ทฤษฎีไมโครคอนโทรลเลอร์ MCS-51 ชิปเดี่ยว

1.1 แนะนำไมโครคอนโทรลเลอร์ชิปเดี่ยว

ก่อนที่จะทำความรู้จักกับไมโครคอนโทรลเลอร์ เราควรจะทำความรู้จักกับที่มาที่ไปของชิปตัวนี้ เพื่อความเข้าใจที่ดีในการกล่าวถึงเกี่ยวกับไมโครคอนโทรลเลอร์กันซะก่อนเดิมทีเดียวนั้นในงานควบคุมระบบต่างๆ ได้นำไมโครโปรเซสเซอร์หรือไมโครคอมพิวเตอร์เข้ามาใช้ในงานควบคุม โดยในการใช้งานนั้นจุต้องมีการต่อร่วมกับอุปกรณ์ภายนอก จำพวกพอร์ต I/O หน่วยความจำข้อมูล (RAM) หน่วยความจำโปรแกรม (ROM) ทั้งยังมีชิปจำพวก UART (Universal Asychronous Receiver Transmitter) เพื่อให้ในการรับส่งข้อมูลแบบอนุกรม โดยมีหน้าที่เปลี่ยนข้อมูลจากรูปแบบขนาน (Parallel) ไปเป็นข้อมูลในรูปแบบอนุกรม (Series) สำหรับทำการส่งชุดข้อมูล และแปลงข้อมูลจากรูปแบบอนุกรมกลับเป็นรูปแบบขนานในขั้นตอนการรับข้อมูล ซึ่งจะเห็นว่ากว่าจะนำไมโครโปรเซสเซอร์ หรือไมโครคอมพิวเตอร์ดังกล่าวมาใช้ในระบบควบคุมโดยเฉพาะนั้น จะค่อนข้างยุ่งยากสิ้นเปลืองอุปกรณ์ร่วมต่างๆ เพื่อที่จะให้คลอบคลุมการใช้งานควบคุมที่มีประสิทธิภาพ ฉะนั้นจากปัญหาต่างๆที่ผ่านมา จึงทำให้มีบริษัทผู้ผลิตชิปไมโครโปรเซสเซอร์หรือไมโครคอมพิวเตอร์ดังกล่าวได้ทำการพัฒนา และผลิตชิปที่มีการรวมคุณสมบัติต่าง ๆ เอาไว้ครบเพื่องานควบคุมโดยเฉพาะ

โดยในชิปจะประกอบไปด้วย ไมโครโปรเซสเซอร์,พอร์ต, I/O,หน่วยความจำข้อมูลขนาดเกล็กตลอดจนอาจจะมีพอร์ตใช้งานในการรับส่งข้อมูล (UART) และหน่วยความจำโปรแกรม (ROM) เข้าไปด้วยภายในชิปเพื่อใช้ในระบบควบคุมอย่างเต็มประสิทธิภาพสามารถประยุกต์ใช้งานง่ายและประหยัดค่าใช้จ่ายเกี่ยวกับอุปกรณ์เสริมต่างๆ ชิปที่รวมคุณสมบัติต่างๆดังที่เกล่าวมาข้างต้นจึงถูกเรียกว่าไมโครคอนโทรลเลอร์ชิปเดี่ยว(Microcontroller Unit; MCU)นั่นเอง โดยบริษัทผู้ผลิตก็ได้ผลิต MCU ออกมาหลายรูปแบบ ซึ่งจะมีคุณสมบัติและความสามารถที่แตกต่างกันเพื่อให้เหมาะสมกับงานในรูปแบบต่างๆ จึงทำให้มีการแยกประเภทของ MCU ออกเป็นตระกูลๆ หรือเบอร์ต่างๆ โดยขึ้นอยู่กับคุณสมบัติของ MCU นั้นและบริษัทผู้ผลิตเอง

โดยตระกูลที่ได้รับความนิยมและเป็นที่รู้จักกันดีในหมู่นิสิตนักศึกษา ตลอดทั้งนักอิเล็คทรอนิกส์สมัครเล่นและมืออาชีพก็คือไมโครคอนโทรลเลอร์ MCS-51 เนื่องจากมีข้อมูลหรือหนังสือที่เกี่ยวกับ MCU ตระกูลนี้อยู่พอสมควร โดยหนังสือเล่มนี้ก็เป็นอีกเล่มหนึ่งที่นำเสนอข้อมูลการประยุกต์ใช้งาน MCU ตระกูลนี้เช่นกัน โดยเนื้อหาจะกล่าวถึงการใช้งานทางด้านฮาร์ตแวร์เป็นหลัก ต่อไปจะเป็นการกล่าวถึง สถาปัตยกรรมและคุณสมบัติของ MCS-51 เพื่อเป็นพื้นฐานในการทำความรู้จักและเข้าใจในระดับต้นดังนี้

1.1.1 คุณสมบัติของ MCS-51

1. ใช้เทคโนโลยีชั้นสูงในการสร้าง โดยมีทั้งประเภท HMOS, CMOS และ CHMOS ทำงานด้วยแหล่งจ่ายไฟ+5 Vdc เพียงแหล่งเดียว

2. มีหน่วยประมวลผลขนาด 8 บิต

3. สามารถติดต่อกับหน่วยความจำภายนอกทั้งหน่วยความจำโปรแกรม และหน่วยความจำข้อมูลได้สูงสุด 64 Kbyte

4. มีพอร์ต I/O แบบขนานสองทิศทางจำนวน 4 พอร์ต พอร์ตละ 8 บิต รวมทั้งหมดเป็น 32 บิต จะใช้ในการเข้าถึงแอดเดรสและข้อมูลสำหรับติดต่อกับหน่วยความจำภายนอก

5. พอร์ตใช้งานทุกพอร์ตและมีลักษณะเป็นพอร์ตแล็ซ (Latch) คงสภาวะ

6. มีขาพอร์ตที่ใช้สำหรับการรับส่งข้อมูลแบบอนุกรม

7. หนึ่ง Machine Cycle จะใช้เวลา 1 ไมโครวินาที โดยใช้ X-TAL 12 MH_Z

8. สามารถกำหนดการใช้งานพอร์ต อนุกรม I/O ได้ในระดับไบต์หรือบิตได้โดยตรง

9. ตัวเลขทางคณิตศาสตร์ใช้ได้ทั้งระบบฐานสอง และฐานสิบหก

ตระกูลของ MCS-51 จะมีทั้งแบบมี EPROM ในตัวหรือไม่มี EPROM ภายใน (ในการกล่าวถึงหน่วยความจำโปรกแกรมต่อไปนั้นผู้เขียนจะกล่าวถึง EPROM เป็นหลัก เนื่องจากเป็นชนิดประเภทของหน่วยความจำโปรแกรมที่สามารถนำมาประยุกต์ใช้งาน และเป็นที่รู้จักกันดี โดยขอให้ผู้อ่านเข้าใจว่าหน่วยความจำโปรแกรม EEPROM ได้เช่นกัน ข้อแตกต่างหรือรายละเอียดนั้นจะกล่าวไว้ในบทที่ 2) ซึ่งก็จะมีตำแหน่งขาที่เหมือนกัน ตารางที่ 1-1 แสดงถึงรายละเอียดของไมโครคอนโทรลเลอร์ตระกูล MCS-51 เบอร์ 89XX ของบริษัท ATMEL และเบอร์ 80C31,80C32 ของบริษัท INTEL

1.1.2 ลักษณะการจัดขาภายนอกของ MCS-51

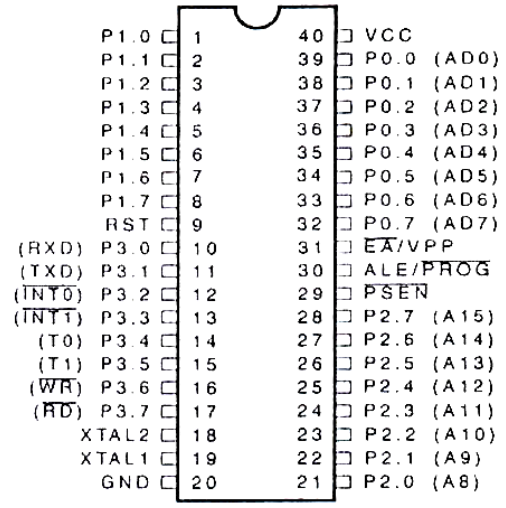
รูปที่ 1-1 แสดงการจัดขาตามลักษณะภายนอกของชิป MCS-51 ซึ่งจะมีการแบ่งกลุ่มจัดขาของ MCS-51 มีอยู่ 4 กลุ่ม คือ

1. กลุ่มขาแหล่งจ่ายไฟเลี้ยง และสัญญาณนาฬิกา

2. กลุ่มขาสำหรับอ้างแอดเดรสและรับส่งข้อมูล

3. กลุ่มขาที่ใช้ในการควบคุม

4. กลุ่มขาพอร์ตใช้งานแบบขนานและอนุกรมพอร์ตใช้งานบางพอร์ตจะทำหน้าที่ได้



รูปที่ 1-1

กลุ่มขาพอร์ตแบบขนาน I/O ปกติในกรณี MCU ตัวนั้นมี EPROM ภายในตัวเป็นต้น รายละเอียดการทำงานตลอดจนโครงสร้างภายใน Microcontroller MCS-51 สามารถศึกษาได้จากคู่มือหรือ Data Sheet การใช้งาน Microcontroller MCS-51 โดยในที่นี้จะแนะนำรายละเอียดและคุณสมบัติในเบื้องต้นสำหรับนำมาประยุกต์ใช้งานต่อไป

1.1.3 **ขาที่สำคัญของไมโครคอนโทรลเลอร์** MCS-51

1. ขา V_{CC} เป็นขารับแรงดันไฟกระแสตรง +5 Vdc
2. ขา GND เป็นขากราวด์
3. พอร์ต 0 (Port 0) มี 8 บิต ได้แก่บิต P0.0 – P0.7 เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง สำหรับใช้งานทั่วไป โดยถ้าใช้งานเป็น

ตารางที่ 1-1 แสดงรายละเอียดของไมโครคอนโทรลเลอร์ตระกูล MCS-51

Features	89C1051	89C2051	80C31	80C32	89C51	89C52	89LV51	89LV52	89C55	89S8252
Flash Memore	1K	2K	-	-	4K	8K	4K	8K	20K	8K
Ram	64	128	128	256	128	256	128	256	256	256
EEPROM	-	-	-	-	-	-	-	-	-	2K
Programming	Yes	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes
Timers/Counters	1	2	2	3	2	3	2	3	3	4
Serial UART	1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Power supply	2.7-6.0	2.7-6.0	4.0-6.0	4.0-6.0	4.0-6.0	4.0-6.0	2.7-6.0	2.7-6.0	2.7-6.0	2.7-6.0
Frequency	0-24	0-24	0-24	0-24	0-24	0-24	0-24	0-24	0-33	0-33
I/O Pins	15	15	32	32	32	32	32	32	32	32
External address/ Data bus	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Pin Count	20	20	40/44	40/44	40/44	40/44	40/44	40/44	40/44	40/44

อินพุตพอร์ตต้องทำการเซ็ตค่า 1 ไปยังพอร์ตเมื่อต้องการใช้งานพอร์ตนั้นทั้งพอร์ตเป็นอินพุตในระดับบิต ก็สามารถกระทำได้โดยการเซ็ตค่า 1 ไปยังแต่ละบิตที่ต้องการใช้งานเป็นพอร์ตอินพุตในระดับ เพื่อกำหนดให้ขาพอร์ตหรือแต่ละบิตเหล่านั้นอยู่ในสถานะปล่อยลอย ซึ่งในสถานะนี้เองที่นำมาใช้เป็นพอร์ตอินพุตอิมพีแดนซ์สูงได้ นอกจากพอร์ตนี้จะใช้งานเป็นพอร์ตอินพุตเอาต์พุตแล้วมันยังถูกใช้งานในการติดต่อกับหน่วยความจำภายนอก (EPROM,RAM) ได้อีกด้วย โดยทำหน้าที่ในการกำหนดตำแหน่งแอดเดรสไบต์ต่ำ (A0-A7) ซึ่งจะใช้งานเป็นแบบมัลติเพล็กซ์สำหรับการรับส่งข้อมูลขนาด 8 บิต (D0-D7)

4. พอร์ต 1 (Port 1) มี 8 บิต ได้แก่บิต P1.0-P1.7 เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง สำหรับใช้งานทั่วไป โดยถ้าใช้งานเป็นอินพุตพอร์ตหรืออินพุตในระดับบิต สามารถกระทำได้โดยวิธีเช่นเดียวกันกับพอร์ต 0 ข้างต้น

5. พอร์ต 2 (Port 2) มี 8 บิต ได้แก่ บิต P2.0-P2.7) เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง สำหรับใช้งานทั่วไป โดยถ้าใช้งานเป็นอินพุตพอร์ตเหรืออินพุตในระดับบิต สามารถกระทำได้เช่นเดียวกันกับพอร์ต 0 ข้างต้น เช่นเดียวกันกับพอร์ต 0 นอกจากใช้งานเป็นพอร์ตอินพุตเอาต์พุตแล้ว มันยังถูกใช้งานในการติดต่อกับหน่วยความจำภายนอก (EPROM,RAM) ได้อีกด้วย โดยทำหน้าที่ในการอ้างตำแหน่งแอดเดรสไบต์สูง (A8-A15)

6. พอร์ต (Port 3) มี 8 บิต ได้แก่บิต P3.0-P3.7 เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง สำหรับใช้งานทั่วไป โดยถ้าใช้งานเป็นอินพุตพอร์ตหรืออินพุตในระดับบิต สามารถกระทำได้เช่นเดียวกันกับพอร์ต 0 ข้างต้น นอกจากจะใช้งานเป็นพอร์ตอินพุตเอาต์พุตแล้วมันยังสามารถใช้งานในหน้าที่พิเศษต่างๆ ดังตารางที่ 1-2

7. ขารีเซต (RST) ใช้สำหรับการรีเซตการทำงานของไมโครคอนโทรลเลอร์ โดยการรีเซตต้องคงสภาวะ high อย่างน้อยนาน 2 Machine cycle ในขณะที่ออสซิลเลเตอร์ยังทำงานอยู่

8. ขา ALE / $\overline{\text{PROG}}$ เป็นขาสัญญาณเพื่อทำหน้าที่ควบคุมการแล็ซ (Latch) ค่าตำแหน่งแอดเดรสไบต์ต่ำ (Address Latch Enable) เมื่อต้องการติดต่อกับหน่วยความจำภายนอก นอกจากนี้ขานี้ยังทำหน้าที่เป็นอินพุตรับพัลส์ในการโปรแกรม (Program Pulse Input)) ในส่วนของหน่วยความจำ EPROM สำหรับ

ตารางที่ 1-2 แสดงหน้าที่พิเศษของแต่ละขาของพอร์ต 3 ของไมโครคอนโทรลเลอร์

Pin Port	Discription
P3.0	RXD (Serial Input Port)
P3.1	TXD (Serial Output Port)
P3.2	$\overline{\text{INT0}}$ (External Interrupt 0)
P3.3	$\overline{\text{INT1}}$ (External Interrupt 1)
P3.4	T0 (Timer 0 External Input)
P3.5	T1 (Timer 1 External Input)
P3.6	$\overline{\text{RW}}$ (External Data Memory Write Strobe)
P3.7	$\overline{\text{RD}}$ (External Data Memory Read Strobe)

ไมโครคอนโทรลเลอร์ในตระกูล MCS-51 ที่มีหน่วนความจำโปรแกรมภายในเป็น EPROM

9. ขา PSEN (Program Store Enable) ทำหน้าที่เป็นสัญญาณสโตรบเพื่ออ่านคำสั่งจากหน่วยความจำโปรแกมภายนอก เมื่อไมโครคอนโทรลเลอร์ประมวลผลคำสั่งจากหน่วยความจำภายนอก เมื่อไมโครคอนโทรลเลอร์ประมวลผลคำสั่งจากหน่วยความจำภายนอก ขานี้จะส่งสัญญาณสโตรบจำนวน 2 ครั้ง ในแต่ละ Machine Cycle แต่ในขณะที่ติดต่อกับหน่วยความจำข้อมูลภายนอกจะไม่มีการส่งสัญญาณ Strobe แต่อย่างใด

10. ขา EA/Vcc (External Access Enable/Vcc) เป็นขาสำหรับการเลือกใช้หน่วยความจำโปรแกรมจากภายในหรือจากภายนอก โดยมีสถานะเป็น 0 หรือ 1 จะหมายถึงให้ไมโครคอนโทรลเลอร์ รับคำสั่งจากหน่วยความจำภายนอก และภายในตามลำดับ อย่างไรก็ตามถ้าบิตป้องกัน (Security Bit) ในหน่วยความจำ EPROM ถูกโปรแกรมไว้ ไมโครคอนโทรลเลอร์จะไม่รับคำสั่งจากหน่วยความจำภายนอกเลย นอกจากนี้ ขานี้ยังทำหน้าที่รับแรงดันไฟสำหรับการโปรแกรม (Vcc) ขนาด 12 โวลต์ เพื่อใช้ในระหว่างการโปรแกรมหน่วยความจำโปรแกรม (EPROM) ภายในตัว MCU

11. ขา XTAL1 และขา XTAL2 เป็นขาใช้งานของวงจรอินเวอร์ติ้งออสซิลเลเตอร์แอมปลิไฟเออร์ (Inverting Oscillator Amplifier) สำหรับใช้ต่อร่วมกับคริสตอลภายนอก

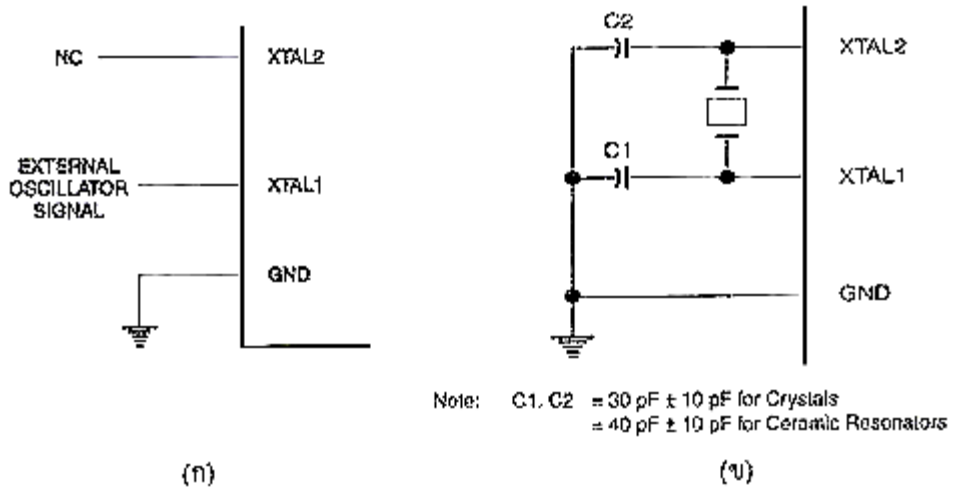
1.1.4 Machine Cycle

ในการทำงานของ MCU นั้นจะถูกกำหนดความเร็วในการทำงานหรือการประมวลผลต่างๆ ด้วยสัญญาณนาฬิกา (Clock) ที่ป้อนให้กับ MCU ไม่ว่าจะเป็นการต่อสัญญาณนาฬิกาโดยตรง ซึ่งวิธีการนี้จะต้องคำนึงถึงชนิดหรือประเภทของ MCU นั้นๆ ด้วยว่าเป็น CMOS,CHMOS หรือ HMOS โดยแต่ละชนิดประเภทจะมีการต่อสัญญาณนาฬิกาโดยตรงที่แตกต่างกันไป จากรูปที่ 1-2 (ก) นั้นจะเป็นตัวอย่างการต่อสัญญาณนาฬิกาโดยตรงจากภายนอกสำหรับ MCU ชนิดประเภท CMOS,CHMOS โดยวิธีการต่อสัญญาณนาฬิกาจากภายนอกวิธีนี้ออกจะยุ่งยาก และต้องระมัดระวังในการต่อใช้งาน ผู้เขียนจึงขอแนะนำให้ใช้วิธีที่ 2 ดังแสดงในรูปที่ 1-2 (ข) จะเป็นการใช้ X-TAL ในการกำเนิดสัญญาณนาฬิกาให้กับ MCU ซึ่งจะมีความแน่นอนและมีวงจรการใช้งานที่มีความซับซ้อนน้อยกว่าวิธีแรก แต่ขอให้ผู้อ่านคำนึงถึงการเลือกใช้ความถี่ของ X-TAL ให้เหมาะสมกับประเภทและชนิดของงานด้วย เพราะ X-TAL นั้นยิ่งมีความถี่มากก็จะเป็นตัวกำเนิดสัญญาณรบกวนมากเช่นกัน (การแก้ไขสัญญาณรบกวนในเบื้องต้นนั้นให้ทำการ ชีลด์ (Shield) หรือ ต่อตัวถัง X-TAL ลงกราวด์ก็จะแก้ปัญหาสัญญาณรบกวนได้บ้าง)

สัญญาณนาฬิกาจะเป็นตัวควบคุมการทำงาน การประมวลผลคำสั่งต่างๆ ของ MCU โดยในการประมวลผลคำสั่งของ MCS-51 จะทำงานเป็นรอบวัฏจักรของภายเครื่อง หรือที่เรียกว่า Machine Cycle โดย 1 Machine Cycle จะใช้ช่วงเวลาในการทำงานเท่ากับคาบเวลาของสัญญาณนาฬิกาจำนวน 12 ลูก (12 Oscillator period) จากคุณสมบัติข้างต้นเราสามารถคำนวณเวลาการทำงานใน 1 Machine Cycle ของ MCS-51 จากความถี่ของ X-TAL ที่เลือกนำมาใช้งานได้จากสมการ

$$1 \text{ Machine Cycle} = 12 \frac{12}{X-TAL} \text{ oscillator (Second)}$$

โดยเมื่อเรารู้ระยะเวลาในการทำงานใน 1 Machine Cycle ของ CPU แล้วจะช่วยให้สามารถประมาณช่วงเวลาในการทำงานของคำสั่งแต่ละคำสั่งในการเขียนโปรแกรมได้ ซึ่งคำสั่งแต่ละคำสั่งจะถูกกำหนดการทำงานเป็น Machine Cycle เช่นกัน โดยรายละเอียดการทำงานของคำสั่งในรูป Machine Cycle หรือ Oscillator Period สามารถดูได้จากชุดคำสั่งของไมโครคอนโทรลเลอร์ MCS-51 ในหัวข้อที่ 1.1.6 หรือสรุปการใช้งานชุดคำสั่งของไมโครคอนโทรลเลอร์ MCS-51 (Instruction Definitions) ในภาคผนวกของหนังสือเล่มนี้



รูปที่ 1-2 (ก) แสดงตัวอย่างการต่อสัญญาณนาฬิกาโดยตรงจากภายนอก

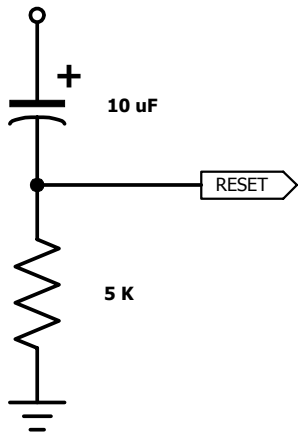
(ข) แสดงการใช้ X-TAL ในการกำเนิดสัญญาณนาฬิกาให้กับ MCU

จากที่ผ่านมาคงจะพอเป็นที่รู้จัก MCU กันพอสมควรแล้ว แต่สิ่งที่ต้องคำนึงถึงอีกประการหนึ่งในการนำ MCU ไปใช้งานนั่นก็คือขั้นตอนในการรีเซ็ตตัว MCU เพื่อให้ MCU ทำงานได้ถูกต้อง ไม่เกิดความผิดพลาด หรือเมื่อเกิดสิ่งผิดปกติก็สามารถทำการรีเซ็ต MCU ให้เริ่มต้นการทำงานต่อไปใหม่ได้

1.1.5 Reset Operation

ในการรีเซ็ต ไมโครคอนโทรลเลอร์ MCS-51 นั้นจะต้องทำให้สภาวะที่ขา RST ของตัว MCU อยู่ในสภาวะ High ที่ค่าเวลาที่เหมาะสมโดยตลอบคลุมช่วงเวลา 2 Machine Cycle ของการทำงานเป็นอย่างน้อย (ทำนองว่าเกินได้แต่ต่ำกว่าไม่ได้) แล้วกลับสู่สภาวะ Low โดย Timming Diagram ต่างๆ ถ้าผู้อ่านสนใจก็ศึกษาได้จาก Data Sheet ของ MCU นั้นๆ แต่หนังสือเล่มนี้จะเน้นภาคปฏิบัติทางด้านการนำไปใช้งานจึงจะไม่ขอกล่าวถึงการคำนวณต่างๆ เท่าใดนัก เอาเป็นว่าการรีเซ็ต MCU ที่นิยมใช้มีด้วยกัน 2 วิธี วิธีแรกก็คือ การใช้อุปกรณ์ R,C ในการรีเซ็ต MCU แต่วิธีนี้ดูจะล้าสมัยไปหน่อยเพราะโดยส่วนมากจะนิยมใช้วิธีที่ 2 คือการใช้อุปกรณ์ Semiconductor หรือ IC ในการรีเซ็ต MCU ซึ่งจะมีความแน่นอน และใช้งานง่ายกว่าวิธีแรกมาก

วิธีที่ 1 R,C Reset

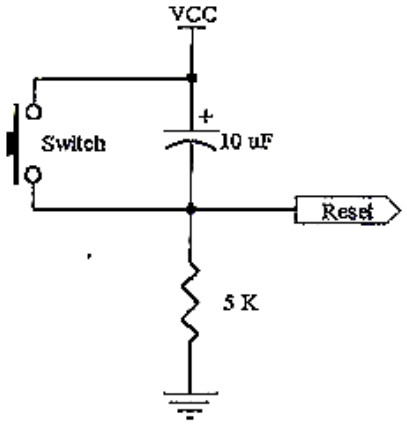


รูปที่ 1-3 วงจรรีเซ็ตอัตโนมัติ

จากรูปที่ 1-3 นั้นจะเป็นการต่อวงจร R,C ใช้ในการรีเซ็ตอัตโนมัติให้กับ MCS-51 โดยการทำงานของวงจรนั้น เมื่อมีการจ่ายไฟเลี้ยงให้กับระบบวงจรจะกำเนิดสัญญาณ Active High เพื่อทำการรีเซ็ต MCS-51 โดยอัตโนมัติ ท่านผู้อ่านคงจะสงสัยว่า จากวงจรนั้นขา RST ของ MCU ต่อผ่าน R-5K ลงกราวด์ ทั้งยังมี C ต่อกั้นระหว่างขา RST และ Vcc อีก จะทำให้ขา RST ของ MCU มีสภาวะ High ได้อย่างไร ด้วยคุณสมบัติของ C ก็จะทำหน้าที่ Block ไหกระแสตรงไม่ให้ผ่านตัวมันอยู่แล้ว ฉะนั้นเพื่อแก้ข้อสงสัยจากวงจรในรูปที่ 1-3 ขา RST ของ ตัว MCU ก็จ่าจะมีสภาวะ Low ตลอดเวลา ฉะนั้นผู้เขียนจะขออธิบายการทำงานเบื้องต้นของวงจรดังนี้

จากวงจรรูปที่ 1-3 ที่สภาวะปกติยังไม่จ่ายไฟเลี้ยงให้กับระบบ การต่อวงจรดังรูปจะทำให้ C รักษาสภาวะเป็นกลาง คือ C จะถูก Discharge ประจุออกจนหมด จากนั้นเมื่อเริ่มจ่ายไฟให้กับระบบจะทำให้มีกระแสไหลผ่าน C ได้ในช่วงเวลาหนึ่ง (Initial State) โดยขึ้นอยู่กับค่าความจุของ C นั่นคือยิ่ง C มีค่ามากช่วงเวลาในการไหลของกระแสผ่าน C ก็จะยิ่งมีค่ามาก (เป็นคุณสมบัติของ C) ฉะนั้นขณะที่มีกระแสไหลผ่าน C กระแสก็จะไหลผ่าน R ซึ่งก็จะทำให้เกิดแรงดันตกคร่อม R ขึ้น เป็นผลให้สภาวะที่ขารีเซ็ตของ MCU มีสภาวะ High จนกระทั่งกระแสหยุดไหลผ่าน C (เมื่อ C ถูก Charge ประจุเต็มที่แล้ว , Steady State) ก็จะทำให้แรงดันที่ตกคร่อม R มีค่าเป็นศูนย์ เป็นผลให้ขารีเซ็ตของ MCU เปลี่ยนสภาวะจาก High เป็น Low ในที่สุดเสมือนเป็นการส่งสัญญาณ Active High ให้กับขา RST ของ MCU ทำให้ MCU กลับมาเริ่มต้นการทำงานต่อไป โดยจากวิธีนี้จะเห็นว่าระยะเวลาสภาวะ High ของขา RST นั้นขึ้นอยู่กับคุณสมบัติของ C และ R โดยถ้า C และ R มีค่าที่ไม่สัมพันธ์กัน ก็จะทำให้สภาวะของขา RST นั้นผิดเพี้ยน อาจส่งผลให้ตัว MCU ทำงานผิดพลาด

จากการที่ผู้เขียนได้ทำการทดลองต่อวงจรรีเซ็ต โดยเลือกใช้ค่า R และ C ค่าต่างๆ พบว่าค่า R และ C ที่เหาะสมในการรีเซ็ต MCU จะช้าค่า C ประมาณ 10 uF ชนิดอิเล็คโตรไลต์ และ R ค่าประมาณ 5-10 KΩ ¼ W 5%



รูปที่ 1-4 วงจรรีเซ็ตอัตโนมัติพร้อม Switch รีเซ็ต

จากวงจรในรูปที่ 1-4 นั้นยังจัดอยู่ในรูปแบบการรีเซ็ตแบบเดียวกับวิธีแรก แต่จะต่อสวิตช์ตามวงจรเพื่อทากรรีเซ็ต MCU แบบ Manual Reset กรณี MCU เกิดการทำงานผิดพลาดในระบบแล้วไม่ต้องการเปิด / ปิด การจ่ายไฟให้แก่ระบบใหม่

จากวงจรในรูปที่ 1-4 เมื่อทำการกดสวิตช์จะทำให้มีแรงดันตกคร่อม R จึงเกิดสภาวะ High ขึ้นที่ขารีเซ็ต โดยระยะเวลาก็จะขึ้นกับช่วงเวลาการกดสวิตช์นั่นเอง ถ้าใช้วิธีนี้คงจะให้ความแน่นอนน้อยกว่าวิธีแรกแต่ก็สามารถรีเซ็ต MCU ได้เช่นกัน จากที่กล่าวมาแล้วสภาวะ High ที่ขารีเซ็ต นั้นอาจจะนานเกินได้แต่ห้ามน้อยกว่าช่วงเวลาการรีเซ็ต นั่นก็คือการกดสวิตช์ด้วยมือนั้นช่วงเวลาจะเกินช่วงเวลาที่ทำให้ MCU รู้ว่าถูกรีเซ็ตแน่นอน จึงไม่มีปัญหาอะไรสำหรับวงจรนี้ แต่จากการทดลองใช้การีเซ็ตแบบ Manual Reset ผลปรากฏว่าไม่สามารถรีเซ็ต MCU ได้เป็นบางครั้ง ทั้งนี้น่าจะขึ้นอยู่กับชนิดและคุณสมบัติของสวิตช์ที่นำมาใช้ การกระเด้งของหน้าสัมผัส ความสนิทของหน้าสัมผัส (Bounce) เป็นต้น