บทที่ 4

การออกแบบโปรแกรมควบคุมระบบแสวงหาข้อมูล

4.1 การทำงานของระบบแสวงหาข้อมูล

การทำงานของระบบแสวงหาข้อมูลจะเริ่มทำงานเมื่อมีการรีเซ็ทระบบไมโครคอนโทรลเลอร์จะไปดึงเอาโปรแกรมควบคุมใน EPROM ตำแหน่งที่ 0000H มาทำงาน ซึ่งในตำแหน่งเริ่มต้นนี้จะเป็นการกำหนดค่าเริ่มต้นให้แก่ระบบ เช่น การกำหนดตำแหน่งพอร์ตและหน่วยความจำ การเคลียร์ค่าในบัฟเฟอร์ที่ใช้งาน และการกำหนดการทำงาน (initialized) ให้กับอุปกรณ์ต่อพ่วงต่างๆ เช่นจอแสดงผลแบบ LCD เป็นต้น

นอกจากนี้ยังมีการกำหนดการทำงานให้กับตำแหน่งแอดเดรสที่เป็นส่วนของการบริการการขัดจังหวะประเภทต่างๆ เช่น ที่ตำแหน่ง 0003H กำหนดไว้สำหรับการขัดจังหวะของ Real Time Clock, ตำแหน่ง 0013H สำหรับการขัดจังหวะของคีย์บอร์ด และตำแหน่ง 0023H สำหรับการขัดจังหวะของพอร์ตสื่อสารแบบอนุกรม

ในส่วนของฟังก์ชั่นหลัก (Main) จะเป็นเพียงการวนรอบรอข้อมูลควบคุมการทำงานที่ส่งมาจากคอมพิวเตอร์ที่เป็นตัวควบคุม เมื่อข้อมูลควบคุมส่งมา (เกิดการขัดจังหวะของพอร์ตสื่อสารอนุกรม) โปรแกรมควบคุมไมโครคอนโทรลเลอร์จะตรวจสอบว่าเป็นคำสั่งสำหรับอะไรและกระโดดไปทำงานในฟังก์ชั่นที่เหมาะสม

ในส่วนของคอมพิวเตอร์ที่ใช้ในการควบคุม ผู้ใช้จะสามารถควบคุมการทำงานของทั้งระบบโดยผ่านวินโดว์และเมนู ซึ่งข้อมูลที่อ่านได้จะถูกเก็บลงในฮาร์ดดิสค์ของเครื่องเพื่อให้สามารถนำไปใช้ในการวิเคราะห์ต่อไป

4.2 โปรแกรมควบคุมการทำงาน

โปรแกรมควบคุมการทำงานของระบบแสวงหาข้อมูลจะแบ่งออกเป็นสองส่วนหลัก คือ

1. โปรแกรมควบคุมไมโครคอนโทรลเลอร์ MCS-51
2. โปรแกรมควบคุมการทำงานของคอมพิวเตอร์

ส่วนของโปรแกรมที่ใช้ควบคุมการทำงานของไมโครคอนโทรลเลอร์ จะเป็นโปรแกรมภาษาแอสแซมบลี้ของไมโครคอนโทรลเลอร์ตระกูล MCS-51 ซึ่งประกอบด้วยโปรแกรมที่ใช้ควบคุมวงจรแปลงสัญญาณแอนะล็อกเป็นดิจิทัล โปรแกรมควบคุมการอ่านและส่งสัญญาณแบบดิจิทัล โปรแกรมควบคุมการรับส่งข้อมูลกับระบบคอมพิวเตอร์หรือไมโครคอนโทรลเลอร์ตัวอื่น เป็นต้น โปรแกรมในส่วนนี้จะถูกเก็บไว้ใน EPROM เบอร์ 27256 ตำแหน่ง 0000H (ขนาด 32 กิโลไบต์) ซึ่งเมื่อไมโครคอนโทรลเลอร์เริ่มทำงาน (รีเซ็ทระบบ) ไมโครคอนโทรลเลอร์จะไปดึงเอาโปรแกรมใน EPROM มาทำงาน

ส่วนโปรแกรมที่ใช้ควบคุมการทำงานของคอมพิวเตอร์ จะเป็นโปรแกรมภาษา C โปรแกรมในส่วนนี้จะประกอบด้วยฟังก์ชั่นต่างๆ ได้แก่ ฟังก์ชั่นการกำหนดค่าเริ่มต้นฟังก์ชั่นการส่งคำสั่ง เพื่อควบคุมไมโครคอนโทรลเลอร์ ฟังก์ชั่นการรับและส่งข้อมูลกับไมโครคอนโทรลเลอร์ และการให้ผู้ใช้สามารถดูข้อมูลและควบคุมการทำงานของระบบโดยผ่านวินโดว์และเมนูเป็นต้น

4.2.1 กำหนดค่าเริ่มต้นให้กับระบบ

การกำหนดค่าเริ่มต้นให้กับระบบนั้น ในส่วนของโปรแกรมที่ใช้ควบคุมการทำงานของไมโครคอนโทรลเลอร์ จะต้องมีการกำหนดตำแหน่งพอร์ตให้กับพอร์ตที่จะใช้เชื่อมต่อกับวงจรภายนอก เช่นกำหนดให้จะแสดงผล LCD อยู่ที่ตำแหน่ง 0E060H - 0E062H เป็นต้น ซึ่งตำแหน่งพอร์ตที่ใช้ในระบบแสดงไว้ในตารางที่ 3.2

นอกจากนี้การที่ระบบต้องมีการตอบสนองต่อการขัดจังหวะต่างๆที่เกิดขึ้นและต้องมีการรับส่งข้อมูลกับระบบคอมพิวเตอร์ที่ใช้ควบคุมจึงต้องมีการกำหนดค่าให้กับรีจีสเตอร์ที่ใช้เป็นตัวควบคุม การทำงานของไมโครคอนโทรลเลอร์ดังต่อไปนี้

- กำหนดให้บิต EA ในรีจีสเตอร์ใช้งานเฉพาะ IE เป็น 1 : เพื่อให้ไมโครคอนโทรลเลอร์ตอบสนองต่อสัญญาณขัดจังหวะที่เกิดขึ้น

- กำหนดให้บิต EX0 และ EX1 มีค่าเป็น 1 : เพื่อให้ไมโครคอนโทรลเลอร์ตอบสนองต่อสัญญาณขัดจังหวะที่เกิดขึ้นจากสัญญาณภายนอกที่ขา INT0 และ INT1

- กำหนดให้บิต IT0 และ IT1 ในรีจีสเตอร์ใช้งานเฉพาะ TCON เป็น 1 : เพื่อให้ไมโครคอนโทรลเลอร์ตอบสนองต่อสัญญาณขัดจังหวะที่เกิดขึ้นจากสัญญาณภายนอกที่เกิดขึ้นที่ขา INT0 และ INT1 โดยตรวจสอบจากการเปลี่ยนระดับสัญญาณจาก 0 เป็น 1 (ใช้กับการเชื่อมต่อคีย์บอร์ด)

- รีจิสเตอร์ SCON มีค่า 50H : เพื่อกำหนดการทำงานของพอร์ตสื่อสารอนุกรมโหมด 1

- รีจิสเตอร์ TMOD มีค่า 20H : เพื่อกำหนดการทำงานของไทม์เมอร์ 1 โหมด 2

- รีจิสเตอร์ TH1 (รีจีสเตอร์ที่ใช้เป็นไทม์เมอร์ 1) มีค่า FDH : เพื่อกำหนดค่า baud rate ของการสื่อสารอนุกรมเป็น 9600 bps

- กำหนดให้บิต TR1 ในรีจีสเตอร์ TCON มีค่าเป็น 1 : ให้ไทม์เมอร์ 1 เริ่มทำงานเพื่อกำหนด baud rate

ในการกำหนดค่าเริ่มต้นในส่วนของเครื่องคอมพิวเตอร์ที่ใช้ควบคุมการทำงาน ค่าเริ่มต้นนี้จะเป็นตัวกำหนดว่ามีอุปกรณ์หรือวงจรภายนอกที่ต่อเชื่อมอยู่กับระบบเป็นจำนวนเท่าใด นอกจากนี้การใช้งานวงจรที่มาเชื่อมต่อบางอย่างวงจร ADC อาจจำเป็นต้องมีการกำหนดค่าบางอย่างด้วย เช่น ชนิดของสัญญาณแอนะล็อกที่จะอ่าน ซึ่งก็สามารถกำหนดได้ในส่วนนี้

โปรแกรมควบคุมการทำงานของเครื่องคอมพิวเตอร์การกำหนดค่าเริ่มต้นจะกระทำเมื่อมีการทำงานเป็นครั้งแรกหรือเมื่อต้องการกำหนดค่าเริ่มต้นใหม่เท่านั้น ในการทำงานครั้งต่อไปถ้า ไม่มีการแก้ไขค่าเริ่มต้นใหม่โปรแกรมจะนำค่าที่ถูกกำหนดไว้แล้วมาใช้

4.2.2 โปรแกรมย่อยสำหรับการแสดงผล

โปรแกรมย่อยสำหรับการแสดงผลจะแบ่งออกเป็น 2 ส่วนคือ ส่วนที่ใช้แสดงผลทางจอคอมพิวเตอร์ที่ใช้ควบคุม และ ส่วนที่ใช้แสดงผลทางจอแสดงผล LCD

ส่วนที่ใช้แสดงผลทางจอคอมพิวเตอร์ที่ใช้ควบคุมจะเป็นการส่งข้อมูลจากวงจรควบคุมระบบแสวงหาข้อมูลผ่านทางพอร์ตสื่อสารอนุกรมไปยังคอมพิวเตอร์ โปรแกรมที่ใช้ควบคุมคอมพิวเตอร์

จะนำข้อมูลที่รับเข้ามา (เป็นตัวอักษร) มาแปลงเป็นค่าที่เหมาะสมแล้วส่งออกไปแสดงบนจอภาพ

สำหรับคอนโทรลเลอร์เบอร์ HD44780 ที่ใช้ในการวิจัยครั้งนี้ จะทำหน้าที่ควบคุมการแสดงผลของ LCD ทั้งอักษร และสัญลักษณ์พิเศษต่างๆ รวมทั้งสามารถออกแบบรูปร่างต่างๆ ได้เอง และสามารถต่อใช้งานแบบ 4 และ 8 บิต

การกำหนดฟังก์ชั่น ในการเตรียม LCD จะต้องมีการกำหนดฟังก์ชั่นให้พร้อมก่อน โดยมีการกำหนดค่าดังนี้

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	X	X

DL : เป็นการเลือกว่าจะให้ติดต่อแบบ 4 หรือ 8 บิต โดยถ้า DL = '0' เป็นการสั่งให้มีการติดต่อแบบ 4 บิต และ DL = '1' เป็นการติดค่อแบบ 8 บิต

N : เป็นการกำหนดบรรทัดการแสดงผล ถ้า N = '0' เป็นการแสดงผล 1 บรรทัด และถ้า N = '1' เป็นการแสดงผลแบบ 2 บรรทัด

F : เป็นการกำหนดขนาดตัวอักษร ถ้าให้ F = '0' ตัวอักษรจะมีขนาด 5 X 7 แต่ถ้าให้ F = '1' ตัวอักษรจะมีขนาด 5 X 10

ข้อมูลที่จะนำมาใช้เขียนโปรแกรมจะนำมาจากค่าในบิต DB0 - DB7 เท่านั้น

ตัวอย่างเช่น ถ้าหากต้องการติดต่อกับ LCD เป็นแบบ 8 บิต มีการแสดงผล 2 บรรทัด และกำหนดขนาดตัวอักษรเท่ากับ 5 X 7 จะต้องเขียนโปรแกรมกำหนดค่า ดังนี้

MOV P1 , #00111000B หรือ MOV P1 , #38H

การควบคุมลักษณะการแสดงผล

ในการควบคุมลักษณะการแสดงผลจะมีการกำหนดค่าดังนี้

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

D : เป็นการเปิด / ปิดจอ LCD ถ้า D = '0' จอ LCD จะดับ และ D = '1' จอ LCD จะสว่าง

C : ควบคุมการแสดงเคอร์เซอร์ ถ้า C = '0' จะไม่แสดงเคอร์เซอร์ และ c = '1' จะแสดงเคอร์เซอร์

B : เป็นการควบคุมการกระพริบของเคอร์เซอร์ ถ้ากำหนด B = '0' เคอร์เซอร์จะไม่กระพริบ และถ้า B = '1' เคอร์เซอร์จะกระพริบ

ตัวอย่างเช่น ถ้าต้องการให้จอ LCD สว่าง มีการแสดงเคอร์เซอร์และตัวเคอร์เซอร์ต้องกระพริบด้วย จะต้องเขียนโปรแกรมเพื่อควบคุมดังนี้

MOV P1 , #00001111B หรือ MOV P1 , #0FH

การกำหนดโหมด ENTRY เป็นการตั้งค่าเพื่อกำหนดการทำงานของ LCD ว่าจะให้อ่านเขียนข้อมูล หรือกำหนดตำแหน่งเคอร์เซอร์ ซึ่งจะมีรูปแบบการกำหนดค่าดังนี้

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

I / D : เป็นการควบคุมคอนโทรลเลอร์ให้เลื่อนตำแหน่งของเคอร์เซอร์ขึ้นหรือลง หลังจากสั่งให้อ่านหรือเขียนข้อมูลแล้ว ถ้า I / D = '0' เคอร์เซอร์จะลดลง 1 ตำแหน่งและถ้ากำหนด I / D = '1' เคอร์เซอร์จะเลื่อนขึ้น 1 ตำแหน่ง

S : ถ้า S = '0' ข้อมูลจะอยู่กับที่ แต่ตัวเคอร์เซอร์จะเลื่อนไปทางขวา ถ้า S = '1' เคอร์เซอร์จะอยู่กับที่ แต่ข้อมูลจะเลื่อนไปทางซ้าย

เช่น ถ้าต้องการกำหนดให้เคอร์เซอร์เลื่อนขึ้น 1 ตำแหน่ง หลังจากอ่านหรือเขียนข้อมูลแล้ว โดยข้อมูลจะอยู่กับที่ และเคอร์เซอร์เลื่อนไปทางขวา จะต้องเขียนโปรแกรมกำหนดค่า ดังนี้

MOV P1 , #00000110B หรือ MOV P1, #0 6H

การเคลียร์จอแสดงผล เมื่อต้องการเคลียร์ข้อมูลบนจอ LCD จะต้องกำหนดค่าดังนี้

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

จากค่าดังกล่าวคอนโทรลเลอร์จะทำการลบข้อมูลบนจอหมด แล้วเลื่อนเคอร์เซอร์ไปอยู่ที่มุมบนซ้ายของจอ

คอนโทรลเลอร์ของ LCD เมื่อระดับไฟเลี้ยงสูงถึง 4.5 V โดยประมาณ 10 มิลลิวินาที มันจะทำการเซตตัวเอง ดังนี้

1. ลบข้อมูลที่อยู่บนจอ
2. กำหนดฟังก์ชั่นให้มีค่าดังนี้

   DL = '1' : กำหนดให้มีการติดต่อแบบ 8 บิต

   N = '0' : เป็นการแสดงผลแบบ 2 บรรทัด

   F = '0' : แสดงผลขนาด 5 X 7 จุด ต่อ 1 ตัวอักษร
4. กำหนดการแสดงผลให้เป็นดังนี้

   I / D = '1' : เพิ่มขึ้นทีละ 1 ตำแหน่ง

   S = '0' : ไม่เลื่อน

จากหลักการดังกล่าวทำให้การแสดงผลทางจอแสดงผล LCD ต้องมีโปรแกรมย่อยที่เกี่ยวข้องหลายส่วนได้แก่

- โปรแกรมย่อยสำหรับการเตรียมจอแสดงผล LCD (LCD_INIT) สำหรับใช้งาน
- โปรแกรมย่อยสำหรับตรวจสอบจอ LCD ว่าพร้อมทำงานหรือไม่ (WAITBF)
- โปรแกรมย่อยสำหรับการส่งค่าออกไปแสดงผลทางจอ LCD 1 อักษร (WRITE)

<u>โปรแกรมย่อยสำหรับการเตรียมจอ LCD (LCD_INIT) สำหรับใช้งาน</u>

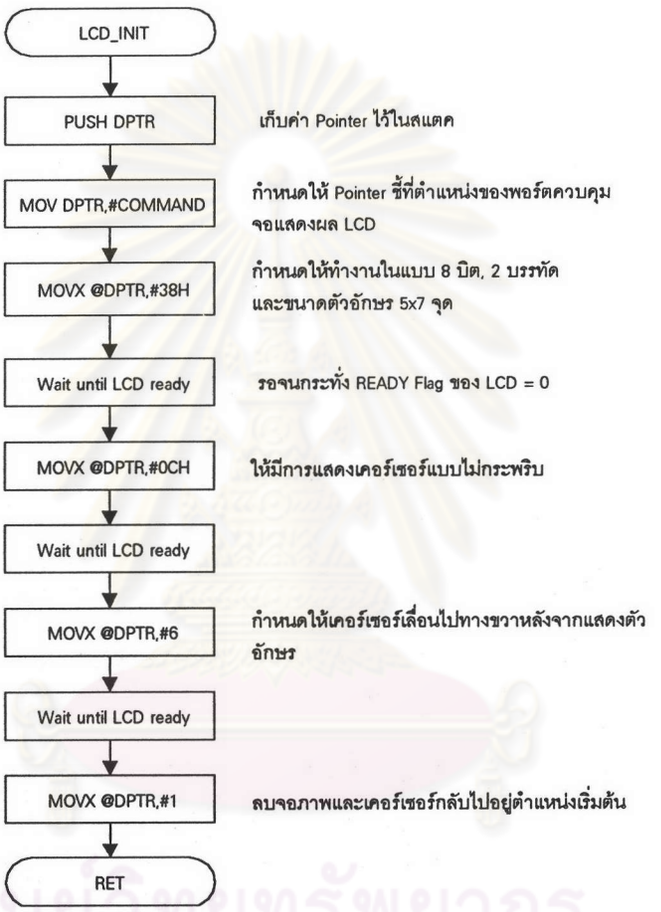
ในการเตรียมจอ LCD เราจะกำหนดให้จอ LCD ทำงานในลักษณะมีการติดต่อแบบ 8 บิต, ขนาดตัวอักษร 5 x 7 จุด, จำนวน 2 บรรทัด, เคอร์เซอร์ไม่กระพริบ, เมื่อเขียนตัวอักษรแล้วเคอร์เซอร์

จะเลื่อนไปทางขวามือ จากข้อกำหนดดังกล่าวจะได้ค่าที่นำไปใช้ในการกำหนดค่าเริ่มต้นของจอ LCD คือ ค่าของการเซ็ทฟังก์ชั่น = 00111000B หรือ 3BH

ค่าในการควบคุมการแสดงผล = 00001111B หรือ 0FH

ค่าในโหมด ENTRY = 00000110BH หรือ 06H และทำการลบจอภาพ

ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับการเตรียมจอ LCD จะแสดงในรูปที่ 4.1



รูปที่ 4.1 การทำงานของโปรแกรมย่อยสำหรับการเตรียมจอ LCD

<u>โปรแกรมย่อยสำหรับตรวจสอบจอ LCD ว่าพร้อมทำงานหรือไม่ (WAITBF)</u>

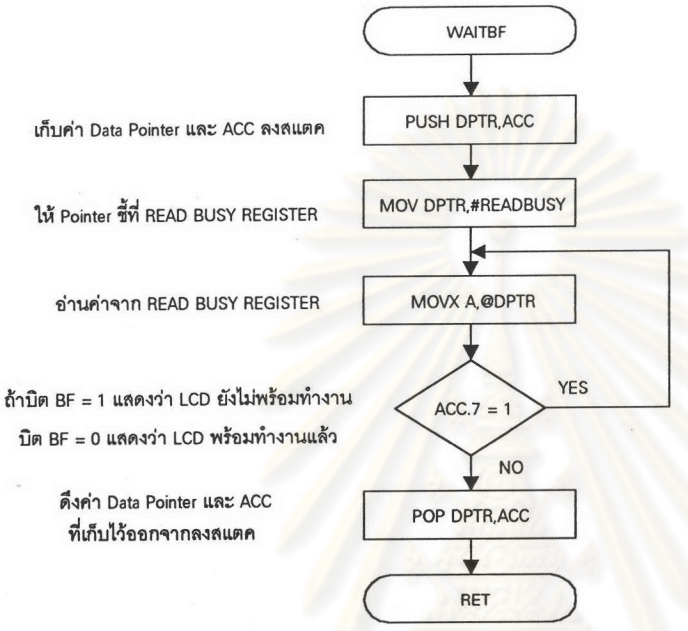
ในการใช้งานจอแสดงผล LCD ไม่ว่าจะเป็นการแสดงตัวอักษรหรือส่งคำสั่งควบคุมต้องมีการตรวจสอบว่าตัวคอนโทรลเลอร์ที่ใช้ควบคุมการแสดงผล LCD พร้อมที่จะทำงานหรือยัง โดย

การดูได้จาก Busy Flag (บิตBF) ของตัวคอนโทรลเลอร์ (ที่ตำแหน่ง 0E61H) โดยที่

BF = 1 หมายถึง อยู่ในขบวนการทำงานภายในไม่พร้อมจะรับข้อมูลหรือคำสั่ง

BF = 0 หมายถึง พร้อมจะรับข้อมูลหรือคำสั่งได้

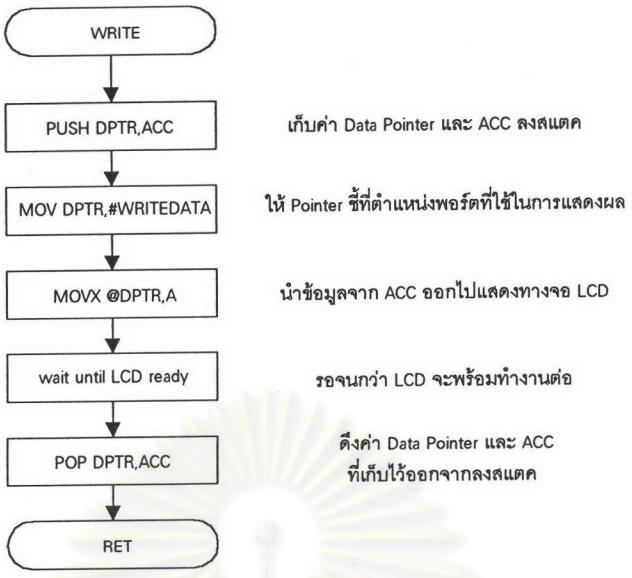
ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับการตรวจสอบ Busy Flag แสดงในรูปที่ 4.2



รูปที่ 4.2 ขั้นตอนการทำงานของโปรแกรมย่อยหรับการตรวจสอบ Busy Flag

<u>โปรแกรมย่อยสำหรับการส่งค่าออกไปแสดงผลทางจอ LCD 1 อักษร (WRITE)</u>

การแสดงตัวอักษรทางจอแสดงผล LCD นั้นสามารถทำได้โดย การส่งค่าตัวอักษรที่ต้องการไปยังส่วนควบคุมการแสดงผลของตัวคอนโทรลเลอร์ (ตำแหน่ง 0E62H) ขั้นตอนการทำงานของโปรแกรมย่อยในการแสดงตัวอักษรทางจอ LCD แสดงในรูปที่ 4.3



รูปที่ 4.3 โปรแกรมย่อยในการแสดงตัวอักษรทางจอ LCD

4.2.3 โปรแกรมบริการการเกิดการการขัดจังหวะ

ในโปรแกรมที่ใช้ควบคุมไมโครคอนโทรลเลอร์ได้กำหนดให้ไมโครคอนโทรลเลอร์ตอบสนองต่อสัญญาณขัดจังหวะที่เกิดขึ้น 3 ประเภทด้วยกัน คือ

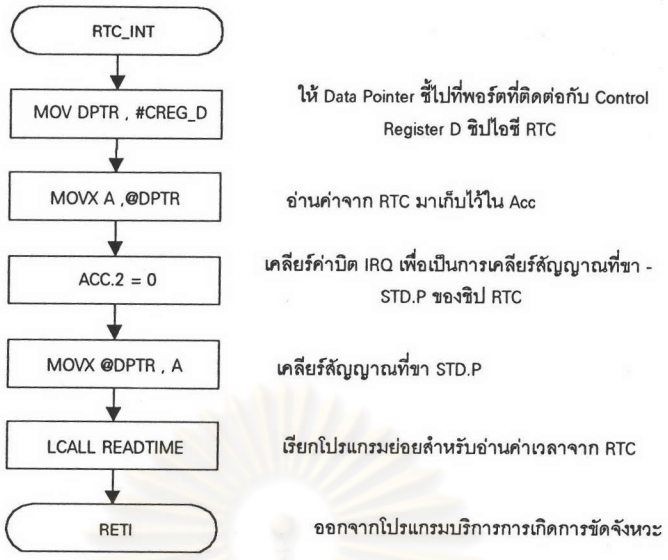
- การขัดจังหวะที่เกิดจากพอร์ตสื่อสารอนุกรม การการขัดจังหวะจากพอร์ตสื่อสารอนุกรมจะเกิดขึ้นเมื่อไมโครคอนโทรลเลอร์พบว่าบิต RI ในรีจีสเตอร์ใช้งานเฉพาะ SCON มีค่าเป็น 1 (บิต RI ถูกเซ็ทเมื่อได้รับข้อมูลจากภายนอก) ไมโครคอนโทรลเลอร์จะย้ายการทำงานมายังโปรแกรมบริการการขัดจังหวะที่ตำแหน่ง 0023H ซึ่งขั้นตอนการทำงานในโปรแกรมบริการการขัดจังหวะของ พอร์ตสื่อสารอนุกรมจะเป็นเพียงการนำค่าจากรีจีสเตอร์ใช้งานเฉพาะ SBUF มาไว้ใน ACC (ค่าที่อยู่ใน ACC จะเป็นคำสั่งจากเครื่องคอมพิวเตอร์ที่ใช้ควบคุมว่าจะให้ไมโครคอนโทรลเลอร์ทำงานอะไร) จากนั้นก็จะกระโดดไปยังฟังก์ชั่น CHECK ที่ทำหน้าที่ตรวจสอบว่าค่าใน ACC เป็นค่าอะไรเพื่อให้ไมโครคอนโทรลเลอร์ไปทำงายยังฟังก์ชั่นที่เหมาะสมต่อไป

การทำงานของโปรแกรมบริการการขัดจังหวะที่เกิดจากพอร์ตสื่อสารอนุกรมแสดงในรูปที่ 4.4

รูปที่ 4.4 การทำงานของโปรแกรมบริการการเกิดการขัดจังหวะที่เกิดจากพอร์ตสื่อสารอนุกรม

- การขัดจังหวะที่เกิดจากสัญญาณภายนอกชนิดที่ 0

ไมโครคอนโทรลเลอร์จะทราบถึงการเกิดการขัดจังหวะชนิดที่ 0 โดยการตรวจสอบจากบิต IE0 ในรีจีสเตอร์ใช้งาน TCON วิธีการตรวจสอบชนิดของสัญญาณขัดจังหวะจะถูกกำหนดให้เป็นแบบตรวจสอบจากการเปลี่ยนระดับของสัญญาณจาก 1 เป็น 0 (บิต IT0 = 1) ในการวิจัยครั้งนี้ การขัดจังหวะที่เกิดจากสัญญาณภายนอกชนิดที่ 0 จะนำไปใช้ในการขัดจังหวะที่เกิดจากชิปไอซี Real Time Clock ซึ่งเมื่อเกิดการขัดจังหวะขึ้น (บิต IE0 ถูกเซ็ท) ไมโครคอนโทรลเลอร์ก็จะย้ายการทำงานไปยังโปรแกรมบริการการเกิดการขัดจังหวะที่ตำแหน่ง 0003H ซึ่งจะมีคำสั่งควบคุมให้กระโดดไปยังโปรแกรมย่อย RTC_INT ในโปรแกรมย่อยส่วนนี้จะมีการเคลียร์ค่าของบิต IRQ Flag (D2) ใน Control D Register แล้วกระโดดไปทำงานยังโปรแกรมย่อยสำหรับอ่านค่าจากวงจร Real Time Clock (READTIME) จึงจะออกจากโปรแกรมบริการการเกิดการขัดจังหวะ (สัญญาณที่ขา STD.P ของชิป Real Time Clock MSM6242 และบิต IRQ Flag จะเป็น 1 เมื่อเกิดการขัดจังหวะและจะคงค่าอยู่เพื่อดูค่าอยู่จนกว่าบิต IRQ Flag จะถูกเคลียร์) การทำงานของโปรแกรมบริการการเกิดการ ขัดจังหวะจากสัญญาณภายนอกชนิดที่ 0 แสดงในรูปที่ 4.5



รูปที่ 4.5 การทำงานของโปรแกรมบริการการเกิดการขัดจังหวะจากชิป Real Time Clock

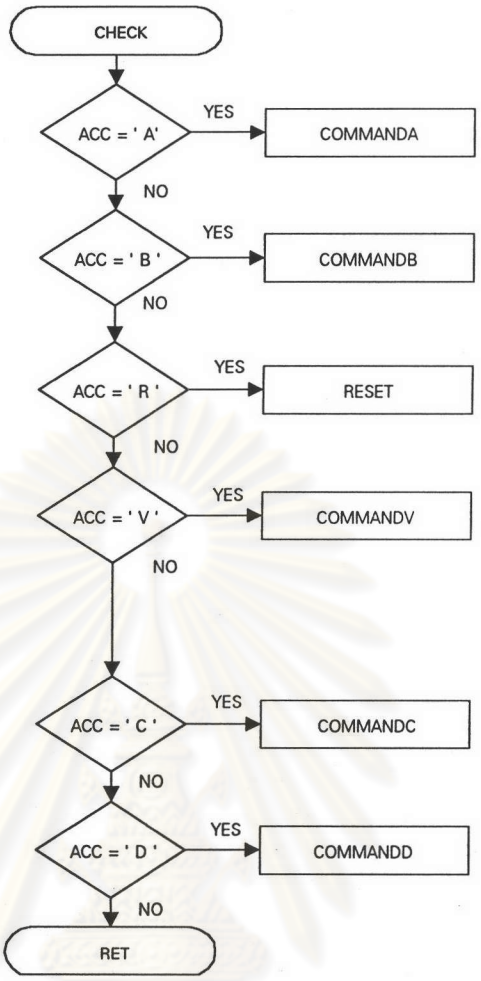
- การขัดจังหวะที่เกิดจากสัญญาณภายนอกชนิดที่ 1

วิธีการที่ไมโครคอนโทรลเลอร์จะตรวจสอบถึงการการการขัดจังหวะจากสัญญาณภายนอกชนิดที่ 1 จะมีขั้นตอนเหมือนกับการตรวจสอบการขัดจังหวะของสัญญาณภายนอกชนิดที่ 0 ซึ่งในการวิจัยครั้งนี้จะใช้กับการขัดจังหวะที่เกิดขึ้นจากคีย์บอร์ด เมื่อเกิดสัญญาณขัดจังหวะจากชิปไอซี 74C923 (ขา DA เปลี่ยนจาก 0 เป็น 1) ไมโครคอนโทรลเลอร์จะย้ายการทำงานไปยังโปรแกรมบริการการเกิดการขัดจังหวะที่เกิดจากสัญญาณภายนอกชนิดที่ 1 ตำแหน่ง 0013H ที่มีคำสั่งให้กระโดดไปยังโปรแกรมย่อย KEY_INT ในโปรแกรมย่อยส่วนนี้จะมีการเคลียร์ค่าของบิต EA เพื่อเป็นการป้องกันไม่ให้ไมโครคอนโทรลเลอร์ตอบสนองต่อสัญญาณขัดจังหวะที่เกิดขึ้นตามมาก่อนที่โปรแกรมบริการการเกิดการขัดจังหวะนี้จะถูกกระทำเสร็จ จากนั้นก็จะทำการอ่านค่าของ คีย์บอร์ดที่ถูกกดมาเก็บไว้ใน ACC (เลือกใช้แค่ 4 บิต โดยการ AND ด้วยค่า 0FH) และรีจีสเตอร์ใช้งานทั่วไป R5 จึงออกจากโปรแกรมบริการการขัดจังหวะ การทำงานของโปรแกรมบริการการขัดจังหวะที่เกิดจากคีย์บอร์ดแสดงในรูปที่ 4.6

รูปที่ 4.6 การทำงานของโปรแกรมบริการการเกิดการการขัดจังหวะที่เกิดจากคีย์บอร์ด

4.2.4 โปรแกรมย่อยสำหรับตรวจสอบข้อมูลควบคุมที่ส่งมาทางพอร์ตสื่อสารอนุกรม

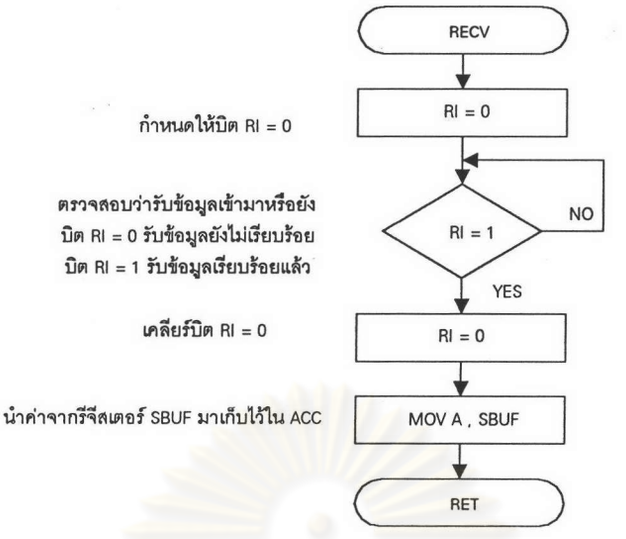
โปรแกรมย่อยที่ใช้ในการตรวจสอบข้อมูลควบคุมที่ส่งมาทางพอร์ตสื่อสารอนุกรมนั้น จะใช้โปรแกรมย่อยที่ชื่อ CHECK โดยจะถูกเรียกใช้จากโปรแกรมบริการการเกิดการขัดจังหวะที่ เกิดจากพอร์ตสื่อสารอนุกรม การทำงานของโปรแกรมย่อยในส่วนนี้จะนำค่าใน ACC (จากโปรแกรมบริการการเกิดการขัดจังหวะ) มาทำการเปรียบเทียบกับค่าที่กำหนด ซึ่งถ้าเปรียบเทียบแล้วพบว่าตรงกับค่าใดก็จะกระโดดไปทำโปรแกรมย่อยที่เหมาะสมต่อไป การทำงานของโปรแกรมย่อยในส่วนนี้จะแสดงในรูปที่ 4.7



รูปที่ 4.7 แสดงขั้นตอนของการตรวจสอบข้อมูลควบคุมการทำงานที่ส่งมาจากคอมพิวเตอร์

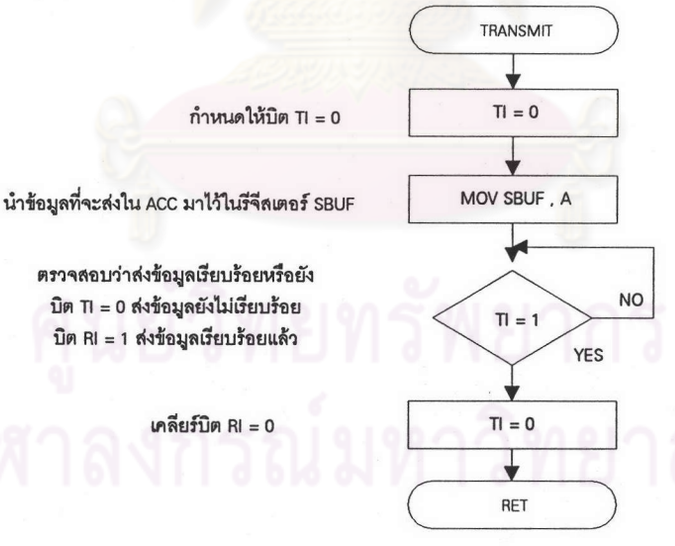
4.2.5 โปรแกรมย่อยสำหรับรับและส่งข้อมูลกับคอมพิวเตอร์

การทำงานของโปรแกรมย่อยในส่วนที่ทำหน้าที่รับข้อมูลนั้นเมื่อมีข้อมูลส่งเข้ามาทางพอร์ตสื่อสารอนุกรม (บิต RI จะถูกเซ็ท) ข้อมูลดังกล่าวจะเก็บอยู่ในรีจีสเตอร์ ใช้งานเฉพาะ SBUF ซึ่งต้องมีการย้ายค่าที่อยู่ในรีจีสเตอร์ SBUF ไปเก็บไว้ในรีจีสเตอร์ A เพื่อนำไปประมวลผลต่อไป ขั้นตอนของการรับข้อมูลทางพอร์ตสื่อสารอนุกรมแสดงในรูปที่ 4.8



รูปที่ 4.8 การรับข้อมูลทางพอร์ตสื่อสารอนุกรม

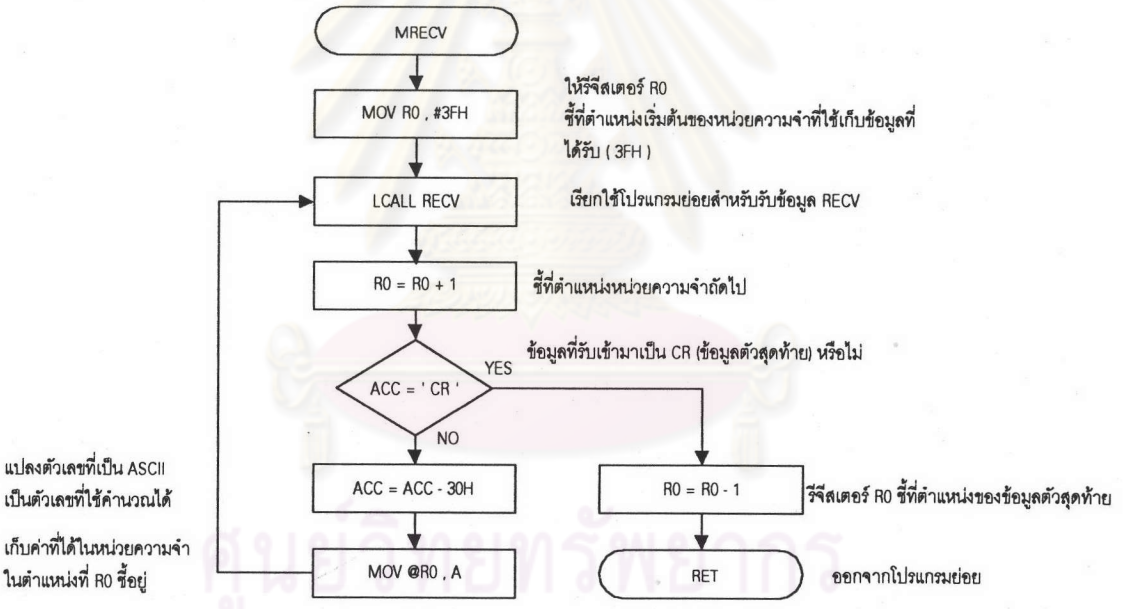
ส่วนในการส่งข้อมูลออกทางพอร์ตสื่อสารอนุกรมก็จะมีหลักการทำงานที่ตรงกันข้ามกัน คือ เมื่อต้องการส่งข้อมูล ข้อมูลที่ต้องการส่งที่อยู่ในรีจีสเตอร์ A จะถูกย้ายไปที่รีจีสเตอร์ SBUF เมื่อส่งข้อมูลเรียบร้อยแล้วบิต TI จะถูกเซ็ท การส่งข้อมูลทางพอร์ตสื่อสารอนุกรมแสดงในรูปที่ 4.9



รูปที่ 4.9 การส่งข้อมูลทางพอร์ตสื่อสารอนุกรม

โปรแกรมย่อยสำหรับรับข้อมูลอีกส่วนหนึ่งที่ใช้ในโปรแกรมควบคุมการทำงานของไมโครคอนโทรลเลอร์ คือโปรแกรมย่อยที่ใช้รับข้อมูลที่เป็นตัวเลขที่เครื่องคอมพิวเตอร์ที่ใช้เป็นตัวควบคุมส่งมา เนื่องจากการสื่อสารระหว่างเครื่องคอมพิวเตอร์และไมโครคอนโทรลเลอร์ผ่านทางพอร์ตสื่อสารอนุกรมนั้น ข้อมูลที่ส่งจะเป็นแบบตัวอักษรเท่านั้น (character) เช่น สมมุติว่าต้องการส่งข้อมูลที่เป็นค่าตัวเลข 255 ข้อมูลที่จะส่งออกไปจะเป็น '2', '5', '5' และตามด้วยข้อมูลที่บอกว่าหมดข้อมูลที่จะส่ง ในที่นี้จะเป็นรหัส 'CR' เมื่อได้ค่าที่ส่งมาแล้วจึงแปลงค่าที่รับได้เป็นค่าตัวเลขจริงต่อไป

การทำงานของโปรแกรมย่อยที่ใช้รับข้อมูลจากคอมพิวเตอร์นี้ ใช้ตำแหน่งหน่วยความจำของไมโครคอนโทรลเลอร์ตำแหน่งตั้งแต่ 40H - 44H เป็นตัวเก็บข้อมูล โดยใช้รีจีสเตอร์ R1 เป็นตัวชี้ตำแหน่ง เมื่อรับข้อมูลเข้ามา ค่าที่รับเข้ามาจะถูกลบด้วยค่า 30H เพื่อแปลงค่าตัวอักษรให้เป็นตัวเลขแล้วเก็บไว้ในหน่วยความจำที่รีจีสเตอร์ R1 ชี้อยู่ (ข้อมูลตัวแรกถูกเก็บที่ตำแหน่ง 40H) ในโปรแกรมจะมีการตรวจสอบว่าข้อมูลนั้นมีค่าเป็น CR หรือไม่ ถ้าไม่ใช่ จะคอยรับข้อมูลตัวต่อไปที่จะส่งมา ถ้าข้อมูลเป็นรหัส 'CR' แสดงว่าเป็นข้อมูลตัวสุดท้ายจึงออกจากโปรแกรม

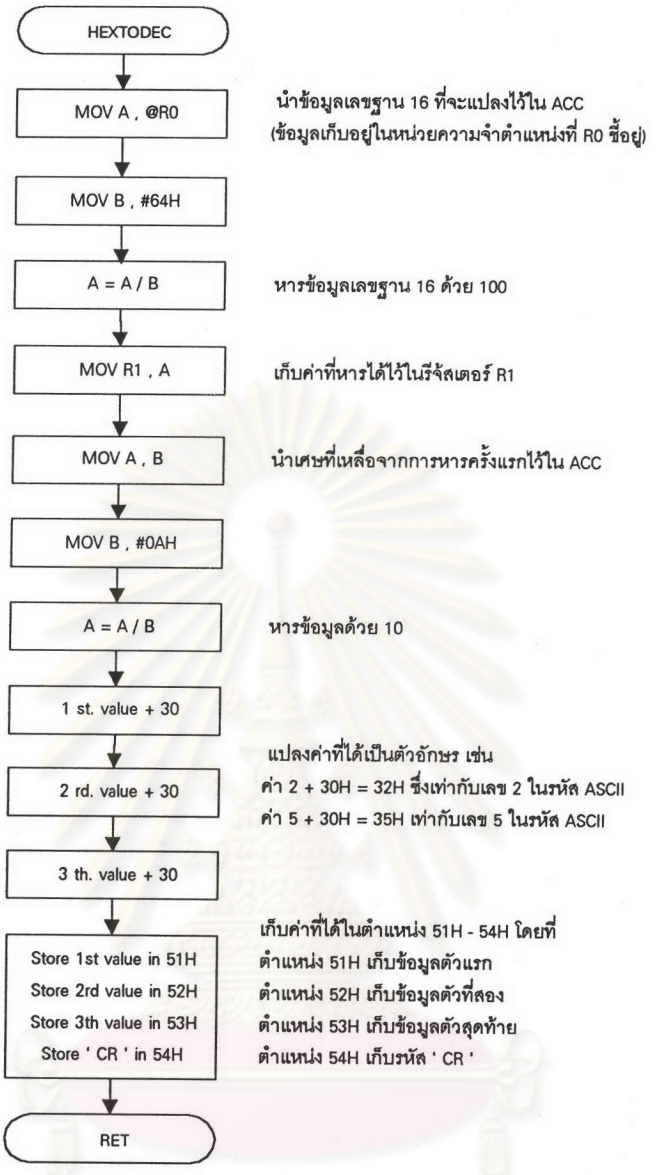


รูปที่ 4.10 การทำงานของโปรแกรมย่อยสำหรับรับข้อมูลที่ส่งจากคอมพิวเตอร์

4.2.6 โปรแกรมย่อยสำหรับแปลงเลขฐาน 16 เป็นฐาน 10 (HEXTODEC)

ในการแสดงข้อมูลทางจอภาพไม่ว่าจะเป็นบนจอคอมพิวเตอร์หรือบนจอ LCD การแสดงข้อมูลในรูปของเลขฐาน 10 จะทำให้ผู้ใช้เข้าใจได้ดีกว่าเลขฐาน 16 ดังนั้นในโปรแกรมย่อยบางโปรแกรมที่มีการแสดงผล จึงต้องมีการแปลงข้อมูลจากเลขฐาน 16 ให้เป็นเลขฐาน 10 ก่อน โดยเรียกใช้โปรแกรม HEXTODEC

การทำงานของโปรแกรม จะนำข้อมูลเลขฐาน 16 ที่ต้องการแปลง ในตำแหน่งหน่วยความจำตำแหน่งที่รีจีสเตอร์ R0 ชี้อยู่มาเก็บที่ ACC ข้อมูลใน ACC จะถูกหารด้วย 100 เพื่อให้ได้ค่าตัวแรกที่เป็นหลักร้อย จากนั้นเศษจากการหารครั้งแรกจะถูกหารด้วย 10 อีกครั้งจากการหารทั้งสองครั้งจะได้ข้อมูลที่เป็นหลักร้อย, หลักสิบ และหลักหน่วย เก็บอยู่ในรีจีสเตอร์ R1, ACC และ B ตามลำดับ ซึ่งค่าเหล่านี้จะบวกด้วย 30H เพื่อแปลงให้เป็นรหัส ASCII แล้วเก็บไว้ในหน่วยความจำตำแหน่งที่ 51H,52H, 53H ส่วนในตำแหน่งที่ 54H จะใช้เก็บรหัส CR เพื่อเป็นตัวบอกจุดสิ้นสุดของข้อมูล เมื่อต้องการแสดงผลข้อมูลจะสามารถนำข้อมูลที่ตำแหน่งดังกล่าวไปแสดงได้



รูปที่ 4.11 แสดงขั้นตอนการทำงานของโปรแกรมย่อยสำหรับแปลงเลขฐาน 16 เป็นฐาน 10

4.2.7 โปรแกรมย่อยสำหรับอ่านสัญญาณอินพุตแบบแอนะล็อก

ในการอ่านข้อมูลอินพุตแบบแอนะล็อกจะใช้โปรแกรมย่อย COMMANDV เป็นโปรแกรมควบคุม รีจีสเตอร์ที่ใช้ในโปรแกรมส่วนนี้มีดังต่อไปนี้

รีจีสเตอร์ R4 : ใช้เป็นตัวบอกลำดับของวงจรแปลงสัญญาณแอนะล็อกที่กำลังใช้งาน

รีจีสเตอร์ R3 : เป็นตัวกำหนดการทำงานให้กับวงจร ADC (เป็นตัวส่งสัญญาณ CS)

รีจีสเตอร์ R1 : นับจำนวนครั้งในการหาค่าเฉลี่ย

การทำงานของวงจรจะเริ่มจากการส่งค่าในรีจีสเตอร์ R3 มายังชิปไอซี 8255 ทำให้ค่าที่ PB1, PB2, PB3, PB4 มีค่าตามค่าของ R3 ซึ่งสถานะที่ขา PB1, PB2, PB3 ของ 8255 จะเป็นตัวกำหนดว่าจะให้วงจร ADC วงจรใดทำงานโดยจะเริ่มจากตัวที่ 1 แล้วค่อยๆไล่ไปจนครบ 8 ตัวจึงค่อยวนกลับมาที่ตัวที่ 1 อีกครั้ง ส่วนสถานะที่ขา PB4 จะเป็นตัวกำหนดจังหวะการทำงานของชิป ADC โดยเมื่อสถานะที่ขา PB4 เป็น 0 แล้วกลับเป็น 1 ใหม่ (ต้องหน่วงเวลาให้มีสถานะเป็น 0 อย่างน้อย 1 ms) ชิป ADC จะเริ่มทำการแปลงข้อมูล วงจร ADC จะถูกกำหนดให้อ่านค่าจากอินพุตทั้งหมด 8 ครั้งเพื่อที่จะนำมาหาค่าเฉลี่ยซึ่งค่าที่อ่านได้แต่ละครั้งจะเก็บไว้ที่หน่วยความจำตำแหน่งที่ R0 ชี้อยู่ (D0H - D7H) ค่าทั้งหมดจะถูกนำไปใช้หาค่าเฉลี่ยโดยโปรแกรมย่อย AVERAGE และค่าเฉลี่ยที่วงจร ADC แต่ละวงจรอ่านได้จะเก็บไว้ในหน่วยความจำตำแหน่ง 90H - 9FH เพื่อนำไปใช้งานต่อไป

เพื่อเป็นการกำหนดจังหวะการทำงานระหว่างคอมพิวเตอร์ที่ใช้ควบคุมกับไมโครคอนโทรลเลอร์ทุกครั้งที่การทำงานในแต่ละส่วนสิ้นสุดลงไมโครคอนโทรลเลอร์จะส่งสัญญาณที่อาจเป็นรหัส 'CR' ไปยังคอมพิวเตอร์ที่เป็นตัวควบคุม และคอมพิวเตอร์ที่เป็นตัวควบคุมจะส่งสัญญาณควบคุมกลับมาเพื่อให้ไมโครคอนโทรเลอร์ทำงานต่อโดยข้อมูลที่คอมพิวเตอร์ที่เป็นตัวควบคุมส่งมาเพื่อใช้ควบคุมการทำงานของโปรแกรมย่อยที่ใช้อ่านค่าอินพุตแบบแอนะล็อกจะมีดังต่อไปนี้

- ค่าที่ส่งมาเป็นตัวอักษร W หมายถึง ให้ทำโปรแกรมย่อยสำหรับตรวจสอบว่าค่าที่อ่านได้เกินกว่าที่กำหนดหรือไม่ โปรแกรมจะไปทำงานในโปรแกรม LIMIT

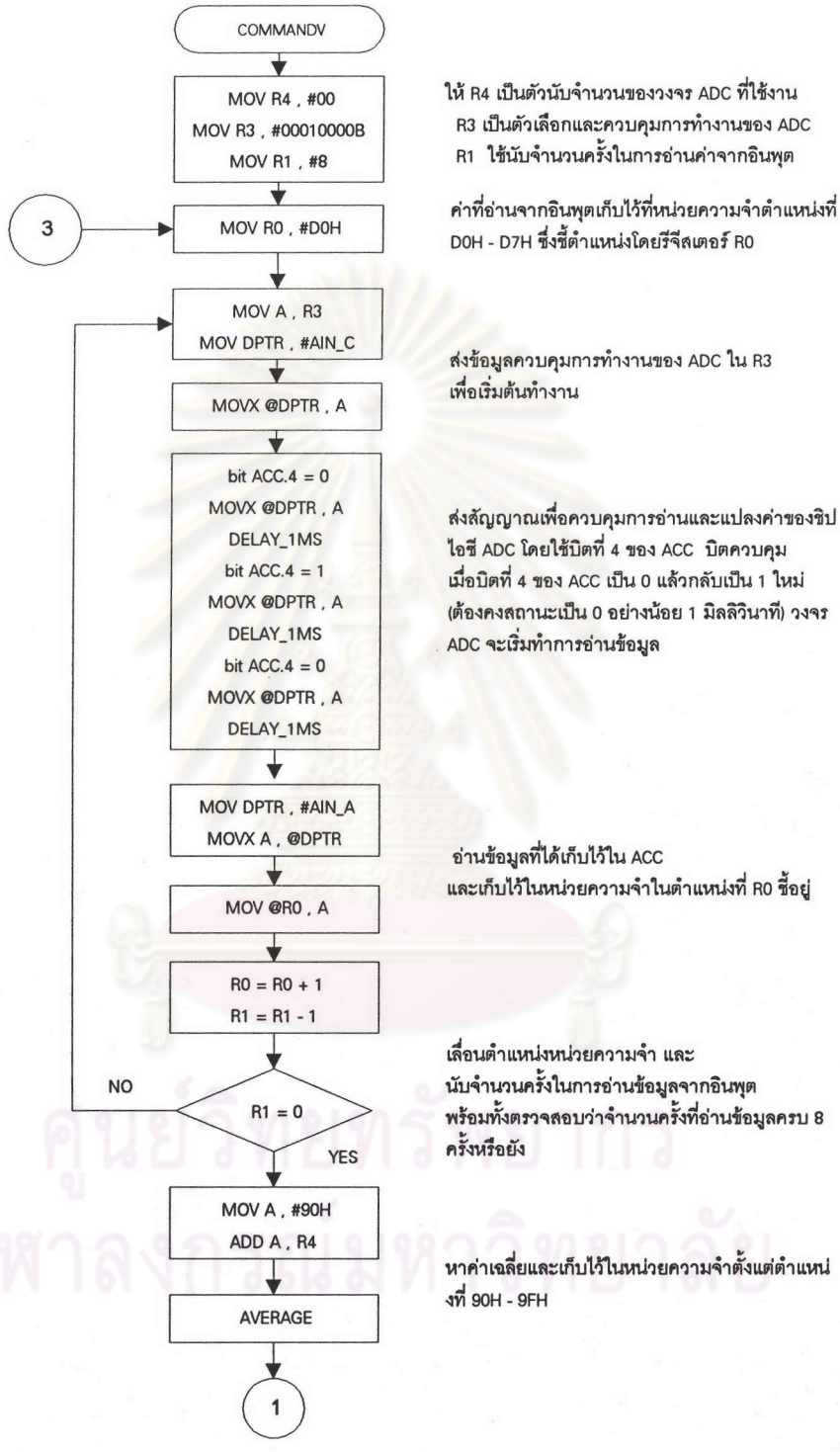
- ค่าที่ส่งมาเป็นตัวอักษร L หมายถึง ให้ไมโครคอนโทรลเลอร์ส่งค่า Limit ของวงจร ADC วงจรนั้นไปแสดงผลทางจอคอมพิวเตอร์

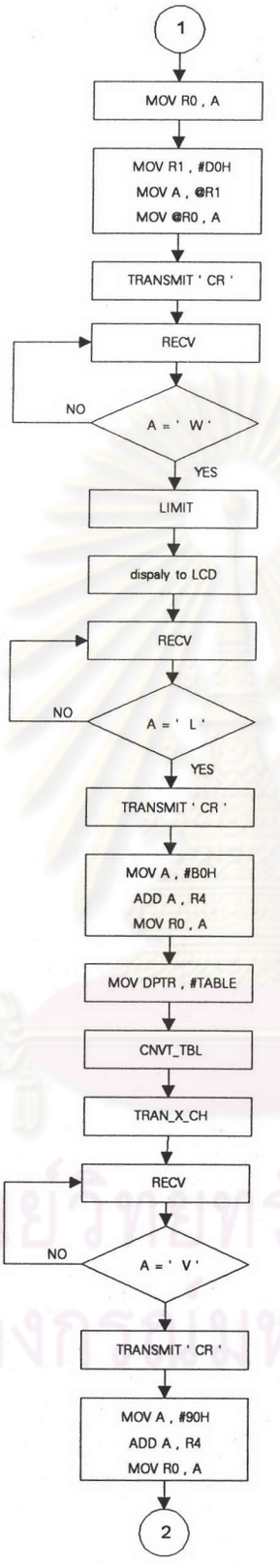
- ค่าที่ส่งมาเป็นตัวอักษร V หมายถึง ให้ไมโครคอนโทรลเลอร์ส่งค่าเฉลี่ยของวงจร ADC วงจรนั้นไปแสดงผลทางจอคอมพิวเตอร์

- ค่าที่ส่งมาเป็นตัวอักษร M หมายถึง ให้ไมโครคอนโทรลเลอร์ส่งค่าสูงสุดของวงจร ADC วงจรนั้นไปแสดงผลทางจอคอมพิวเตอร์

เมื่อทำการอ่านข้อมูลจากวงจร ADC วงจรแรกเสร็จ ค่าของ R3 (PB1, PB2, PB3, PB4) จะถูกกำหนดให้เป็นค่าใหม่เพื่ออ่านค่าจากวงจร ADC ตัวต่อๆไปจนกว่าจะครบตามจำนวนของ

วงจร ADC ที่กำหนดไว้ (จำนวนวงจร ADC จะเก็บไว้ในหน่วยความจำตำแหน่งที่ 70H) ขั้นตอนการทำงานของโปรแกรมจะแสดงในรูปที่ 4.12





ส่งตัวอักษร 'CR'
เพื่อบอกคอมพิวเตอร์พร้อมทำงานต่อ

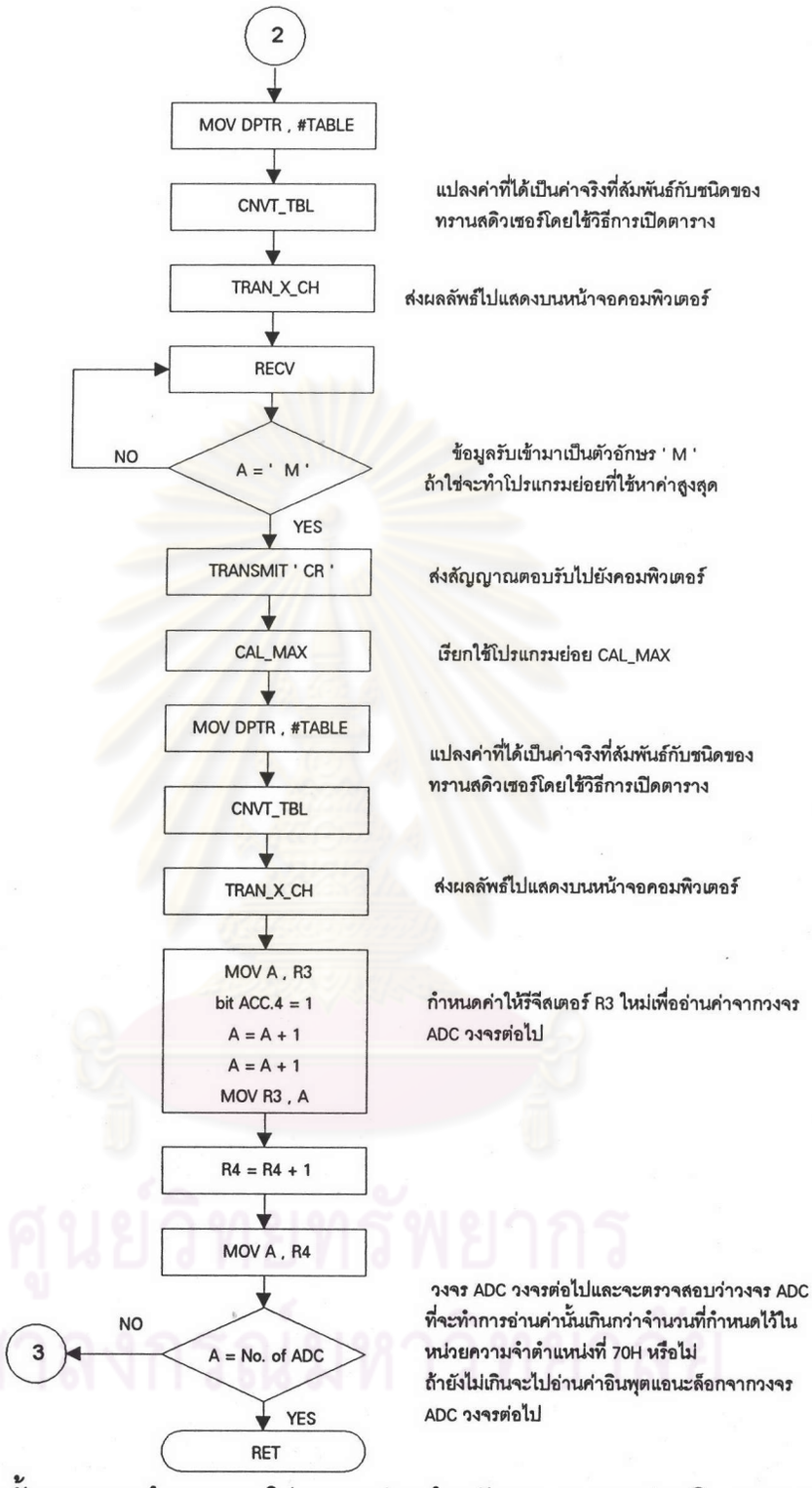
คอมพิวเตอร์ส่งข้อมูลเพื่อให้ทำการคำนวณว่าค่า
อินพุตที่อ่านเข้ามาเกินกำหนดหรือไม่

แสดงข้อมูลทางจอ LCD

คอมพิวเตอร์ส่งข้อมูลมาเพื่อให้ส่งค่า limit
ไปแสดงบนจอคอมพิวเตอร์

แปลงค่า Limit ที่เก็บไว้แล้วแสดงส่งไปแสดง
บนจอคอมพิวเตอร์

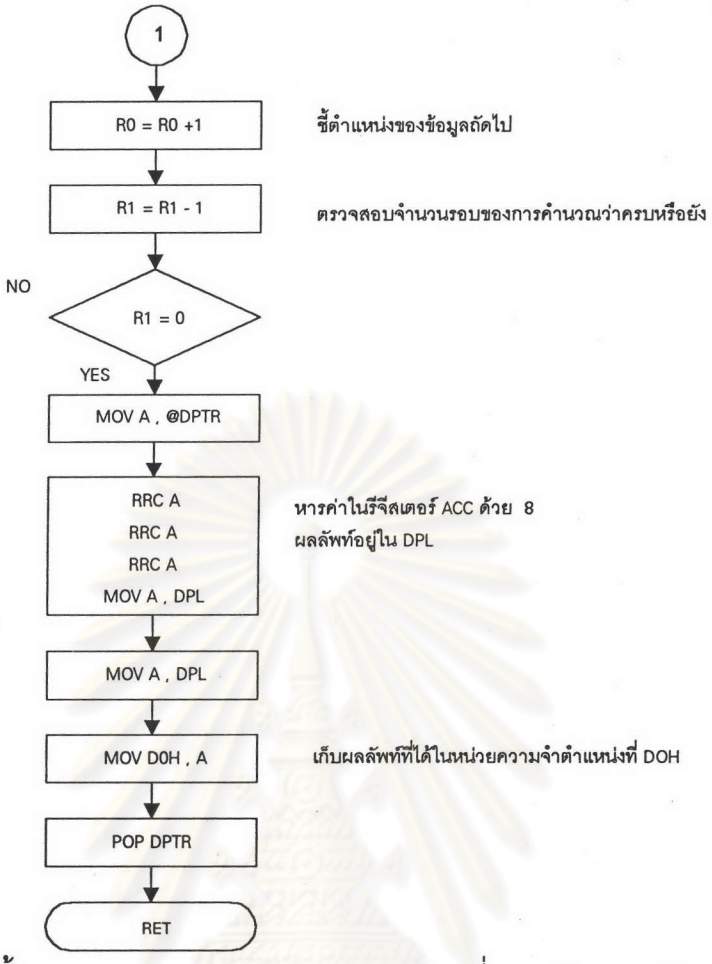
คอมพิวเตอร์ส่งข้อมูลมาเพื่อให้ส่งค่าที่อ่านได้
(ค่าเฉลี่ย) ไปแสดงบนจอคอมพิวเตอร์



รูปที่ 4.12 แสดงขั้นตอนการทำงานของโปรแกรมย่อยสำหรับควบคุมการอ่านอินพุตแบบแอนะล็อก

4.2.8 โปรแกรมย่อยสำหรับหาค่าเฉลี่ยของวงจร ADC

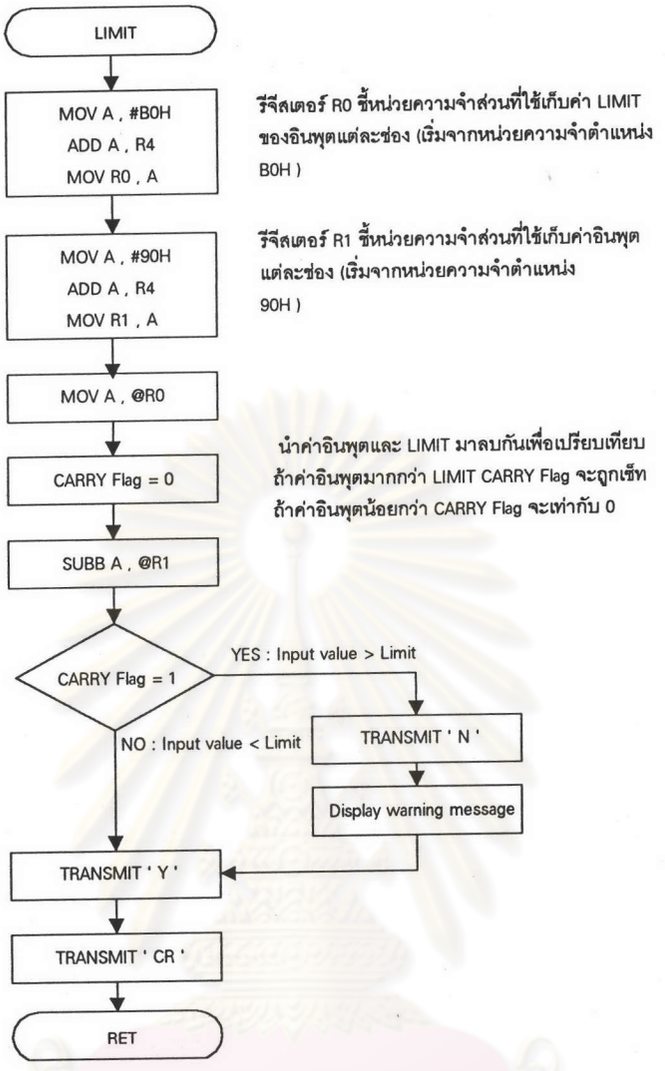
ในการแปลงค่าอินพุตแบบแอนะล็อกเป็นดิจิทัลนั้น อาจเกิดความผิดพลาดขึ้นเนื่องจากค่าความผิดพลาดที่เกิดจากอุปกรณ์ที่นำมาประกอบเป็นวงจร ดังนั้นเพื่อให้ข้อมูลที่ได้มีความเที่ยงตรงยิ่งขึ้น การอ่านค่าจากเซนเซอร์ของวงจร ADC จะอ่านเซนเซอร์ละ 8 ครั้ง โดยเก็บข้อมูลไว้ในหน่วยความจำตำแหน่ง D0H - D7H นำมาหาค่าเฉลี่ย แล้วเก็บไว้ในหน่วยความจำตำแหน่งที่ D0H ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับหาค่าเฉลี่ยแสดงในรูปที่ 4.13



รูปที่ 4.13 แสดงขั้นตอนการทำงานของโปรแกรมหาค่าเฉลี่ยของสัญญาณอินพุตแบบแอนะล็อก

4.2.9 โปรแกรมย่อยสำหรับตรวจสอบสัญญาณอินพุตแบบแอนะล็อกว่าเกินกำหนดหรือไม่

หน้าที่ของโปรแกรมในสิอนนี้จะทำหน้าที่ตรวจสอบว่าสัญญาณอินพุตแบบแอนะล็อกที่วงจร ADC อ่านเข้ามามีค่าเกินกว่าค่าที่กำหนดเอาไว้หรือไม่ โดยนำค่า อินพุตที่อ่านได้มาเปรียบเทียบกับค่า LIMIT ของอินพุตช่องนั้น (LIMIT ของอินพุตแต่ละช่องจะเก็บไว้ในหน่วยความจำตำแหน่ง B0H - BFH) ถ้าสัญญาณอินพุตที่อ่านได้มีค่าสูงกว่ากำหนด โปรแกรมส่วนนี้จะส่งข้อมูล (ตัวอักษร 'N') ไปแจ้งให้คอมพิวเตอร์ทำหน้าที่ควบคุมทราบ ถ้าค่าอินพุตที่อ่านได้ไม่เกินกว่าค่าที่กำหนดโปรแกรมจะส่งตัวอักษร 'Y' ไปยังคอมพิวเตอร์ที่เป็นตัวควบคุมเช่นกัน การทำงานของโปรแกรมย่อยในส่วนนี้แสดงในรูปที่ 4.14

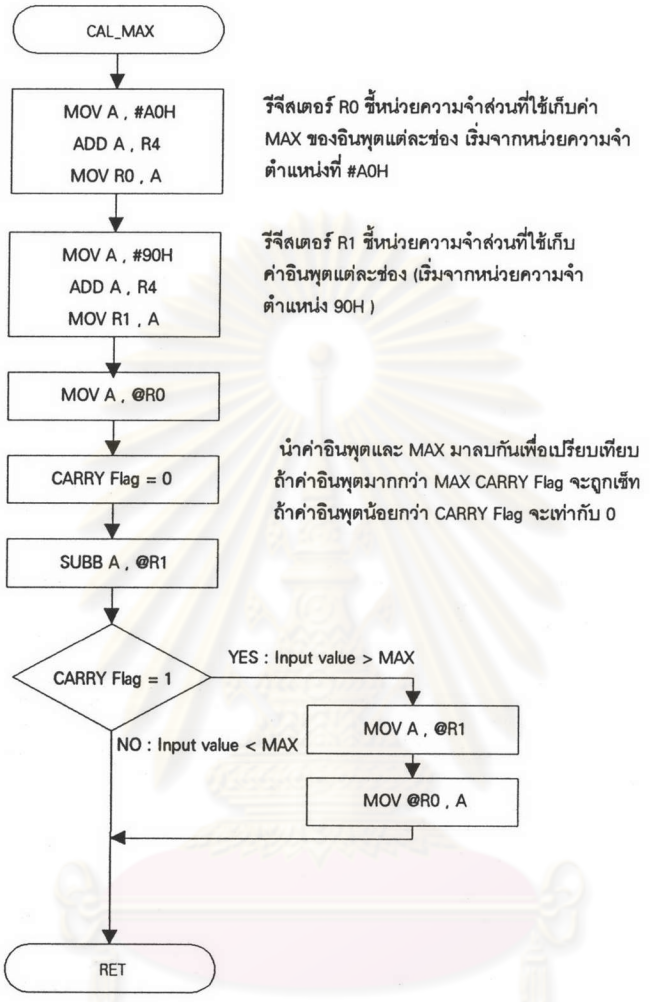


รูปที่ 4.14 ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับเปรียบเทียบค่าอินพุตกับ LIMIT

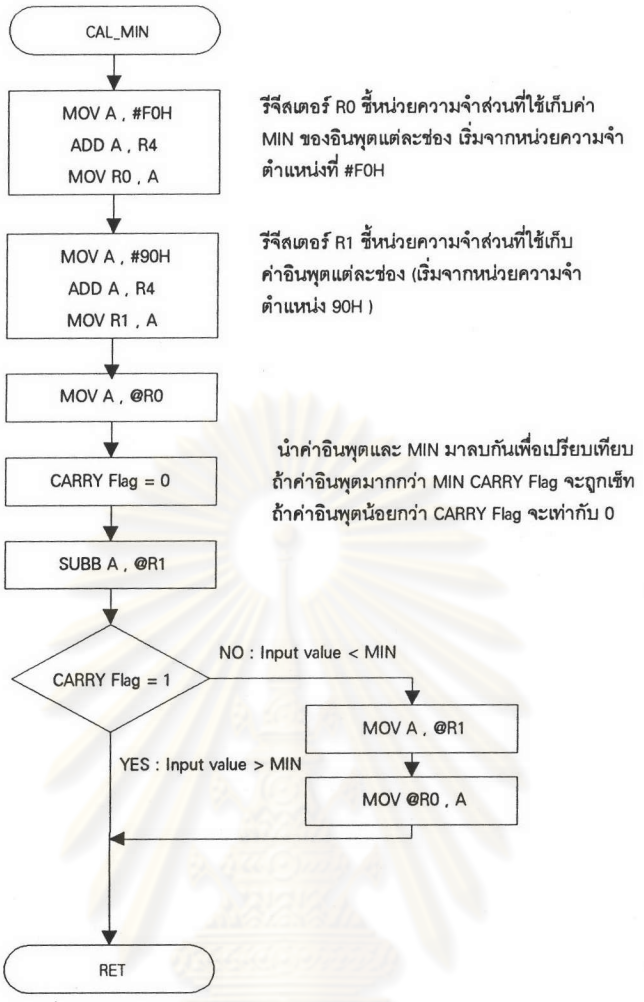
4.2.10 โปรแกรมย่อยสำหรับหาค่าสูงสุดและค่าต่ำสุดของสัญญาณอินพุตแบบแอนะล็อก

ในโปรแกรมส่วนนี้ทำหน้าที่เปรียบเทียบค่าของสัญญาณอินพุตแบบแอนะล็อกว่ามีค่าสูงสุดและต่ำสุดเป็นเท่าใร ค่าที่คำนวณได้ (ค่าสูงสุดหรือค่าต่ำสุด) เก็บอยู่ในหน่วยความจำตำแหน่งที่ A0H - AFH สำหรับค่าสูงสุด และที่ตำแหน่ง F0H - FFH สำหรับค่าต่ำสุด (แล้วแต่ว่าจะต้องการอ่าน

ข้อมูลอินพุตของวงจร ADC ตัวใด) ขั้นตอนการทำงานของโปรแกรมแสดงในรูปที่ 4.15 และ 4.16



รูปที่ 4.15 ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับคำนวณหาค่าสูงสุด

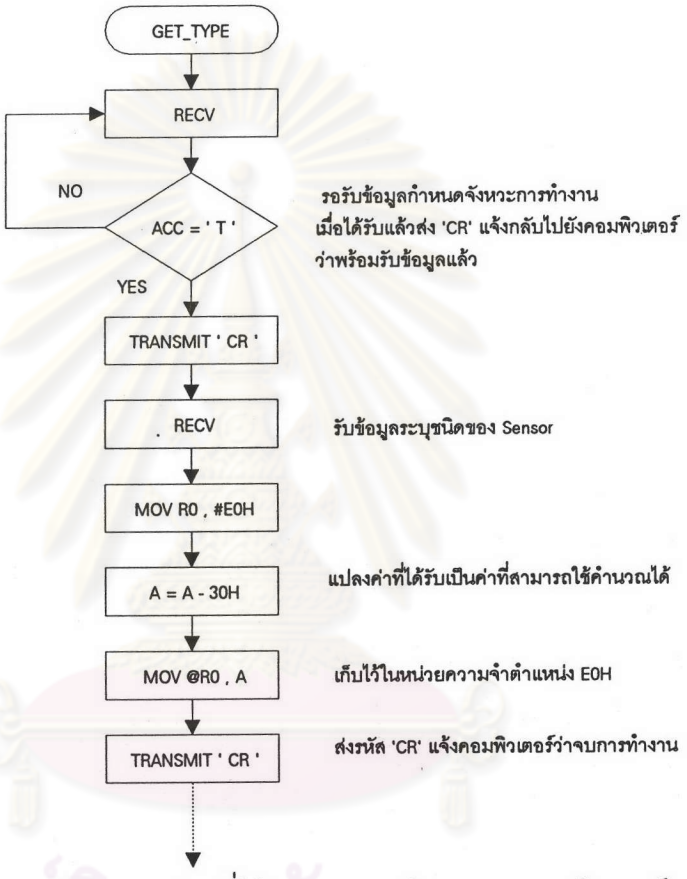


รูปที่ 4.16 ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับคำนวณหาค่าต่ำสุด

4.2.11 โปรแกรมสำหรับตรวจสอบชนิดของอินพุตแบบแอนะล็อก (GET_TYPE)

โปรแกรมส่วนนี้ใช้ในการตรวจสอบชนิดของอินพุตแต่ละชนิดว่ากำลังทำงานกับทรานสดิวเซอร์ชนิดใดอยู่เพื่อสามารถแปลงสัญญาณแอนะล็อกที่อ่านได้เป็นค่าจริงตรงตามชนิดของทรานสดิวเซอร์ชนิดนั้นได้ถูกต้อง ทุกครั้งที่เป็นรอบของการอ่านค่าจากอินพุตแบบแอนะล็อก โปรแกรมย่อยส่วนนี้จะถูกใช้งาน เพื่อรอรับข้อมูลที่เป็นตัวระบุชนิดของทรานสดิวเซอร์จากคอมพิวเตอร์ที่เป็นตัวควบคุม

การทำงานของโปรแกรม ก่อนที่คอมพิวเตอร์จะส่งข้อมูลที่เป็นชนิดของทรานสดิวเซอร์มา มันจะส่งข้อมูล (ตัวอักษร ' T ') มาแจ้งยังวงจรควบคุมก่อนว่า ข้อมูลที่จะส่งต่อไปเป็นชนิดของทรานสดิวเซอร์เพื่อเป็นการกำหนดจังหวะการทำงาน เมื่อวงจรควบคุมได้รับข้อมูลระบุชนิดของทรานสดิวเซอร์แล้ว (ข้อมูลที่ได้จะอยู่ในรูปของตัวเลข) จะทำการแปลงค่าที่ได้รับเป็นค่าที่สามารถใช้คำนวณได้แล้วเก็บไว้ในหน่วยความจำตำแหน่งที่ E0H ขั้นตอนการทำงานของโปรแกรมแสดงในรูปที่ 4.17



รูปที่ 4.17 โปรแกรมส่วนที่ใช้ตรวจสอบชนิดของทรานสดิวเซอร์

4.2.12 โปรแกรมย่อยสำหรับแปลงค่าอินพุตแอนะล็อกที่อ่านได้เป็นค่าที่สัมพันธ์กับชนิดของทรานสดิวเซอร์

ในการอ่านค่าจะอินพุตแบบแอนะล็อก ค่าที่อ่านได้จะอยู่ในรูปเลขฐาน 16 ค่าตั้งแต่ 00H ถึง FFH ซึ่งผู้ใช้งานจะไม่ทราบว่าค่าที่อ่านได้มีค่าเป็นเท่าไร ดังนั้นหน้าที่ข้องโปรแกรมย่อยส่วนนี้จะทำหน้าที่แปลงค่าที่อ่านได้จากอินพุตแบบแอนะล็อกเป็นค่าจริงที่สัมพันธ์กับชนิดของทรานสดิวเซอร์ชนิดนั้น

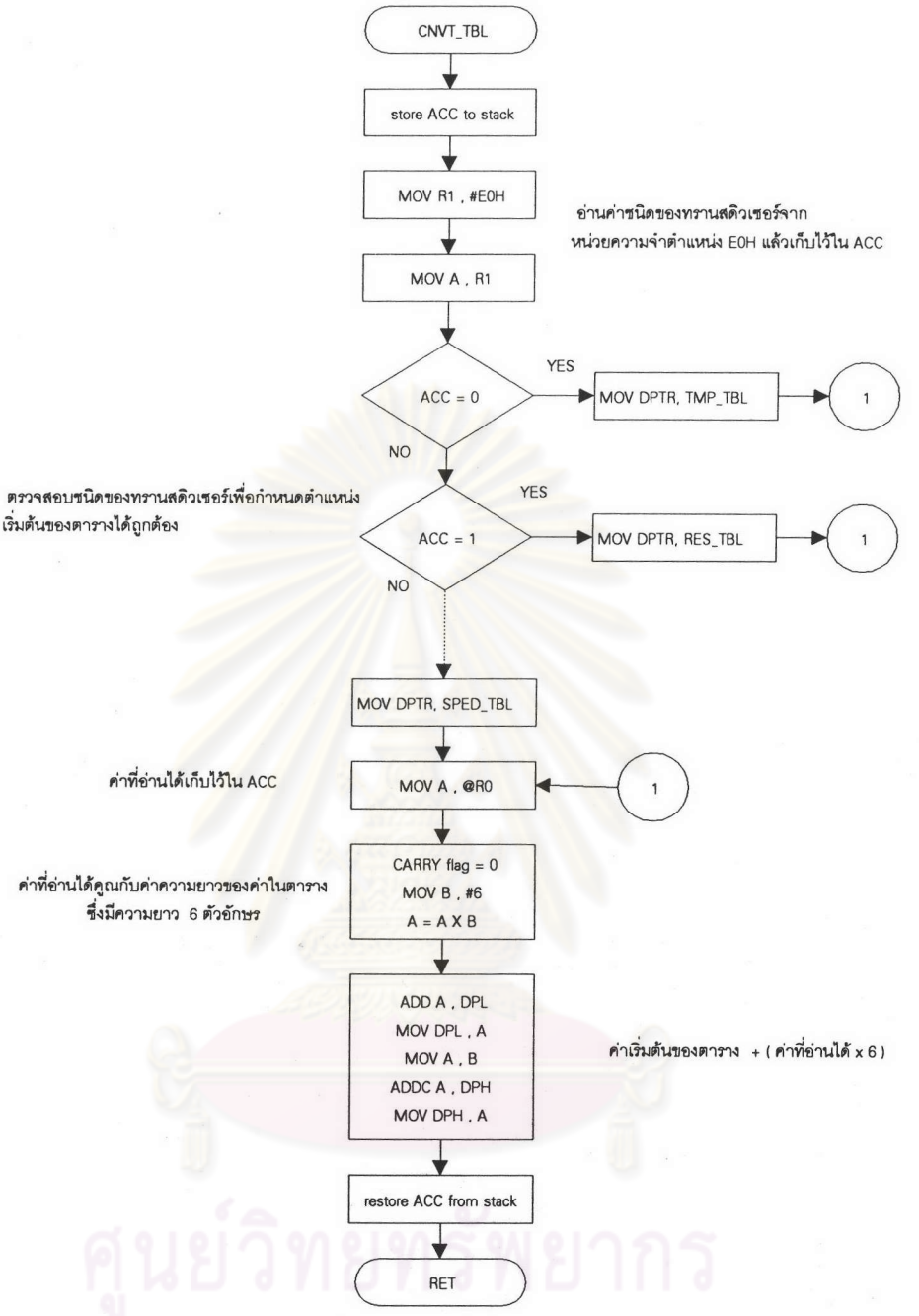
การทำงานในการแปลงค่าอินพุต จะใช้วิธีการเปิดตาราง (Table Look up) ซึ่งตารางแต่ละตารางจะมีค่าจริงที่เป็นไปได้ทั้งหมดของทรานสดิวเซอร์เก็บไว้ เช่นตารางที่ใช้แปลงค่าเป็นค่าแรงดันไฟฟ้าจะเก็บค่าตั้งแต่ 0 โวลต์ ถึง 5 โวลต์

ขั้นตอนการทำงาน โปรแกรมจะตรวจสอบชนิดของทรานสดิวเซอร์ที่กำลังอ่านค่าอยู่ว่าเป็นชนิดใด โดยดูจากหน่วยความจำตำแหน่งที่ E0H นำค่าที่ได้ไปตรวจสอบว่าควรจะใช้ตารางแปลงค่าตารางใด (ตำแหน่งของตารางชี้โดย DPTR) นำค่าที่อ่านได้คูณกับ 6 (อยู่ในรีจีสเตอร์ B) เนื่องจากค่าที่เก็บอยู่ในตารางมีความยาว 6 หลัก แล้วบวกค่าที่ได่กับค่าของ DPTR ซึ่งจะทำให้ได้ตำแหน่งเริ่มต้นของค่าที่ต้องการใช้งาน

วิธีการคำนวณค่าที่ต้องการในตารางมีหลักดังนี้

ตำแหน่งเริ่มต้นของค่าที่ต้องการใช้งาน (DPTR) = ค่าเริ่มต้นของตาราง (DPTR)
+ (ค่าที่อ่านได้จากอินพุต x
ค่าความยาวของค่าในตารางในที่นี้คือ 6)

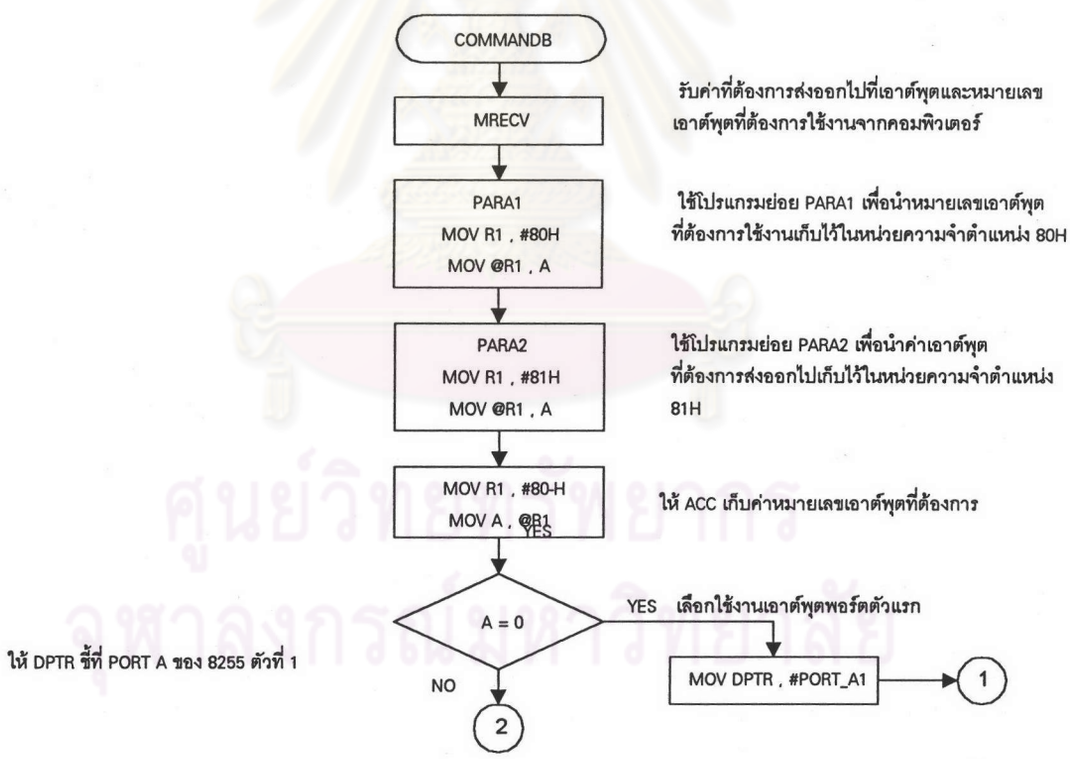
ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับแปลงค่าอินพุตแอนะล็อกที่อ่านได้เป็นค่าที่สัมพันธ์กับชนิดของทรานสดิวเซอร์แสดงในรูปที่ 4.18

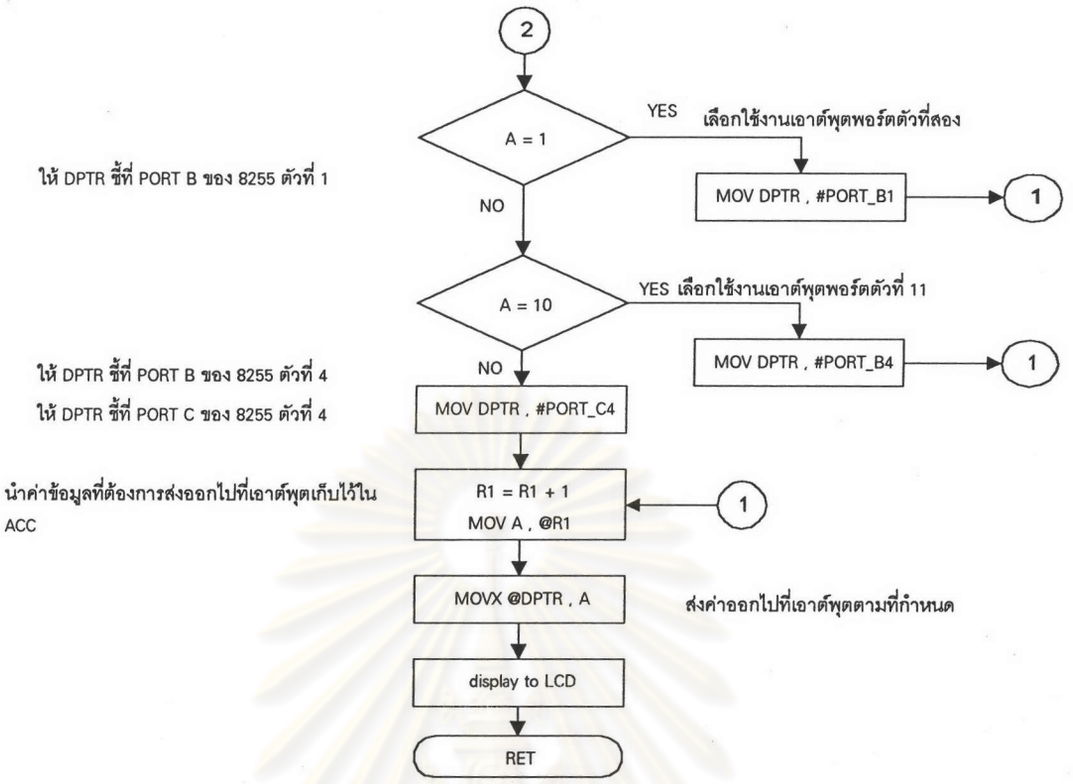


รูปที่ 4.18 โปรแกรมย่อยสำหรับแปลงค่าอินพุตแอนะล็อกที่อ่านได้เป็นค่าที่สัมพันธ์กับชนิดของทรานสดิวเซอร์

4.2.13 โปรแกรมย่อยสำหรับการเขียนค่าไปที่เอาต์พุตแบบดิจิทัล

โปรแกรมส่วนนี้มีหน้าที่ควบคุมการส่งค่าไปที่เอาต์พุตแบบดิจิทัลโดยมีขั้นตอนการทำงาน คือ เมื่อคอมพิวเตอร์ที่ใช้ควบคุมส่งสัญญาณมาเพื่อให้เริ่มต้นทำงาน (ข้อมูลที่ส่งมาคือตัวอักษร B) โปรแกรมจะเริ่มทำงานโดยรอรับค่าที่คอมพิวเตอร์จะส่งตามมาได้แก่ ค่าที่จะส่งออกไปที่เอาต์พุต และให้ส่งออกไปที่เอาต์พุตตัวใด โดยหมายเลขของเอาต์พุตที่จะส่งค่าออกไป จะเก็บไว้ที่หน่วยความจำตำแหน่งที่ 80H ส่วนค่าที่จะส่งออกไปเก็บไว้ในตำแหน่งที่ 81H เนื่องจากในการวิจัยครั้งนี้ใช้ชิปไอซี 8255 เป็นตัวควบคุมอินพุตและเอาต์พุตแบบดิจิทัล ซึ่ง 8255 หนึ่งตัวจะใช้ควบคุมการทำงานของอินพุตและเอาต์พุตแบบดิจิทัลถึง 3 ตัว ดังนั้นโปรแกรมในส่วนนี้จะต้องนำ หมายเลขของเอาต์พุตที่ต้องการไปตรวจสอบว่าจะให้พอร์ตใดของ 8255 ตัวใดทำงาน เมื่อทราบพอร์ตที่จะใช้งานแล้วก็จะส่งค่าที่ต้องการออกไปยังพอร์ตของ 8255 ตัวนั้น โปรแกรมการส่งค่าออกไปที่เอาต์พุตแบบดิจิทัลจะแสดงในรูปที่ 4.19

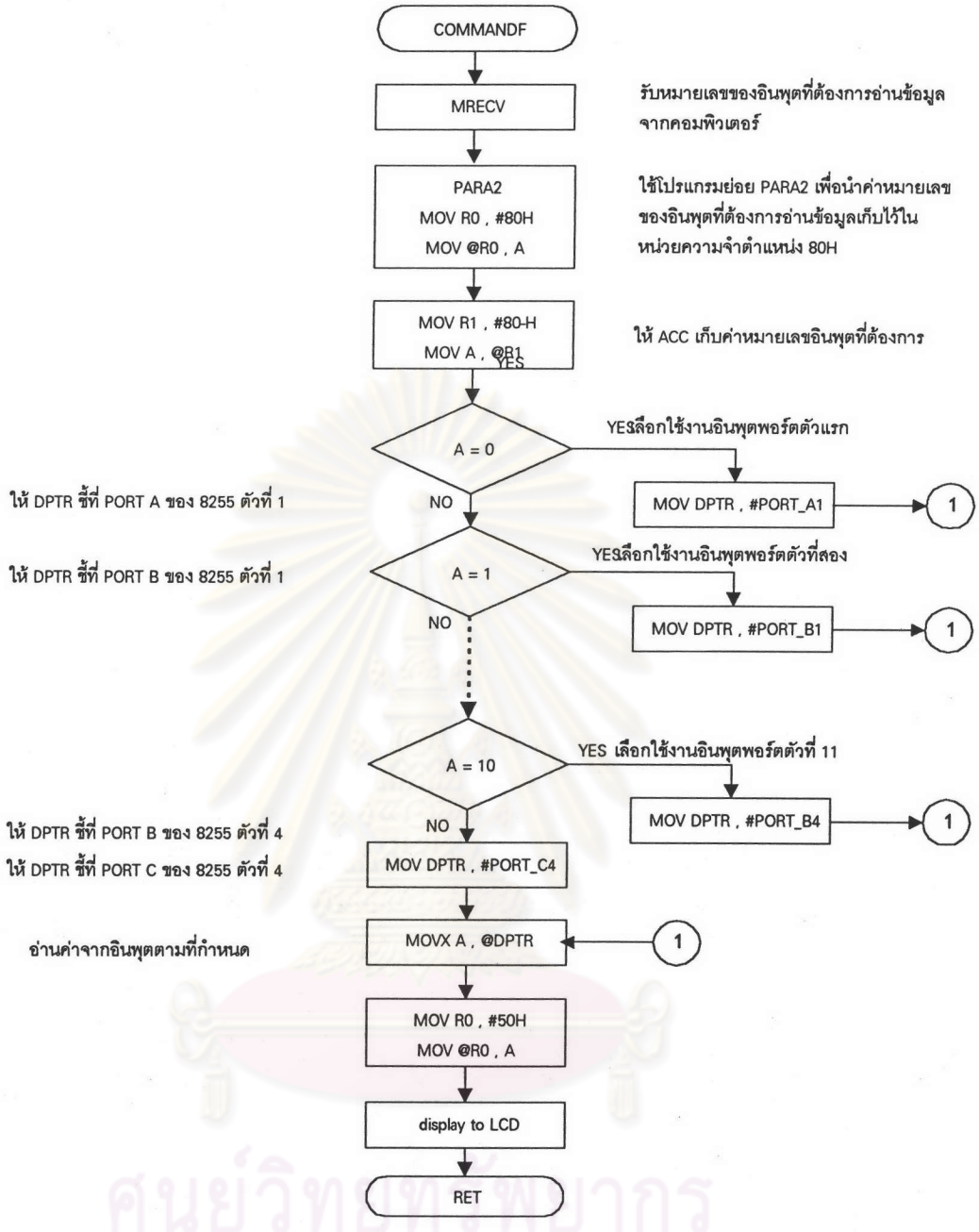




รูปที่ 4.19 โปรแกรมย่อยสำหรับการส่งค่าออกไปที่เอาต์พุตแบบดิจิทัล

4.2.14 โปรแกรมย่อยสำหรับการอ่านข้อมูลจากอินพุตแบบดิจิทัล

โปรแกรมย่อยในส่วนนี้คล้ายกับโปรแกรมย่อยสำหรับส่งข้อมูลออกไปที่เอาต์พุตแบบดิจิทัล ต่างกันก็เพียงข้อมูลที่ส่งมาจากคอมพิวเตอร์จะมีเพียงหมายเลขของอินพุตที่ต้องการอ่านค่าเท่านั้น ซึ่งโปรแกรมจะเก็บหมายเลขอินพุตไว้ในหน่วยความจำตำแหน่ง 80H ขั้นตอนในการตรวจสอบว่าอินพุตที่ต้องการใช้งานตรงกับพอร์ตใดของชิปไอซี 8255 ตัวใดจะเหมือนกับโปรแกรมการส่งค่าออกไปที่เอาต์พุตทุกประการ ขั้นตอนการทำงานแสดงในรูปที่ 4.20



รูปที่ 4.20 ขั้นตอนการทำงานของโปรแกรมย่อยสำหรับอ่านค่าจากอินพุตแบบดิจิทัล