	B	A	A	B
6	A	B	B	B	A	A
7	A	B	A	B	A	B
8	A	B	A	B	B	A
9	A	B	B	A	B	A

รหัสบาร์โค้ด EAN 8 ซึ่งมีข้อความเป็น 80123453 สามารถแทนด้วยเลขไบนารีดังนี้ 101 / 0110111 / 0001101 / 0011001 / 0010011 / 01010 / 1000010 / 1011100 / 1001110 / 100010 / 101 (เพื่อให้ดูง่ายๆ เราจะใช้เครื่องหมาย "/" คั่นระหว่างรหัสหรืออักขระแต่ละตัว)

อักขระตรวจสอบสามารถหาได้โดยการสมมติว่าตัวอักขระขวาสุดเป็นตำแหน่งคี่ (odd)==>EOEOEO แล้วบวกอักขระทั้งหมดในตำแหน่งคี่ (odd) และคูณด้วย 3 ได้เป็นผลลัพธ์แรก ส่วนผลลัพธ์ที่สองหาจากผลรวมของอักขระทั้งหมดในตำแหน่งคู่ (even) ดังนั้นผลลัพธ์ที่ได้คือผลรวมของทั้งสองกรณีข้างต้น

อักขระตรวจสอบคือ จำนวนเลขที่น้อยที่สุดที่บวกเข้ากับผลลัพธ์ (ดังกล่าวแล้วในข้างต้น) แล้วสามารถหารจำนวนนั้นด้วย 10 ได้ลงตัว (หรือบวกให้หลักหน่วยเป็นศูนย์)

ตัวอย่างรหัส EAN 13

ตัวอย่างรหัส EAN 13 กำหนดให้อักขระแฟล็กคือ 97 และมีข้อความเป็น 7095983300 ดังนั้นเมื่อรวมกันแล้วจะได้ข้อความเป็น 977095983300 ตำแหน่งของคู่/คี่ (even/odd) เป็นดังนี้ EOEOEOEOEOEO ดังนั้นผลรวมของเลขในตำแหน่งคี่คูณด้วย 3 ได้เท่ากับ 69 [(7+0+5+8+3+0) *3] และผลรวมของเลขในตำแหน่งคู่เท่ากับ 37(9+7+9+9+3+0) จากนั้นนำผลรวมของทั้งสองข้างต้นมาบวกกันได้เป็นผลลัพธ์เท่ากับ 106 (69+37) ดังนั้นเมื่อนำ 4 บวกเข้ากับ 106 จะได้เท่ากับ 110 ซึ่งหารด้วย 10 ได้ลงตัวพอดี ฉะนั้นอักขระตรวจสอบคือ 4 และได้ข้อความเต็ม ๆ ดังนี้ 9770959833004

รูปที่ 2.15 EAN 13 ใช้กับข้อความที่มีความยาว 10 หลัก

รหัสเพิ่ม 2 หลัก

รหัสบาร์โค้ด 2 หลักที่เพิ่มขึ้นมา (two digit supplement) ซึ่งอยู่ด้านหน้าของบาร์โค้ดหลักนั้น จะแสดงหมายเลขเดือน โดยเริ่มจาก 01 สำหรับเดือนมกราคม (January)

รหัส 2 หลักที่เพิ่มขึ้นมานี้ จะประกอบด้วยส่วนต่าง ๆ ตามลำดับ ดังนี้

-แถบรหัสกั้นข้าง (guard bars) เข้ารหัสด้วย 1011

-ข้อมูลอักขระตัวแรก เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

-อักขระแยก (character delineator) เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

ข้อมูลอักขระที่ถูกเข้ารหัสโดยใช้คอลัมน์ซ้ายมือ A หรือ B นั้น ขึ้นอยู่กับหลักของอักขระนั้น ตัวอย่างเช่น ถ้าส่วนที่เพิ่มขึ้นมานั้นคือ 13 เราจะใช้ตารางที่ 2.20 ในการอ้างอิงซึ่งจะได้คอลัมน์ซ้ายมือ A ใช้สำหรับเลข 1 และ 3 ก็จะเข้ารหัสจากคอลัมน์ซ้ายมือ B (เลข 13 อยู่ในคอลัมน์ A-B)

ตารางที่ 2.20 พาริตี้แพ็ตเทิร์นของ 2 หลักที่เพิ่มมาของ EAN

A-A	A-B	B-A	B-B
00	01	02	03
04	05	06	07
08	09	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59
60	61	62	63
64	65	66	67
68	69	70	71
72	73	74	75
76	77	78	79
80	81	82	83
84	85	86	87
88	89	90	91
92	93	94	95
96	97	98	99

รูปที่ 2.16 ตัวอย่างบาร์โค้ด 5 หลัก ที่เพิ่มมาของ EAN

<u>รหัสเพิ่ม 5 หลัก</u>

ส่วนบาร์โค้ดที่เพิ่มขึ้นมา 5 หลัก (five digit supplement) นั้น ส่วนมากมักจะพบเห็นกันบนหนังสือหรือนิตยสารที่ป้ายบอกราคาหรือปกหนังสือ ใน 5 หลักที่เพิ่มมานั้นประกอบไปด้วยส่วนต่าง ๆ ตามลำดับดังนี้

-แถบเส้นด้านข้าง ซึ่งเข้ารหัสด้วย 1011

-ข้อมูลอักขระตัวแรก เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

-อักขระแยก เข้ารหัสด้วย 1

-ข้อมูลอักขระตัวที่สอง เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

-อักขระแยก เข้ารหัสด้วย 01

-ข้อมูลอักขระตัวที่สาม เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

-อักขระแยก เข้ารหัสด้วย 1

-ข้อมูลอักขระตัวที่สี่ เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

-อักขระแยก เข้ารหัสด้วย 01

-ข้อมูลอักขระตัวที่ห้า เข้ารหัสด้วยคอลัมน์ซ้ายมือ A หรือ B

เช่นเดียวกัน ตัวอักขระจะถูกเข้ารหัสโดยการใช้คอลัมน์ซ้ายมือ ซึ่งคอลัมน์ที่ใช้นั้นคิดจากค่า check-sum ที่คิดเหมือนกับอักขระตรวจสอบของข้อความหลัก โดยสมมติให้อักขระด้านขวาสุดเป็นตำแหน่งคี่ จากนั้นนำอักขระที่อยู่ในตำแหน่งคี่มาบวกกันแล้วคูณด้วย 3 และนำอักขระที่อยู่ในตำแหน่งคู่มาบวกกันแล้วคูณด้วย 9 แล้วนำผลลัพธ์ทั้งสองข้างต้นมาบวกกัน จะได้ค่า ๆ หนึ่ง โดยสนใจเฉพาะเลขในหลักหน่วยของค่าผลลัพธ์ที่ได้ ซึ่งก็คือหมายเลขของพาริตี้แพ็ตเทิร์นในตารางที่ 2.21

ตัวอย่างเช่น ใน 5 หลักที่เพิ่มมาคือ 12345 มีตำแหน่งคี่-คู่ คือ OEOEO จะมีผลรวมของตำแหน่งคี่ทั้งหมดคูณด้วย 3 เท่ากับ 27 และผลรวมของตำแหน่งคู่คูณด้วย 9 เท่ากับ 54 เสร็จแล้ว นำผลลัพธ์ทั้งสองมารวมกันได้เท่ากับ 81 (27+54) ดังนั้นหมายเลข (No.) ของพาริตี้แพ็ตเทิร์นคือ หลัก

หน่วย ซึ่งคือ 1 นั่นเอง ฉะนั้นเราสามารถรู้ได้ว่าจะใช้คอลัมน์ซ้ายมือ A หรือ B ในการเข้ารหัสโดยดูตารางที่ 2.21 หมายเลข 1 (ตามตัวอย่าง) ซึ่งคอลัมน์ที่ใช้เข้าข้อมูลที่ 1-5 ได้เรียงตามลำดับดังนี้ BABAA

ตารางที่ 2.21 พาริตี้แพ็ตเทิร์นของ 5 หลักที่เพิ่มมาของ EAN 13

NO	Data 1	Data 2	Data 3	Data 4	Data 5
0	B	B	A	A	A
1	B	A	B	A	A
2	B	A	A	B	A
3	B	A	A	A	B
4	A	B	B	A	A
5	A	A	B	B	A
6	A	A	A	B	B
7	A	B	A	B	A
8	A	B	A	A	B
9	A	A	B	A	B

Interleaved 2 of 5

Interleaved 2 of 5 เป็นรหัสบาร์โค้ดที่ใช้สำหรับการขนส่งพัสดุหีบห่อ

รหัสที่ใช้มีเพียงตัวเลข 0-9 ดังที่แสดงไว้ในตารางที่ 6 โดยมี 2 ระดับ (กว้างหรือแคบ) ดังนั้นทั้งแถบเส้นหรือช่องว่าง จึงเป็นสัญลักษณ์ที่ใช้บอกความหมายของรหัสนั้น ๆ โดยแต่ละแถบเส้นและช่องว่าง จึงเป็นสัญลักษณ์ที่ใช้บอกความหมายของรหัสนั้น ๆ โดยแต่ละแถบเส้นและช่องว่างนั้นสามารถเป็นได้เพียงแถบเส้นหรือช่องว่างที่แคบหรือกว้างเท่านั้น อัตราส่วนของความกว้างต่อความแคบอยู่ระหว่าง 2 ถึง 3 ต่อ 1

อักขระแต่ละตัวถูกแทนด้วยส่วนกว้าง 2 ส่วน ในจำนวนทั้งหมด 5 ส่วน โดยที่แถบเส้นแคบและช่องว่างแคบถูกแทนด้วยไบนารี "0" เช่นเดียวกับแถบเส้นรือช่องว่างที่กว้าง แทนด้วยไบนารี "1"

รหัส Interleaved 2 of 5 นี้จะเริ่มต้นด้วยรหัส 0000 และปิดท้ายด้วยรหัส 100 โดยข้อความจะบรรจุอยู่ระหว่างรหัสเริ่มต้นและรหัสสิ้นสุด คือ ถูกสอดแทรก (interleaved) นั่นเอง ดังนั้นอักขระตัวแรกจะตามหลังรหัสเริ่มต้น และอักขระตัวที่สองจะแทรกอยู่ในช่องว่างของตัวอักขระตัวแรก (FSFSFSFSFS) โดยให้ F แทนรหัสของอักขระตัวแรก และ S แทนรหัสของอักขระตัวที่สอง

บาร์โค้ดแบบ Interleaved 2 of 5 มีความจุสูงนั่นคือ จำนวนตัวเลขต่อตัวอักขระน้อย ตามความจริงนั้นตัวอักขระที่ถูกแทรกเข้าไปนั้นยังหมายถึงการแบ่งแยกที่ไม่ต้องการ (รหัสเป็นลักษณะต่อเนื่อง) ตัวอักขระจะมีการตรวจสอบตัวเอง โดยอักขระแต่ละตัวประกอบไปด้วยส่วนกว้าง 2 ส่วน และส่วนแคบ 3 ส่วน

รหัสนี้จะมีอักขระ check-sum ที่เลือกขึ้นมาจากเกณฑ์การคูณด้วย 10 ตัวอย่างเช่น ข้อความ 57654823 ตัว check-sum ถูกคำนวณ (เหมือนข้างต้น) โดยกำหนดให้ตัวอักขระด้านขวาสุดเป็นตำแหน่ง E (even) ดังนั้นผลรวมของตัวอักขระในตำแหน่ง O (odd) เท่ากับ 17 ส่วนผลรวมในตำแหน่ง E (even) เท่ากับ 23 แล้วนำไปคูณด้วย 3 ได้เท่ากับ 69 จากนั้นนำผลการคำนวณทั้งสองมารวมกันได้ผลลัพธ์เท่ากับ 86 (69+17) ดังนั้นจะได้ check-sum เท่ากับ 4 [check-sum คือ เลขจำนวนที่น้อยที่สุด ซึ่งบวกเข้ากับผลลัพธ์ทั้งหมดที่หาได้แล้วหารด้วย 10 ลงตัว (86+4=90)]

ดังนั้นข้อความที่เพิ่มตัวอักขระตรวจสอบคือ 576548234 อย่างไรก็ตาม เพราะว่าตัวอักขระที่จะเข้ารหัสต้องเป็นคู่ ๆ ดังนั้นถ้าข้อความนั้นไม่เป็นคู่ เราจะทำให้เป็นคู่ได้โดยเติมเลข 0 ข้างหน้าข้อความนั้น ๆ ฉะนั้นข้อความใหม่จะถูกเข้ารหัสเป็น 0576548234

ตัวอย่างการเข้ารหัสของข้อความ 2345 โดยไม่มีตัว check-sum ทำได้โดยการแบ่งข้อความออกเป็นคู่ คู่แรกของข้อความคือ อักขระ 23 และคู่หลังก็คือ 45 จากนั้นก็แปลงอักขระคู่แรกคือ 23 เป็นรหัสไบนารี โดยอักขระเลข 2 แทนด้วยรหัส 01001 และตัวเลข 3 เท่ากับ 11000 ซึ่งดูได้จากตารางที่ 2.22 แล้วนำรหัสที่ได้มารวมกันแบบสอดแทรก นั่นคือให้อักขระตัวหลัง (3) อยู่ที่ตำแหน่งคี่ และอักขระตัวแรก (2) อยู่ที่ตำแหน่งคู่ ฉะนั้นจะได้รหัสของคู่แรกคือ 23 เป็น 0111000010

ส่วนการแปลงรหัสในคู่หลังนั้น ก็เหมือนกับในคู่แรกคือ อักขระเลข 4 แทนด้วยรหัส 00101 และอักขระเลข 5 แทนด้วย 10100 แล้วนำมาสอดแทรกกันได้รหัสของคู่หลังเป็น 0100110010

ผลสุดท้ายก็นำรหัสที่ได้จากข้อความในคู่แรกกับคู่หลังมารวมกัน (เขียนต่อกัน) โดยเพิ่มรหัสเริ่มต้นคือ 0000 ไว้ข้างหน้า และรหัสสิ้นสุดคือ 100 ไว้ข้างหลังสุด ดังนี้ 000 / 01110001001001100 10 / 100

ตารางที่ 2.22 อักขระของรหัส Interleaved 2 of 5

ASCII	เลขฐานสอง
1	10001
2	01001
3	11000
4	00101
5	10100

ตารางที่ 2.22 (ต่อ) อักขระของรหัส Interleaved 2 of 5

ASCII	เลขฐานสอง
6	01100
7	00011
8	10010
9	01010
0	00110

Codabar

บาร์โค้ดแบบ Codabar นี้มี 2 ระดับ (กว้างหรือแคบ) ประกอบด้วยตัวเลข 0-9, อักขระพิเศษ 6 ตัว และตัวอักขระอีก 4 ตัว ให้เลือกใช้ซึ่งเป็นอักขระเริ่มต้นและสิ้นสุดในแต่ละตัวอักขระประกอบด้วยรหัสไบนารี 7 บิต (ดังแสดงในตารางที่ 2.23) โดยจะเป็นเลขไบนารี "1" 2 หรือ 3 บิต แต่ละตัวอักขระจะถูกแยกจากกันด้วยช่องว่างแคบ ๆ ไบนารี "1" เข้ารหัสด้วยแถบเส้นกว้างหรือช่องว่างกว้าง และไบนารี "0" ก็แทนด้วยแถบเส้นหรือช่องว่างแคบ

การที่มี 20 ตัวอักขระที่ใช้งานจากที่เป็นไปได้ 128 ตัวนั้น ทำให้รหัสชนิดนี้มีการตรวจสอบตัวเองเป็นลักษณะประจำตัว และไม่มีการกำหนดค่าของ check-sum

ข้อความหนึ่ง ๆ สามารถจะเข้ารหัสแบบ Codabar โดยจะประกอบด้วยอักขระเริ่มต้นและสิ้นสุด 4 ตัว (A,B,C หรือ D) ตัวใดตัวหนึ่ง ตัวอย่างเช่น ข้อความ 2345 ใช้ A เป็นอักขระเริ่มต้นและปิดท้าย ดังนั้นจะได้รหัส Codabar เป็น A2345A ซึ่งแปลงเป็นรหัสไบนารีได้ดังนี้ 0011010 / 0 / 0001001 / 0 / 1100000 / 0 / 0010010 / 0 / 1000010 / 0 / 0011010

โดยจะมีช่องว่างแคบๆ เป็นตัวแบ่งแยกอักขระแต่ละตัว ซึ่งก็คือเลขไบนารี "0"

ABC Codebar หรือที่รู้จักกันในชื่อ NW7 และค่อนข้างใช้กันอย่างกว้างขวาง ABC เป็นชื่อย่อจาก American Blood Commission และรหัสชนิดนี้เป็นมาตรฐานที่ยอมรับของนานาชาติ ซึ่งใช้เป็นบาร์โค้ดในงานเกี่ยวกับการถ่ายโลหิต และส่วนใหญ่แล้วจะใช้กันในด้านเวชกรรมการแพทย์

ยังมีรหัสกลุ่มอื่น ๆ ที่มีลักษณะคล้าย ๆ กับแบบ Code 39 ซึ่งเรียกว่า UPC (Universal Product Code) และ EAN/JAN (European/Janpan Article Number)

รูปที่ 2.17 Interleaved 2 of 5 พบบ่อยในด้านการขนส่งพัสดุหีบห่อ

ตารางที่ 2.23 อักขระของรหัส Codebar

ASCII	เลขฐานสอง
0	0000011
1	0000110
2	0001001
3	1100000
4	0010010
5	1000010
6	0100001
7	0100100
8	0110000
9	1001000
-	0001100
$	0011000
:	1000101
/	1010001
.	1010100
+	0011010
A	0011010
B	0101001
C	0001011
D	0001110

ลักษณะที่คล้ายคลึงกันของ UPC และ EAN/JAN

ระบบ UPC นั้นได้ถูกพัฒนาขึ้นในยุโรปและญี่ปุ่นตามลำดับ ซึ่งแต่ละระบบนั้นจะพบเห็นได้ตามหนังสือนิตยสาร หรือสัญลักษณ์ที่พิมพ์อยู่บนผลิตภัณฑ์ต่าง ๆ ตามซุปเปอร์มาร์เก็ต

แน่นอนที่ข้อความต่าง ๆ นั้น ไม่ได้ถูกพิมพ์ด้วยบาร์โค้ดที่เหมือนกันซึ่งบริษัททั้งหลายจะต้องประยุกต์ใช้กับตัวเลขเฉพาะอย่างของ Article Number Association (ANA) ของประเทศนั้น ๆ

ระบบ UPC A จะมีรหัสอยู่ 12 หลัก สามารถบอกความหมายได้หลายอย่าง โดยอาศัยหลักแรกของรหัสนี้

3 [illegible]

4 ข้อความที่แสดงว่า ไม่ใช่อาหาร

5 ใช้ในพวกคูปอง

ระบบ UPC E มีรหัส 6 หลัก ใช้สำหรับการแสดงข้อความเล็ก ๆ น้อย ๆ

ระบบ EAN/JAN 13 จะมีรหัส 13 หลัก ซึ่งสองอักขระแรกหมายถึงประเทศที่เป็นต้นแบบ เช่น รหัส 50 = UK

ระบบ EAN/JAN 8 จะมีรหัส 8 หลัก ใช้สำหรับข้อความสั้น ๆ เมื่อใช้กับหนังสือและวารสาร รหัส 2 ตัวแรกจะถูกตั้งค่า (set) ให้เป็น 97 ส่วนในเรื่องของประเทศต้นกำเนิดนั้นไม่ได้คำนึงถึง เพราะว่าได้จัดรูปแบบตามมาตรฐาน ISSN ซึ่งได้ถูกพัฒนามาก่อนแบบ UPC/EAN/JAN

รูปที่ 2.18 Codebar ใช้กันมากในด้านเวชกรรมการแพทย์

การเลือกใช้รหัสให้เหมาะกับงาน

ปกติการเลือกรหัสบาร์โค้ดใช้งานจะถูกบังคับตามการใช้งานเฉพาะอย่าง ถ้าการใช้งานแบบใหม่หรือผิดไปจากทั้งหมดที่ได้กล่าวมาแล้ว รหัสที่ควรจะพิจารณาถึงก็คือ Code 39 ที่ใช้ตัวเลขและตัวอักษร A-Z หรือแบบ Interleaved 2 of 5 ซึ่งใช้กับข้อมูลที่เป็นตัวเลขล้วนๆ

Interleaved 2 of 5 ยังให้ความหนาแน่นของรหัสมากที่สุดอีกด้วย ทั้ง Code 39 และ Interleaved 2 of 5 เป็นรหัสแบบ 2 ระดับ ที่ให้จำนวนตัวแปรมากกว่าและพิมพ์ได้คุณภาพดีกว่าแบบรหัส 4 ระดับอย่าง EAN

ลักษณะประจำตัวของรหัสทั้ง 2 แบบคือ Code 39 และ Interleaved 2 of 5 นั้น จะมีการตรวจสอบตัวเอง ด้วยการใช้จำนวนคอมบิเนชั่นเพียงเล็กน้อยจากจำนวนมากมายที่เป็นไปได้ และยังมีตัว check-sum เพิ่มมาอีกด้วย

ข้อเสียของรหัส Code 39 คือ มีรหัสยาว ซึ่งแต่ละตัวอักขระจะประกอบไปด้วยรหัสไบนารี ถึง 9 บิต ส่วนรหัส Interleaved 2 of 5 นั่นคือ ข้อมูลที่ใช้ถอดรหัสต้องมีจำนวนเป็นคู่

ตัวอ่านรหัสบาร์โค้ด

ตัวอ่านรหัสบาร์โค้ดหรือสแกนเนอร์(scanner) ที่ใช้กันตามซุปเปอร์มาเก็ตในการคิดราคาสินค้า (check-out) นั้น โดยใช้วิธีการกวาดลำแสงเลเซอร์สีแดงบนผลิตภัณฑ์สินค้าด้วยการกวาดลำแสงแบบเคลื่อนที่ไปมาได้เอง ด้วยอัตราประมาณ 40 ครั้งต่อวินาที ซึ่งระยะห่างของการอ่านบาร์โค้ดอาจไกลเกิน 30 เซนติเมตร โดยสามารถอ่านรหัสบาร์โค้ดบนผลิตภัณฑ์เกือบทุกประเภทที่ปรากฏจากรอยต่อ

สแกนเนอร์แบบสัมผัส

ปัจจุบันสแกนเนอร์แบบสัมผัส (Contact CCD scanner) จะพบเห็นกันส่วนมากตามร้านค้าทั่วไปสามารถอ่านรหัสบาร์โค้ดโดยใช้แสงจาก LED สีแดงส่องไปที่บาร์โค้ด และมีดีเท็กเตอร์ (Detector) คอยบันทึกการสะท้อนของแสงจากแถบเส้นและช่องว่างบนบาร์โค้ด แล้วแปลงให้เป็นสัญญาณไฟฟ้า

ถึงแม้ว่าสแกนเนอร์จะเป็นแบบสัมผัส แต่สามารถยอมรับการอ่านบาร์โค้ดในระยะที่มากกว่า 1 เซนติเมตร และสามารถอ่านรหัสแม้กระทั่งเส้นเทปพลาสติกของคาสเซตต์เทปและแผ่น CD

เลเซอร์สแกนเนอร์

นอกจากนี้ยังมีเลเซอร์สแกนเนอร์ (Laser scanner) ซึ่งหาได้ทั่วไปในลักษณะที่ถูกออกแบบมาให้คล้าย ๆ กับปืน โดยสามารถอ่านรหัสบาร์โค้ดได้ในระยะห่างที่น้อยกว่า 200 เซนติเมตร สแกนเนอร์แบบนี้เป็นลักษณะการสแกนทางแนวนอน โดยมีกระจกสะท้อนแสงซึ่งหมุนได้มากกว่า 4 ทิศทาง เพื่อบันทึกการสะท้อนของลำแสงจากบาร์โค้ด ซึ่งให้ความแม่นยำมากในการอ่านรหัสบาร์โค้ด โดยผู้ใช้เพียงแต่ส่องลำแสงให้ตรงกับบาร์โค้ดที่จะอ่าน แล้วยิงลำแสงเข้าไปตามแนวขวางตัดกับแนวเส้นของบาร์โค้ด

สแกนเนอร์แบบใช้มือนั้นจะให้ความแม่นยำในการสแกนรหัสบาร์โค้ดเหมือนกัน และที่ต่างจากแบบอื่นคือ ราคาที่ต่ำที่สุดอีกทั้งยังต้องการพลังงานที่ต่ำที่สุด ส่วนมากสแกนเนอร์ชนิดมือถือนี้ใช้ตัวเปล่งแสงที่แผ่กว้าง และตัวดีเท็กเตอร์แบบรวมแสง และตัวดีเท็กเตอร์ที่ตรวจจับพื้นที่ได้กว้าง

แต่สแกนเนอร์แบบใช้มือนั้น ยังสามารถทำให้เกิดข้อผิดพลาดได้ด้วยปัญหาในเรื่องความเร็ว การทำซ้ำและความเร็วคงที่ในการรับข้อมูล ซึ่งผลที่ได้จากสแกนเนอร์แบบใช้มือนี้จะขึ้น

อยู่กับผู้ใช้เองว่าเคลื่อนสแกนเนอร์ตัดกับบาร์โค้ดในแนวโค้ง หรือสแกนด้วยความเร็วไม่คงที่หรือไม่ ซึ่งข้ออยู่กับการสะบัดของข้อมือ

2.7 ไมโครคอนโทรลเลอร์

ความหมายของไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (microcontroller) เป็นชื่อของอุปกรณ์อิเล็กทรอนิกส์แบบหนึ่งที่บรรจุความสามารถมากมายไม่ว่าจะเป็นหน่วยประมวลผล หน่วยคำนวณทางคณิตศาสตร์และลอจิก วงจรรับสัญญาณอินพุต วงจรขับสัญญาณออกทางเอาต์พุต หน่วยความจำ วงจรกำเนิดสัญญาณนาฬิกา ทำให้ไมโครคอนโทรลเลอร์สามารถนำไปประยุกต์ใช้งานแทนวงจรอิเล็กทรอนิกส์ที่ซับซ้อนได้เป็นอย่างดีโดยช่วยลดจำนวนของอุปกรณ์และขนาดของระบบลง ในขณะที่มีขีดความสามารถสูงขึ้น ภายใต้งบประมาณที่เหมาะสม

ไมโครคอนโทรลเลอร์มาจากคำ 2 คำรวมกันคือ "ไมโคร" (micro) ซึ่งหมายถึง ไมโครโปรเซสเซอร์ (Microprocessor) ซึ่งเป็นอุปกรณ์ประมวลผลข้อมูลขนาดเล็ก ซึ่งภายในประกอบด้วยหน่วยประมวลผลกลางหรือซีพียู (CPU : Central Processing Unit) หน่วยคำนวณทางคณิตศาสตร์และลอจิก (ALU : Arithmetic Logic Unit) วงจรเชื่อมต่อหน่วยความจำ และวงจรเชื่อมต่อสัญญาณนาฬิกาอีกคำหนึ่งคือคำว่า "คอนโทรลเลอร์" (controller) หมายถึงอุปกรณ์ควบคุม ดังนั้นไมโครคอนโทรลเลอร์จึงเป็นอุปกรณ์ที่ใช้ในการควบคุม โดยที่สามารถเขียนโปรแกรมเพื่อกำหนดรูปแบบการควบคุมได้อย่างอิสระ

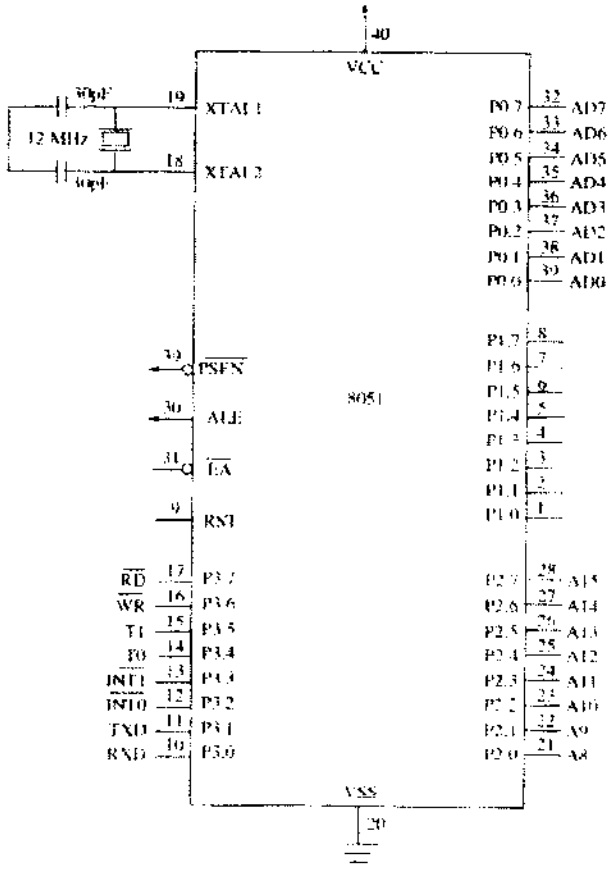
โครงสร้างของ MCS-51

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 มีด้วยกันหลายหมายเลขขึ้นอยู่กับโครงสร้างภายในบางหมายเลขจะมีหน่วยความจำภายในเป็นแบบ ROM บางเบอร์เป็นแบบ EPROM บางหมายเลขมี RAM ภายใน 128 ไบต์ บางเบอร์มี 256 ไบต์ เป็นต้น ซึ่งรายละเอียดจะศึกษาได้จากคู่มือโดยตรง และลักษณะของขาต่าง ๆ จะเหมือนกัน คุณสมบัติที่สำคัญของ MCS-51 มีดังนี้

- มีหน่วยความจำ ROM 4K bytes
- มีหน่วยความจำ RAM 128 bytes
- มีพอร์ต I/O ขนาด 8 บิต 4 พอร์ต
- มี Timer 16 บิต 2 ตัว
- สามารถอินเทอร์รัพท์ได้ 5 แหล่ง
- มีวงจรออสซิลเลเตอร์และวงจรนาฬิกาบนชิป
- มีพอร์ตอนุกรมที่สามารถรับส่งข้อมูลแบบ Full Duplex ความเร็วสูง
- อ้างหน่วยความจำโปรแกรมภายนอกได้ 64K

- อ้างหน่วยความจำข้อมูลทีละบิตได้ 64K
- สามารถประมวลผลทีละบิตได้
- สามารถอ้างหน่ายความจำแบบบิตได้ 210 ตำแหน่ง
- หนึ่งวัฏจักรคำสั่งกินเวลาประมาณ 1 ไมโครวินาที ขณะทำงานด้วย Clock 12 MHz

การจัดขาต่าง ๆ ของ MCS-51



รูปที่ 2.19 แสดงขาต่าง ๆ ของ 8051

ความหมายของขาต่าง ๆ มีดังนี้

1.พอร์ต 0 (Port 0) พอร์ต 0 ได้แก่ขาที่ 32-39 ของ MCS-51 สามารถใช้เป็นอินพุตได้ นอกจากนี้ในการติดต่อกับหน่วยความจำภายนอกยังใช้เป็นขา Address Bus และ Data Bus อีกด้วย

2.พอร์ต 1 (Port 1) พอร์ต 1 ได้แก่ขาที่ 1-8 เป็นพอร์ต 8 บิต สามารถอ้างทีละบิตได้ คือ P1.0, P1.1,...etc

3.พอร์ต 2 (Port 2) พอร์ต 2 ได้แก่ขาที่ 21-28 จะใช้งาน 2 หน้าที่ คือใช้เป็นพอร์ท 8 บิตกับใช้เป็นขาแอดเดรส 8 บิตในการอ้างหน่วยความจำภายนอก

4.พอร์ต 3 (Port 3) พอร์ต 3 ได้แก่ขาที่ 10-17 จะใช้งานสองหน้าที่คือ เป็นพอร์ตอินพุตและเอาต์พุต และใช้เป็นขาควบคุมต่าง ๆ ดังตาราง

ตารางที่ 2.24 แสดงบิตและหน้าที่ต่าง ๆ ของพอร์ต 3

บิต	ชื่อ	หน้าที่พิเศษ
P3.0	RXD	ใช้รับข้อมูลทางพอร์ตอนุกรม
P3.1	TXD	ใช้ส่งข้อมูลทางพอร์ตอนุกรม
P3.2	INT0	อินเทอร์รัพท์ภายนอกหมายเลข 0
P3.3	INT1	อินเทอร์รัพท์ภายนอกหมายเลข 1
P3.4	T0	ตัวจับเวลา / ตัวนับ ตัวที่ 0
P3.5	T1	ตัวจับเวลา / ตัวนับ ตัวที่ 1
P3.6	WR	สัญญาณเขียนข้อมูลหน่วยความจำภายนอก
P3.7	RD	สัญญาณอ่านข้อมูลหน่วยความจำภายนอก

1.PSEN (Program Store Enable) ขา PSEN เป็นขาที่ส่งสัญญาณออกคือขา 29 ขานี้จะแอคทีฟเมื่อ MCS-51 ต้องการอ่าน Code โปรแกรมภายนอก โดยปกติถ้าหน่วยความจำภายนอกเป็น EPROM ขา PSEN จะต่อกับขา Output Enable (OE) ของ EPROM

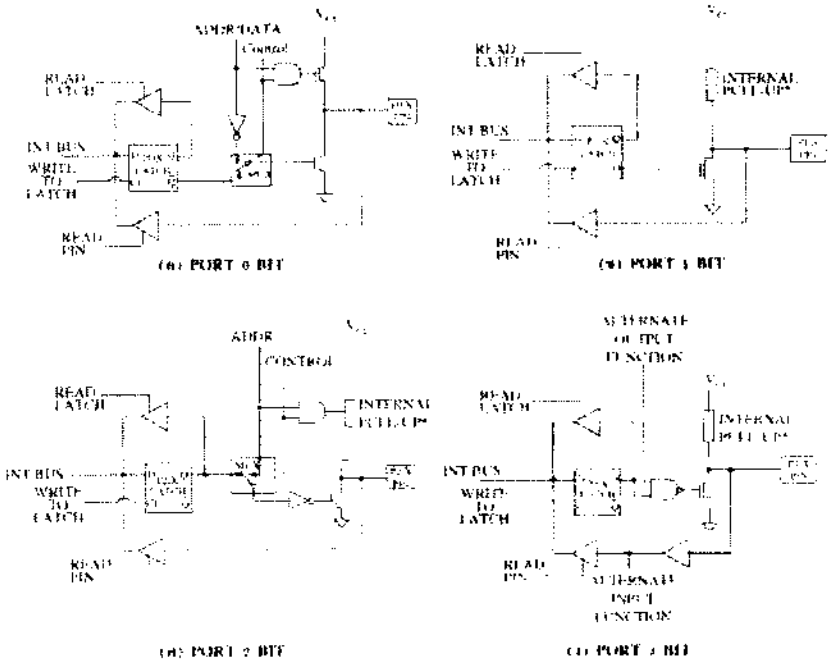
2.ALE (Address Latch Enable) เนื่องจากพอร์ต 0 สามารถใช้เป็นขาอ้างตำแหน่ง และขาข้อมูล MCS-51 จะมีขา ALE ได้แก่ขา 30 ขานี้จะใช้ Multiplex สัญญาณ Address Bus ของ Port 0 ในการใช้งานระบบ MCS-51 นั้น จะต้องมีอุปกรณ์มาต่อกับ Port 0 ที่ทำหน้าที่ latch สัญญาณ Address Bus เมื่อ MCS-51 ต้องการติดต่อกับหน่วยความจำภายนอก MCS-51 จะส่งสัญญาณ Address Bus ออกมาก่อนทาง Port 0 จากนั้นจะส่งสัญญาณ ALE มา latch อุปกรณ์ภายนอก ให้เก็บค่า Address Bus ของ Port 0 ไว้เพื่อใช้ Port 0 เป็น Data Bus ต่อไป

3.EA (External Access) ขา EA ได้แก่ขาที่ 31 ถ้าขานี้เป็นลอจิก "1" จะใช้กับเบอร์ 8051/8052 เพื่อบอกว่าให้อ่านโปรแกรมจากหน่วยความจำโปรแกรมภายใน แต่ถ้าเป็นลอจิก "0" จะบอกว่าให้ MCS-51 ทำโปรแกรมโดย อ่านจากหน่วยความจำโปรแกรมภายนอก (ถ้าขา EA เป็น "0" ขา PSEN จะแอคทีฟ) ถ้าหากเป็นเบอร์ 8031 หรือ 8032 ขา EA จะเป็น "0" เสมอ เพราะว่าไม่มีโปรแกรมหน่วยความจำภายใน แต่ถ้าใช้เบอร์ 8051 / 8052 ซึ่งมีหน่วยความจำโปรแกรมภายในและให้ขา EA เป็น "0" ซึ่งจะ Disabled ROM ภายในและจะอ่านโปรแกรมจาก EPROM ภายนอกแทน

4.RST (Reset) ขา RST ได้แก่ขา 9 จะใช้ในการรีเซต MCS-51 โดยจะให้ขานี้เป็นลอจิก "1" อย่างน้อย 2 Machine Cycles จึงจะรีเซตระบบให้

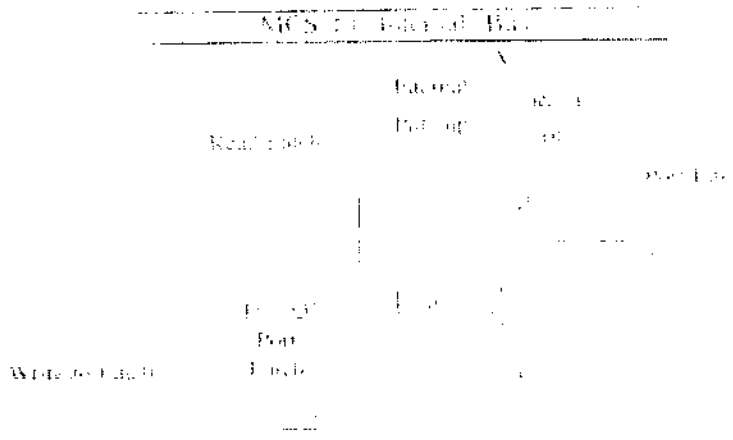
โครงสร้างของพอร์ตอินพุตเอาต์พุต (I/O Port Structure)

ขาของพอร์ตจะแสดงโครงสร้างภายในได้ดังรูปโดยจะมีโครงสร้างเป็น Field-effect Transistor ต่ออยู่กับขาภายนอก และมีความต้านทานต่อ Pull-up อยู่สำหรับพอร์ต 1,2,3 แต่ถ้าเป็นพอร์ท 0 จะไม่มีตัวต้านทาน Pull-up ภายในเพราะว่าต้องใช้เป็นขา Address Bus และ Data Bus



รูปที่ 2.20 โครงสร้างพอร์ททั้ง 4 ของ MCS-51

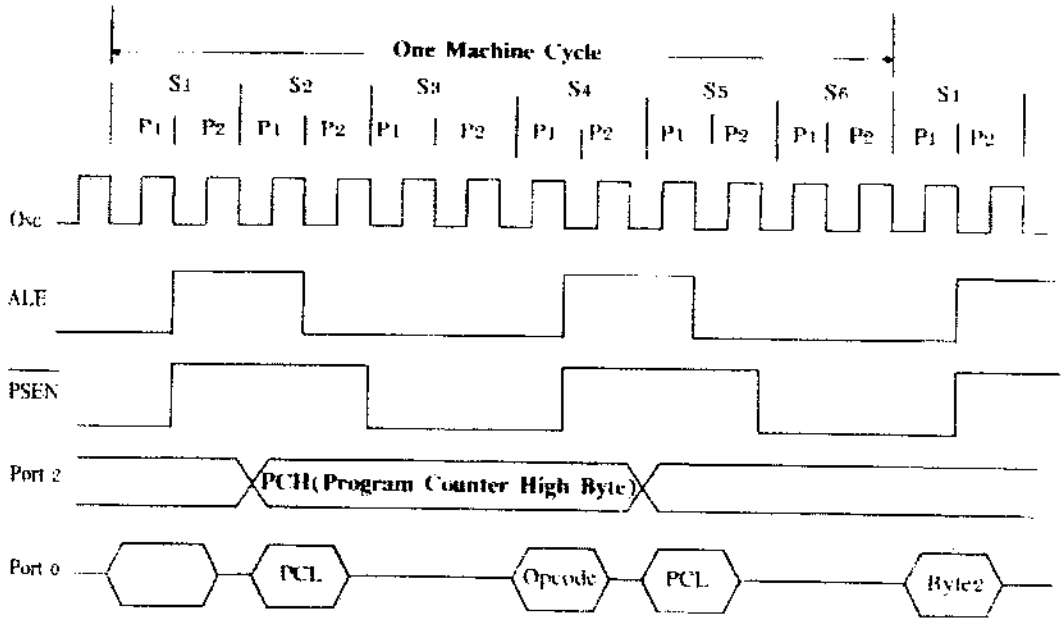
พอร์ตนี้สามารถใช้เป็นอินพุตเอาต์พุตกับอุปกรณ์ภายนอกได้ ในการอ่านข้อมูลจากพอร์ตจะอ่านได้สองแบบคือ Read Latch และ Read Pin โดย Read Latch หมายถึงการอ่านข้อมูลที่ถูก Latch เอาไว้เข้าสู่บัสภายในของ MCS-51 เช่นการทำคำสั่ง CPL P1.5 แต่ถ้าเป็นการ Read Pin จะเป็นการใช้พอร์ตเป็นอินพุต โดยจะอ่านค่าจากขาของไอซีเข้าสู่บัสภายในโดยการอ่านแบบ Read Latch และ Read Pin จะมีสัญญาณมาควบคุมที่บัฟเฟอร์ดังรูปที่ 2.21



รูปที่ 2.21 การต่อพอร์ตเข้ากับระบบบัสภายในของ MCS-51

การติดต่อกับหน่วยความจำโปรแกรมภายนอก

ในการอ่านข้อมูลจากหน่วยความจำโปรแกรมภายนอก MCS-51 จะส่งค่าตำแหน่งของหน่วยความจำออกไปก่อน ซึ่งค่าตำแหน่งจะเก็บอยู่ใน PC โดยส่งออกไปทางพอร์ต 0 และ พอร์ต 2 จากนั้นเวลาต่อมาจะส่งขา ALE ให้เป็นลอจิก "0" เพื่อ Latch ขา address ของ 8 บิตต่ำ คือ พอร์ต 0 จากนั้นจะส่งสัญญาณทางขา PSEN ให้เป็นลอจิก "0" เพื่ออ่านข้อมูลซึ่งจะได้ Opcode เข้าไปทางขา Data Bus คือพอร์ต 0 ไดอะแกรมเวลาการอ่านข้อมูลจากหน่วยความจำภายนอกแสดงได้ดังรูปที่ 2.22



รูปที่ 2.22 ไดอะแกรมเวลาการอ่านข้อมูลจากหน่วยความจำโปรแกรมภายนอก

ชุดคำสั่ง instruction set

โหมดการอ้างแอดเดรส (Addressing Modes)

การติดต่อกับหน่วยความจำในการกระทำคำสั่งต่าง ๆ เรียกว่า Addressing ใน MCS-51 สามารถติดต่อกับหน่วยความจำได้หลายแบบ เช่น ในไบต์ที่ 2 ของชุดคำสั่ง, ในรีจิสเตอร์ R4, ในตำแหน่ง 35 H ของหน่วยความจำภายใน ในตำแหน่งของหน่วยความจำภายนอกที่มีรีจิสเตอร์ DPTR ชี้อยู่ เป็นต้น การติดต่อกับหน่วยความจำจะเป็นการอ่านข้อมูลออกมา หรือเขียนข้อมูลเข้าไปใน MCS-51 สามารถติดต่อกับหน่วยความจำได้ 8 โหมด คือ

- Register
- Direct
- Indirect
- Immediate
- Relative
- Absolute
- Long
- Indexed

Registers Addressing

เป็นการติดต่อข้อมูลที่อยู่ในรีจิสเตอร์โดยตรง ซึ่งจะเป็นรีจิสเตอร์ชุดที่กำลังใช้งานอยู่ คือ $R_n(R_0\text{-}R_7)$ ตัวอย่างเช่น

MOV A , R_0

หมายความว่า ย้ายข้อมูลที่เก็บไว้ใน R_0 มาเก็บไว้ที่ A ถ้าดูรหัสคำสั่งของ MOV A , R_n ซึ่งดูได้จากตารางชุดคำสั่งจะพบว่ารหัสคำสั่งคือ 11101rrr โดยที่ rrr จะหมายถึงค่ารีจิสเตอร์ R_n ส่วน 11101 5 บิตบนจะเป็น opcode บอกว่าให้ย้ายข้อมูลจากความหมายของ 3 บิตถัดไปมาเก็บใน A

ถ้าหากจะบวกข้อมูลในรีจิสเตอร์ A กับข้อมูลใน R_7 จะเขียนคำสั่งได้ดังนี้

ADD A , R_7

นอกจากการติดต่อกับ R_n แล้วยังสามารถติดต่อกับรีจิสเตอร์พิเศษได้อีกด้วย เช่น Accumulator จะใช้ "A" Data Pointer จะใช้ "DPTR" Program Counter จะใช้ "PC" Carry Flag จะใช้ "C" Accumulator-B Register Pair จะใช้ AB ตัวอย่างเช่น

INC DPTR

จะเป็นการบวกค่า 1 กับ 16 – Bit Fata Pointer

Direct Addressing

เป็นการกำหนดตำแหน่งของหน่วยความจำโดยตรง ซึ่งจะเป็นการติดต่อกับหน่วยความจำภายในของ MCS – 51 เท่านั้น เช่น

```
MOV  A , 35H  ; ย้ายข้อมูลที่เก็บในตำแหน่ง 35H มาเก็บใน A
ADD  A , 20H  ; บวกข้อมูลที่เก็บในตำแหน่ง 20H กับข้อมูลใน A
```

Indirect Addressing

การอ้างตำแหน่งแบบนี้เป็นการอ้างตำแหน่งโดยทางอ้อม ซึ่งจะใช้รีจิสเตอร์ R_0 และ R_1 เป็นตัวชี้ข้อมูลโดยการเขียนรหัส Mnemonic จะใช้เครื่องหมาย @ นำหน้า เช่น

```
MOV  A , @R0
```

การเขียนแบบนี้ หมายความว่าย้ายข้อมูลจากตำแหน่งที่ R_0 ชี้อยู่มาเก็บในรีจิสเตอร์ A ถ้าในรีจิสเตอร์ R_1 เก็บค่า 40H และในตำแหน่ง 40H ของหน่วยความจำภายในเก็บค่า 55H เอาไว้ แล้วเขียนคำสั่งดังนี้

```
MOV  A , @R1
```

ผลลัพธ์ที่ได้จะเป็นการนำค่า 55H ไปใส่ในรีจิสเตอร์ A

Immediate Addressing

วิธีนี้เป็นการกำหนดค่าของข้อมูลโดยตรง โดยจะใช้เครื่องหมาย # นำหน้าตัวเลข เช่น

```
MOV  A , #12
```

เป็นการกำหนดค่า 12 ให้กับรีจิสเตอร์ A หรือค่า 0CH ในระบบเลขฐานสิบหก นอกจากนี้ยังสามารถกำหนดค่าให้กับรีจิสเตอร์ 16 บิตได้ ซึ่งคำสั่งนี้จะมีขนาด 3 ไบต์ เช่น

```
MOV  DPTR , #8000H
```

Relative Addressing

การอ้างตำแหน่งแบบนี้มักจะใช้กับคำสั่งกระโดด โดยค่า Relative Address (Offset) จะมีขนาด 8 บิต ซึ่งสามารถกระโดดกลับหลังหรือไปข้างหน้าได้ -128 ถึง +127 ตำแหน่ง ชุดคำสั่งจะมีลักษณะ 2 ไบต์

Absolute Addressing

เป็นการอ้างตำแหน่งเมื่อทำคำสั่ง ACALL และ AJMP ซึ่งเป็นคำสั่งที่มีขนาด 2 ไบต์ โดยตำแหน่งที่จะกระโดดอยู่ในช่วง 2K ตำแหน่ง ซึ่งจะใช้หน่วยความจำในการเก็บตำแหน่ง 11 บิต โดยในไบต์ของ Opcode จะเก็บค่า A8-A10 ส่วนไบต์ที่ 2 จะเก็บค่า A0-A7

Long Addressing

เป็นการอ้างตำแหน่งเมื่อทำคำสั่ง LCALL และ LJMP โดยชุดคำสั่งนี้ใช้เนื้อที่เก็บ 3 ไบต์ โดยไบต์ที่ 2 และ 3 ใช้ในการอ้างตำแหน่งขนาด 16 บิต ซึ่งเท่ากับ 64 K byte การอ้างตำแหน่งแบบนี้สามารถอ้างได้ตลอดที่ MCS-51 มีหน่วยความจำภายนอกต่ออยู่

Indexed Addressing

การอ้างตำแหน่งแบบนี้ เป็นการติดต่อกับหน่วยความจำภายนอกในลักษณะการเปิดตาราง โดยใช้รีจีสเตอร์ พิเศษ (Program Counter หรือ Data Pointer) เป็นตัวชี้ตำแหน่งเริ่มต้นและบวกด้วยค่า Offset ซึ่งเก็บไว้ในรีจีสเตอร์ A (Accumulator) ซึ่งจะใช้กับชุดคำสั่ง MOVC หรือ JMP
ตัวอย่างเช่น

MOVC A , @A+PC

เป็นการอ่านข้อมูลจากตำแหน่งที่ PC ชี้อยู่บวกกับค่าใน A มาเก็บในรีจีสเตอร์ A ถ้า PC ชี้อยู่ที่ 1000H และ A = 3 จะเป็นการอ่านค่าจากตำแหน่ง 1003H มาเก็บในรีจีสตอร์ A

ประเภทของชุดคำสั่ง

ใน MCS – 51 จะแบ่งชุดของคำสั่งได้ดังนี้

ชุดคำสั่งทางคณิตศาสตร์ (Arithmetic Instructions)

เป็นกลุ่มคำสั่งที่ทำงานด้านคณิตศาสตร์ ซึ่งจะเกี่ยวข้องกับรีจีสเตอร์ A และมักจะใช้รีจีสเตอร์ A เก็บผลลัพธ์ เช่นคำสั่ง ADD A คำสั่งนี้จะอ้างตำแหน่งได้หลายแบบ เช่น

ADD A , 7FH (Direct Addressing)

ADD A , @R0 (Indirect Addressing)

ADD A , R7 (Register Addressing)

ADD A , #35H (Immediate Addressing)

นอกจากนี้ MCS-51 ยังมีคำสั่งที่ใช้เพิ่มค่า และลดค่าในหน่วยความจำภายในได้โดยตรง โดยใช้การอ้างตำแหน่งแบบ Direct Addressing เช่น ถ้าหน่วยความจำภายในตำแหน่ง 7FH เก็บค่า 40H ไว้ถ้ามีคำสั่ง

INC 7FH

ค่าในหน่วยความจำ 7FH จะมีค่าเป็น 41H

สำหรับคำสั่งคูณ เช่น MUL AB จะเป็นการคูณเลขที่อยู่ในรีจีสเตอร์ A กับเลขที่เก็บไว้ในรีจีสเตอร์ B ซึ่งผลลัพธ์ที่ได้จะมีขนาด 16 บิต โดยจะเก็บค่าไบต์สูงที่รีจีสเตอร์ B และเก็บค่าไบต์ต่ำในรีจีสเตอร์ A ถ้าเป็นคำสั่งหาร เช่นคำสั่ง DIV AB จะเป็นการหาค่าใน A ด้วยค่าใน B โดยผลลัพธ์

ที่ได้จะเก็บใน A และเศษที่เหลือจะเก็บใน B เช่นถ้าใน A มีคาเท่ากับ 25(19H) และค่าใน B มีค่า 6(6H) ถ้าถูกกระทำด้วยคำสั่ง

DIV AB

ผลลัพธ์ที่ได้จะเป็น 4 เหลือเศษ 1 โดยค่า 4 จะถูกเก็บใน A และ B จะเก็บค่า 1

ชุดคำสั่งทางลอจิก (logical Instructions)

MCS-51 มีคำสั่งกระทำทางลอจิกซึ่งจะคล้ายกับ Boolean Operations (AND , OR, Exclusive OR และ NOT) ซึ่งสามารถกระทำแบบไบต์ หรือ บิตต่อบิตได้ ถ้าค่าใน A มี 00110101B และทำคำสั่ง AND สามารถเขียนได้ดังนี้

ANL A , #01010011B

ผลลัพธ์ที่ได้จะเก็บไว้ใน A

ชุดคำสั่งทางลอจิกสามารถอ้างตำแหน่งได้หลายแบบ พิจารณาชุดคำสั่งต่อไปนี้ ซึ่งจะกระทำการ AND ทางลอจิก

ANL A , 55H (Direct Addressing)

ANL A , @R0 (Indirect Addressing)

ANL A , R6 (Register Addressing)

ANL A , #33H (Immediate Addressing)

คำสั่งทางลอจิกส่วนมากจะใช้รีจีสเตอร์ A กับค่าต่าง ๆ และผลลัพธ์ที่ได้เก็บใน A

กลุ่มคำสั่งการโอนย้ายข้อมูล (Data Transfer Instructions)

หน่วยความจำข้อมูลภายใน การโอนย้ายข้อมูลของหน่วยความจำภายในจะใช้ 1 หรือ 2 Machine Cycle รูปแบบของคำสั่งจะเป็น

MOV <Destination > , < Source>

โดยจะเป็นการโอนย้ายข้อมูลของ 2 ตำแหน่งจากหน่วยความจำภายใน นอกจากนี้ยังมีคำสั่งสลับข้อมูล เช่น

XCH A , <Source>

ซึ่งจะสลับข้อมูลที่เก็บใน A กับข้อมูลในตำแหน่งที่กำหนด นอกจากนี้ยังมีคำสั่งสลับข้อมูลแบบ "Digit" เช่น

XCHD A , @Ri

เป็นการสลับข้อมูลของ 4 บิตล่าง

หน่วยความจำข้อมูลภายนอก (External RAM)

การย้ายข้อมูลระหว่างหน่วยความจำข้อมูลภายในกับหน่วยความจำภายนอก จะใช้การอ้างตำแหน่งแบบ Indirect Addressing โดยใช้ @Ri เป็นตัวชี้ตำแหน่ง (เมื่อ Ri คือ R0 และ R1 ของ Bank ที่กำลังทำงาน) หรือใช้ @DPTR เป็นตัวชี้ซึ่งจะอ้างตำแหน่งแบบ 16 บิต หรืออ้างได้ 64K

การเปิดตาราง (Look-up Tables)

การเขียนโปรแกรมแบบเปิดตารางจะใช้ในการติดต่อกับหน่วยความจำโปรแกรม โดยคำสั่งที่ใช้คือ MOVC หรือ "Move Constant" การใช้คำสั่งนี้จะใช้ Program Counter หรือ Data-pointer ในการอ้างตำแหน่ง และใช้รีจิสเตอร์ หรือ Accumulator เป็นค่า Offset เช่นชุดคำสั่ง

MOVC A , @A+DPTR

โดยค่าของ DPTR จะชี้ที่ตำแหน่งเริ่มต้นของตาราง และ ค่ารีจิสเตอร์ A จะเป็นค่า Offset ถ้า DPTR มีค่า 2000H และ A มีค่า 08H จะเป็นการอ่านค่าจากตำแหน่ง 2008H มาใส่ในรีจิสเตอร์ A

Boolean Instructions

MCS-51 มีคำสั่งที่ทำงานแบบ Boolean ได้ ซึ่งประกอบด้วยชุดคำสั่ง Set, Clear, Complement, OR และ AND โดยจะเป็นการกระทำแบบบิตต่อบิต โดยตำแหน่ง 00H – 7FH และตำแหน่ง 80H – FFH ใน SFR จะเป็นการอ้างตำแหน่งแบบ Direct Addressing สำหรับตำแหน่งไบต์ที่ 20H – 2FH สามารถอ้างตำแหน่งเป็นตำแหน่งบิตได้ โดยเริ่มตั้งแต่บิต 0 ของตำแหน่ง 20H (บิต 00H) ถึงบิต 7 ของตำแหน่ง 2FH (บิต 7FH) นอกจากนี้ใน SFR บางตำแหน่งก็สามารถอ้างแบบบิตได้

ชุดคำสั่งกระโดดเรียกโปรแกรมย่อย

ชุดคำสั่งกระโดด เป็นชุดคำสั่งที่ทำให้โปรแกรมไปทำคำสั่งที่ตำแหน่งที่กำหนด คำสั่งกระโดดแบ่งออกได้ 2 ชนิดคือ

คำสั่งกระโดดแบบไม่มีเงื่อนไข

คำสั่ง AJMP เป็นคำสั่งที่มีขนาด 2 ไบต์ โดยจะกระโดดไปยังตำแหน่งที่กำหนด ซึ่งตำแหน่งที่ใช้จะเป็นเลขฐานสองแบบ 11 บิต ทำให้สามารถกระโดดไปได้ 2024 ตำแหน่ง หรือ 2K ตำแหน่ง รูปแบบคำสั่งเป็นดังนี้

AJMP ADDR

คำสั่ง LJMP เป็นคำสั่งกระโดดที่มีขนาด 3 ไบต์ โดยไบต์แรกจะเป็น Operation Code ส่วนอีกสองไบต์จะใช้อ้างตำแหน่งที่จะกระโดด ซึ่งอ้างได้ 16 บิต ทำให้สามารถกระโดดได้ 64K

ตำแหน่ง หรืออาจเรียกได้ว่าจะกระโดดไปยังตำแหน่งใด ๆ ก็ได้ที่ MCS-51 สามารถอ้างถึง รูปแบบคำสั่งเป็นดังนี้

LJMP ADDR

คำสั่ง SJMP เป็นคำสั่งกระโดดขนาด 2 ไบต์ โดยไบต์แรกจำเป็น Operation Code ส่วนไบต์ที่เหลือจะใช้อ้างตำแหน่งแบบ Relative ซึ่งสามารถกระโดดไปข้างหน้าได้ +127 ไบต์และกระโดดไปข้างหลังได้ –128 ไบต์

คำสั่ง JMP @A+DPTR คำสั่งนี้จะใช้กับการกระโดดแบบเปิดตาราง โดยให้ DPTR ชี้ตำแหน่งเริ่มต้นของตาราง และใส่ค่าที่ต้องการในตารางไว้ในรีจีสเตอร์ A โปรแกรมจะกระโดดไปยังตำแหน่งที่เกิดจากผลรวมของ DPTR กับ A

คำสั่งกระโดดแบบมีเงื่อนไข ชุดคำสั่งนี้จะกระโดดไปยังตำแหน่งใด ๆ แบบ Relative โดยจะมีการตรวจสอบเงื่อนไขก่อนทำคำสั่ง ถ้าเงื่อนไขไม่จริงจะทำคำสั่งถัดไป ชุดคำสั่งเหล่านี้ได้แก่

JZ Rel Jump on Zero คำสั่งนี้จะกระโดดไปตำแหน่ง Rel ถ้าค่าในรีจีสเตอร์ A เป็น 0

JNZ Rel Jump on not Zero คำสั่งนี้จะกระโดดไปตำแหน่ง Rel ถ้าค่าในรีจีสเตอร์ A ไม่เป็น 0

DJNZ <byte>, Rel Decrement and Jump if not Zero คำสั่งนี้จะลดค่าในตำแหน่ง byte ลงหนึ่ง ถ้า ค่าใน byte ยังไม่เป็น 0 จะกระโดดไปตำแหน่ง Rel แต่ถ้าค่าเป็น 0 จะทำคำสั่งถัดไป

CJNE A, <byte>, Rel Compare and Jump if not Equal รำสั่งนี้จะเปรียบเทียบค่าในรีจีสเตอร์ A กับค่าในตำแหน่ง byte ถ้าไม่เท่ากันจะกระโดดไปตำแหน่ง Rel

CJNE <byte>, #Data, Rel รำสั่งนี้จะเปรียบเทียบข้อมูล #Data กับข้อมูลในตำแหน่ง byte ถ้าไม่เท่ากันจะกระโดดไปตำแหน่ง Rel

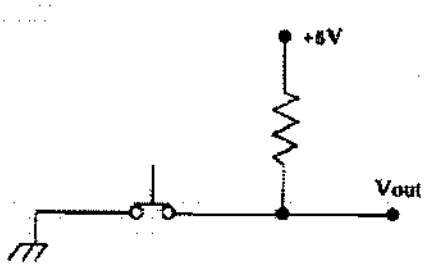
กลุ่มคำสั่งเรียกโปรแกรมย่อย คำสั่งเรียกโปรแกรมย่อยจะใช้คำสั่ง CALL โดยจะเรียก CALL ไปยังตำแหน่งที่เก็บโปรแกรมย่อยนั้นอยู่ โดยคำสั่ง CALL มีสองคำสั่งดังนี้

ACALL เป็นคำสั่งแบบ 2 ไบต์ โดยจะใช้การอ้างตำแหน่งโปรแกรมย่อย 11 บิต ซึ่งโปรแกรมย่อยจะอยู่ห่างได้ไม่เกิน 2K ตำแหน่ง

LCALL เป็นคำสั่งแบบ 3 ไบต์ โดยใช้การอ้างตำแหน่งโปรแกรมย่อย 16 บิต ซึ่งสามารถเขียนโปรแกรมย่อยได้ทุกตำแหน่งที่อยู่ในหน่วยความจำโปรแกรมของ MCS-51

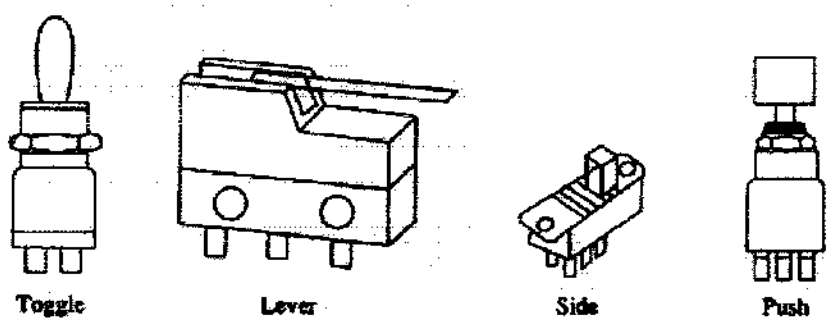
การเชื่อมต่อสวิตช์กับระบบไมโครคอนโทรลเลอร์

สวิตช์หรือ Keyboard เป็นอุปกรณ์อินพุตพื้นฐานที่ระบบไมโครคอนโทรลเลอร์สามารถรับข้อมูลได้ในที่นี้จะกล่าวถึงการต่อสวิตช์แบบต่าง ๆ ตั้งแต่มีจำนวนน้อยตัวจนถึงการต่อสวิตช์หลาย ๆ ตัวแบบเมทริกซ์การสร้างสวิตช์ให้กับระบบไมโครคอนโทรลเลอร์นั้น อาจทำได้ดังรูปที่ 2.23 โดยการต่อเข้ากับแต่ละบิตของพอร์ตอินพุต ถ้าสวิตช์ ON จะให้ลอจิก "0" แต่ถ้าสวิตช์ OFF จะให้ลอจิก "1"

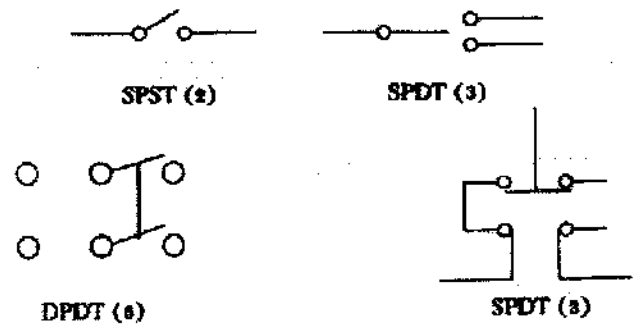


รูปที่ 2.23 แสดงการสร้างลอจิกจากสวิตช์

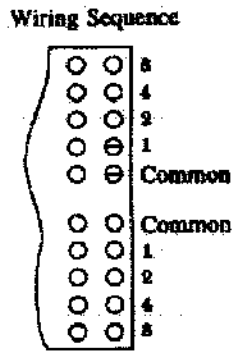
สวิตช์ที่นิยมใช้กันมีหลายชนิดดังรูปที่ 2.24 แต่ละแบบอาจแบ่งได้จากจำนวนหน้าสัมผัสและจำนวนขั้นของมัน ได้แก่ แบบSingle-pole/Single-throw (SPST), Single-pole/Double-throw(SPDT), Double-pole/Double-throw (DPDT) เป็นต้น สัญลักษณ์แสดงได้ดังรูปที่ 2.25 นอกจากนี้ยังมีสวิตช์ที่มีโครงสร้างภายในเป็นแบบหมุน ซึ่งจะให้สัญญาณออกมาเป็นรหัส BCD ได้เลย เรียกว่า Thumbwheel Switch การใช้สวิตช์แบบนี้จะต้องป้อนสัญญาณเข้าที่ขา Common ดังรูปที่ 2.26 ในที่นี้จะกล่าวถึงการต่อสวิตช์แบบ ON-OFF เข้ากับระบบไมโครคอนโทรลเลอร์



รูปที่ 2.24 แสดงตัวอย่างสวิตช์ที่นิยมใช้กัน



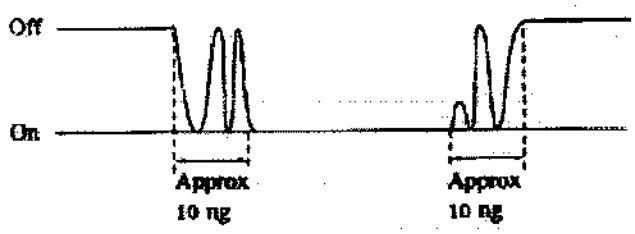
รูปที่ 2.25 แสดงสัญลักษณ์ของสวิตช์แบบต่าง ๆ



Wheel Printing		0	1	2	3	4	5	6	7	8	9
Output	8	0	0	0	0	0	0	0	0		
	4	0	0	0	0					0	0
	2	0	0			0	0			0	0
	1	0		0		0		0		0	
	Common	0	0	0	0	0	0	0	0	0	0
	Common	x	x	x	x	x	x	x	x	x	x
	1		x		x		x		x		x
	2			x		x		x	x		
	4					x	x	x	x		
	8									x	x

รูปที่ 2.26 แสดงไบนารีสวิตช์

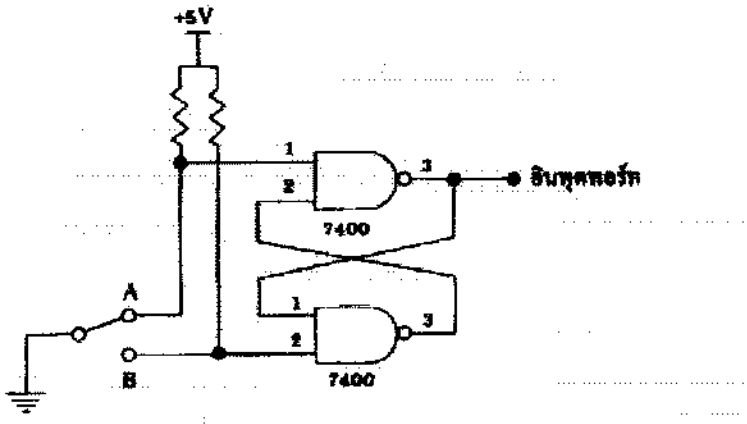
การต่อสวิตช์เข้ากับระบบไมโครคอนโทรลเลอร์จะต้องให้หน้าสัมผัสของสวิตช์สะอาดอยู่เสมอพิจารณาตามรูปที่ 2.23 ถ้าสวิตช์ไม่มีการกดสัญญาณที่ได้จะมีค่าเป็น "1" ถ้าสวิตช์ทีการกดข้อมูลที่ได้จะมีค่าเป็น "0" แต่โดยทั่วไปแล้วการเปลี่ยนระดับสัญญาณจาก "1" เป็น "0" จะมีสัญญาณที่ไม่ต้องการเกิดขึ้นดังรูปที่ 2.27 ซึ่งเกิดจากการสั่นของหน้าสัมผัสของสวิตช์ ทำให้เกิดการแกว่งของสัญญาณซึ่งเรียกว่า บาวนซ์ (Bounce) อยู่ชั่วระยะหนึ่ง โดยปกติจะมีเวลาประมาณ 5 ถึง 50 มิลลิวินาที ดังนั้นสิ่งที่ MCS-51 ได้รับจากการกดสวิตช์จะไม่ใช่สัญญาณหนึ่งลูก แต่เป็นสัญญาณหลาย ๆ ลูก ซึ่ง MCS-51 อาจได้รับข้อมูลผิดพลาด การแก้ปัญหานี้เรียกว่า การทำ Debounce ซึ่งอาจทำได้ 2 วิธีคือ Hardware Debounce และ Software Debounce



รูปที่ 2.27 แสดงสัญญาณที่เกิดจากการสั่นของหน้าสัมผัสของสวิตช์

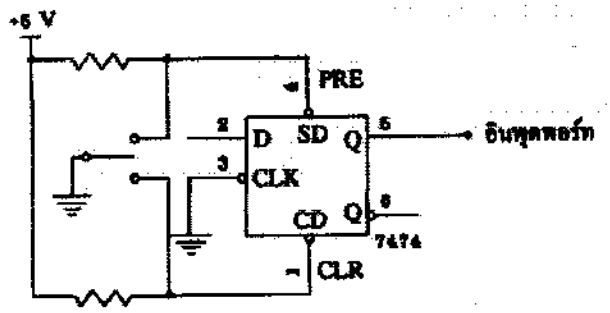
Hardware Debounce

การแก้โดยวิธีทางฮาร์ดแวร์วิธีง่าย ๆ วิธีหนึ่งคือการใช้ แนนเกต มาต่อเป็น R – S ฟลิปฟลอป หรืออาจใช้ไอซีเบอร์ 74LS279 ก็ได้ พิจารณารูปที่ 2.28 เมื่อสวิตช์ถูกโยกมาที่ตำแหน่ง A ขาอินพุตของแนนเกตตัวบนจะได้รับลอจิก "0" ซึ่งจะทำให้เอาต์พุตของเกตตัวบนมีลอจิกเป็น "1" เอาต์พุตนี้จะถูกนำไปใช้งาน ขณะเดียวกันก็จะป้อนกลับมาให้กับอินพุตของแนนเกตตัวล่าง ทำให้เอาต์พุตของแนนเกตตัวล่างมีค่าเป็น "0" เมื่อมีการแกว่งของสัญญาณที่จุด A แนนเกตตัวบนจะได้รับลอจิกอินพุตที่เปลี่ยนแปลงแต่ไม่มีผลทำให้เอาต์พุตเปลี่ยนแปลง



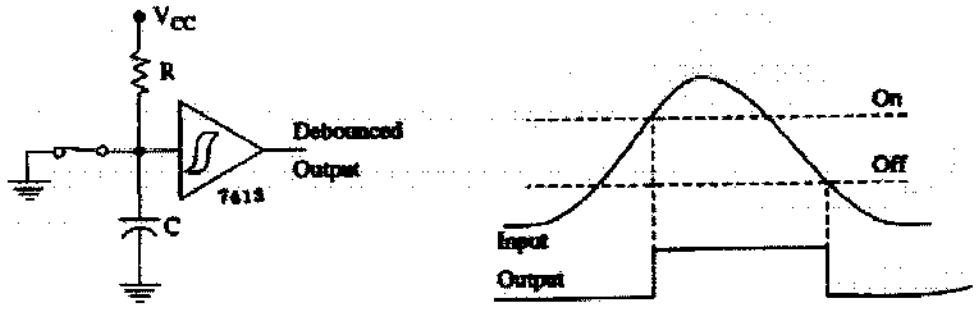
รูปที่ 2.28 แสดงการนำแนนเกตมาต่อเป็น R-S ฟลิปฟลอป เพื่อแก้การสั่นของสวิตช์

การแก้ปัญหาอีกวิธีหนึ่งอาจนำเอา D ฟลิปฟลอป มาช่วยดังแสดงในรูปที่ 2.29 โดยใช้ไอซีเบอร์ 7447 โดยให้อินพุตจากสวิตช์ต่อเข้าที่ขา Preset และ Clear ถ้าขา Preset เป็น "0" เอาต์พุตจะเป็น "1" ถ้ามีการแกว่งของสัญญาณเกิดขึ้นจะไม่มีผลต่อเอาต์พุต และเมื่อสวิตช์ถูกทำให้ Clear เป็น "0" เอาต์พุตที่ได้จะเป็นลอจิก "0"



รูปที่ 2.29 การใช้ฟลิปฟลอปมาแก้ปัญหาสวิตช์

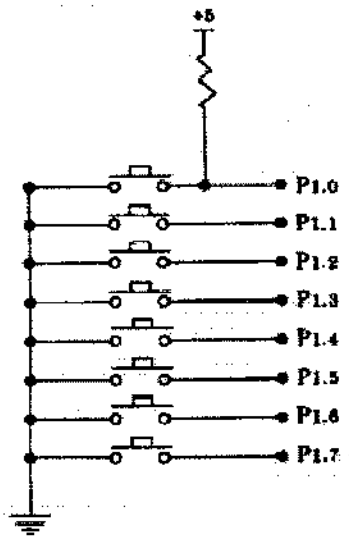
ถ้าใช้สวิตช์แบบ SPST อาจแก้ได้โดยใช้ D ฟลิปฟลอป โดยจะได้สัญญาณอินพุตเมื่อมี Clock เข้าไปหรืออาจใช้ Schmitt Trigger ถ้าสัญญาณอินพุตจากสวิตช์มีลักษณะไม่เหมาะกับระบบดิจิตอล ดังรูปที่ 2.30



รูปที่ 2.30 การแก้ปัญหาสวิตช์โดยใช้ Schmitt Trigger

การต่อสวิตช์เข้ากับพอร์ตของ MCS-51 โดยตรง

ตามที่ทราบมาแล้วว่า MCS-51 มีพอร์ตให้ใช้งานอยู่หลายพอร์ต ในที่นี้จะยกตัวอย่างการต่อสวิตช์ 8 ตัวเข้ากับพอร์ต 1 ดังแสดงในรูปที่ 2.31 ระบบนี้เป็นการต่อสวิตช์แบบง่ายที่สุดเหมาะสำหรับระบบที่ไม่ต้องการสวิตช์มากนัก สำหรับการเขียนโปรแกรมจะต้องอ่านค่าจากสวิตช์คือ อ่านค่าจากพอร์ต 1 จากนั้นต้องเขียนโปรแกรมตรวจสอบการสั่นของสวิตช์ เมื่อได้ค่าจากการกดสวิตช์แน่นอนแล้วก็ทำการตรวจสอบว่าค่าที่อ่านได้นั้นเป็นค่าจากกการกดสวิตช์ใด



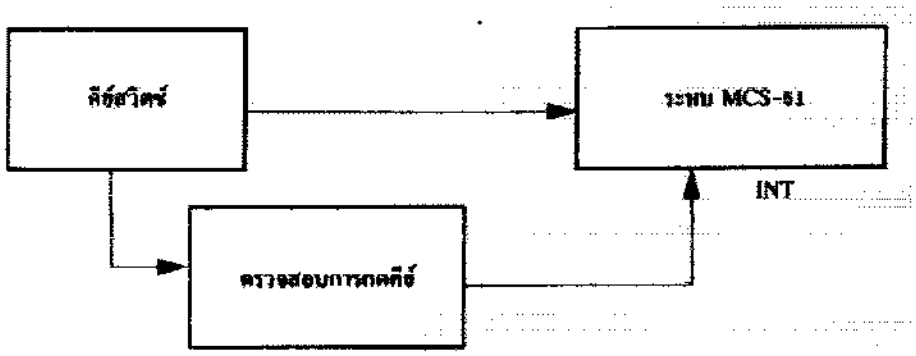
รูปที่ 2.31 แสดงการต่อสวิตช์เข้ากับพอร์ต P1

ต่อไปจะยกตัวอย่างการเชื่อมต่อสวิตช์เข้ากับพอร์ตของ 8255 โดยใช้สวิตช์แบบดิปสวิตช์เชื่อมต่อกับพอร์ท A ของ 8255 การเขียนโปรแกรมอย่างง่าย ๆ อาจทำได้โดยอ่านค่าทางพอร์ต A แล้วทำการตรวจสอบว่าสวิตช์ใด ON โดยนำค่าที่อ่านได้สั่งให้โปรแกรมประมวลผล ซึ่งการเขียนโปรแกรมอาจทำได้หลายวิธีในที่นี้จะยกตัวอย่างง่าย ๆ โดยอ่านค่าเข้ามาแล้วตรวจสอบว่า สวิตช์แรก ON หรือไม่ ถ้าไม่ใช้ตรวจสวิตช์ที่สอง ทำไปเรื่อย ๆ จนครบ สมมติว่าถ้าเรามีโปรแกรมต่าง ๆ หลาย ๆ โปรแกรมและเก็บโปรแกรมเหล่านั้นเป็นโปรแกรมย่อย จากนั้นจะเลือกทำโปรแกรมต่าง ๆ โดยการกดสวิตช์ เราอาจเชื่อมต่อสวิตช์เข้ากับพอร์ต A ของ 8255 จากนั้นเขียนโปรแกรมให้อ่านค่าจากสวิตช์เพื่อเลือกทำโปรแกรมย่อยต่าง ๆ อาจเขียนได้ดังนี้

```
PORT_A      EQU   0FC00H
MAIN:       ACALL APORT CJNE A,#01,KEY1
            ACALL PRO1
            SJMP  MAIN
KEY1:       CJME  A,#02,KEY2
            ACALL PRO2
            SJMP  NAIN
KEY2:       CJNE  A,#04,KEY3
            ACALL PRO3
            SJMP  MAIN
```

```
.....
.....
KEY7:       CJNE A,#128,MAIN
            ACALL PRO7
            SJMP MAIN
;*****************************************
APORT:      MOV  DPTR,#PORT_A
            MOVX A,@DPTR
            RET
;*****************************************
```

การเชื่อมต่อสวิตช์และการเขียนโปรแกรมแบบที่ผ่านมาเราต้องเขียนโปรแกรมวนลูปเพื่อตรวจสอบการกด KEY ตลอดเวลา ซึ่งจะทำให้ระบบไมโครคอนโทรลเลอร์ต้องเสียเวลาส่วนใหญ่ในการตรวจสอบการกดเราอาจต่อสวิตช์อีกแบบได้โดยวิธีอินเทอร์รัพท์ (Interrupt) ในภาวะปกติให้ไมโครคอนโทรลเลอร์ทำงานตามปกติไป เช่น อาจทำโปรแกรมแสดงผลต่าง ๆ ถ้ามีการกด KEY จะมีสัญญาณไปอินเทอร์รัพท์ MCS-51 จากนั้น MCS-51 จะไปทำโปรแกรมตอบสนองการอินเทอร์รัพท์ต่าง ๆ ตามต้องการ ซึ่งอาจต่อวงจรได้ดังรูปที่ 2.32 ถ้ามีการกดสวิตช์จะมีสัญญาณมาอินเทอร์รัพท์ MCS-51 จากนั้น MCS-51 จะทำโปรแกรมตอบสนองการอินเทอร์รัพท์คือ อ่านค่าจากพอร์ตที่สวิตช์ต่ออยู่ จากนั้นจะตรวจสอบว่าสวิตช์ใดกด จากนั้นไปทำโปรแกรมย่อยต่าง ๆ ต่อไป แต่อย่าลืมว่าถ้าเป็นสวิตช์แบบกดติดปล่อยดับจะต้องมีการแก้การสั่นของสวิตช์ด้วย



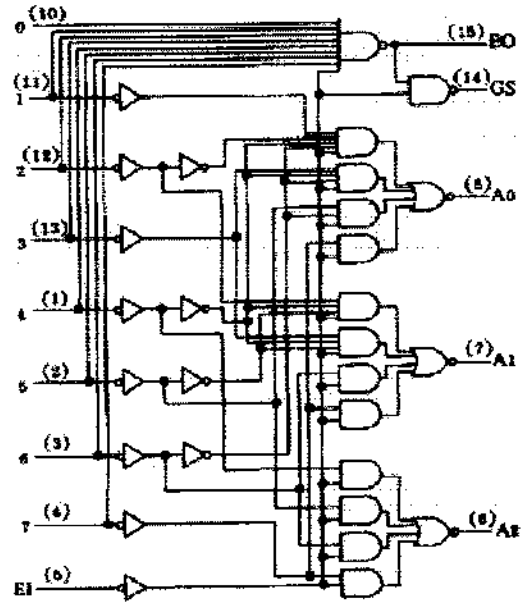
รูปที่ 2.32 แสดงไดอะแกรมการใช้อินเทอร์รัพท์

การต่อสวิตช์เป็นจำนวนมาก

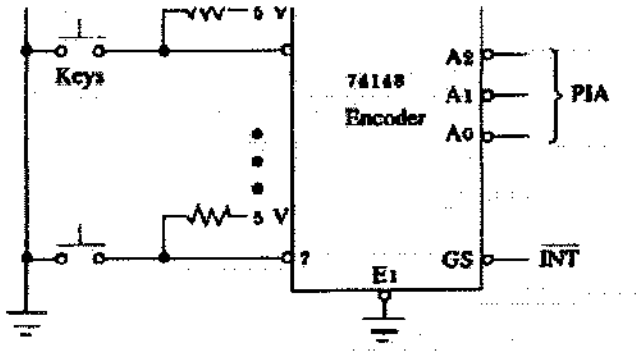
จากที่กล่าวมาข้างต้นจะเห็นว่าสวิตช์หนึ่งตัวจะใช้พอร์ท 1 บิต ดังนั้นพอร์ตหนึ่งพอร์ตจะต่อสวิตช์ได้ 8 ตัว ถ้าหากระบบของเราต้องการสวิตช์มากกว่านี้อาจออกแบบได้หลายวิธี ในที่นี้จะยกตัวอย่างการเข้ารหัสสวิตช์และการต่อสวิตช์แบบเมทริกซ์

การเข้ารหัสสวิตช์ (Encoding Switches) ในที่นี้จะยกตัวอย่างการใช้ไอซีเข้ารหัสเบอร์ 74LS148 ที่ทำงานเป็นแบบ 8 To 3 ซึ่งเป็นไอซีเข้ารหัสเป็นเลขไบนารี 3 บิตถ้าเราให้อินพุตขาใดขาหนึ่งเป็นลอจิก "0" จะให้เอาต์พุตเป็นไบนารีที่มีค่าสัมพันธ์กับขานั้น การทำงานของไอซีเบอร์นี้ดูได้ดังรูปที่ 2.33 การใช้ไอซีเบอร์นี้มาสร้างเป็นวงจรเข้ารหัสสวิตช์เราสามารถนำเอาต์พุตไบนารีต่อโดยตรงเข้ากับพอร์ทได้และอินพุตได้เลยในขณะเดียวกันถ้ามีการกดสวิตช์ใดสวิตช์หนึ่งสัญญาณที่ขา GS จะเป็นลอจิก "0" ทันทีซึ่งเราสามารถนำสัญญาณนี้ไปอินเทอร์รัพท์ MCS-51 ได้ ดังนั้นการใช้ไอซีเบอร์นี้สามารถเขียนโปรแกรมได้ทั้งแบบ Polling และแบบ Interrupt การต่อดีย์สวิตช์เข้ากับไอซีเบอร์นี้แสดงได้ดังรูปที่ 2.34

Input									Outputs				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	X	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H



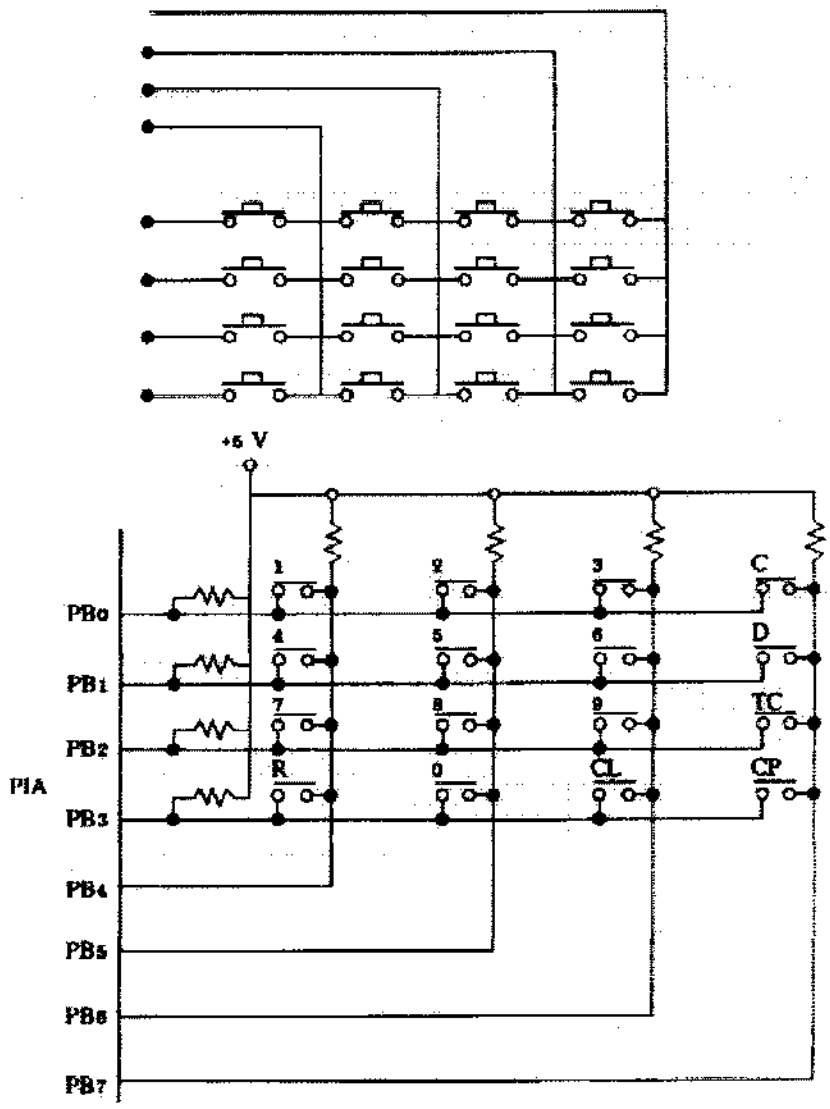
รูปที่ 2.33 แสดงลักษณะไอซีเบอร์ 74LS148



รูปที่ 2.34 การเชื่อมต่อสวิตช์เข้ากับ 74LS148

การต่อสวิตช์แบบเมทริกซ์

หลายคนคงเคยเห็นคีย์สวิตช์แบบ Matrix ที่เรียกว่า Keypad การต่อสวิตช์แบบนี้จะใช้พอร์ท 2 พอร์ตคือ พอร์ตเอาต์พุตและพอร์ทอินพุต ในที่นี้จะยกตัวอย่างการต่อ Keypad ขนาด 4 * 4 Key เข้ากับพอร์ท C ของ 8255 ซึ่งโปรแกรมให้พอร์ต C ล่างเป็นพอร์ตอินพุตและพอร์ต C บนเป็นพอร์ตเอาต์พุต การต่อสวิตช์ลักษณะนี้จะต้องโปรแกรมสแกนคีย์สวิตช์เพื่อที่จะเช็กว่าสวิตช์ใดถูกกด ซึ่งทำได้โดยส่งค่าออกไปที่พอร์ตเอาต์พุตให้เป็นลอจิก "0" ทีละบิตแล้วอ่านค่าเข้าทางพอร์ตอินพุต จากนั้นตรวจเช็คดูว่าคีย์สวิตช์ใดถูกกดหรือไม่ จากนั้นให้บิตต่อไปของพอร์ตเอาต์พุตเป็น "0" แล้วอ่านค่าเข้ามาใหม่ การส่งค่าสแกนไปที่พอร์ตเอาต์พุตนี้จะต้องทำอย่างเร็วเพื่อที่จะตรวจสอบการกดสวิตช์ได้ทัน จากตัวอย่างวงจรในรูปที่ 2.35 จะทำการสแกนคีย์สวิตช์ได้โดย ขั้นแรกจะส่ง ข้อมูลไปที่พอร์ต C บนโดยส่งค่าบิต PC4 เป็น"0" ก่อนคือส่งค่า 1110 จากนั้นอ่านค่าเข้าทางพอร์ต C ล่าง ถ้าสวิตช์ 1 ถูกกดค่าที่อ่านได้จะเป็น 1110 ถ้าสวิตช์ 4 ถูกกด ค่าอ่านได้จะเป็น 1101 ถ้าไม่มีสวิตช์ใดในหลักที่ 1 ถูกกดเลยค่าที่อ่านเข้ามาได้จะเป็น 1111 จังหวะต่อไปให้ PB5 เป็น "0" โดยส่งค่าออกไปเป็น 1101 แล้วอ่านค่าเข้ามา ถ้าอ่านมาได้เป็น 1111 แสดงว่าหลักที่ 2 ไม่มีการกด ถ้าอ่านเข้ามาได้เป็น 1101 แสดงว่าสวิตช์ 5 ถูกกด แล้วทำการสแกนแบบนี้ไปเรื่อย ๆ ค่าของพอร์ต C ของ 8255 เมื่อมีการกดคีย์สวิตช์ใด ๆ จะเป็นไปตามตารางที่ 2.25



รูปที่ 2.35 แสดงการต่อพอร์ตแบบเมทริกซ์เข้ากับพอร์ต C ของ 8255

KEY	PC7 - PC0	KEY	PC7 - PC0
1	1110 1110	7	1110 1011
2	1101 1110	8	1101 1011
3	1011 1110	9	1011 1011
C	0111 1110	TC	0111 1011
4	1110 1101	R	1110 0111
5	1101 1101	0	1101 0111
6	1011 1101	CL	1011 0111
D	0111 1101	CP	0111 0111

ตารางที่ 2.25 แสดงค่าจากพอร์ต C เมื่อกดสวิตช์