เครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อม

(TEMPERATURE DATA LOGGER)

นายชวสุ นุรักษ์

นายอานนท์ อักษัส

อาจารย์ที่ปรึกษา

อาจารย์ชวลิต เนตุจางคประเสริฐ

ปริญญานิพนธ์เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาอุตสาหกรรมศาสตร์บัณฑิต

คณะวิศวกรรมศาสตร์

สาขาเทคโนโลยีอิเลคทรอนิคส์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2537

เครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อม

(TEMPERATURE DATA LOGGER)

นายวสุ นุรักษ์

นายอานนท์ ลักบัส

ได้รับพิจารณาอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษา

ตามหลักสูตรปริญญาอุตสาหกรรมศาสตร์บัณฑิต

สาขาวิชาเทคโนโลยีอีเลคทรอนิคส์

คณะกรรมการตรวจสอบปริญญานิพนธ์

..... ประธานกรรมการ

( )

..... กรรมการ

( )

..... กรรมการ

( )

ชื่อเรื่องปริญญานิพนธ์ : เครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อม

ชื่อผู้เขียน : นายวสุ นุรักษ์
นายอานนท์ อักบัส

อาจารย์ที่ปรึกษา : อาจารย์ชวลิต เบญจางคประเสริฐ

ปริญญานิพนธ์ : อุตสาหกรรมศาสตร์บัณฑิต สาขาเทคโนโลยีอีเลคทรอนิคส์

บทคัดย่อ

ในการสร้างเครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อม ออกแบบโดยการใช้ไมโคร-คอนโทรลเลอร์ ควบคุมการทำงานของเครื่องและตัวเซนเซอร์ในการตรวจวัดค่าอุณหภูมิ และแปลงเป็นข้อมูลดิจิตอลโดยเอทูดีคอนเวอร์เตอร์ และค่าอุณหภูมิที่ถูกบันทึกไว้จะสามารถแสดงผลบนจอคอมพิวเตอร์ สำหรับการติดต่อรับส่งข้อมูลระหว่างเครื่องได้กล่าวถึงระบบของเครื่องและขั้นตอนการทำงาน

คุณสมบัติของเครื่อง เป็นเครื่องที่สามารถวัดและบันทึกค่าของอุณหภูมิแวดล้อม โดยที่ค่าของอุณหภูมิจะแสดงบนหน้าปัทม์แสดงผล และสามารถแสดงข้อมูลทั้งหมด บนจอภาพคอมพิวเตอร์

PROJECT REPORT TITLE : TEMPERATURE DATA LOGGER

NAME : MR. WASU NURAK

MR. ANON AKBUS

PROJECT REPORT ADVISOR : MR. CHAOVALIT BENJARKPRASAT

DEPARTMENT OF : ELECTRONICS TECHNOLOGY

ACADEMIC YEAR : 1994

ABSTRACT

THE TEMPERATURE DATA LOGGER WAS DESIGNED BY USING MICRO CONTROLLOR (8051) WHICH WILL CONTROL TEMPERATURE SENSOR AND ALL PARTS OF DATA LOGGER. A-D CONVERTOR USED FOR CONVERSING DIGITAL DATA OF TEMPERATURE MEASUREMENT AND THE SUCH CONVERTED TEMPERATURE DATA COULD BE DISPLAY ON PC. THE SERIAL PORT (COM1) CAN ALSO USE FOR COMMUNICATE BETWEEN DATA LOGGER AND PC. THIS PROJECT HAS MANY DETAILS THAT EXPLAIN ABOUT TEMPERATURE DATA LOGGER AND SEQUENCE OF WORK

THE QUALITY OF TEMPERATURE DATA LOGGER. CAN MEASURE AND RECORD AMBIENT TEMPERATURE BY READ FROM PANAL AND DISPLAY ON PC.

กิตติกรรมประกาศ

ในการทำปริญญานิพนธ์ เรื่อง "เครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อม" ต้องขอขอบคุณท่านอาจารย์ชวลิต เบญจางคประเสริฐ ที่ได้ให้คำแนะนำและข้อมูลที่เป็นประโยชน์แก่การทดลอง และเป็นอาจารย์ที่ควบคุมในการทำปริญญานิพนธ์ในครั้งนี้ด้วย

บทนำ

ปัจจุบันความก้าวหน้าทางเทคโนโลยี ทำให้มนุษย์มีความเป็นอยู่เปลี่ยนไป โดยเฉพาะเรื่องของความสะดวกสบาย ความรวดเร็วตลอดจนความสามารถใช้แรงงานมนุษย์ให้น้อยลง แต่ประสิทธิภาพของงานสูงขึ้น ระบบอัตโนมัติจึงเข้ามามีบทบาทสำคัญในชีวิตของคนเรา อาทิเช่น แต่ก่อนถ้าเราต้องการทราบการเปลี่ยนแปลงของสภาพอุณหภูมิ (ซึ่งเป็นสิ่งที่มนุษย์สัมผัสได้ทางร่างกายและผิวหนัง) ณ สถานที่ใดที่หนึ่ง เราจะต้องใช้คนเฝ้าคอยจดบันทึก ตลอดเวลา แต่ในปัจจุบันระบบอัตโนมัติเข้ามามีบทบาทสำคัญอันเป็นจุดเริ่มต้นของโครงการนี้ โดยเครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อม จะทำงานในการเก็บบันทึกค่าของอุณหภูมิแวดล้อม ณ สถานที่นั้น ในระยะห่างของเวลาที่เก็บข้อมูลซึ่งถูกควบคุมการทำงานด้วยไมโครคอนโทรลเลอร์ และข้อมูลที่ถูกบันทึกรวบรวมไว้สามารถนำไปแสดงบนจอภาพคอมพิวเตอร์ เพื่อแสดงค่าของอุณหภูมิที่ได้จากทางบันทึกไว้

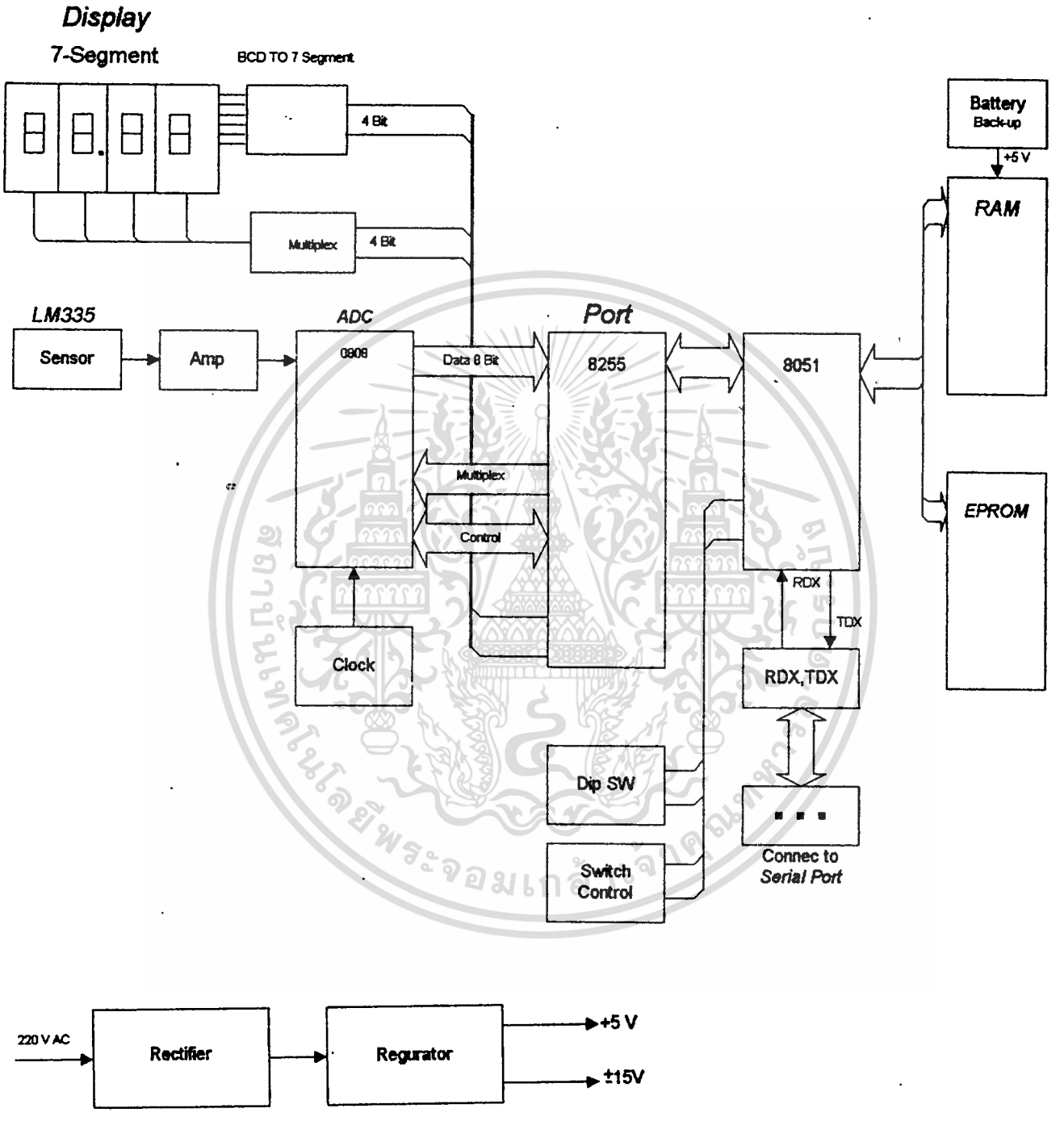
ในฉบับนี้ จะกล่าวถึงเครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อม โดยใช้ไมโครคอนโทรลเลอร์เป็นส่วนควบคุมการทำงานของเครื่อง โดยเนื้อหานี้จะประกอบด้วย ตัวตรวจจับอุณหภูมิ ระบบไมโครคอนโทรลเลอร์ บล๊อกไดอะแกรม และการทำงานของเครื่อง ซึ่งทั้งหมดนี้เป็นส่วนหนึ่งของโปรเจคทั้งหมด

ในโปรเจค 2 นี้ จะเป็นการใช้ซอร์ฟแวร์ ควบคุมการทำงานของฮาร์ดแวร์ ซึ่งซอร์ฟแวร์ที่ใช้จะเป็นแอสแซมบี้ของ MCS 51

สารบัญ

บทที่ 1

โครงสร้างของระบบ



LM335

ตัวตรวจวัดอุณหภูมิ

ในการวัดอุณหภูมิในปัจจุบันมีอยู่หลายประเภท เช่น การใช้เทอร์โมมิเตอร์แบบปรอท แต่ก็มีข้อเสียอยู่คือ อ่านอุณหภูมิได้ยาก และใช้วัดอุณหภูมิที่จุดห่างออกไปไม่ได้ หรือ เทอร์โมมิเตอร์ที่ใช้หลักการการขยายของโลหะต่างชนิดกัน โดยขดอยู่ในลักษณะแบบก้นหอย ซึ่งสามารถใช้ได้สะดวกแต่มีข้อเสีย คือ ขาดความเที่ยงตรง ส่วนเทอร์โมมิเตอร์แบบอิเล็กทรอนิคส์นั้น จะแสดงค่าอุณหภูมิออกมาในรูปของตัวเลข

รูปที่ 1 ลักษณะรูปร่าง และการต่อขาของ LM335 ฯ. 1

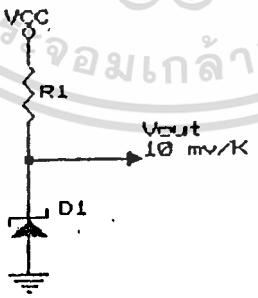
LM 335 เป็นอุปกรณ์ทางอิเล็กทรอนิคส์ที่ออกแบบมาในการสำหรับการตรวจจับอุณหภูมิ ซึ่งใช้อยู่ในย่านอุณหภูมิตั้งแต่ 0° ถึง 100° โดยออกแบบมาอยู่ในตัวถังพลาสติกสีดำ ซึ่งมีลักษณะการต่อขาดังรูปที่ 1ก และในลักษณะตัวถังโลหะแบบ TO-46 ดังในรูปที่ 1ข

LM 335 เป็นอุปกรณ์ที่ออกแบบมาใช้ในงานวัดอุณหภูมิ โดยเฉพาะซึ่งใช้งานในย่านอุณหภูมิตั้งแต่ -25° ถึง +100° โดย IC ตัวนี้มีค่าความคลาดเคลื่อนจากรายละเอียดทางเทคนิคต่ำมาก

โดยพื้นฐานแล้ว LM 335 มีหลักการทำงานคล้ายกับ ZENER DIODE ดังแสดงในรูปที่ 2 โดยแรงดันพังทลายซึ่งหมายถึง VOLTAGE OUTPUT จากวงจรจะแปรค่าโดยตรงตามค่าอุณหภูมิสัมบูรณ์ โดยมีค่าเท่ากับ 10 mv/K ในย่านอุณหภูมิที่ออกแบบมาให้ใช้งาน

ค่าของตัวต้านทาน R_1 ในรูปที่ 2 จะทำหน้าที่เป็นตัวกำหนดกระแสที่ไหลผ่านอุปกรณ์ตัวนี้ แต่เนื่องจากค่าไดนามิกอิมพีแดนซ์ที่กระแส 1 mA จะมีค่าประมาณ 0.6 โอห์ม อุปกรณ์ตัวนี้จึงสามารถทำงานได้ในย่านกระแสตั้งแต่ 400 uA ถึง 5 mA มีข้อน่าสังเกตก็คือค่ากระแสฟอร์เวิร์ดหรือกระแสรีเวอร์สสูงสุด ซึ่งไหลผ่านอุปกรณ์ตัวนี้อย่างปลอดภัย ซึ่งค่าสูงสุดชั่วขณะหนึ่งควรไม่เกิน 10 mA ถ้ากระแสที่ไหลผ่านมากกว่านี้จะทำให้ตัว IC เสียหายได้

ที่อุณหภูมิที่ 25° และที่กระแสรีเวอร์ส 1 mA แรงดันเอาท์พุตจากวงจรในรูปที่ 2 จะมี ค่าความตามที่ออกแบบไว้เท่ากับ 2.98 V ซึ่งได้จาก



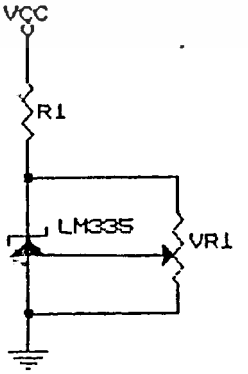
รูปที่ 2 วงจรพื้นฐานในการใช้งานของ LM 335

$(273 + 25) * 10^{-3}$ โดยมีข้อจำกัดทางด้านแรงดันต่ำสุดและสูงสุด มีค่าเท่ากับ 2.98 V และ 3.04 V ตามลำดับ ค่าของ R_1 ที่ใช้สามารถคำนวณได้จากกระแสที่ ไหลผ่าน IC ตัวนี้ เท่ากับ 1 mA โดยใช้สมการ

$$R_1 = \frac{V - Vout}{0.001}$$

โดยที่ V คือ แรงดันไฟเลี้ยงวงจรและ Vout คือ แรงดันเอาท์พุตของ IC นี้ ซึ่ง R_1 นี้ จะมีค่าประมาณ (V - 3) K ประโยชน์ที่สำคัญของ IC LM 335 นี้ก็คือ จะให้เอาท์พุตที่เป็นเชิงเส้นซึ่ง ไม่เหมือนกับเอาท์พุตที่ได้จากการตรวจจับอุณหภูมิ ส่วนใหญ่ซึ่งเอาท์พุตจะไม่เป็นเชิงเส้นกับอุณหภูมิ ซึ่งที่จริงแล้วถ้าเรานำเอาค่าแรงดันเอาท์พุตมาเขียนเป็นกราฟ ระหว่างค่าแรงดันกับอุณหภูมิ ตลอดย่านการทำงานของ IC ตัวนี้ เมื่อลากเส้นกราฟไป ตัดแกนอุณหภูมิที่ศูนย์องศาสัมบูรณ์ ค่าแรงดันที่อ่านได้จากกราฟจะเป็นศูนย์โวลท์

แม้ว่าเอาท์พุตของ LM 335 จากวงจรในรูปที่ 2 จะอยู่ภายในข้อจำกัดที่กำหนดมา เราก็ควรจะมีการต่อขาสำหรับปรับแต่งความถูกต้อง ซึ่งมีอยู่ด้วยโดยการต่อตัวต้านทานปรับค่าเพียงตัวเดียวคร่อม LM 335 ดังแสดงในรูปที่ 3 ในการใช้งานให้ปรับค่า VR_1 ให้ OUTPUT เท่ากับ 2.982 V เมื่ออุณหภูมิอยู่ที่ 25° เพื่อที่จะให้เกิดความเที่ยงตรงเพิ่มขึ้น ตลอดย่านอุณหภูมิที่ใช้งาน

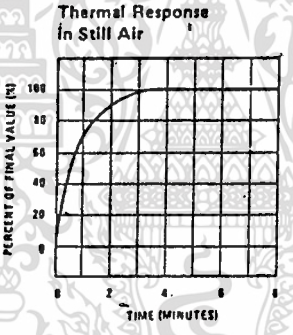


รูปที่ 3 การปรับแต่งความถูกต้องโดยใช้ความต้านทานเพียงตัวเดียว

การปรับความถูกต้องของอุณหภูมิเพียงครั้งเดียว จะให้ถูกต้องตลอดย่านอุณหภูมิที่ใช้งานนั้นก็เนื่องจากเอาท์พุตจะแปรผันโดยเป็นสัดส่วนกับอุณหภูมิสัมบูรณ์โดยเอาท์พุต จะลดลงเป็นศูนย์โวลท์ที่อุณหภูมิศูนย์องศาสัมบูรณ์ ดังนั้นการปรับค่าความลาดชันที่อุณหภูมิค่าหนึ่งให้ถูกต้องจะทำให้เกิดความถูกต้องตลอดย่านอุณหภูมิ ซึ่งการปรับค่าความถูกต้องจะทำได้ง่ายกว่าพวกอุปกร์ที่ไม่เป็นเชิงเส้น ดังเช่น เทอร์โมคัปเปิล เป็นต้น

ความร้อนที่เกินขึ้นในตัว

ไม่ว่าระบบตรวจจับอุณหภูมิใด ๆ ก็ตาม ความร้อนใด ๆ ที่เกิดขึ้นจากกระแสที่ไหลผ่านอุปกรณ์ ที่ตรวจจับจะมีผลต่อค่าอุณหภูมิของตัวมัน ตลอดจน



รูปที่ 4 เวลาการตอบสนองของ LM 335 ต่อการเปลี่ยนแปลงของอุณหภูมิในอากาศ

แรงดันเอาท์พุตที่เกิดขึ้น สำหรับ LM 335 นั้นควรจะให้ทำงานที่กระแสต่ำสุด ซึ่งเพียงพอที่จะชับให้วงจรภายใน IC ทำงานได้ เมื่อคำนวณค่าของ R_1 จะยอมให้กระแสไหลผ่านตัวต้านทานปรับค่าได้ที่ต่อขนานกับ IC สำหรับที่กระแสเอาท์พุตใด ๆ โดยกระแสประมาณ 400 uA จะเป็นกระแสต่ำสุดที่ IC นี้ทำงานได้ปกติ

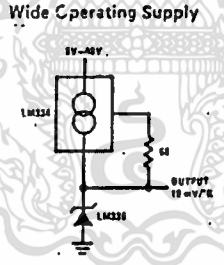
ถ้าตัวตรวจจับถูกใช้ในสถานที่ ซึ่งค่าความต้านทานทางอุณหภูมิต่อสิ่งแวดล้อมนั้นมีค่าคงที่ ค่าความผิดพลาดจากความร้อนที่เกิดขึ้นในตัวเองสามารถที่จะปรับให้ถูกต้องได้ ซึ่งจะทำให้อุปกรณ์นั้นทำงานด้วยกระแสคงที่ โดยไม่ขึ้นอยู่กับอุณหภูมิความร้อนเกิดขึ้นกับอุปกรณ์จะแปรผันโดย

ตรงกับ แรงดันซีเนอร์และอุณหภูมิสัมบูรณ์ ดังนั้นค่าความผิดพลาดที่เกิดขึ้นจากความร้อนในตัวเอง จะแปรผันโดยตรงกับค่าอุณหภูมิสัมบูรณ์ และความเป็นเชิงเส้นของสเกลอุณหภูมิที่รักษาไว้

คุณสมบัติเฉพาะตัว

ในวงจรทั่วไปของ LM 335 ซึ่งไม่ได้ปรับค่าความถูกต้องไว้ให้ทำงานที่กระแส 1 mA ค่าความผิดพลาดเนื่องจากอุณหภูมิจะเท่ากับ 2° (สูงสุด 6°) ที่อุณหภูมิ 25° หรือ 4° (สูงสุด 9°) ตลอดย่านการทำงาน เมื่อได้มีการปรับค่าความถูกต้องไว้ ค่าความผิดพลาดที่อุณหภูมิจำกัดไว้เป็น 2° ความไม่เป็นเชิงเส้นที่ค่ากระแส1 mA จะเท่ากับ 0.3° ตลอดย่านที่ใช้งาน

ในอากาศอุปกรณ์ตัวนี้ต้องใช้เวลาประมาณ 3 นาที ที่จะมีค่าอุณหภูมิสุดท้าย



รูปที่ 5 วงจรที่สามารถทำงานกับแหล่งจ่ายไฟได้ในย่านกว้าง

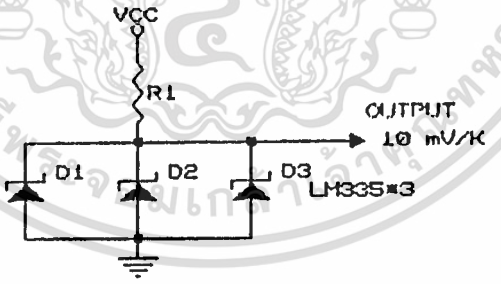
หลักจากที่อุณหภูมิได้เปลี่ยนไป (ในรูปที่ 4) ค่าคงที่ของเวลาจะมีค่าตามที่ออกแบบไว้เท่ากับ 80 วินาที ในน้ำมันที่ไหลวนให้เคลื่อนไปมา ค่าอุณหภูมิสุดท้ายจะถึงภายในเวลาประมาณ 3 วินาที (ค่าคงที่ของเวลาจะเท่ากับ 1 วินาที ตามที่ออกแบบไว้) อุปกรณ์จะคงที่อยู่ภายในช่วง 0.2° ตลอด 1,000 ชั่วโมง ถึงแม้ว่าที่อุณหภูมิ 125°

ค่าไดนามิคอิมพีแดนซ์จะน้อยกว่า 1 โอห์ม ที่ความถี่สูงกว่า 1 KHz, (ตามที่ออกแบบไว้) แต่จะมีค่าเพิ่มขึ้นเป็น 20 - 30 โอห์ม ที่ความถี่ 100 KHz

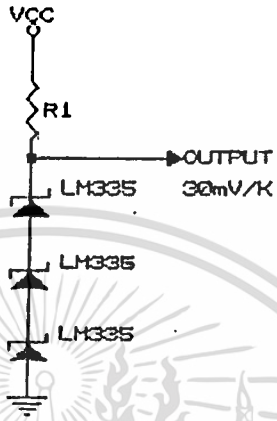
วงจรที่ใช้งาน

วงจรในรูปที่ 2 และ 3 เหมาะสำหรับใช้เมื่อแรงดันไฟเลี้ยงวงจรมีค่าค่อนข้างคงที่ ถ้าคาดว่าแรงดันไฟเลี้ยงวงจรจะเปลี่ยนแปลงในย่านกว้าง ควรจะใช้ LM 334 ซึ่งเป็นตัวจ่ายกระแสคงที่ร่วมกับตัวต้านทานภายนอกดังในรูปที่ 5 เพื่อกำหนดค่ากระแสของ LM 335 ให้ค่าประมาณ 1 mA สำหรับทุก ๆ ค่าของไฟเลี้ยงวงจร

ถ้าน่าเอา LM 335 หลาย ๆ ตัวมาต่อขนานกันดังรูปที่ 6 ค่าเอาท์พุตจะขึ้นอยู่กับอุปกรณ์ตัวที่มีอุณหภูมิต่ำสุดซึ่งจะแสดงถึงค่าอุณหภูมิที่ต่ำสุดระหว่าง 3 ตำแหน่งได้และเช่นเดียวกันเมื่อต่อ LM 335 อย่างอนุกรมกันดังรูปที่ 7 ในกรณีนี้ค่าเอาท์พุตจะ



รูปที่ 6 วงจรนี้จะบันทึงานได้ในย่านกระแสตั้งแต่ 400 uA ถึง 5 mA

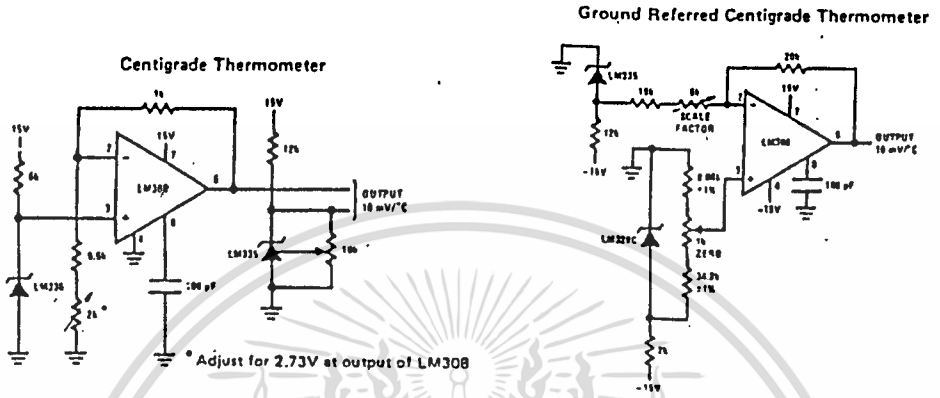


รูปที่ 7 วงจรนี้จะบันทึกอุณหภูมิเฉลี่ยของอุณหภูมิของไอซีทั้ง 3 ตัว

แทนค่าเฉลี่ยของอุณหภูมิของ IC ทั้ง 3 ตัวนั้นแต่จะมีค่าเพิ่มขึ้นด้วยสัมประสิทธิ์เท่ากับจำนวนของ IC ที่ใช้

ใช้เป็นเทอร์โมมิเตอร์

วงจรต่าง ๆ ที่กล่าวมานั้นเป็นวงจรพื้นฐาน ซึ่งจะให้ค่าแรงดันเอาท์พุทแปรผันโดยตรงกับค่าอุณหภูมิสมบูรณ์ ซึ่งไม่เหมาะสมที่จะป้อนเข้าสู่ DIGITAL METER เพื่อที่จะอ่านค่าอุณหภูมิออกมาเป็นองศาเซลเซียส วงจรออปแอมป์ที่เพิ่มขึ้นมาดังรูปที่ 8 จะเป็นตัวแก้ปัญหานี้ได้



รูปที่ 8 (ก) วงจรเทอร์โมมิเตอร์แบบเซลเซียสซึ่งเอาท์พุตไม่ต่อส่งกราวด์

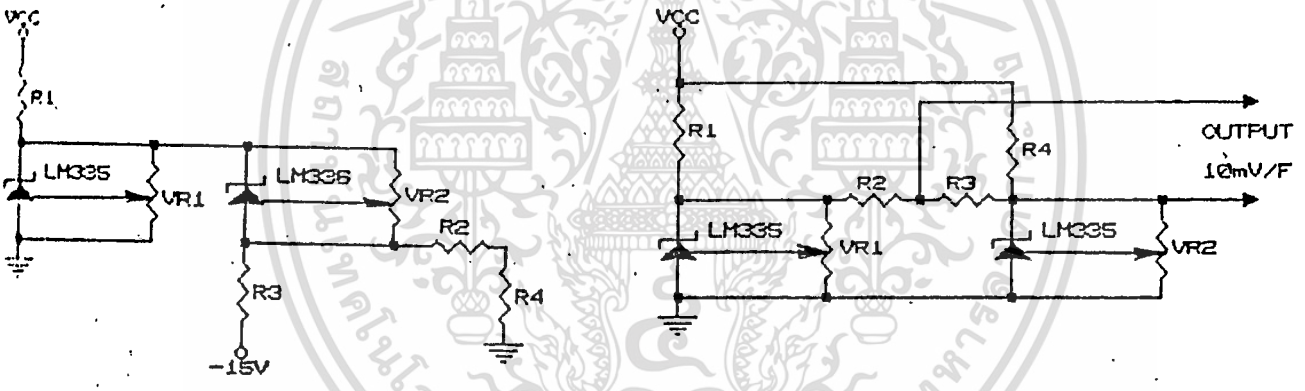
ส่วนใน 8 (ข) เอาท์พุตที่ได้จะอ้างอิงกับกราวด์

ตามวงจรในรูปที่ 8ก. IC LM 335 จะเป็นตัวกำหนดแรงดันอ้างอิงขนาด 5 VOLT ที่ขา 3 ของออปแอมป์ LM 308 การป้อนกลับที่ขาลบที่ขา 2 จะถูกปรับได้ตัว ต้านทานปรับค่าได้ ค่า 2K เพื่อให้เอาท์พุตของวงจรขยายมีค่าเท่ กับ 2.73 VOLT แรงดันที่แตกต่างระหว่างเอาท์พุตของออปแอมป์ จะแทนด้วยการลบค่า 273° จากค่าอุณหภูมิสัมบูรณ์ เพื่อที่จะให้เอาท์พุตจาก LM 335 แสดงค่าออกมาเป็นองศาเซลเซียสต่อไป (อุณหภูมิสัมบูรณ์ ช 273 + อุณหภูมิเป็นองศาเซลเซียส)

วงจรในรูปที่ 8ก.นั้นไม่มีขาใดเทียบเป็น GROUND ได้เลยขาเอาท์พุตจะต่ออยู่ลอย ๆ ถ้าเราเปลี่ยนวงจรให้ซับซ้อนขึ้นดังในรูปที่ 8ก. จะให้เอาท์พุตเท่ากับ 10 mV/° เทียบกับ GROUND โดยใช้ IC LM 29C เป็นตัวกำเนิดแรงดันอ้างอิงขนาด 6.9 V ในการกำหนดแรงดัน

ที่ปรับค่าได้ ที่ป้อนเข้าสู่ขานอนอินเวิร์ทติ้ง (ขา+) ของออปแอมป์ต่อไป ส่วนเอาท์พุตของ LM 335 ก็จะป้อนเข้าสู่ขาอินเวิร์ทติ้ง (ขา-) ของออปแอมป์ต่ ไป ตัวต้านทานปรับค่าได้ 5 k นี้มีไว้ปรับค่าสเกลแฟคเตอร์หรือ ค่าความลาดชันระหว่างอุณหภูมิกับค่าแรงดันเอาท์พุตที่เกิดขึ้น เพื่อการปรับค่าให้ถูกต้อง

นอกจากนี้ LM 335 นี้ยังสามารถทำการต่อวัดค่าอุณหภูมิออกมาเป็นองศาฟาเรนไฮท์ ได้อีก โดยการต่อวงจรตามรูปที่ 9ก. LM 336 เป็นตัวกำเนิดแรงดันอ้างอิงขนาด 5 VOLT โดยจะต่อตัวต้านทานปรับค่าได้ 10 K ขนานกับ LM 336 ทำการปรับค่า VR_2 นี้ให้แรงดันที่ตกคร่อม LM 336 ท่ากับ 2.554 V ส่วนตัวต้านทาน



รูปที่ 9 (ก) วงจรเทอร์โมมิเตอร์แบบฟาเรนไฮท์ ซึ่งเอาท์พุตที่ได้จะอ้างอิงกับกราวด์ ส่วนในรูปที่ 9 (ข) เอาท์พุตที่ได้จะลอยและใช้แหล่งจ่ายไฟชุดเดียว

VR_1 นั้นปรับให้เอาท์พุตที่ตกคร่อม LM 335 นั้นให้เท่ากับ 2.992 V เมื่อ LM 335 อยู่ที่อุณหภูมิ 77^F วงจรนี้จะให้ค่าเอาท์พุตออกมา 1 mV/1^F วงจรในรูปที่ 9ก. นี้เอาท์พุตที่ได้จะเทียบ GROUND แต่ต้องใช้แหล่งจ่ายไฟถึง 2 ชุด ส่วนวงจรในรูปที่ 9ข นั้นก็คล้ายกับวงจรในรูปที่ 9ก. เพียงแต่วงจรนี้เอาท์พุตที่ได้จะไม่เทียบกับ GROUND และใช้แหล่งจ่ายไฟเพียงชุดเดียว การปรับค่าความต้านทานทั้งสองตัวก็เช่นเดียวกับวงจรในรูปที่ 9ก.

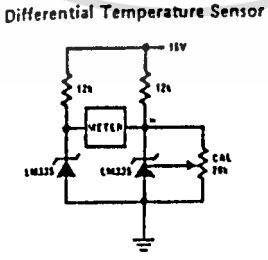
ตรวจจับความแตกต่างของอุณหภูมิ

IC LM 335 สองตัวว่างในตำแหน่งที่ต่างกันแล้วใช้วงจรในรูปที่ 10 ก็สามารถวัดความแตกต่างของอุณหภูมิระหว่างตำแหน่งทั้ง 2 นี้ได้ และใช้ตัวต้านทานปรับค่าได้ 20k ในการปรับแต่ง ห้เข็มของมิเตอร์อยู่ในตำแหน่งศูนย์กลางของหน้าปัด หอดีเมื่อ IC ทั้ง 2 ตัว อยู่ในตำแหน่งที่อุณหภูมิเท่ากัน

ในรูปที่ 11 ใช้ออปแอมป์ในการเปรียบเทียบแรงดันของเอาท์พุตของ IC ทั้ง 2 ตัว ซึ่งต่อกันเหมือนกับในรูปที่ 11 แต่ออปแอมป์ถูกจัดให้มีการป้อนกลับทางลบ เพื่อให้มีอัตราขยายเท่ากับ 10 ดังนั้นเอาท์พุตของออปแอมป์จะให้ความสัมพันธ์ของอุณหภูมิเป็นองศาเซลเซียสกับแรงดัน 100 mV/°

ตรวจจับการไหลของอากาศ

ในวงจรรูปที่ 12 ใช้หลักการของความร้อนที่เกิดขึ้นในตัวเองมาตรวจจับการไหลของอากาศ โดยที่กระแสจำนวนมากจะไหลผ่าน LM 335 ตัวบนทำให้อุปกรณ์ตัวนี้อุ่นขึ้น ถ้ามีอากาศไหลผ่าน IC ตัวนี้อย่างเร็ว IC ตัวนี้จะเย็นลงทำให้แรงดันเอาท์พุตลดลง เนื่องจากแรงดันนี้ป้อนเข้าสู่ขาอินเวิร์ทติ้งของ LM 301A จึงทำให้



รูปที่ 10 เป็นวงจรตรวจจับความแตกต่างของอุณหภูมิ 2 ตำแหน่ง

เอาท์พุตของออปแอมป์นี้มีค่าสูง เมื่อมีอากาศไหลผ่านอย่างรวดเร็ว ส่วน LM 335 ตัวล่าง (ไม่ได้มีอากาศไหลผ่าน) จะใช้เป็นตัวกำหนดแรงดันเปรียบเทียบโดยรับอุณหภูมิของสิ่งแวดล้อมโดยรอบแล้ว ปรับของอุณหภูมิที่เริ่มทำงาน

การประยุกต์ใช้งานกับเทอร์โมคัปเปิล

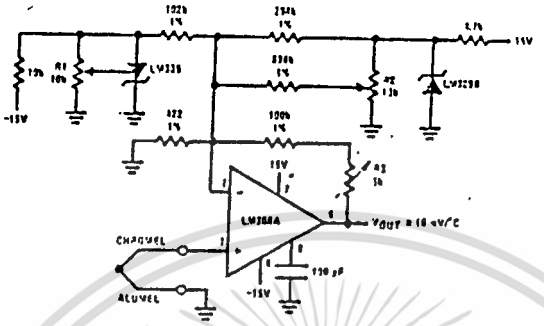
เทอร์โมคัปเปิลใช้กันมากในการวัดอุณหภูมิในย่านที่กว้างกว่าที่ IC LM 335 จะทำได้ ส่วนหนึ่งก็เนื่องมาจากมันมีราคาถูกและใช้งานได้ง่ายกว่า แม้ว่าเทอร์โมคัปเปิลจะใช้วัดอุณหภูมิในย่านที่สูงถึง 1,000° โดยใช้หลักการของรอยต่อระหว่างโลหะ 2 ชนิดที่ต่างกัน แต่ก็ต้องมีรอยต่ออ้างอิงที่เย็นเป็นตัวเปรียบเทียบ ส่วนใหญ่แล้วจะใช้อ่างน้ำแข็ง ยกเว้นเมื่อใช้วัดแบบความแตกต่างเท่านั้น

แทนที่จะใช้อ่างน้ำแข็งก็จะเป็นการสะดวกกว่า ถ้าเราใช้เทคนิคที่รู้จักกันในนามของ "การชดเชยของรอยต่อที่เย็น" แรงดันที่ชดเชยจะรวมเข้ากับเอาท์พุตจากเทอร์โมคัปเปิล ดังนั้นความต่างศักย์ของรอยต่ออ้างอิงจะดูเหมือนอยู่ที่ 0° แม้ว่าจริง ๆ แล้วมันจะอยู่ที่อุณหภูมิอื่น แรงดันที่เพิ่มขึ้นมานั้นสามารถทำให้การแปรโดยตรงกับอุณหภูมิด้วย ค่าคงที่ของการแปรผันเช่นเดียวกับเทอร์โมคัปเปิล ดังนั้นการเปลี่ยนแปลงของอุณหภูมิของบรรยากาศจึงไม่มีผลต่อแรงดันเอาท์พุต

การใช้ LM 335 เป็นตัวตรวจจับอุณหภูมินั้น เหมาะสมสำหรับใช้เป็นวงจรชดเชยรอยต่อที่เย็น เนื่องจากความเป็นเชิงเส้นของคุณสมบัติระหว่างแรงดัน และอุณหภูมิ นอกจากนั้นแรงดันของ LM 335 ยังลดลงเป็นศูนย์องศาสัมบูรณ์สัมประสิทธิ์ของอุณหภูมิของวงจรชดเชยสามารถที่จะปรับให้เท่ากับอุณหภูมิห้อง ได้โดยปราศจากวัฏจักรของอุณหภูมิ

เทอร์โมคัปเปิลที่ใช้เทอร์เมอร์คัปเบิ้ล และ ปรับค่าให้วัดออกมาเป็นองศาเซลเซียส แสดงในรูปที่ ๗1 รอยต่ออ้างอิงของเทอร์โมคัปเปิลควรจะอยู่ใกล้กับLM 335 ให้มากที่สุด เพื่อที่อุณหภูมิของอุปกรณ์ของอุปกรณ์สองชิ้นนี้จะได้ไม่แตกต่างกันนัก โดยเริ่มแรกควรจะป้อนสัญญาณเข้าไปแทนที่เทอร์โมคัปเปิล และปรับ VR_3 ให้ได้อัตราขยายเท่ากับ 245.7 เท่า เสร็จแล้วจึงต่อขานอนอินเวิร์ทติ้งของ LM 308 ลง GROUND แล้วปรับ VR_1 ให้แรงดันเอาท์พุตเท่ากับ 2.982 V ที่อุณหภูมิ 25° แล้วปลดสายที่ต่อลง GROUND ออกแล้วปรับ VR_2 ให้ได้เอาท์พุต 246 mV ที่ 25° แล้วจึงต่อเทอร์โมมิเตอร์เข้าที่ขานอนอินเวิร์ตติ้งตามเดิม เทอร์โมมิเตอร์

แบบอิเล็กทรอนิคส์นี้จะให้เอาท์พุตออกมา 10 mV/° ตลอดย่าน 0° ถึง 1300° โดยจำเป็นต้องใช้อุปกรณ์ในวงจรที่มีคุณภาพและความเที่ยงตรง



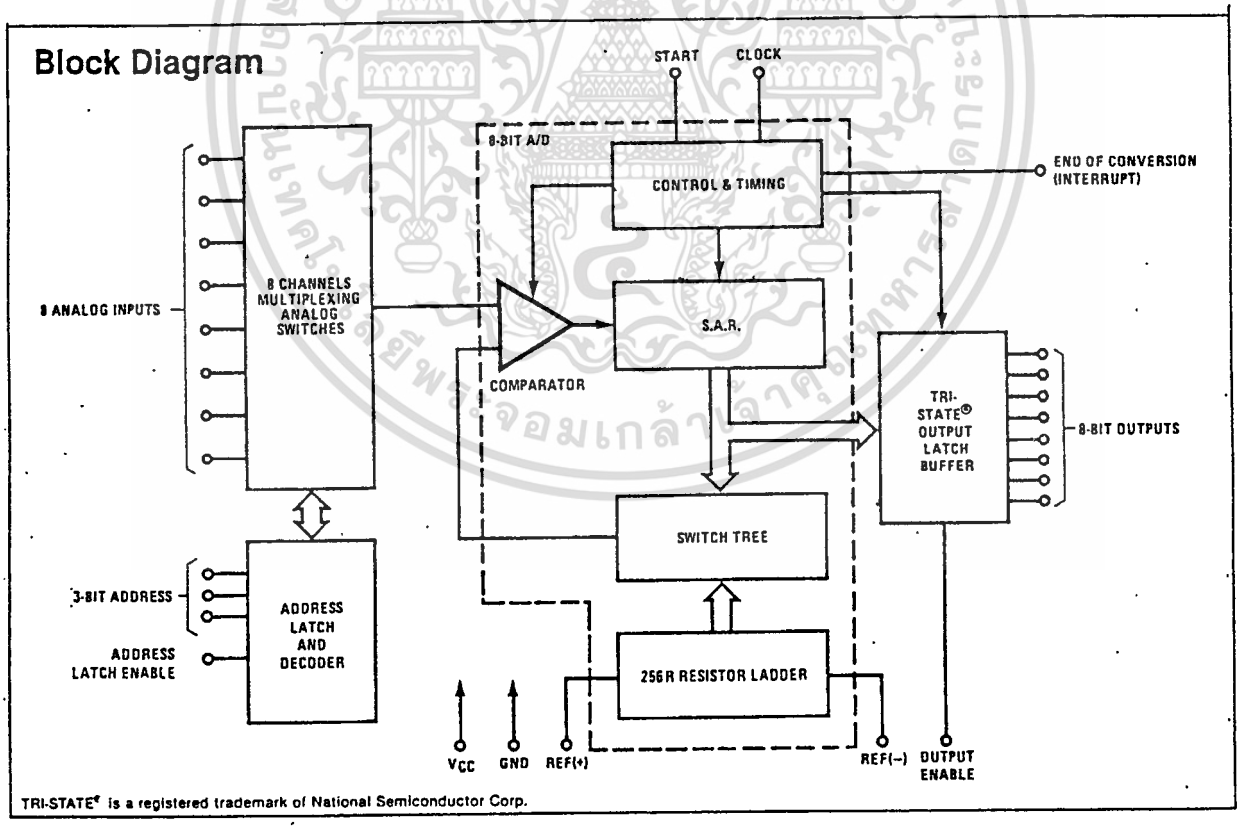
รูปที่ 11 วงจรเทอร์โมมิเตอร์แอธเซลเซียส

บทที่ 2

เอ/ดี คอนเวอร์เตอร์ ADC 0808

ADC 0808 เป็น IC แบบ CMOS ที่ใช้แปลงสัญญาณ Analog เป็นข้อมูล Digital ที่มีขนาดของข้อมูล 8 bit ซึ่งการทำงานภายในแบบ Soccessive Approximation ที่มีความเร็วปานกลาง และความเที่ยงตรงสูง

และ ADC 0808 มี Analog input ถึง 8 channel ซึ่งสามารถต่อกับสัญญาณ Analog ที่ต้องการแปลงเป็นข้อมูล Digital ได้ถึง 8 ช่องสัญญาณ โดยมีขา multiplex ในการเลือก Analog input อยู่ 3 ขา คือ ADD A, ADD B และ ADD C ดังรูปเป็นโครงสร้างภายใน ADC 0808

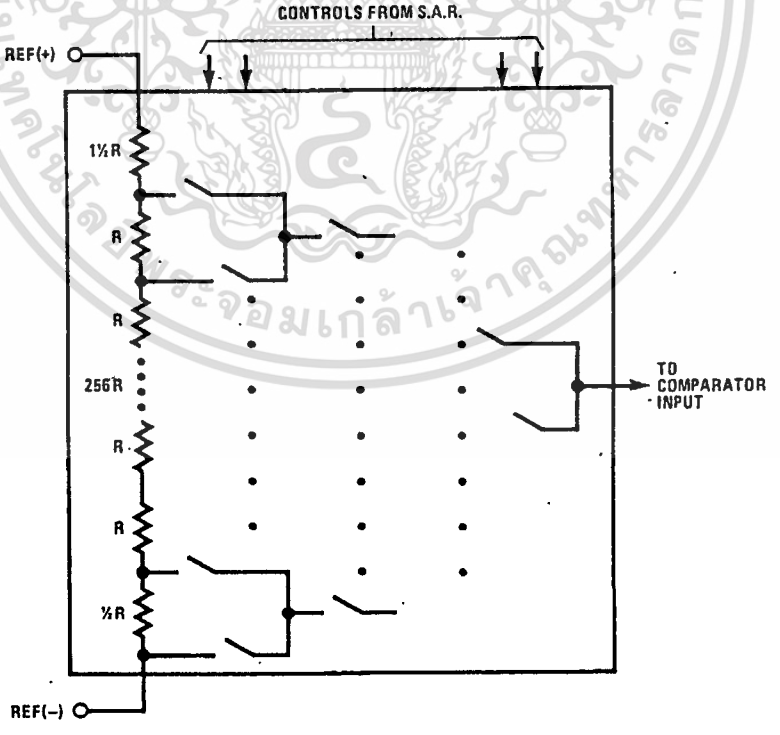


รูปที่ 11 โครงสร้างของ ADC 0808

ลักษณะสำคัญ

- สามารถเปลี่ยนเป็นสัญญาณดิจิตอลได้ถึง 8 บิท
- ความไม่ลิเนียร์ประมาณ $\pm$1/2 LSB และ $\pm$ 1 LSB
- แหล่งจ่ายไฟ 5 VDC
- 8 channel Analog input และ multiplex with Latchd control logic
- ง่ายในการต่อใช้งาน และ Interface กับ microprocessor
- ย่านอุณหภูมิ -20°C to + 85°C หรือ -55°C to +125°C
- Latched TRI-STATE output

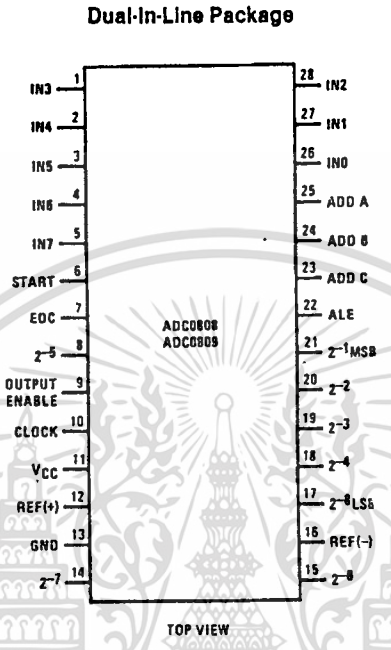
ลักษณะของ ADC แบบ Successive Approximation Register (SAR) โดย ADC ภายในเป็นแบบ 256R ladder network โดยผ่าน switch three โดยควบคุมจาก SAR



Resistor Ladder and Switch Tree

รูปที่ 11.2

รูปที่ 11.3 เป็นรูปแสดงขาของ IC ADC 0808



รูปที่ 11.3

การทำงานของขา ADC 0808

IN0 - IN7	เป็นขา Analog signal ที่สามารถต่อสัญญาณ analog ได้ 8 สัญญาณ
ADD A,B และ C	เป็นขา multiplex สัญญาณ Analog Input ตามที่ต้องการแปลงสัญญาณ Analog to Digital
ALE	(ADDRESS LATCH BNABLE) เป็นขา input ที่รับสัญญาณเพื่อ Latch ข้อมูลขนาด 3 bit ที่ขา ADD, ADD B และ ADD C เพื่อในการ multiplex
START	เป็นขา input เพื่อรับสัญญาณในการ START ให้ ADC 0808 ทำงานในแปลงสัญญาณ Analog เป็น Digital
2^{-8} - 2^{-1}	เป็นขาของข้อมูล Digital ขนาด 8 bit
OE	(output ENABLE) เป็นขาที่ควบคุมข้อมูล Digital ขนาด 8 bit ว่าต้องการที่จะให้ส่งออกทางขา output ของ ADC 0808 หรือไม่
EOC	(END OF CONVERSION) เป็นขา output ที่จะส่งสัญญาณเพื่อบอกว่าการทำงานของ ADC 0808 ทำการแปลงข้อมูลเสร็จแล้วหรือยัง

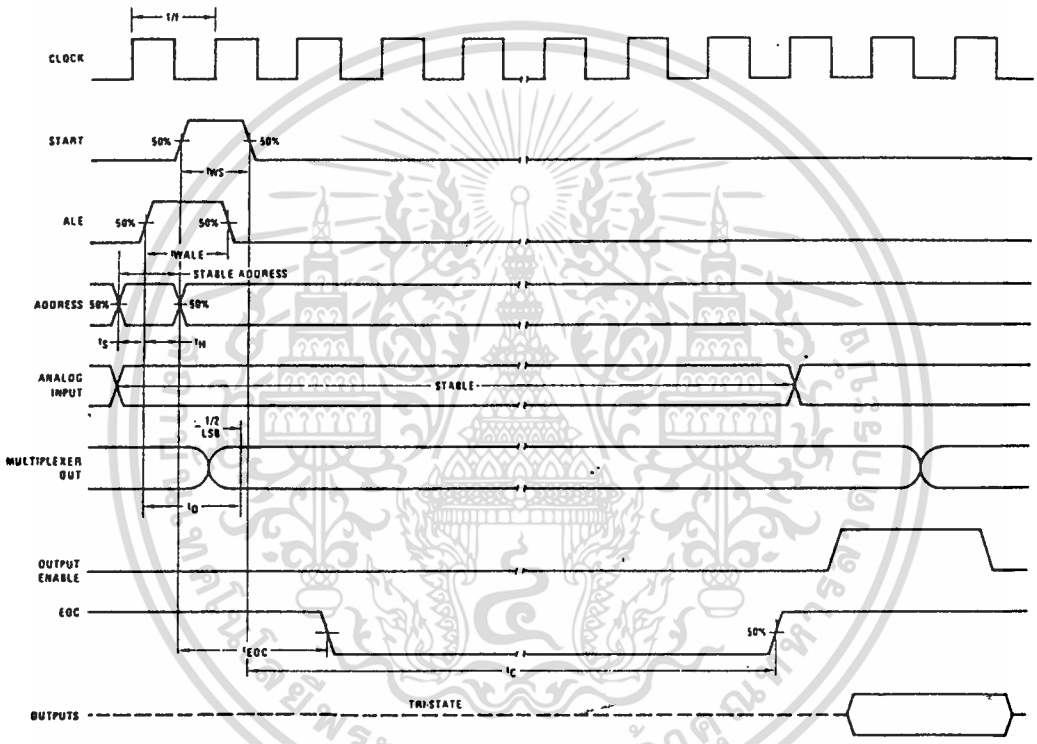
จากโครงสร้างของ ADC 0808 จะพบว่า ADC 0808 มี Analog input อยู่ 8 channel ที่สามารถต่อ analog signal เพื่อที่จะแปลงเป็น Digital 8 bit ได้ถึง 8 channel และในการ multiplex channel ที่จะทำการแปลง จะต้องกำหนดข้อมูลขนาด 3 bit ให้กับขา ADD A, ADD B และ ADD C ดังตารางรูปที่ 12

SELECTED ANALOG CHANNEL	ADDRESS LINE		
	C	B	A
IN0	L	L	L
IN1	L	L	H
IN2	L	H	L
IN3	L	H	H
IN4	H	L	L
IN5	H	L	H
IN6	H	H	L
IN7	H	H	H

รูปที่ 12

การทำงานตามแผนผังเวลา

Timing Diagram

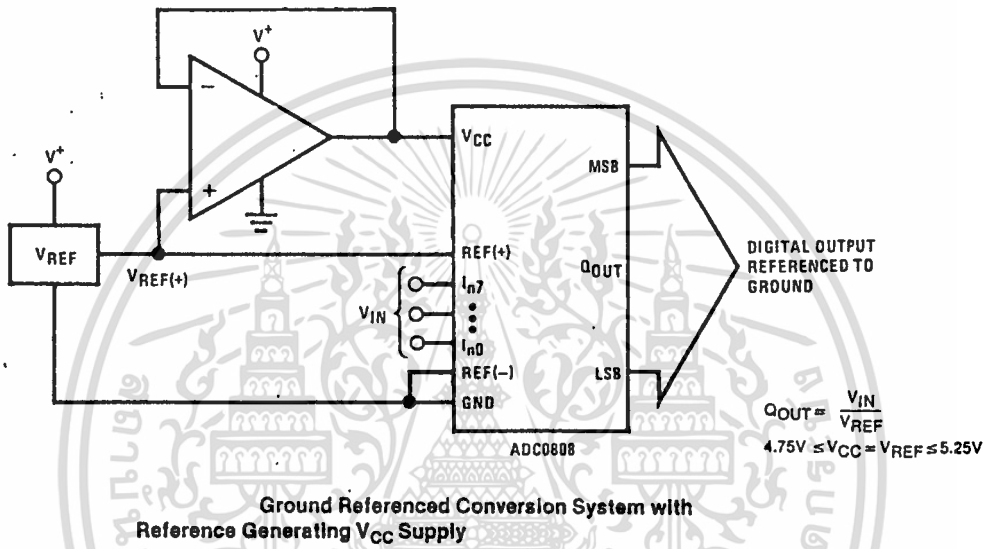


จากแผนผังเวลา (timing diagram) การทำงานของวงจรจะต้องสัมพันธ์กับสัญญาณนาฬิกา (clock) การทำงานเป็นไปตามลำดับดังนี้

ขั้นตอนการทำงาน

1. กำหนด channel ของ analot input ที่ต้องการโดยป้อนเป็นสัญญาณ digital 3 Bit เข้าที่ขา ADD A, ADD B, ADD C
2. ป้อนพัลซ์บวกเข้าที่ขา ALE (ADDRESS LATCH ENABLE) เพื่อทำการ Latch ข้อมูลขนาด 3 bit ที่ป้อนเข้าที่ขา ADD A, ADD B, ADD C และวงจรจะทำการ multiplex analog channel ตามข้อมูล 3 bit ตามตารางรูปที่ 12
3. ป้อนพัลซ์บวกที่ขา START เพื่อให้วงจรทำงานแปลงข้อมูล
4. เมื่อทำตามขั้นตอนที่ 1, 2 และ 3 เสร็จ ADC 0808 จะทำการเลือก channel ที่เป็น analog signal ตาม Data 3 bit มายังวงจรภายใน เข้ากับ Comparator in สัญญาณ analog ก็จะถูกทำการแปลงเป็นสัญญาณ digital โดยมีขนาด 8 bit ขณะที่ทำการแปลงสัญญาณนั้น ที่ขา EOC (END OF CONVERSION) จะ Active Low จนกระทั่งวงจรภายในแปลงสัญญาณเสร็จ EOC ก็จะเปลี่ยนสถานะเป็น High นั้นแสดงว่าข้อมูลที่นำมาแปลงเป็นสัญญาณ Digital เรียบร้อยแล้ว
5. OUTPUT ของ ADC 0808 โดยมี 3 สถานะ โดยปกติจะอยู่สถานะ High Impedance ดังนั้นถ้าต้องการให้ ADC แสดงข้อมูลออกทาง o/p ก็ให้ป้อนพัลซ์บวกที่ขา OUTPUT ENABLE ด้าน OUTPUT จึงแสดงค่าข้อมูลที่เป็น digital ออกมาโดย OUTPUT จะแสดงค่านานเท่าใดนั้น ขึ้นอยู่กับความกว้างของพัลซ์บวกที่ป้อนให้ที่ขา OE (OUTPUT ENABLE)

ในการนำไปใช้งาน วงจร เอ/ดี คอนเวอร์เตอร์ สามารถต่อวงจรได้ ดังนี้

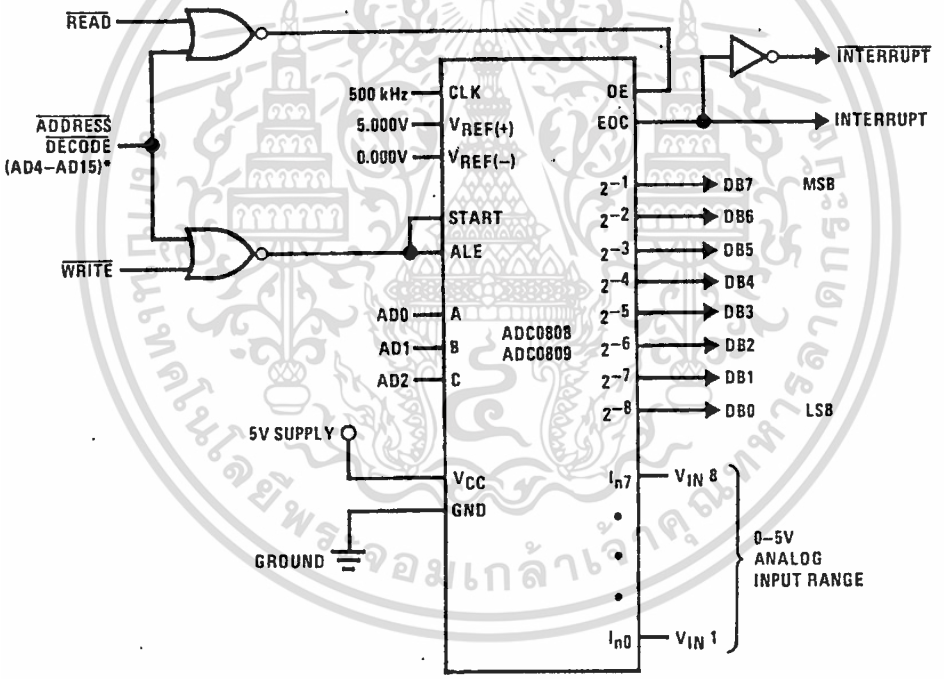


Ground Referenced Conversion System with
Reference Generating V_{CC} Supply

รูปที่ 14 ระบบการแปลงแบบเป็นอัตราส่วน

จากรูปที่ 14 เราจะเห็นว่า ขา ref (+) ต่อเข้ากับแหล่งจ่าย (Vcc) และ ref (-) ต่อเข้ากับ ground ทำให้เราใช้งานได้ในช่วงจำกัด คือ อินพุทที่ป้อนเข้ามาจะอยู่ระหว่างแหล่งจ่าย (Vcc) กับแท่น ground

สำหรับการนำไปใช้งานในการ interface กับ microprocessor นั้น จะแสดงดังรูปที่ 15 โดยในกรณีที่จะให้วงจรทำการแปลงสัญญาณ ซึ่งจะใช้สัญญาณ Write จาก Microprocessor มาทำการ Latch Channel ที่ต้องการแปลงและพร้อมทำการ Start และที่ขา EOC ซึ่งนำไปใช้ประโยชน์ในการ interrup Microprocessor ให้ Microprocessor รับรู้ว่าขณะนี้ วงจรได้ทำการแปลงสัญญาณเสร็จเรียบร้อยแล้ว และเมื่อ Microprocessor ต้องการอ่านข้อมูล ก็จะใช้สัญญาณจากขา READ มาทำการ Output Enabel (OE) เพื่อให้วงจรแสดงข้อมูล ขนาด 8 bit ที่ขา Output



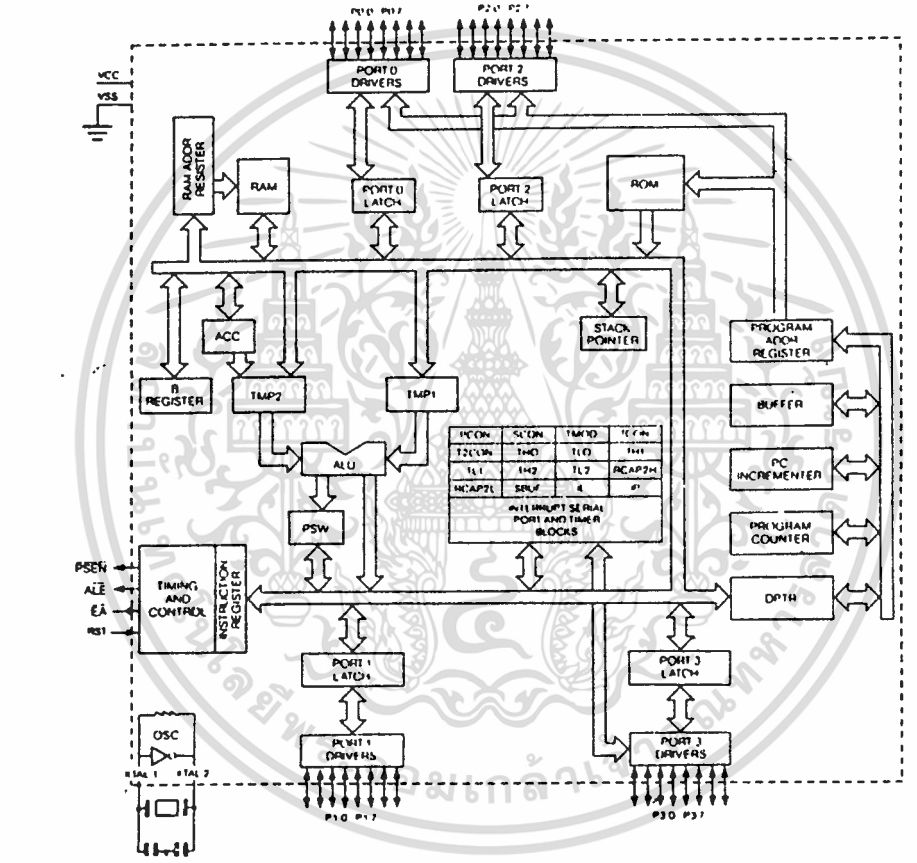
* Address latches needed for 8085 and SC/MP Interfacing the ADC0808 to a microprocessor

รูปที่ 15

บทที่ 3

MCS 51

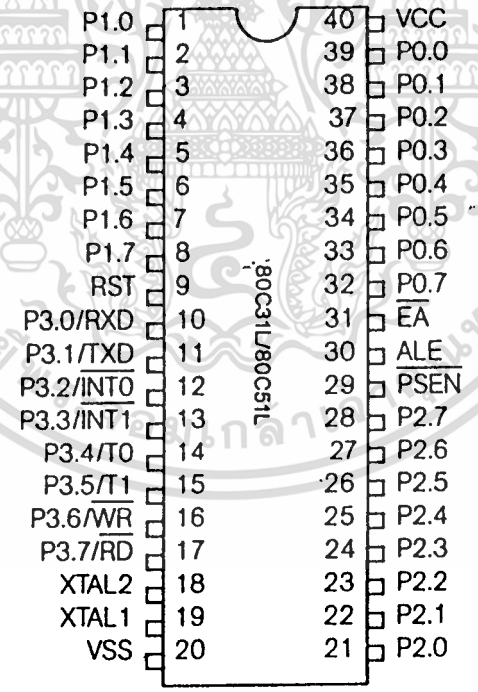
ไมโครคอนโทรลเลอร์ที่ใช้ในโครงงานนี้ คือ MCS-51 เป็นไมโครคอนโทรลเลอร์แบบชิพเดี่ยว ซึ่งผลิตโดยบริษัท Intel โดยมีการประมวลผลแบบ 8 บิต



รูปที่ 16 สถาปัตยกรรมภายใน 8051

ข้อมูลเกี่ยวกับ MCS-51

- วงจรรวมแบบ Dual Inline Package (DIP) ขาทั้งหมด 40 ขา
- ใช้ไฟ VCC +5 โวลท์ สามารถต่อเข้ากับอุปกรณ์ลอจิกแบบ TTL ได้โดยตรง
- มีหน่วยความจำสำหรับโปรแกรมภายในขนาด 4 กิโลไบต์
- ความถี่สัญญาณนาฬิกา 11.0592 เมกะเฮิรตซ์
- มีพอร์ทแบบขนาน (Porallel Port) สำหรับข้อมูลเข้าออกจำนวน 32 บิต
- มีพอร์ทอนุกรมมาตรฐาน RS232 แบบ Full Duplex
- สามารถอ้างหน่วยความจำภายนอกสูงสุด 64 กิโลไบท์



รูปที่ 17 ไดอะแกรมขาของ 8051 แบบ DIP

หน้าที่การทำงานของขา MCS-51

VCC ขาป้อนไฟเลี้ยว + 5

VSS ขาที่ต่อกับกราวด์ของแหล่งจ่ายไฟ

Port 0 เป็น พอร์ทขนาด 8 บิต โดยใช้ได้ทั้งการรับ-ส่ง ตำแหน่งและข้อมูลกับหน่วยความจำโดยพอร์ท 0 นี้ จะส่งข้อมูลเพียงชั่วขณะหนึ่ง แล้วจะกลับมาทำหน้าที่รับข้อมูลต่อ

Port 1 นี้จะทำหน้าที่รับและส่งข้อมูลเท่านั้น

Port 2 ใช้ส่งค่าตำแหน่งหน่วยความจำภายนอกที่ต้องการติดต่อและใช้เป็นพอร์ทรับและส่งข้อมูลกับภายนอก

Port 3

P3.0/RXD (Serial Input Port) เป็นขาที่ใช้รับข้อมูลแบบอนุกรม

P3.1/TXD (Serial Output Port) เป็นขาที่ใช้ส่งข้อมูลแบบอนุกรม

P3.2/INTO (External Interrupt) ใช้รับสัญญาณขัดจังหวะจากภายนอก

P3.3/INT1 (External Interrupt) ใช้รับสัญญาณขัดจังหวะจากภายนอก

P3.4/TO (Timer/Counter O External Input) ขารับสัญญาณเข้าไปยังวงจร Timer/Counter O ทำหน้าที่ที่นับจำนวนไซเคิลของสัญญาณ TO นี้ หรือสัญญาณนาฬิกาก็ได้

P3.5/T1 (Timer/Counter 1 External Input) ขารับสัญญาณเข้าไปยัง Timer/Counter 1 ซึ่งมีการทำงานเหมือนกับ TO

P3.6/WR (External Data Memory Write Strobe) ขาสัญญาณควบคุมการเขียนข้อมูลไปยังหน่วยความจำสำหรับข้อมูลภายนอก 8051

P3.7/RD (External Data Memory Read Strobe) ขาสัญญาณควบคุมการอ่านข้อมูลจากหน่วยความจำสำหรับข้อมูลภายนอก

RST ขารีเซ็ทจะใช้ทำการรีเซ็ทการทำงานของ 8051 ที่ขา RST ภายในจะมีตัวต้านทานต่อระหว่างขานี้กับกราวด์ถ้าป้อนสภาวะลอจิก 1 ก็จะทำการรีเซ็ทการทำงาน จึงสามารถต่อตัวเก็บบรรจุภายนอกระหว่างขา RST กับไฟเลี้ยง + 5 โวลท์ เพื่อ Power on reset

ALE ADDRESS LATCH ENABLE

สัญญาณนี้จะใช้บอกกับอุปกรณ์ภายนอก 8051 ว่าขณะนี้สัญญาณนี้ Active (LOOIC 1) จะมีการส่งข้อมูลที่เป็น 8 บิตล่างของตำแหน่งหน่วยความจำภายนอก 8051 ที่ต้องการติดต่อออกไปทางพอร์ท 0 อุปกรณ์ภายนอกจะใช้สัญญาณ ALE Latch ข้อมูลไว้เฉพาะพอร์ท 0

PSEN PROGRAM STORE ENABLE

ขานี้ปกติจะเป็นลอจิก 1 แต่เมื่อเป็นลอจิก 0 ก็คือต้องการอ่านคำสั่ง (Fetch Instruction) ที่จะนำไปทำงานมาจากหน่วยความจำสำหรับโปรแกรมภายนอก 8051 แต่กรณีอ่านคำสั่ง ซึ่งเก็บอยู่ภายใน 8051 สัญญาณนี้จะไม่เปลี่ยนสภาวะ

EA External Access

ถ้าเป็นลอจิก 0 ที่ขา EA นี้ แสดงว่าโปรแกรมที่ต้องการให้ทำงานถูกเก็บไว้ภายนอก 8051 ถ้าเป็นลอจิก 1 แสดงว่าโปรแกรมที่ต้องการทำงานถูกเก็บไว้ภายใน ROM 8051

XTAL 1

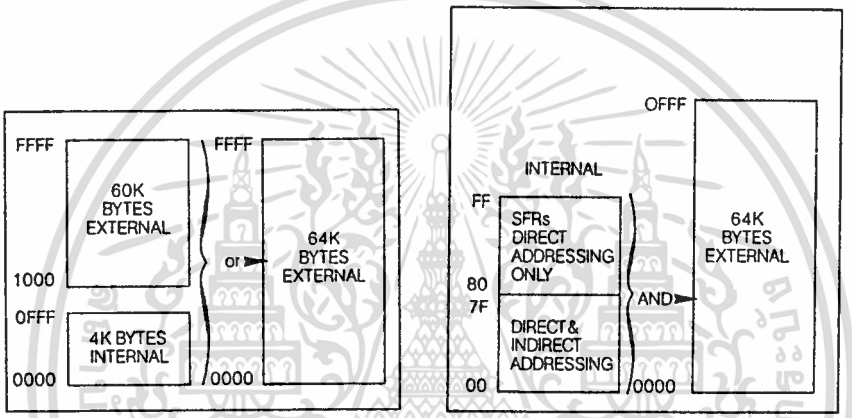
ถ้าต้องการใช้สัญญาณนาฬิกาจากภายนอกควบคุม 8051 ก็ให้ป้อนเข้ามาขานี้แต่ต้องการใช้วงจรออสซิลเลเตอร์ภายในให้ต่อ Crystal ร่วมกับคาปาซิเตอร์ ซึ่งมีค่าประมาณ 20 pF

หน่วยความจำภายใน 8051

หน่วยความจำภายใน 8051 แบ่งออกเป็น 2 แบบ คือ

- หน่วยความจำสำหรับโปรแกรม (Program Area)
- หน่วยความจำสำหรับเก็บข้อมูล (Data Area)

ดังแสดงในรูปที่ 18



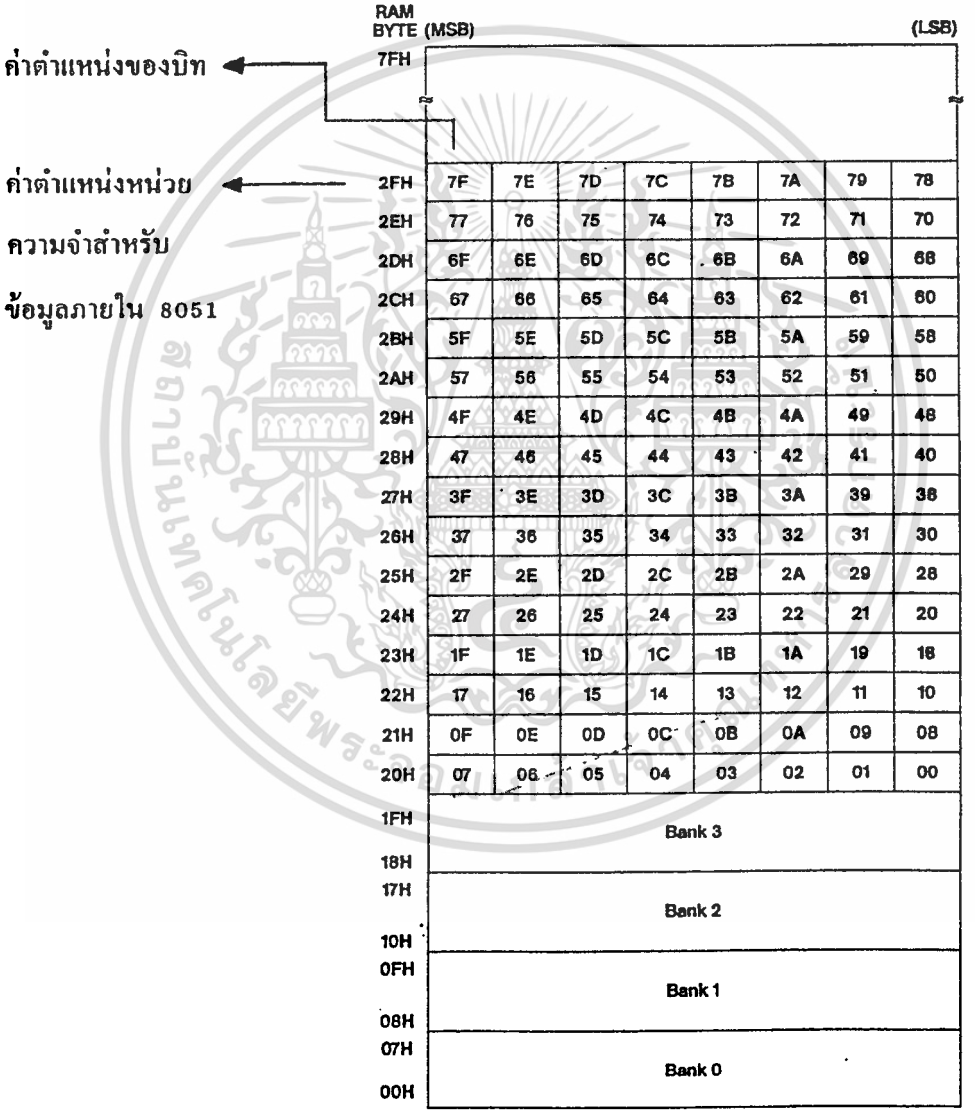
รูปที่ 18 แสดงค่าของหน่วยความจำ 8051

หน่วยความจำสำหรับโปรแกรม เป็นหน่วยความจำที่ 8051 ใช้สำหรับเก็บโปรแกรมภาษาเครื่องที่ 8051 จะทำงานเมื่อเริ่มป้อนไปเลี้ยงให้กับ 8051 ซึ่งหน่วยความจำสำหรับโปรแกรมนี้ สามารถเลือกได้ว่า จะเป็นหน่วยความจำที่อยู่ภายใน 8051 หรือภายนอก 8051

หน่วยความจำสำหรับข้อมูลที่ 8051 ใช้สำหรับเก็บหรือพักข้อมูลระหว่างการทำงาน ซึ่งหน่วยความจำนี้มี 2 แบบ

- แบบที่หนึ่งมีขนาด 128 ไบท์ อยู่ภายใน 8051
- แบบที่สองมีขนาด 64 K ไบท์ ต้องต่อเพิ่มเติมเข้าไปภายนอก 8051

รูปที่ 19 เป็นหน่วยความจำภายใน 8051 ซึ่งมีขนาด 128 byte โดยมีค่าตำแหน่งของหน่วยความจำ ตั้งแต่ 00H ถึง 7FH โดยแบ่งออกเป็น 3 กลุ่ม ดังนี้



รูปที่ 19

1.Register bank 0, bank 1, bank 2 และ bank 3 ช่วงตำแหน่งของหน่วยความจำที่ 00H ถึง 1FH โดยหน่วยความจำแบบนี้แบ่งออกเป็น 4 ชุด ชุดละ 8 byte แต่ละชุดเรียกว่า BANK แต่ละ byte ใน 1 BANK จะมีชื่อของ Register ว่า R0, R1, R2, R3 R4, R5, R6 และ R7 Register เหล่านี้จะเรียกใช้งานระหว่างการทำงานของโปรแกรมได้อย่างสะดวก และ Register เหล่านี้จะมีชื่อซ้ำกันทุก BANK โดยต่างกันที่ ตำแหน่งของหน่วยความจำ ในการใช้งานจะใช้ได้ครั้งละ1 BANK เท่านั้น โดยการกำหนดค่าใน Register PSW รูปที่ 20 เป็น Register แต่ละ BANK และค่าตำแหน่งของหน่วยความจำ

รีจิสเตอร์	ตำแหน่งหน่วยความจำ			
	BANK 0	BANK 1	BANK 2	BANK 3
R0	0	8	10	18
R1	1	9	11	19
R2	2	A	12	1A
R3	3	B	13	1B
R4	4	C	14	1C
R5	5	D	15	1D
R6	6	E	16	1E
R7	7	F	17	1F

รูปที่ 20

2. Bit Address Area เป็นหน่วยความจำในช่วงตำแหน่ง 20H ถึง 2FH หน่วยความจำแต่ละบิทในพื้นที่นี้ จะสามารถตรวจสอบหรือตั้งค่า 1 หรือ 0 ได้โดยการโปรแกรม และในแต่ละบิทของข้อมูลในหน่วยความจำจะมีค่าตำแหน่งดังใน Memory Map รูปที่ 20

3. Scratched Pod Area เป็นช่วงหน่วยความจำตำแหน่ง 30H ถึง 7FH หน่วยความจำในช่วงนี้สามารถใช้งานในการเก็บข้อมูลทั่วไป เช่น Stact Pointer

Flag Register เป็น Register ที่ใช้เก็บสภาวะที่เกิดขึ้นระหว่างการคำนวณ หรือจะใช้เลือก Bank ของ 8051 Register นี้คือ PSW Program Status Word มีขนาด 8 bit

(MSB) (LSB)

CY	AC	F0	RS1	RS0	OV	–	P

Symbol	Position	Name and Significance
CY	PSW.7	Carry flag.
AC	PSW.6	Auxiliary Carry flag. (For BCD operations).
F0	PSW.5	Flag 0 (Available to the user for general purposes).
RS1 RS0	PSW.4 PSW.3	Register bank select control bits 1 & 0. Set/cleared by software to determine working register bank (see Note).
OV	PSW.2	Overflow flag.
–	PSW.1	User definable flag.
P	PSW.0	Parity flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of "one" bits in the Accumulator, i.e., even parity.

Note :
The contents of (RS1, RS0) enable the working register banks as follows :

(0.0)—Bank 0 (00H–07H)
(0.1)—Bank 1 (08H–0FH)
(1.0)—Bank 2 (10H–17H)
(1.1)—Bank 3 (18H–1FH)

รูปที่ 21

- PSW.0 บิท 0 คือ Parity bit โดยจะใช้บอกว่าใน Register Accumulator มี 1 เป็นจำนวนคู่หรือจำนวนคี่
- PSW.1 บิท 1 บิทที่ไม่ได้ใช้งาน
- PSW.2 บิท 2 คือ Overflow Flag เป็นบิทที่บอกการคำนวณนั้น ทำให้เกิดตัวทดขึ้นในระหว่าง การคำนวณตัวทดนี้เกิดจาก บิทที่ 6 ไปยังบิทที่ 7
- PSW.3, PSW.4 สำหรับ 2 บิทนี้ จะใช้งานร่วมกันเพื่อเป็นตัวบอกว่าขณะนี้ใช้ Register R0 ถึง R7 ใน BANK ใด ดังตารางข้างล่าง

บิท 4 (RB1)	บิทที่ 3 (RB0)	Register bank	address
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

รูปที่ 22

- PSW.5 เป็นเอนกประสงค์
- PSW.6 คือ Auxiliary Flag
- PSW.7 คือ Carry Flag

- Spectial Funtion Register,SFR

ใน 8051 มี Register ที่สำหรับใช้งานเฉพาะ คือข้อมูลที่ถูกนำไปเก็บไว้ใน Register เหล่านี้จะมีความหมายเฉพาะตัวของ Register โดยแต่ละ Register จะมีตำแหน่งของตัวมันเอง ดังแสดงในตารางข้างล่าง ซึ่งจากตารางจะแสดง Symbol และชื่อของ Register และช่องสุดท้ายคือ ตำแหน่งของ Register

Symbol	Name	Address
*ACC	Accululator	OEOH
*B	B Register	OFOH
*PSW	Program Status Word	ODOH
SP	Stack Pointer	81H
DPTR	Data Pointer 2 Bytes	
DPL	Low Byte	82H
DPH	High Byte	83H
*P0	Port 0	80H
*P1	Port 1	90H
*P2	Port 2	OAOH
*P3	Port 3	OBOH
*IP	Interrupt Priority Control	OB8H
*IE	Interrupt Enable Control	OA8H
TMOD	Timer/Counter Mode Control	89H
*TCON	Timer/Counter Control	88H
*+T2CON	Timer/Counter 2 Control	0C8H
TH0	Timer/Counter 0 High Byte	8CH
TL0	Timer/Counter 0 Low Byte	8AH
TH1	Timer/Counter 1 High Byte	8DH
TL1	Timer/Counter 1 Low Byte	8BH
+TH2	Timer/Counter 1 High Byte	0CDH
+TL2	Timer/Counter 2 Low Byte	0CCH
+RCAP2H	T/C 2 Capture Reg. High Byte	0CBH
+RCAP2L	T/C Capture Reg. Low Byte	0CAH
*SCON	Serial Control	98H
SBUF	Serial Data Buffer	99H
PCON	Power Control	87H
*IOCON (1)	IO Control	F8H

+ 80C52 and 83C154 only * bit addressable

(1) 83C154 only

รูปที่ 23

และรูปที่ 24. แสดงค่าตำแหน่งหน่วยความจำแต่ละบิท

จากรูปข้างล่างนี้จะเห็นว่า Register SFR จะใช้งานเฉพาะอย่างโดยง่าย SFR สามารถเข้าถึงแบบ BIT ADDRESS เพื่อใช้ในการตรวจสอบสภาวะการทำลายได้รวดเร็วขึ้น

Direct Byte Address	Bit Address (MSB)							(LSB)	Special Function Register Symbol
	WDT	T32	SERR	IZC	P3HZ	P2HZ	P1HZ	ALF	
0F8H	FF	FE	FD	FC	FB	FA	F9	F8	IOCON
0F0H	F7	F6	F5	F4	F3	F2	F1	F0	B
0E0H	E7	E6	E5	E4	E3	E2	E1	E0	ACC
	CY	AC	F0	RS1	RS0	OV	F1	P	
0D0H	D7	D6	D5	D4	D3	D2	D1	D0	PSW
0CDH	Not Bit Addressable								TH2
0CCH	Not Bit Addressable								TL2
0CBH	Not Bit Addressable								RCAP2H
0CAH	Not Bit Addressable								RCAP2L
	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2	
0C8H	CF	CE	CD	CC	CB	CA	C9	C8	T2CON
	PCT		PT2	PS	PT1	PX1	PT0	PX0	
0B8H	BF	–	BD	BC	BB	BA	B9	B8	IP
0B0H	B7	B6	B5	B4	B3	B2	B1	B0	P3
	EA		ET2	ES	ET1	EX1	ET0	EX0	
0A8H	AF	–	AD	AC	AB	AA	A9	A8	IE
0A0H	A7	A6	A5	A4	A3	A2	A1	A0	P2
99H	Not Bit Addressable								SBUF
	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	
98H	9F	9E	9D	9C	9B	9A	99	98	SCON
90H	97	96	95	94	93	92	91	90	P1
8DH	Not Bit Addressable								TH1
8CH	Not Bit Addressable								TH0
8BH	Not Bit Addressable								TL1
8AH	Not Bit Addressable								TL0
89H	Not Bit Addressable								TMOD
	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
88H	8F	8E	8D	8C	8B	8A	89	88	TCON
87H	Not Bit Addressable								PCON
83H	Not Bit Addressable								DPH
82H	Not Bit Addressable								DPL
81H	Not Bit Addressable								SP
80H	87	86	85	84	83	82	81	80	P0

รูปที่ 24

การทำงานของ Special Function Register

1. Accumulater Register มีขนาด 8 bit ที่ใช้สำหรับการคำนวณและเก็บค่าอ้างอิงและให้อ่านค่าของข้อมูลจากภายนอก ซึ่งต้องผ่าน Register นี้

2. B.Register ซึ่งเป็น Register ที่ใช้สำหรับการคูณ และการหาร โดย Register B จะเก็บตัวคูณและผลลัพธ์ บิท 8 ถึง 15 ในคำสั่งการคูณ สำหรับการหาร Register B จะเก็บ ตัวหารและผลการหาร

3. Programs status word ดังที่กล่าวมาแล้ว

4. Stack Pointer Register นี้จะใช้เก็บตำแหน่งของหน่วยความจำภายใน 8051 ที่ใช้เก็บตำแหน่งเดิมของโปรแกรมก่อนทำงานคำสั่ง CALL และ PUSH

5. Data Pointer Register โดย Register นี้มีขนาด 12 Bit หน้าที่ของ Register นี้ใช้สำหรับชี้ตำแหน่งในหน่วยความจำ

6. PORT 0 ถึง 3 เป็นขาที่ใช้ติดต่อรับข้อมูลจากภายนอกกับ 8051

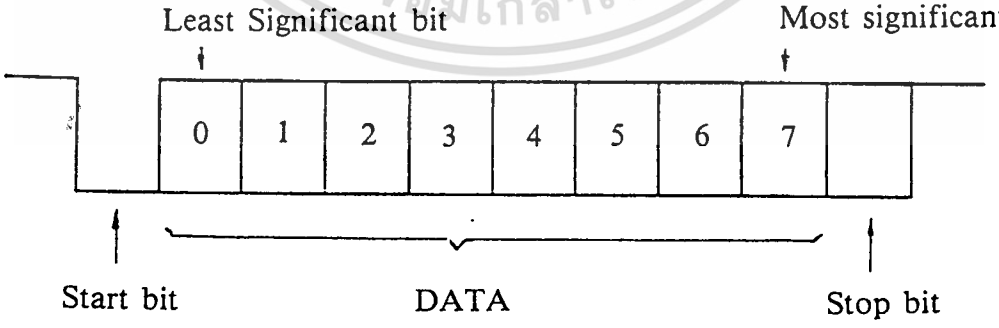
7. Serial Data Buffer

Serial Data Buffer

รีจิสเตอร์นี้มีขนาด 8 บิทและโครงสร้างภายในแล้วรีจิสเตอร์นี้มี 2 ตัวที่มีชื่อเดียวกัน ตัวหนึ่งสำหรับเก็บข้อมูลที่จะส่งแบบอนุกรมออกจาก 8051 และอีกตัวหนึ่งสำหรับรับข้อมูลแบบอนุกรมเข้ามา ดังนั้น Serial Port ของ 8051 จึงเรียกว่ามีการทำงานแบบ Full Duplex เพราะสามารถส่งและรับข้อมูลได้ในเวลาเดียวกันเนื่องจากมีรีจิสเตอร์สำหรับส่งและรับแยกออกจากกัน ข้อมูลที่ต้องการจะส่งออกก็ให้เขียนไปยังรีจิสเตอร์ SBUF แล้วสั่งงานให้ส่งข้อมูลออกมา ข้อมูลในริจิสเตอร์ จะเริ่มส่งออกโดยเริ่มจากบิท 0 ถึง 7 ตามลำดับ ถ้าข้อมูลมีข้อมูลเข้ามาทางขา RED ก็จะถูกเก็บไปไว้ในรีจิสเตอร์นี้โดยถือว่าข้อมูลบิทแรกที่เข้ามาคือบิท 0

Serial Port จะสามารถกำหนดให้การทำงาน รับ-ส่ง ข้อมูลแบบอนุกรมได้ 4 โมด (MODE) โดยการกำหนดในรีจิสเตอร์ SCON (Serial Port Control Register)แต่ละโมด การทำงานของ Serial Port มีดังนี้

MODE 0 : ในโมดนี้จะมีการรับหรือส่งข้อมูลแบบอนุกรมทางขา RXD และขา TXD จะส่งสัญญาณ Clock ที่ใช้สำหรับเลื่อน(Shift) ข้อมูล 1 ชุดของข้อมูลจะประกอบด้วยข้อมูล 8 บิทเท่านั้นและจะเริ่มการรับส่งข้อมูล-ส่งข้อมูลจากบิท 0 จนถึงบิท 7 ตามลำดับ อัตราการส่งข้อมูลแบบอนุกรมจะเท่ากับ 1/12 เท่าของความถี่สัญญาณนาฬิกาที่ใช้กับ 8051



รูปที่ 25 ชุดข้อมูลอนุกรมในโมด 1

MODE 1: ข้อมูลที่รับ-ส่ง 1 ชุดในโมดนี้จะมี 10 บิท ผ่านทางขา RXD และ TXD ตามลำดับเริ่มต้นการรับส่งข้อมูลด้วย Start bit 1 บิท (ลอจิกเป็น 0),ข้อมูล 8 บิท (เริ่มจากบิท 0), STop bit 1 บิท (ลอจิก 1) การส่งข้อมูลโมดนี้มีดังรูป 25

เมื่อรับข้อมูลอนุกรมเข้ามาข้อมูล 8 บิทจะถูกเก็บในรีจิสเตอร์ SBUF และ Stop Bit จะถูกเก็บไปที่บิท RB8 ในรีจิสเตอร์ SCON ในการส่งข้อมูลออกก็ จะเขียนข้อมูลที่ต้องการส่งไปยังรีจิสเตอร์ SBUF อัตราการส่งข้อมูลในโมดนี้สามารถกำหนดได้ตามต้องการโดยจะขึ้นกับการเกิด Overflow ใน Timer 1

MODE 2: การรับ-ส่งข้อมูลของโมด 2 1 ชุดจะมี 11 บิท ข้อมูลจะส่งออกผ่านทางขา TXD และรับเข้ามาทางขา RXD ข้อมูลแต่ละชุดจะเริ่มต้นด้วย Start bit 1 บิท, ข้อมูล 8บิท (เริ่มจากบิท 0),ข้อมูลบิทที่ 9 จำนวน 1 บิทและ Stop Bit อีก 1 บิทข้อมูลบิทที่ 9 ที่จะส่งออกนี้สามารถกำหนดได้ว่าจะให้เป็น 1 หรือ 0

รูปที่ 26 ชุดข้อมูลอนุกรมในโมด 2

อัตราการส่งข้อมูลจะกำหนดให้เป็น 1/32 หรือ 1/64 เท่าของความถี่สัญญาณนาฬิกาที่ใช้กับ 8051

MODE 3: การส่งข้อมูลโมดนี้ 1 ชุดมี 11 บิท เหมือนกับโมด 2 ทุกประการแตกต่างกันตรงอัตราการส่งข้อมูลเท่านั้น คืออัตราการส่งข้อมูลในโมด 3 นี้สามารถกำหนดได้ตามต้องการ โดยจะขึ้นกับการเกิด Overflow ใน Timer 1 เหมือนกับโมด 1

SCON (Serial Port Control Register)

รีจิสเตอร์ SCON มีขนาด 8 บิท ใช้สำหรับควบคุมการส่งและรับข้อมูลผ่านทาง Serial Port แต่ละบิทของข้อมูลในรีจิสเตอร์นี้ มีความหมายเฉพาะดังรูปที่ 27

SCON : SERIAL PORT CONTROL REGISTER. BIT ADDRESSABLE.

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0	SCON.7	Serial Port mode specifier.(NOTE 1).
SM1	SCON.6	Serial Port mode specifier. (NOTE 1).
SM2	SCON.5	Enables the multiprocessor communication feature in mode 2 & 3. In mode 2 or 3, if SM2 is set to 1 then RI will not be activated if the received 9th data bit (RB8) is 0. In mode 1, if SM2 = 1 then RI will not be activated if a valid stop bit was not received. In mode 0, SM2 should be 0. (See Table 9).
REN	SCON.4	Set/Cleared by software to Enable/Disable reception.
TB8	SCON.3	The 9th bit that will be transmitted in modes 2 & 3. Set/Cleared by software.
RB8	SCON.2	In modes 2 & 3, is the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.
TI	SCON.1	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes. Must be cleared by software.
RI	SCON.0	Receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes (except see SM2). Must be cleared by software.

NOTE 1:

SM0	SM1	Mode	Description	Baud Rate
0	0	0	SHIFT REGISTER	Fosc./12
0	1	1	8-Bit UART	Variable
1	0	2	9-Bit UART	Fosc./64 OR Fosc./32
1	1	3	9-Bit UART	Variable

SERIAL PORT SET-UP : Table 9

MODE	SCON	SM2 VARIATION
0	10H	Single Processor Environment (SM2 = 0)
1	50H	
2	90H	
3	D0H	
0	NA	Multiprocessor Environment (SM2 = 1)
1	70H	
2	B0H	
3	F0H	

รูปที่ 27 Serial Port Control Register (SCON)

ในรูปที่ 27 บิท RI จะเป็นชื่อของบิท 0 และ SMO จะเป็นบิทที่ 7 ของรีจิสเตอร์ SCON ซึ่งความหมายหรือการทำงานของแต่ละบิทมีดังนี้

RI Receive Interrupt Flag

บิทนี้จะถูกกำหนดโดย ฮาร์ดแวร์ให้มีค่าเป็น 0 หรือ 1 โดยที่ในการรับข้อมูลโมด 0 นั้น บิท RB8 จะมีค่าเป็น 1 เมื่อมีข้อมูลเข้ามาครบทั้ง 8 บิท ส่วนในโมดอื่นบิท RB8 จะเป็น 1 ก็ต่อเมื่อข้อมูลเข้ามาถึงเวลาครึ่งหนึ่งของ Stop Bit

TI Transmit Interrupt Flag

ค่าในบิท TI จะถูกกำหนดให้เป็น 1 หรือ 0 ด้วยฮาร์ดแวร์ โดยในการส่งข้อมูลแบบอนุกรมโมด 0 บิทนี้ จะเป็น 1 เพื่อจะบอกว่าการส่งข้อมูลในรีจิสเตอร์ SBUF ออกไปทางพอร์ทอนุกรมครบทั้ง 8 บิท แต่ถ้าเป็นการส่งข้อมูลแบบอนุกรมในโมดอื่น จะทำให้ข้อมูลในบิท TI เป็น 1 เมื่อเริ่มการส่ง Stop Bit

RB8

เมื่อมีการกำหนดให้รับข้อมูลในโมด 2 และ 3 จะใช้บิทนี้สำหรับเก็บข้อมูลบิทที่ 9 ที่เข้ามาทางพอร์ทอนุกรม ส่วนในโมด 1 นั้นบิทนี้จะเก็บ Stop bit ซึ่งมีค่าเป็น 1 นั่นเอง ในโมด 0 บิทนี้จะไม่ถูกใช้งาน

TB8

ในการส่งข้อมูลแบบอนุกรมโมด 2 และ 3 จะใช้บิทนี้เก็บข้อมูลบิทที่ 9 ส่วนโมดอื่นจะไม่ใช้งานบิทนี้

REN Receive Enable

เป็นบิทที่จะใช้กำหนด ให้ทำการรับข้อมูลเข้ามาจากทางพอร์ทอนุกรม หรือไม่ถ้าบิทนี้เป็น 1 ก็จะรับข้อมูลเข้ามา แต่ถ้าเป็น 0 ก็จะไม่รับข้อมูลที่ขา RXD เข้ามา

SM2

เป็นบิทสำหรับควบคุมการทำงานของฮาร์ดแวร์ที่จะทำให้บิท RI เป็น 1 หรือไม่ ในกรณีที่บิท SM2 เป็น 0 ค่าในบิท RI ก็จะเป็นไปตามที่ได้อธิบายมาแล้ว ในเรื่องบิท RI แต่ถ้าบิท SM2 = 1

โมด 2 และ 3 ปกติแล้วบิท RI จะเป็น 1 เมื่อ SM2 เป็น 1 แล้ว RI จะเป็น 1 ก็ต่อเมื่อ ข้อมูลบิทที่ 9 ที่เข้ามามีค่าเป็น 1 ถ้าข้อมูลบิทที่ 9 เข้ามาเป็น 0 จะไม่ทำให้บิท RI มีค่าเป็น 1

ในโมด 1 บิท RI มีค่าเป็น 1 เมื่อข้อมูล Stop Bit เข้ามายังพอร์ทอนุกรมถูกต้องแต่ถ้า Stop bit ไม่เข้ามายังพอร์ทอนุกรม อันอาจเกิดจากปัญหาในการส่งข้อมูลแล้วบิท RI จะมีค่าเป็น 0 ในโมด 0 บิทนี้จะมีค่าเป็น 0 เสมอ

SM0, SM1

เป็น 2 บิทที่ใช้งานร่วมกันเพื่อกำหนดโมดของการรับ-ส่งข้อมูลของพอร์ทอนุกรม ค่าใน 2 บิทนี้จะกำหนดโมดได้ดังนี้

SM0	SM1	MODE	Description
0	0	0	Shift register
0	1	1	8 -bit UART
1	0	2	9 -bit UART
1	1	3	9 -bit UART

การทำงานของแต่ละโมดจะมีดังในข้อ 28

TIMER Register

ใน 8051 จะมีวงจร Timer อยู่ 2 ชุด คือ Timer 0 และ Timer 1 ใน Timer แต่ละชุด จะมี Register ขนาด 8 บิท อยู่ 2 ตัวเพื่อเก็บค่าการนับของ Timer ได้สูงสุดถึง 16 บิท ใน Timer 0 รีจิสเตอร์นี้คือ TH0, TL0 และใน Timer 1 คือรีจิสเตอร์ TH1 TL1 TLx จะเก็บค่าของการนับ 8 บิทล่างและ THx จะเก็บค่าของการนับ 8 บิทบน

TMOD Timer/Counter mode register

GATE	C/T	M1	M0	GATE	C/T	M1	M0
TIMER 1				TIMER 0			

GATE — When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control). When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control).

C/T — Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin).

M1 — Mode selector bit. (NOTE 1)

M0 — Mode selector bit. (NOTE 1)

NOTE 1:

M1	M0	Operating Mode	
0	0	0	13-bit Timer
0	1	1	16-bit Timer/Counter
1	0	2	8-bit Auto-Reload Timer/Counter
1	1	3	(Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits, TH0 is an 8-bit Timer and is controlled by Timer 1 control bits.
1	1	3	(Timer 1) Timer/Counter 1 stopped.

รูปที่ 29 TMOD Timer/counter Mode Register

GATE เป็นบิทที่ใช้ควบคุมให้ Timer ทำงานหรือไม่ ถ้าบิทนี้ของ Timer x ถูกตั้งเป็น 1 จะทำให้ Timer ทำงานก็ต่อเมื่อที่ขา INTx มีสภาวะลอจิกเป็น 1 และบิท TRx ในรีจิสเตอร์ TCON เป็น 1 ด้วย

C/T บิทนี้ใช้สำหรับเลือกการทำงานของ Timer ว่าจะใช้เป็น Timer หรือ Counter

M1, M0 เป็น 2 บิทที่ใช้ร่วมกันเพื่อเลือกโมดการทำงานของ Timer การทำงานโมด 0, 1 และ 2 ของ Timer 0 จะเหมือนกับ Timer 1 แต่ในโมด 3

M1	M0	การทำงาน
0	0	โมด 0 รีจิสเตอร์ THx และ TLx ทำตัวเป็นตัวนับ 13 บิท ค่าจากการนับ 8 บิทบนมาจาก 8 บิทของ THx และอีก 5 บิทล่างมาจากค่า 5 บิทล่างของรีจิสเตอร์ TLx โดยที่ 3 บิทบน ของ TLx จะไม่ต้องสนใจเลย
0	1	โมด 1 รีจิสเตอร์ THx และ TLx ทำตัวเป็นตัวนับ 16 บิทค่าจากการนับ 8 บิทบนอยู่ในรีจิสเตอร์ THx และค่าจากการนับ 8 บิทล่างอยู่ในรีจิสเตอร์ TLx
1	0	โมด 2 ในการนับของรีจิสเตอร์ TLx ขนาด 8 บิทเมื่อนับถึงค่าสูงสุดคือ FFH เมื่อทำการนับต่อไปจะเกิดการ Overflow แล้วก็จะ "Reload" เอาข้อมูลจาก THx เข้าไปยัง TLx เพื่อเป็นค่าเริ่มต้นในการนับครั้งต่อไป
1	1	โมด 3 การทำงานของ Timer 0 และ Timer 1 จะต่างกันดังที่จะกล่าวต่อไป

รูปที่ 30

TCON Timer Control Register

รีจิสเตอร์ขนาด 8 บิท นี้ใช้ควบคุมการทำงานและ บอกสภาวะของ Timer 0 และ Timer 1 แต่ บิทของรีจิสเตอร์นี้จะทำงานต่างกันดังรูปที่ 31

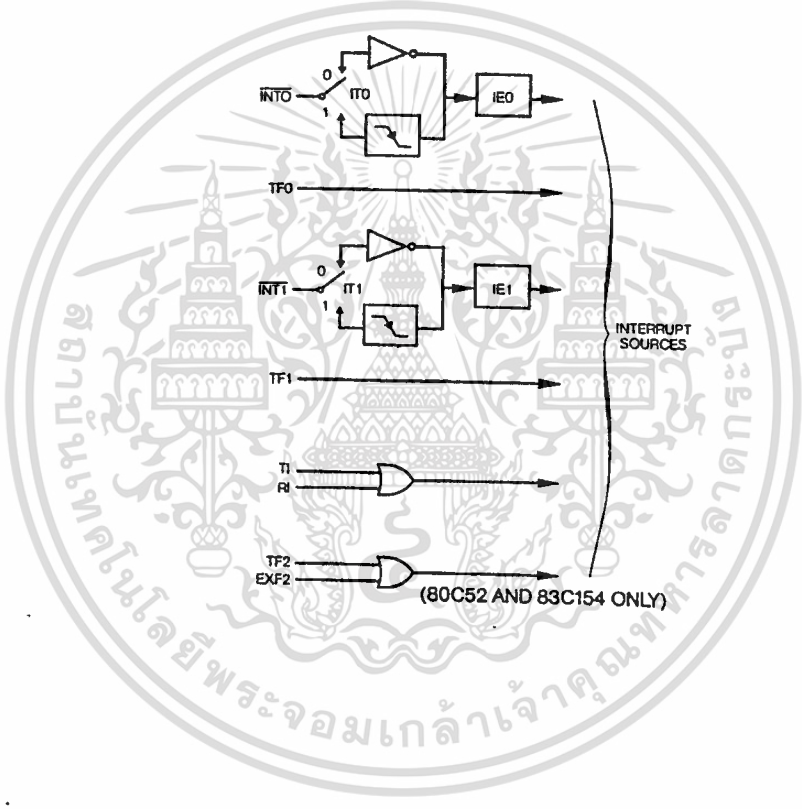
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1	TCON.7	Timer 1 overflow flag. Set by hardware when the Timer/Counter 1 overflows. Cleared by hardware as processor vectors to the interrupt service routine.
TR1	TCON.6	Timer 1 run control bit. Set/cleared by software to turn Timer/Counter 1 ONOFF.
TF0	TCON.5	Timer 0 overflow flag. Set by hardware when the Timer/Counter 0 overflows. Cleared by hardware as processor vectors to the service routine.
TR0	TCON.4	Timer 0 run control bit. Set/cleared by software to turn Timer/Counter 0 ON/OFF.
IE1	TCON.3	External Interrupt 1 edge flag. Set by hardware when External interrupt edge is detected. Cleared by hardware when interrupt is processed.
IT1	TCON.2	Interrupt 1 type control bit. Set/cleared by software to specify failling edge/flow level triggered External Interrupt.
IE0	TCON.1	External Interrupt 0 edge flag. Set by hardware when External Interrupt edge detected. Cleared by hardware when interrupt is processed.
IT0	TCON.0	Interrupt 0 type control bit. Set/cleared by software to specify failling edge/low level triggered External Interrupt.

รูปที่ 31 TCON Timer Control register

IE Interrupt Enable Register

การขัดจังหวะการทำงานเป็นการที่มีสัญญาณหนึ่งหรือคำสั่งหนึ่งที่ ที่จะทำให้การทำงานการปกติของ โปรแกรมถูกขัดจังหวะใน 8051 จะสามารถขัดจังหวะด้วยสัญญาณจาก 6 แหล่งดังรูปที่ 32



รูปที่ 32 แหล่งกำเนิดสัญญาณขัดจังหวะ

สัญญาณขัดจังหวะที่ 5 ในรูปที่ 32 จะสามารถทำให้เกิดการขัดจังหวะได้ 2 วิธี คือมีข้อมูลเข้ามาทางพอร์ทอนุกรมเก็บอยู่ที่รีจิสเตอร์ SBUF และกรณีที่ข้อมูลใน SBUF ส่งออกไปทางพอร์ทอนุกรมหมดแล้ว ไม่ว่าเกิดกรณีใด ๆ ก็ทำให้เกิดการขัดจังหวะขึ้น

PCON (Power Control Register)

Symbol	Position	Name and Function
SMOD	PCON.7	Double Baud rate bit. When set to a 1, the baud rate is doubled when the serial port is being used in either modes 1,2 or 3.
HPD	PCON.6 (83C154 only)	Hard Power Down bit. Setting this bit allows CPU to enter in Power Down state on an external event (1 to 0 transition) on bit T1 (p.3-5) the CPU quit the Hard Power Down mode when bit T1 (p. 3-5) go high or when reset is activated.
RPD	PCON.5 (83C154 only)	Recover from idle or Power Down bit. When 0 RPD has no effect. When 1, RPD permits to exit from idle or Power Down with any non enabled interrupt source (except timex 2). In this case the program start at the next address. When interrupt is enabled the appropriate interrupt routine is serviced.
–	PCON.4	(Reserved)
GF1	PCON.3	General-purpose flag bit.
GF0	PCON.2	General-purpose flag bit.
PD	PCON.1	Power Down bit. Setting this bit activates power down operation.
IDL	PCON.0	Ide mode bit. Setting this bit activates idle mode operation.

รูปที่ 33 PCON : Power Control Register

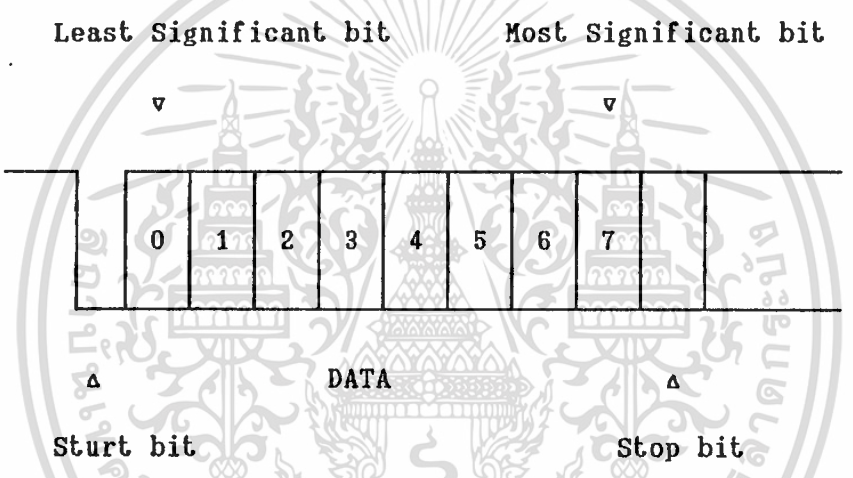
8051 เป็นไมโครโปรคอนโทรลที่สร้างขึ้นด้วยเทคโนโลยีทั้งแบบ CHMOS และ HMOS ซึ่งแบบ CHMOS มีข้อดีตรงที่ใช้กำลังไฟต่ำกว่าแบบ HMOS ดังนั้นต่อไปในอนาคตจึงจะมีแต่เฉพาะรุ่น CHMOS เท่านั้น นอกจากนี้แล้ว 8051 ยังมีข้อดีอีกตรงที่สามารถลดการใช้กำลังไฟลงได้โดยการทำงานใน Idle Mode และ Power Down Mode ใน Idle Mode นั้น สัญญาณนาฬิกาออกจากออสซิลเลเตอร์จะป้อนให้เฉพาะส่วน Interrupt, Serial Port และ Timer ในส่วนอื่นจะไม่มีสัญญาณนาฬิกาไปเลี้ยง แต่มีไฟเลี้ยง ให้กับทุกส่วนในวงจร การใช้กำลังไฟจึงลดลงมาก ส่วนใน Power Down Mode นั้น ออสซิลเลเตอร์จะหยุดทำงาน ทำให้ไม่มีสัญญาณนาฬิกาไปเลี้ยงส่วนใด ๆ ในวงจรเลยแต่ข้อมูลภายในรีจิสเตอร์จะยังคง อยู่ไม่สูญหายไป รายละเอียดของแต่ละโมดจะได้กล่าวต่อไป

SMOD บิท 7 เป็นบิทที่ใช้ร่วมในการกำหนดอัตราการส่งข้อมูล (Baud Rate) ผ่านทางอนุกรม ซึ่งในการรับ-ส่ง ข้อมูลผ่านทางพอร์ทอนุกรม โมด 1 และ 3 จะสามารถกำหนดอัตราการส่งข้อมูลได้ ตามอัตราการเกิด Overflow ใน Timer 1 ถ้าบิทนี้เป็น 1 จะทำให้อัตราการส่งข้อมูลเพิ่มขึ้น 2 เท่า รายละเอียดการส่งข้อมูลผ่านพอร์ทอนุกรม

การส่งและรับข้อมูลแบบอนุกรมผ่าน 8051

สำหรับ MCS 8051 สามารถส่งและรับข้อมูลอนุกรมแบบ Universal Asynchrnous Receive Transmitter (UART) แบบ Full duplex ที่สามารถเลือกรูปแบบในการรับส่งได้ถึง 4 แบบ

สำหรับในโครงงานได้เลือกใช้การส่งและรับข้อมูลใน MODE 1



โดย ข้อมูลที่รับส่ง 1 ชุด มีขนาด 10 Bit

Start bit 1 บิท (Logic 0)

DATA bit 8 บิท

Stop bit 1 บิท (Logic 1)

และอัตราความเร็วในการส่งข้อมูลมีค่าเป็น 9600 b/s.

โดยอัตราความเร็วในการรับส่งข้อมูล คำนวณได้จากสมการ

$$\text{Boud rate} = \frac{2^{SMOD}}{32} \times \frac{\text{Oscillator Frequency}}{12 \times [256 - (TH1)]}$$

จากโครงงานได้กำหนดใช้ timer 1 ทำงานใน Mode 2 และความถี่ระบบเท่ากับ 11.059 MHZ บิท SMOD = 0 และ Register TH1 = FDH. (253) D

$$\text{Baud rate} = \frac{2^{0}}{32} \times \frac{11.059\ M}{12 \times [256 - 253]}$$

Baud rate = 9600 b/s

ในการทำงานในการส่งและรับข้อมูล และอัตราความเร็วในการส่งข้อมูล จะต้องทำการใช้งาน SFR ต่าง ๆ ดังนี้

- SCON (Serial Port Control Register)

  Scon เป็น register เพื่อควบคุมการส่งและรับข้อมูลแบบ Serial Port ในที่นี้ ได้กำหนดให้การรับส่งข้อมูลทำงานใน MODE 1 โดยข้อมูล 1 ชุด มีขนาด 10 bit ซึ่งประกอบด้วย Start bit 1 บิท (Logic 0), DAta bit 8 bit และ Stop bit 1 บิท (Logic 1)

  และเป็น Register เพื่อให้มีการรับข้อมูลเข้ามาทาง RXD.

- TMOD (Timer/Counter Mode Register)

  TMOD เป็น register ที่ทำหน้าที่ควบคุมการทำงานของ Timer หรือ Counter ในการใช้งานนี้ได้กำหนดใช้งานให้ใช้ Timer 1 และเลือกการทำงานของ Timer 1 ใน Mode 2 โดยทำงานแบบ "Reload"

  และเป็น Register ที่เป็นตัวกำหนดให้ วงจรในการรับส่งทำงาน

- TCON (Timer Control Register)

  TCON เป็น register เพื่อใช้ตรวจสอบสภาวะการทำงานของ Timer ในที่นี้ได้ใช้ Register นี้ในการตรวจสอบว่ามีข้อมูลเข้ามาทาง RXD หรือไม่ และ

ใช้ตรวจสอบในการส่งข้อมูล

- IE (Interrupt Enable Register)

  IE เป็น register ที่ใช้ควบคุมในการ Interrupt

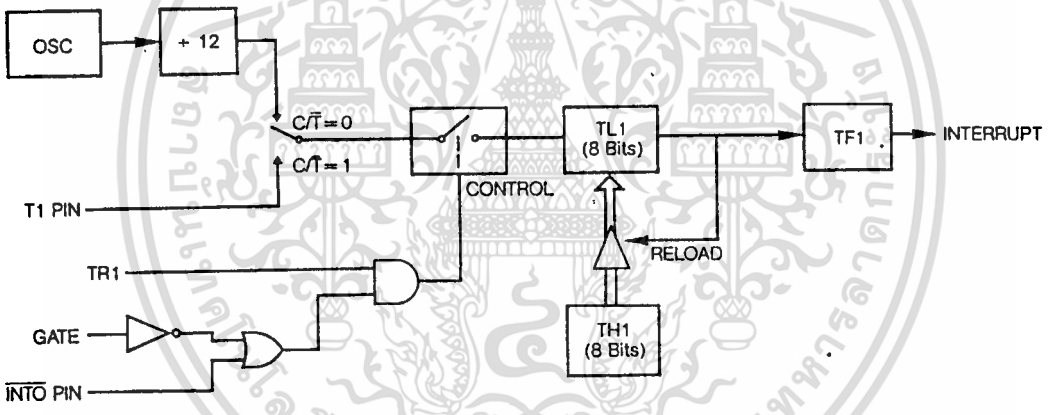
- PCON (Power Control Register)

  PCON เป็น register ที่ควบคุมการใช้งาน ในรูปของการลดกำลังงาน และอัตราเร็วในการส่งข้อมูล

Register ทั้งหมดที่กล่าวมานี้จะต้องทำงานที่สอดคล้องกันหมด จึงจะทำให้การรับส่งข้อมูลแบบ Universal Asynchonous Receive Transmitter (UART) ทำงานในอัตราความเร็ว (Baud Rate) ที่กำหนด

ในรูปที่ 34 เป็น Diagram ของวงจร Timer 1 ใน 8051 ที่ทำงานใน Mode 2 ซึ่งจะใช้ในการกำหนด อัตราความเร็วในการส่งและรับข้อมูล โดยการทำงานต้องกำหนดใน Register SFR.

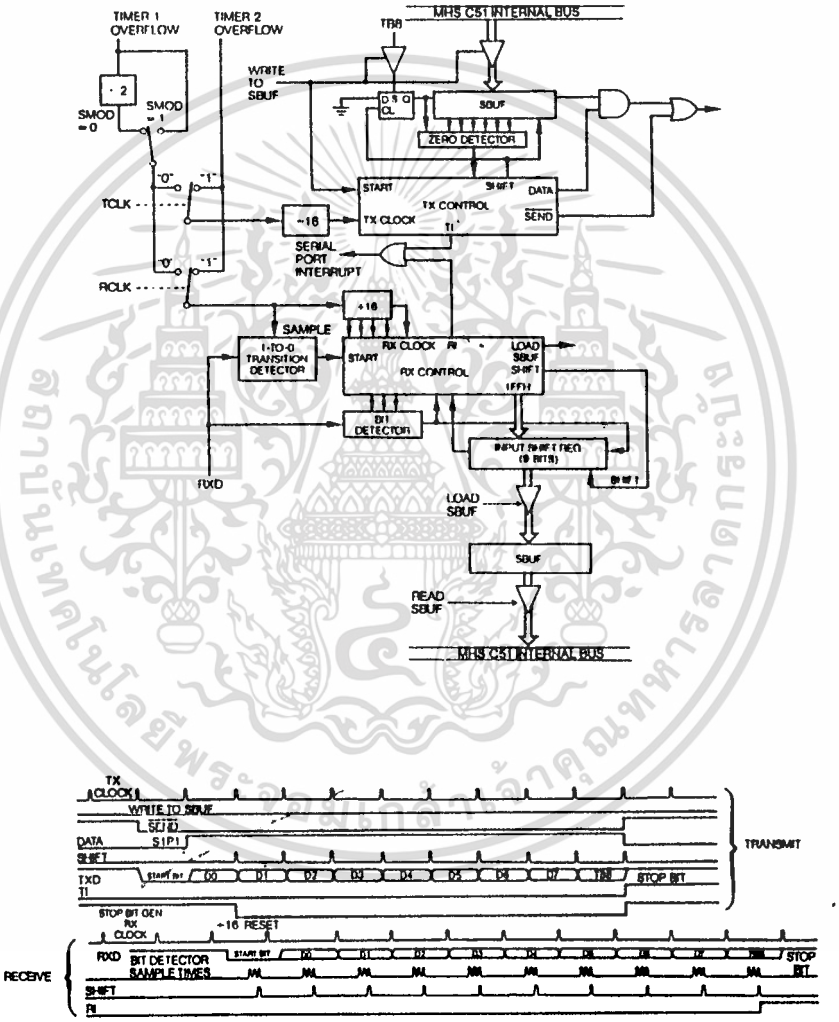
การกำหนด Baud Rate นั้น ได้กำหนดค่าที่ TH1 และวงจรทำงานใน Mode Timer 1 โดยกำหนดที่ขา C/T โดยความถี่ OSC จะถูกวงจรหาร 12 มารออยู่ที่ Control ดังนั้นการ START ให้วงจรทำงาน ก็ขึ้นอยู่กับ output ของ AND gate ที่มี Logic "1" โดยถูกควบคุมด้วย TR1 และ gate หรือ INT



Timer mode 2

รูปที่ 34

รูปที่ 35 เป็น Diagram ของโครงสร้างภายใน 8051 ที่ในการรับและส่งข้อมูล พร้อม Timming Diagram



Serial Port mode 1

การทำงานโมด 1

การส่งข้อมูล

จากรูปที่ 35 บิท SMOD จะเป็นตัวเลือกว่า สัญญาณ Timer 1 Overflow ที่ส่งไปยังวงจรหาร 16 จะถูกหาร 2 ก่อนหรือไม่ ถ้า SMOD เป็น 1 สัญญาณ Timer 1 Overflow จะไม่ถูกหาร แต่ถ้า SMOD เป็น 1 สัญญาณ Timer Overflow จะถูกหาร 2 ก่อนที่จะเข้าวงจรหาร 16 การส่งข้อมูลจะเริ่มจากการที่มีคำสั่งเขียนข้อมูลไปยังรีจีสเตอร์ SBUF จะมีสัญญาณ Write to SBUF เกิดขึ้นเพื่อรับข้อมูลจาก Internal Bus ด้านบนไปเก็บยังรีจีสเตอร์ SBUF และทำให้เอาท์พุทของ D FLIP FLOP ทางซ้ายของ SBUF มีค่าเป็น 1 และเป็นบิทที่ 9 ของการส่งข้อมูลสัญญาณ Write to SBUF ยังส่งไปยัง TX control ด้วย ขณะนี้ข้อมูลในวงจรหาร 16 มีค่าเป็นอะไรไม่ทราบจึงจะรอจนกว่าข้อมูลในวงจรหาร 16 นับเพิ่มขึ้นจนถึงค่าสูงสุดแล้ววนกลับเป็น 0 คือเกิดการวนกลับทำให้เริ่มการส่งข้อมูลที่เวลา S1P1 ของไซเคิลเครื่องถัดไป (การส่งข้อมูลออกจะสัมพันธ์กับการเกิด Overflow ในวงจรหาร 16) สัญญาณ SEND จาก TX Control เปลี่ยนสภาวะลอจิกเป็น 0 แล้วเริ่มส่งข้อมูลที่เป็น Start bit (0) ออกไป เมื่อส่ง Start Bit ออกไปแล้ววงจร Tx Control ก็จะทำให้สัญญาณ DATA เป็น 1 เพื่อเลื่อนข้อมูลใน SBUF ออกไป เริ่มจากบิท 0 จนถึงบิทที่ 7 การส่งข้อมูลนี้จะเกิดขึ้นเมื่อสัญญาณ Tx Clock เปลี่ยนสถานะจาก 0 เป็น 1 ดังในรูปที่ 36 ขณะที่ข้อมูลถูกเลื่อนออกไปนั้น จะมี 0 ถูกเลื่อนเข้ามาทางซ้ายของรีจิสเตอร์ SBUF เมื่อข้อมูลเลื่อนออกไปทั้ง 8 บิท แล้วบิทที่ 9 ซึ่งเป็น 1 และตอนต้นอยู่ทางซ้ายสุดจะถูกเลื่อนมาอยู่ในตำแหน่งสุดท้ายทางขวาของรีจีสเตอร์ SBUF และทางซ้ายของหลักนี้จะมี 0 อยู่ทั้ง 8 บิท ใน SBUF ทำให้ Zero Detector รู้ว่าเป็นข้อมูลบิทสุดท้ายแล้วที่ส่งออกโดยจะมีสัญญาณมาบอกกับวงจร Tx Control ด้วย เมื่อ TX Control ส่งสัญญาณ Shift ออกไปเป็นการส่งข้อมูลบิทสุดท้าย (บิท 7) ออกไป ก็จะรออีก 1 TX Clock (Bit Clock) ก็จะทำให้ขา TXD ส่งข้อมูล Stop Bit (1) ออกมา สัญญาณ DATA ซึ่งมีสภาวะลอจิกเป็น 1 มาตั้งแต่เริ่มส่งข้อมูลบิท 0 ก็จะกลับเป็น 0 และบิท TI จะเป็น 1 เพื่อบอกการสิ้นสุดการส่งข้อมูลทั้งหมดจะสิ้นสุดลงเมื่อสัญญาณ TX Clock ไซเคิลที่ 10 นับตั้งแต่สัญญาณ SEND เปลี่ยนสภาวะลอจิกเป็น 0

การรับข้อมูล

การรับข้อมูลจะขึ้นกับอัตราการเกิด Overflow ใน Timer 1 แล้วหาร 2 หรือไม่ขึ้นกับค่าของบิท SMOD สัญญาณนี้จะไปเข้าวงจรหาร 16 และเป็นตัวกำหนดอัตราการรับข้อมูล การรับข้อมูลจะเริ่มจากวงจร 1-TO-0 Transition Detector พบว่าสัญญาณที่ขา RXD เปลี่ยนจาก 1 เป็น 0 ซึ่งหมายถึงมีข้อมูล Start bit เข้ามา การตรวจสอบนี้จะกระทำด้วยอัตราเดียวกับสัญญาณที่เข้าวงจรหาร 16 เมื่อพบการเปลี่ยนสถานะลอจิกที่ขา RXD ก็จะเริ่มการรับข้อมูล ขณะนี้จะรีเซ็ทวงจรหาร 16 ให้มีค่าเป็น 0 เพื่อสร้างสัญญาณ RX Clock ให้เข้าจังหวะ (Synchronous) กับข้อมูลที่เข้ามาโดยสัญญาณ RX Clock จะเป็น 1 เมื่อการนับของวงจรหาร 16 มีค่าเป็น 15 ขณะที่วงจรหาร 16 นับถึง 7,8 และ 9 จะมีการตรวจสอบข้อมูลที่เข้ามาเป็นค่านั้น ถ้าในการตรวจสอบ Start Bit แล้วพบว่าผิดพลาด คือไม่เป็น 0 ก็จะรีเซ็ทการทำงานเพื่อไปตรวจสอบการเปลี่ยนสถานะจาก 1 เป็น 0 ของข้อมูลที่ขา RXD ใหม่ แต่ถ้าพบ Start bit ก็จะเก็บข้อมูลทั้งหมดที่เข้ามาโดยเลื่อนข้อมูลเข้าไปยัง Input Shift Register ที่มีสัญญาณควบคุมการเลื่อนข้อมูล (Shift) ส่งมาจาก RX control ในตอนเริ่มต้นการรับข้อมูลจะมีการเขียนข้อมูล 1FFH ไปเก็บใน Input Shift Register ขณะที่ข้อมูลถูกเลื่อนเข้าไปทางขวาของ Input Shift Register ก็จะมี 1 ถูกเลื่อนออกไปทางซ้ายทุกครั้งที่มีข้อมูลเข้ามา เมื่อ Start bit ที่รับเข้ามาถูกเลื่อนไปถึงซ้ายสุดของ Input Shift Register ก็จะมีสัญญาณไปบอก RX Control Block หลังจากข้อมูลบิทสุดท้ายเข้ามาแล้วก็จะโหลด (Load) เอาข้อมูล 8 บิท ไปเก็บในรีจิสเตอร์ SBUF พร้อมทั้ง Set ค่าในบิท RI และ RB8 ของรีจิสเตอร์ SCON แต่การโหลดข้อมูลไปเก็บนี้จะเกิดขึ้นได้ก็ต่อเมื่อ

1. RI = 0 และ

2. SM2 = 0 หรือถ้า SM2 = 1 จะต้องได้รับ stop bit เป็น 1

ถ้าไม่มีสภาวะใดสภาวะหนึ่งดังกล่าวแล้ว ข้อมูลที่รับเข้ามาก็จะถูกทิ้งไปคือไม่โหลดไปเก็บในรีจีสเตอร์ SBUF ถ้ามีสภาวะดังกล่าวถูกต้อง stop bit จะถูกนำไปเก็บในรีจิสเตอร์ RB8 และ บิท RI จะเป็น 1

แต่ไม่ว่าทั้ง 2 กรณีจะเกิดหรือไม่ก็จะกลับไปสู่การตรวจสภาวะเปลี่ยนจาก 1 เป็น 0 ที่ขา RXD เพื่อรับข้อมูลต่อไป

ในการรับข้อมูลแบบอนุกรมโมด 1 นี้ อัตราการส่งข้อมูลแต่ละบิท (Baud Rate) จะขึ้นกับอัตราการเกิด overflow ใน Timer 1 ดังสมการ

$$\text{Baud rate} = \frac{2^{SMOD}}{32} \times (\text{Timer 1 Overflow Rate})$$

ในขณะที่ใช้ Timer 1 เป็นตัวกำหนด Baud Rate นี้จะต้อง Disable ไม่ให้เกิดการขัดจังหวะเนื่องมาจากการ Overflow Timer 1 อาจใช้ในโมดของ Timer หรือ Counter ก็ได้ ซึ่งเมื่อการนับในรีจิสเตอร์ตัวนับมีค่าสูงสุดแล้วกลับมาเป็น 0 ก็จะเกิด Overflow เช่นเดียวกัน แต่โดยปกติแล้วจะใช้ Timer 1 นี้ในโมดของ Timer ที่มีการทำงานแบบ Auto Reload โมด 2 เพื่อว่าเมื่อค่าในการนับโดยรีจิสเตอร์ TL1 ถึงค่าสูงสุดก็จะโหลดค่าในรีจิสเตอร์ TH1 มาไว้ใน TL1 สำหรับเป็นค่าเริ่มต้นการนับต่อไปซึ่ง Baud rate จะมีค่า

$$\text{Baud reat} = \frac{2^{SMOD}}{32} \times \frac{\text{Oscillator frequency}}{12 \times [\ 256 - (TH1)\ ]}$$

โดยที่ SMOD เป็นบิทหนึ่งในรีจิสเตอร์ PCON

เช่นความถี่ของออสซิลเลเตอร์เท่ากับ 11.059 MHz บิท SMOD = 0 รีจิสเตอร์ TH1 มีค่า E8H, Timer 1 ทำงานในโมด 2 จะได้ว่าอัตราการส่ง-รับข้อมูลแบบอนุกรม

$$= \frac{2^{0}}{32} \times \frac{11.059 \times 10^{6}}{12 \times [256 - 232]}$$

$$= \frac{1}{32} \times \frac{11.059 \times 10^{6}}{12 \times 24}$$

$$= 1200 \text{ บิท/วินาที}$$

บทที่ 4

PORT 8255

8255 เป็นไอซี LSI ขนาด 40 ขา ดังรูป 37 ซึ่งแสดงตำแหน่งของขาต่าง ๆ ทั้ง 40 ขา สำหรับรูปที่ 36 แสดงแผนผังภายในของ 8255

ซึ่ง 8255 นี้มีพอร์ทสำหรับรับส่งข้อมูลอยู่ด้วยกัน 3 พอร์ท มีชื่อดังนี้ พอร์ท A, B และ C โดยพอร์ท C นี้จะแบ่งออกเป็น 2 ส่วน คือ พอร์ท C บน (CLO) กับพอร์ท C ล่าง (CHI) และยังมีอีกพอร์ทหนึ่ง ซึ่งทำหน้าที่ควบคุมการทำงานของพอร์ท A, B และ C โดยรับคำสั่งมาจาก CPU พอร์ทนี้ เรียกว่าพอร์ทควบคุม (Control port) การทำงานของพอร์ท จะถูกกำหนดโดย CPU โดย CPU จะส่งรหัสควบคุมมาทางดาต้าบัส (Data Bus) ให้แก่พอร์ทควบคุม หน้าที่ของขาต่าง ๆ

CS (Chip Select)

ขาใช้ในการเลือกว่าจะให้ 8255 ตัวนี้ทำงานหรือไม่ โดยถ้าได้รับลอจิก 0 จะทำให้ 8255 เชื่อมต่อเข้ากับระบบบัสต่าง ๆ ของ CPU แต่ถ้าเป็นลอจิก 1 จะอยู่สภาวะ High Impedance

RD (Read Enable)

ถ้าได้รับลอจิก 0 และ CS ได้รับลอจิก 0 เช่นกัน แสดงว่า 8255 ทำการส่งข้อมูลจากพอร์ทที่ CPU ต้องการติดต่อด้วยนั้นให้แก่ CPU ทางดาต้าบัส

WR (Write Enable)

ถ้าได้รับลอจิก 0 พร้อมกับ CS แล้ว 8255 จะส่งข้อมูลจากดาต้าบัสของ CPU ออกไปยังพอร์ทที่ CPU กำหนดไว้

RESET

ทำการ Reset 8255 เข้าสู่โหมดอินพุท ทุก ๆ พอร์ท และเคลียร์สถานะต่าง ๆ ของ 8255

D0-D7

เป็น Data Bus ที่ใช้รับส่งข้อมูลกับ CPU

A0-A1

คือขาแอดเดรสที่ใช้ในการเลือกพอร์ทที่ CPU ต้องการติดต่อ

00 = พอร์ท A

01 = พอร์ท B

10 = พอร์ท C

11 = พอร์ทควบคุม

PA0-PA7

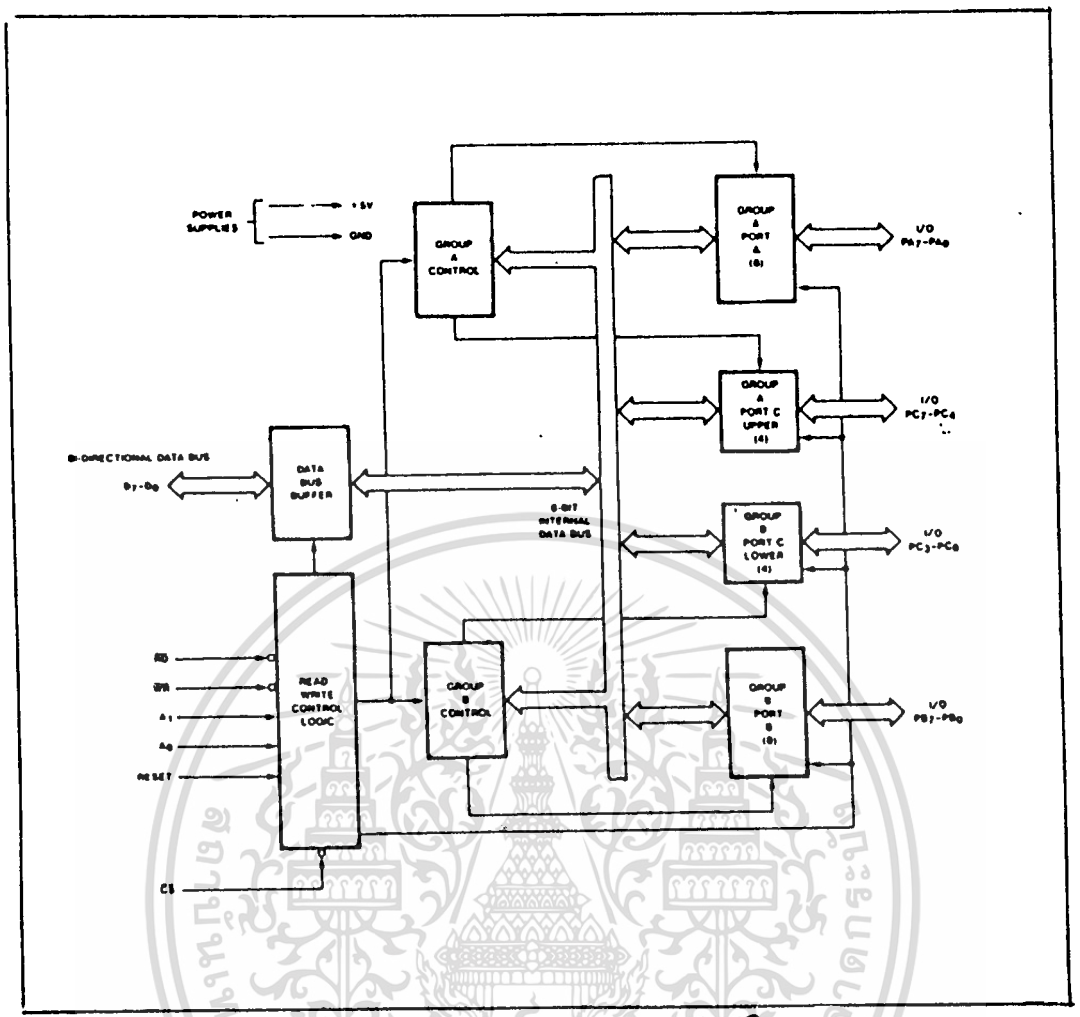
เป็นขาสัญญาณของพอร์ท A

PB0-PB7

เป็นขาสัญญาณของพอร์ท B

PC0-PC7

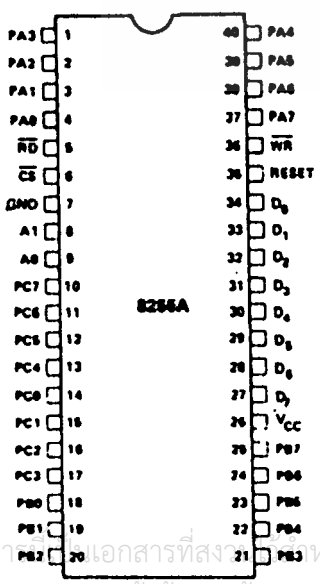
เป็นขาสัญญาณของพอร์ท C โดยแยกเป็น PC0-PC3 และ PC4-PC7 โดยสามารถแยกการทำงานได้โดยอิสระ



3,8

รูปที่ 36

PIN CONFIGURATION

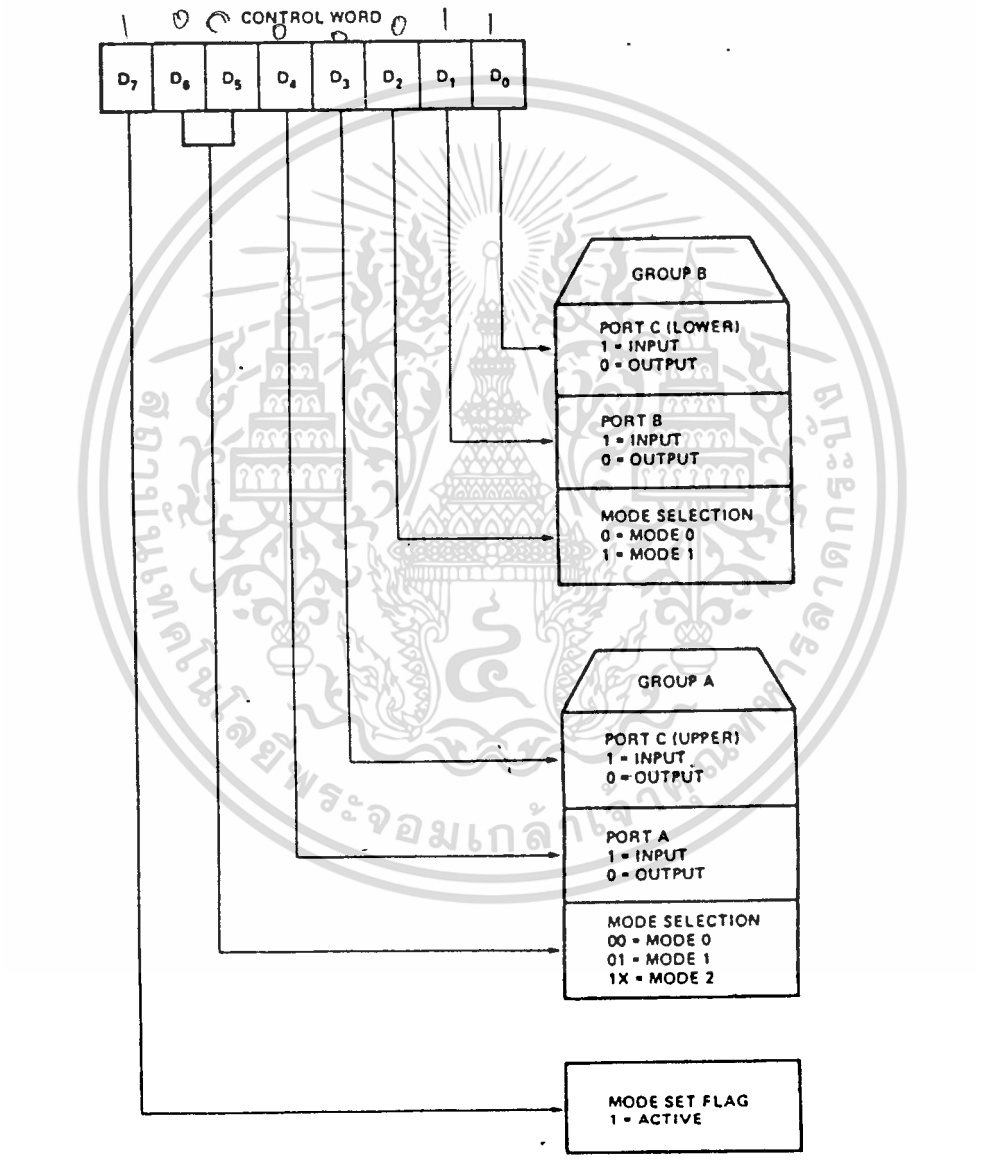


PIN NAMES

D_7-D_0	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

พอร์ทควบคุม

เป็นพอร์ทการกำหนดการทำงานของ 8255 โดยควบคุมจาก CPU โดย CPU ทำการส่งรหัสควบคุม ผ่านทาง คาต้าบัสมายังพอร์ทควบคุมของ 8255 รหัสควบคุมนี้มีขนาด 1 ไบท์ เรียกว่า Control byte และในแต่ละบิตมีความหมายเฉพาะตัวเอง ดังรูป 38



รูปที่ 38 แสดงความหมายของแต่ละบิตของรหัสควบคุม

การใช้งาน 8255

8255 แบ่งการทำงานออกเป็น 3 โหมด ดังนี้

โหมด 0 เป็นโหมดอินพุทหรือเอาท์พุทพอร์ทอย่างใดอย่างหนึ่ง ซึ่งเรียกว่า Simple I/O Port และสามารถทำงานได้ทั้ง 3 พอร์ท คือ A, B, และ C

โหมด 1 เป็นโหมดอินพุทหรือเอาท์พุทอย่างใดอย่างหนึ่ง ซึ่งทำงานแบบ handshaking ซึ่งสามารถทำงานได้เฉพาะพอร์ท A และ B การทำงานแบบ Handshaking ก็คือระหว่าง CPU กับพอร์ทและอุปกรณ์ภายนอกนั้น ขณะที่รับส่งข้อมูลกัน นอกจากจะรับข้อมูลกันแล้ว ยังต้องมีสัญญาณในการตอบรับในแต่ละครั้งของการรับส่งข้อมูล โดยผู้รับกับผู้ส่งจะต้องทำงานสัมพันธ์กันตลอดเวลา โดยพอร์ท A และ B ใช้ในการรับส่งข้อมูล ส่วนพอร์ท C จะเป็นรับและส่งสัญญาณควบคุม ดังตารางข้างล่าง

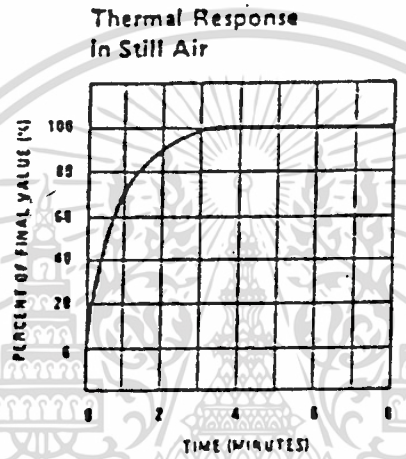
	OUT	IN
PC0	$INTR_B$	$INTR_B$
PC1	IBF_B	OBF_B
PC2	STB_B	ACK_B
PC3	$INTR_A$	$INTR_A$
PC4	STB_A	I/O
PC5	IBF_A	I/O
PC6	I/O	ACK_A
PC7	I/O	OBF_A

โหมด 2 ในโหมด 2 นี้ เป็นการรับส่งข้อมูลแบบ Handshaking เช่นกัน โดยในโหมดนี้ จะใช้พอร์ท A เท่านั้นในการใช้งาน และพอร์ท A ในโหมดนี้ เป็นได้ทั้งอินพุทและเอาท์พุทพอร์ท (Bidirectional) และใช้พอร์ท C เป็นพอร์ทในการส่งสัญญาณควบคุม ดังตารางข้างล่าง การใช้งานในโหมดนี้ ซึ่งใช้เพียงพอร์ท A ขณะเดียวกันนี้ สามารถใช้พอร์ท B ในการทำงานโหมด 0 และ 1 ได้

PORT C LINE	DEFINITION
PC0	I/O
PC1	I/O
PC2	I/O
PC3	$INTR_A$
PC4	STB_A
PC5	IBF_A
PC6	ACK_A
PC7	OBF_A

การตรวจจับอุณหภูมิและแปลงเป็นข้อมูลดิจิตอล

ในการตรวจจับอุณหภูมิ ซึ่งได้ใช้ LM 335 ในการตรวจจับอุณหภูมิโดยคุณสมบัติของ LM 335 นั้น แรงดันจะแปรผันตรงกับอุณหภูมิ ดังตารางกราฟของ LM 335 ซึ่งตารางนี้ได้จากการทดสอบของบริษัทผู้ผลิต



รูปที่ 39

และค่าแรงดันของ LM 335 นี้จะแปรผันตรงกับค่าอุณหภูมิในหน่วย เคลวิน K ตามสมการที่ 1

$$V_{\equiv} = \frac{10mV}{K} T \qquad \ldots\ldots\ldots (1)$$

เพื่อสะดวกในการอ่านค่าอุณหภูมิจึงทำการปรับปรุงวงจรให้แรงดันเอาท์พุทแปรผันตรงตามอุณหภูมิในหน่วยของศาเซลเซียส $^{\circ}C$

ที่ $0\ ^{\circ}C = 273\ K$

จากสมการที่ 1 เมื่อ T=273 K

$$V_{\equiv} = 2.73\ V$$

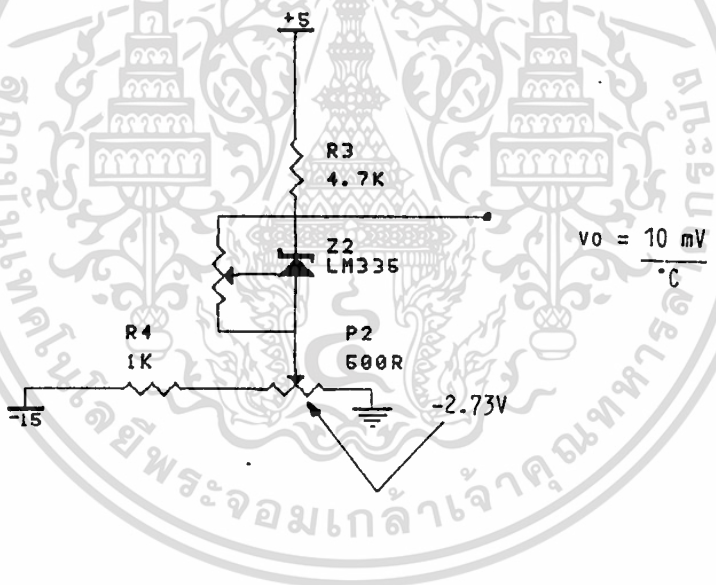
ดังนั้นจึงได้สมการที่ 2 ที่ค่าแรงดันที่เปลี่ยนแปลงตามอุณหภูมิในหน่วยองศาเซลเซียส $^{\circ}_{C}$

$$V_{=} = 2.73 + \frac{10mV}{^{\circ}C} T \quad \text{..... (2)}$$

เช่นเมื่อ $T = 0\ ^{\circ}C$ (273 K)

ก็คือ 2.73 V

ดังนั้นนำสมการที่ 2 มาปรับปรุงวงจรให้แรงดันเอาท์พุทแปรผันตรงกับอุณหภูมิในหน่วยองศาเซลเซียส โดยสร้างแรงดัน offset ที่มีค่า -2.73 V เพื่อให้ค่าแรงดันเอาท์พุทแปรผันตรงกับอุณหภูมิในหน่วยองศาเซลเซียส



วงจรรูปที่ 40

จากรูปจะพบว่าแรงดันเอาท์พุทมีค่า

$$Vout = V_{=} + Voffset$$

โดย $Voffset = -2.73$ V

ดังนั้น $Vout = 2.73 + \frac{10mV}{^{\circ}C} T - 2.73$

$$Vout = \frac{10\ mV\ T}{^{\circ}C} \qquad \ldots\ldots\ldots (3)$$

จากสมการที่ 3 แรงดันเอาท์พุทจะแปรผันตรงกับอุณหภูมิในหน่วยองศาเซลเซียส

ตัวอย่าง 1 เมื่ออุณหภูมิ 30 $^{\circ}C$

$$Vout = V_{z} + Voffset$$

$$V_{z} = 2.73 + \frac{10m\ T}{^{\circ}C}$$

เมื่อ $T=30\ ^{\circ}C$

$$Vout = 2.73 + \frac{10m \times 30\ ^{\circ}C}{^{\circ}C} - 2.73$$

$$Volt = 300\ mV$$

ตัวอย่างที่ 2 เมื่ออุณหภูมิ 1.1 $^{\circ}C$

$$Vout = \frac{10m \times 1.1\ ^{\circ}C}{^{\circ}C}$$

$$= 11\ mV$$

การขยายความละเอียดของค่าอุณหภูมิ

จากตารางกราฟของ LM 335 พบว่าแรงดันกับอุณหภูมิเป็นเชิงเส้นกับแปรผันตรงกับและจากสมการ (3) เมื่ออุณหภูมิมีค่าเป็นทศนิยมทำให้แรงดันก็เปลี่ยนแปลงเช่นกัน

เช่น 30.5 C°

$$V_{out} = \frac{10\ mV}{C^\circ} T$$

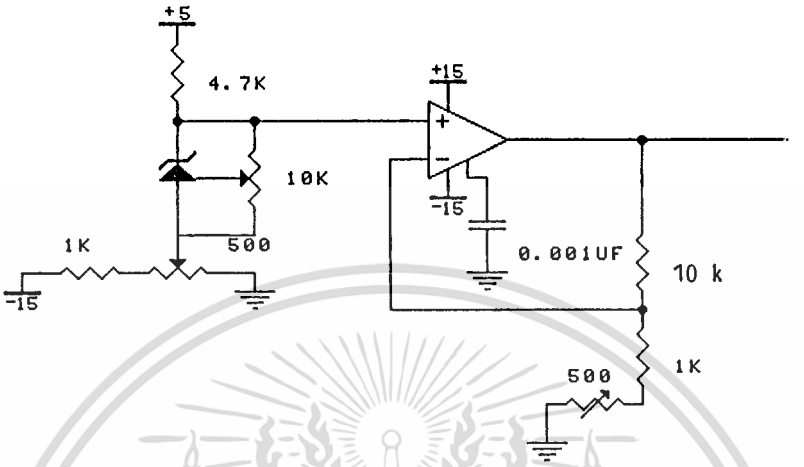
$$= \frac{10\ mV}{C^\circ} \times 30.5\ C^\circ$$

$$= 305\ mV$$

ซึ่งแรงดันจาก LM 335 ซึ่งเป็น Analog ต้องทำการเปลี่ยนแปลงเป็นข้อมูล Digital โดยผ่าน ADC 0808 โดย ADC 0808 จะทำการแปลงข้อมูล 1 Step เมื่อ Analog Input เปลี่ยนแปลงไป 20 mV

เช่น	Analog I/p	Digital O/p
	20 mV	00000001 B
	40 mV	000000010 B

ดังนั้น Analog I/p ซึ่งมาจาก LM 335 ที่ต้องการแปลงเป็น Digital โดยพบ ADC 0808 นั้น เมื่อต้องการความละเอียดของข้อมูล 1 ตำแหน่งนั้น ทำโดยนำ Analog I/p ทำการขยาย 10 เท่าเพื่อเปลี่ยนแปลงเป็นข้อมูล Digital โดยผ่านวงจร op-Amp Noninverting ดังรูปที่ 41



รูปที่ 41

ดังนั้นแรงดันจาก LM 335 ที่แปรผันตรงกับอุณหภูมิในหน่วยของสาเซลเซียส เมื่อผ่านวงจรขยาย op-Amp Noninverting โดยขยาย 10 เท่าซึ่งเป็นแรงดัน Analog ที่พร้อมแปลงเป็นข้อมูล Digital โดยผ่าน ADC 0808 ซึ่งแรงดันที่เป็น I/p ของ ADC 0808 เป็นแรงดันของค่าอุณหภูมิ ซึ่งมีค่าความละเอียดโดยผ่านวงจรขยาย 10 เท่านั้น

เมื่อแปลงข้อมูล Analog เป็น Digital ขนาด 8 bit โดยผ่าน ADC 0808 Data 8 bit เป็นค่าข้อมูลที่มีค่าของทศนิยมอยู่ 1 ตำแหน่ง ซึ่งค่าทศนิยมจะมีความละเอียดของอุณหภูมิเป็นขั้นโดยขั้นละ 0.2 C°

ในการแปลงข้อมูล Digital ให้เป็นค่าอุณหภูมิทศนิยม 1 ตำแหน่ง จะกระทำด้วยโปรแกรมของระบบ โดยประมวลผลภายใน 8051

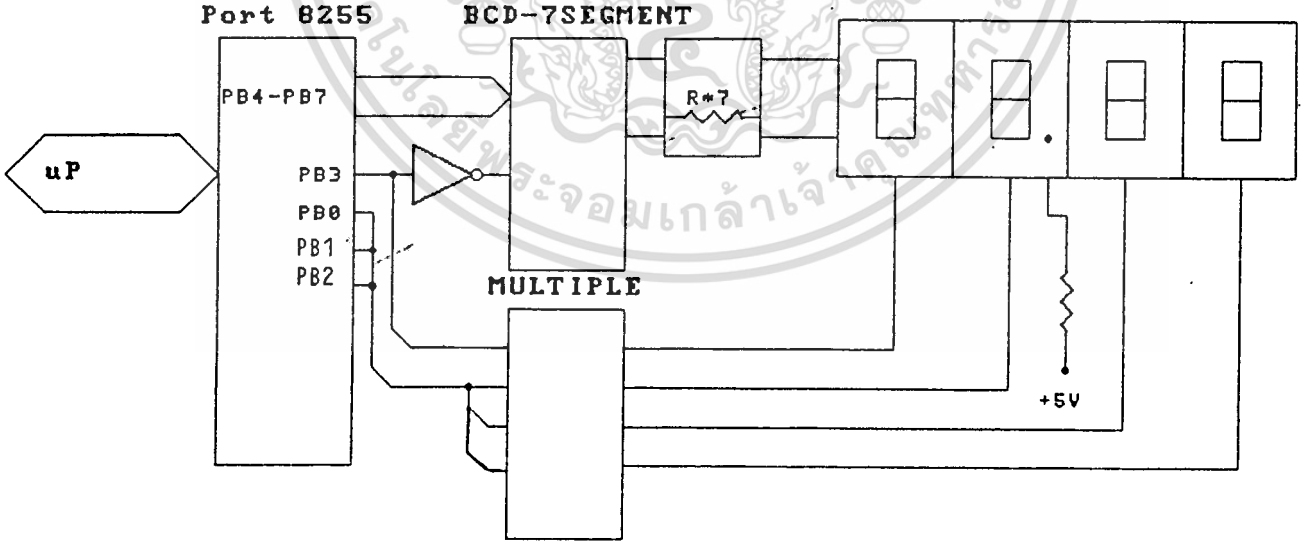
การใช้งาน พอร์ท 8255

จากที่กล่าวมาแล้วว่า Port 8255 ประกอบด้วย 3 Port คือ Port A,B และ C โดยเลือกใช้งานได้ ซึ่งควบคุมทาง Port ควบคุม ในโครงงานนี้ได้ใช้ Port 8255 ออกเป็น 2 สถานะ

สถานะแรก ใช้ Port 8255 ในการแสดงผลซึ่งเป็นค่าอุณหภูมิของ 7-Segment โดยใช้ Port B เป็น Output Port และกำหนดให้ทำงานใน Mode 0 ซึ่งแบ่งการทำงานดังนี้

PB0, PB1, PB2 เป็นขา Data ที่ส่งให้ IC 74138 เพื่อ Multiplex 7-Segment

PB4, PB5, PB6, PB7 เป็นขา Data ข้อมูลจริงที่ส่งให้ IC 74248 เพื่อแปลงจากเลข BCD เป็น 7-Segment Code



โดยในการแสดงผลนี้จะถูกควบคุมโดยขา PB3 ซึ่ง PB3 ถูกกำหนดทำงานในลักษณะ Enable IC

สถานะที่ 2 โดยการทำงานครั้งที่ 2 นี้จะใช้ Port 8255 ทำงานในการเรียกเก็บค่าอุณหภูมิที่ถูกแปลงจาก Analog เป็น Digital จาก ADC 0808 ซึ่งข้อมูลที่เรียกเก็บมีขนาด 8 bit

ในการทำงานครั้งนี้มีการกำหนดให้ Port 8255 ทำงานอยู่ 2 Mode คือ Mode 0 และ Mode 1 ดังนี้

- Mode 1 กำหนดให้ Port B ทำงานเป็น Output Port วัตถุประสงค์เพื่อใช้ Port B เป็นตัวเลือก Analog I/p ที่เข้า ADC 0808 ซึ่งทำงานตามขาดังนี้

  PB0, PB1, PB2 เป็นค่าในการ Multiplex Analog I/p ที่ ADC 0808

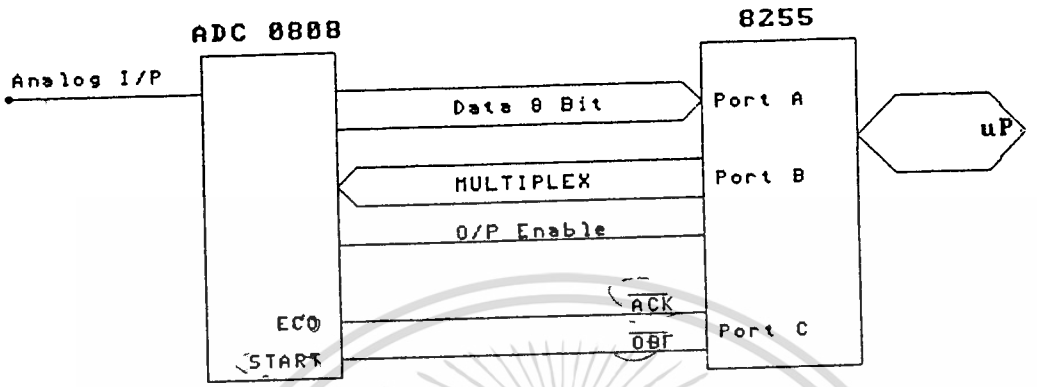
  ซึ่งการทำงานใน Mode 1 นี้เป็นการทำงานแบบ Hand Checking โดยจะทำงานพร้อมกับ Port C บน โดยมีขาที่ทำงานในการตอบรับคือ

  PC1 คือ OBF เป็นขาที่ส่งสัญญาจาก Port 8255 ไปยังอุปกรณ์ภายนอกว่าขณะนี้มีข้อมูลมารอที่อุปกรณ์ภายนอกแล้ว

  PC2 คือ ACK เป็นขาที่รอการตอบรับจากอุปกรณ์ภายนอกมายัง Port 8255 เพื่อบอกกับ Port 8255 ว่าได้รับข้อมูลที่ส่งมาให้แล้ว ซึ่งสัญญาที่ขานี้จะมีผลต่อสัญญาณที่ขา OBF ที่จะกล่าวต่อไป

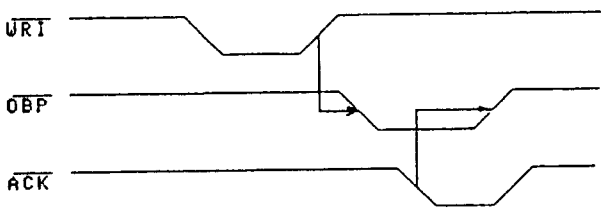
  และ PB3 เป็นขาที่ส่งสัญญาณเพื่อเป็น Output Enable แก่ ADC 0808

- Mode 0 กำหนดให้ Port A เป็น Input Port เพื่อเรียกเก็บข้อมูลจากอุปกรณ์ภายนอก



จากรูปด้านบนเป็น Block Diagram การติดต่อ Port 8255 กับ ADC 0808 เพื่อ START ADC 0808 และ MULTIDEX Analog I/p และตรวจสอบการรับข้อมูล และเก็บข้อมูล

Timing Diagram

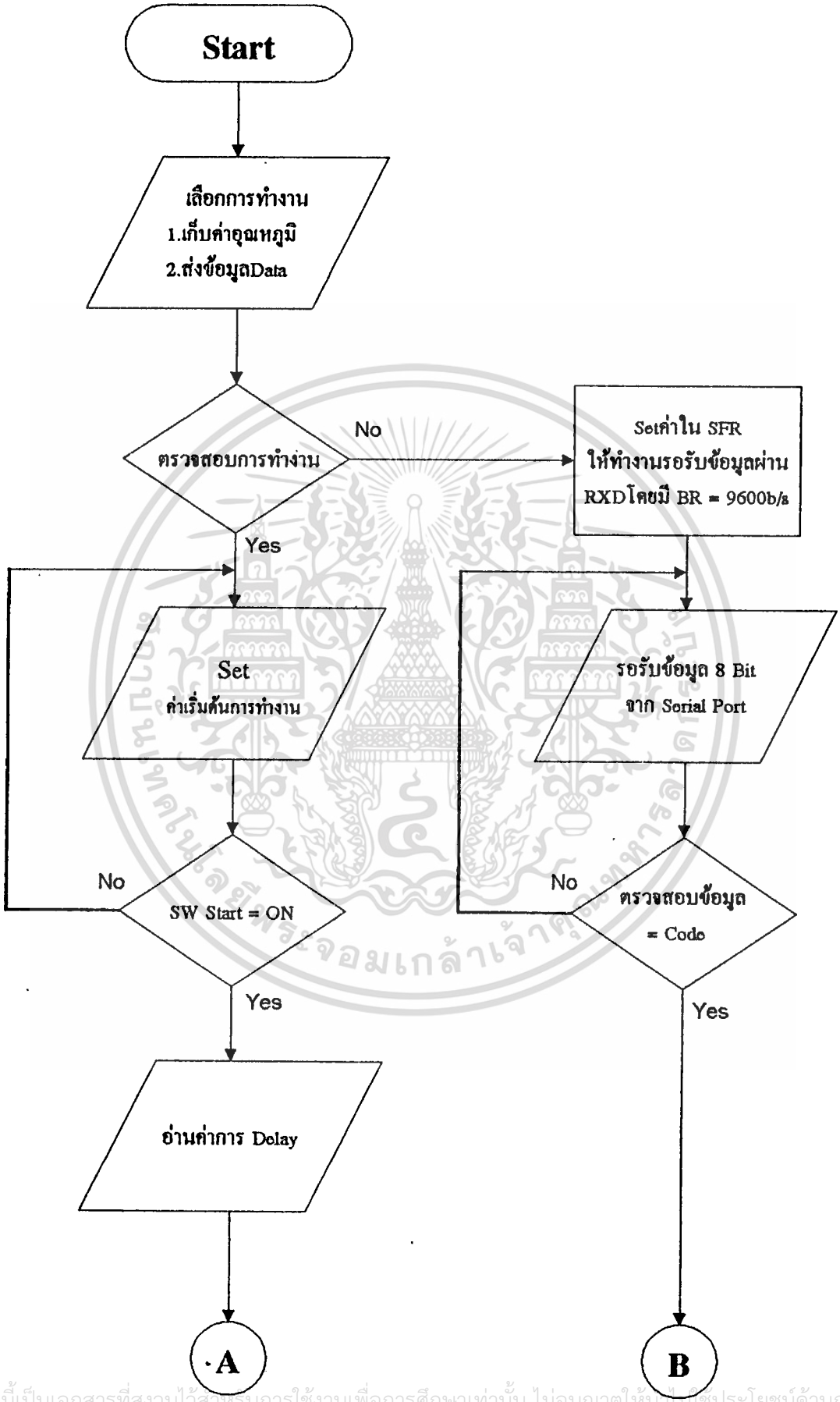


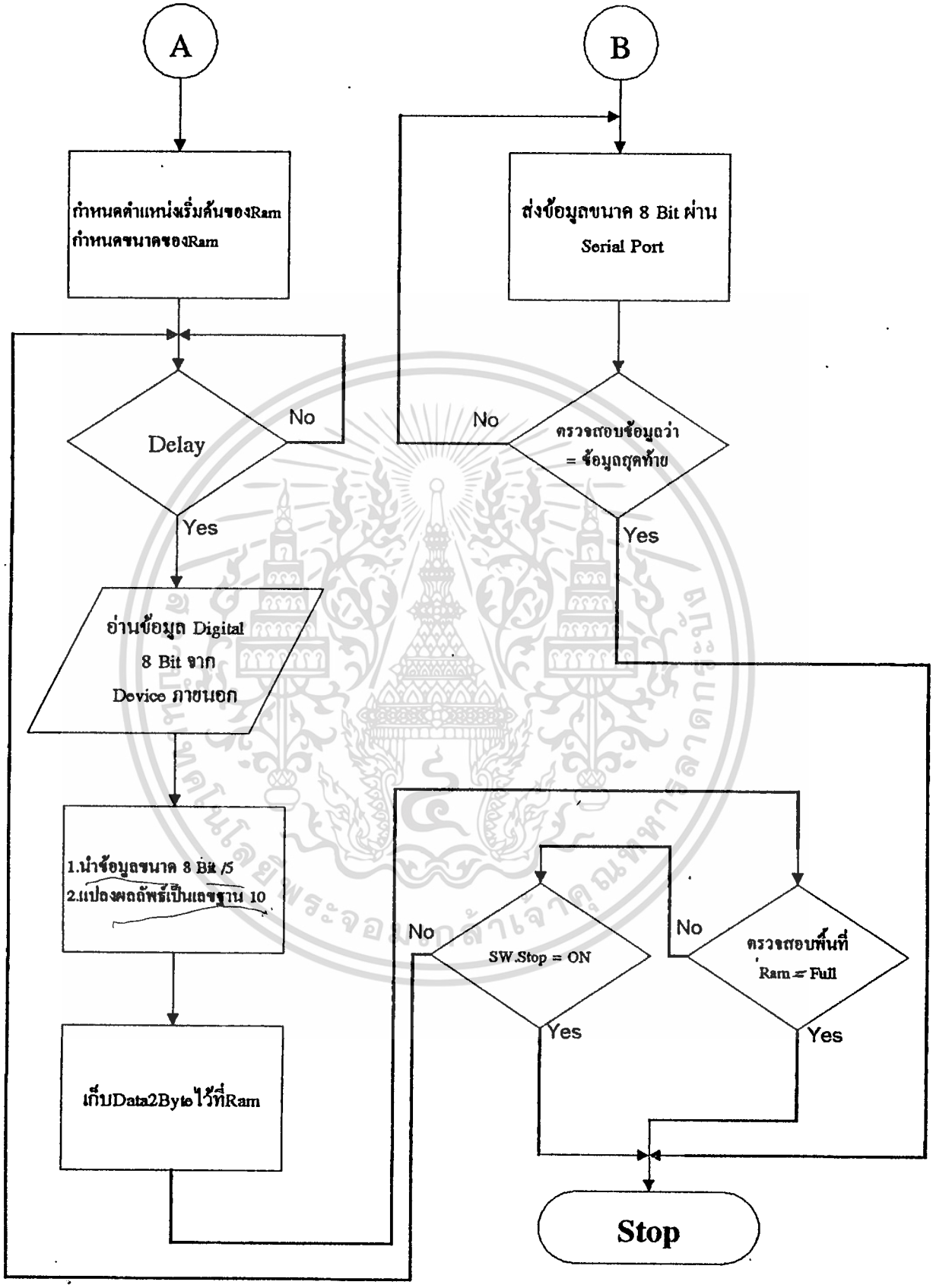
OBF (OUTPUT BOFFER FULL) ซึ่งจะต่อกับขา START แ\ะ ALE ของ ADC 0808 และ ACK (ACKNOWLEDGE) ซึ่งต่อจาก EOC โดยผ่าน INVERTER ของ ADC 0808 เมื่อมีสัญญาณ WRI จาก UP ACTIVE และระยะหนึ่งทำให้ขา OBF ACTIVE ซึ่งขณะก่อนที่ขา OBF จะ ACTIVE นั้น ข้อมูลได้ถูกส่งไปยัง ADC 0808 แล้ว ซึ่งเมื่อขา OBF ACTIVE จาก H เป็น L ซึ่งเป็นสัญญาณ ALE เพื่อ Latch ค่า Multiplex Analog I/p และพร้อมกับ START ให้ ADC 0808 เริ่มทำการแปลงข้อมูลจาก I/p ซึ่งเป็น Analog ให้เป็น Digital เมื่อ ADC 0808 ทำการแปลงสัญญาณเสร็จเรียบร้อยแล้ว ADC 0808 ก็จะสร้างสัญญาณ EOC ออกมา เมื่อนำสัญญาณนี้เป็นสัญญาณตอบรับจาก ADC 0808 ไปยัง Port 8255 แล้วก็สามารถทำให้สัญญาณ OBF กลับสู่สถานะเดิม เมื่อมีการตอบรับจากภายนอก และเราได้นำสัญญาณ OBF ไปใช้ประโยชน์ในการตรวจสอบจากไมโครคอนโทรเลอร์ว่าพร้อมที่สามารถทำงานในขั้นต่อไปได้หรือยัง

บทที่ 5

โปรแกรมควบคุมการทำงาน

เนื่องจากเครื่องวัดและบันทึกค่าอุณหภูมิแวดล้อมได้ใช้ Microcontroller เบอร์ 8051 ควบคุมการทำงานบนเครื่อง เพราะฉะนั้นการเขียนโปรแกรม (SOFTWARE) ให้เครื่องทำงานจึงใช้คำสั่งของ MCS 51 (รายละเอียดของคำสั่ง MCS 51 ได้รวบรวมไว้พร้อมกับข้อมูลของไมโครคอนโทรเลอร์ ท้ายบท) สำหรับโปรกแกรมควบคุมการทำงานและโฟลวชาร์ต (Flowchart) ได้แสดงไว้ต่อไปนี้





```
        CPU  "8051.TBL"
        HOF  "INT8"
;*******************************

        ORG 0000H
P0:     EQU  80H
SP:     EQU  81H
DPTR:   EQU  82H
DPL:    EQU  82H
DPH:    EQU  83H
P1:     EQU  90H
P2:     EQU  0A0H
P3:     EQU  0B0H
PSW:    EQU  0D0H
ACC:    EQU  0E0H
B:      EQU  0F0H
R0:     EQU  00H
R1:     EQU  01H
R2:     EQU  02H
R3:     EQU  03H
R4:     EQU  04H
R5:     EQU  05H
R6:     EQU  06H
R7:     EQU  07H
PORTA:  EQU  0E000H
PORTB:  EQU  0E001H
PORTC:  EQU  0E002H
PORTCON: EQU 0E003H
TL0:    EQU  8AH
TH0:    EQU  8CH
TL1:    EQU  8BH
TH1:    EQU  8DH
TMOD:   EQU  89H
TCON:   EQU  88H
SCON:   EQU  98H
SBUF:   EQU  99H
PCON:   EQU  87H
```

```
IE:     EQU   0A8H
IP:     EQU   0B8H
TR1:    EQU   8EH
TI:     EQU   99H
RI:     EQU   98H
RB8:    EQU   9AH
REN:    EQU   9CH

;******* USE P1 CHECK SW.*******
;P1.0 CONNECT  SW.0  <DELAY 1 MINUTE>
;P1.1   ,,     SW.1  < ,,   2  ,,  >
;P1.2   ,,     SW.2  < ,,   3  ,,  >
;P1.3   ,,     SW.3  <  START SW.  >
;P1.4   ,,     SW.4  <  STOP  SW.  >
;P1.5   ,,     SW.5  <  SET HOUR   >
;P1.6   ,,     SW.6  <  SET MINUET >
;P1.7   ,,     SW.7  <  "ON" SERIAL TRANSFER DATA>

          MOV PSW,#00H
          MOV R1,#00H
      DD1: MOV R0,#00H
      DD:  DJNZ R0,DD
           DJNZ R1,DD1
           MOV 20H,P1
           JB 7,ST
           LJMP RX          ; <RECEIVE DATA FORM SERIAL PORT>

;**** SET TIME ****

       ST: MOV DPTR,#PORTCON
           MOV A,#0F0H
           MOVX @DPTR,A
           MOV 49H,#00H
           MOV 4AH,#00H

       Y4: MOV R0,#0FH
```

```
Y3:  MOV 20H,P1
     JB 5,Y1
     SJMP X1

Y1:  JB 6,Y2
     SJMP Y5

Y2:  DJNZ R0,Y3
     SJMP M1

Y5:  INC 49H
     MOV A,49H
     ANL A,#0FH
     CJNE A,#0AH,B1
     MOV A,49H
     ADD A,#06H
     SJMP X2

B1:  MOV A,49H

X2:  CJNE A,#60H,B2
     MOV 49H,#00H
     SJMP M1

B2:  MOV 49H,A
     SJMP M1

X1:  INC 4AH
     MOV A,4AH
     ANL A,#0FH
     CJNE A,#0AH,B3
     MOV A,4AH
     ADD A,#06H
     SJMP X3

B3:  MOV A,4AH
```

```
X3:  CJNE A,#24H,B4
     MOV 4AH,#00H
     SJMP M1

B4:  MOV 4AH,A
     SJMP M1

;*****DISPLAY****

M1:  MOV R1,#30H

M2:  MOV DPTR,#PORTB
     MOV A,49H
     SWAP A
     ANL A,#0F0H
     ORL A,#04H
     MOV R0,#0FFH

JJ1: MOVX @DPTR,A
     DJNZ R0,JJ1
     MOV A,49H
     ANL A,#0F0H
     ORL A,#03H
     MOV R0,#0FFH

JJ2: MOVX @DPTR,A
     DJNZ R0,JJ2
     MOV A,4AH
     SWAP A
     ANL A,#0F0H
     ORL A,#02H
     MOV R0,#0FFH

JJ3: MOVX @DPTR,A
     DJNZ R0,JJ3
     MOV A,4AH
     ANL A,#0F0H
```

```
                ORL A,#01H
                MOV R0,#0FFH

          JJ4:  MOVX @DPTR,A
                DJNZ R0,JJ4
                DJNZ R1,M2

;************************
;   CHECK START SW.
;************************
                 MOV R0,#0FH
         START:  MOV 20H,P1
                 JNB 3,S0
                 DJNZ R0,START
                 LJMP Y4

;*****************************
;   PROGRAM CHECK DIP SW.
;*****************************
          S0:   MOV 20H,P1
                JNB  0,S1                ;TEST SW1
                JNB  1,S2
                JNB  2,S3
                SJMP S0

;****************************
;   PROGRAM SET TIME
;****************************
          S1:   MOV R5,#1CH
                MOV 4BH,#01H
                SJMP  RAM

          S2:   MOV R5,#3AH
                MOV 4BH,#02H
                SJMP  RAM
```

```
          S3:   MOV R5,#59H
                MOV 4BH,#03H
                SJMP  RAM

;*****SAVE TIME*****
        RAM:    MOV DPL,#00H
                MOV DPH,#20H
                MOV A,4AH
                MOVX @DPTR,A
                INC DPL
                MOV A,49H
                MOVX @DPTR,A
                MOV A,4BH                   ;DATA DELAY
                INC DPL
                MOVX @DPTR,A

;*****************************************
;   SET STACK AND ADD. DATA RAM OF TEM.
;*****************************************
                MOV SP,#07H
                MOV 43H,#02H        ;INITION ADDRESS OF RAM
                MOV 44H,#20H
                PUSH 43H
                PUSH 44H

;************************
;   PROGRAME CHECK RAM
;************************
                MOV 45H,#0F0H       ;AREA DATA RAM FOR SAVE
                MOV 46H,#07H
        LOOP:   MOV DPL,45H
                MOV DPH,46H
                MOV R0,DPH
                CJNE R0,#00H,CHECK1
                MOV R0,DPL
                CJNE R0,#00H,SUB3
                LJMP STOP
```

```
CHECK1:   MOV R0,DPL
          CJNE R0,#00H,SUB1
          DEC DPH

  SUB1:   DEC DPL
          MOV R0,DPL
          CJNE R0,#00H,SUB2
          DEC DPH

  SUB2:   DEC DPL

  SAVE:   MOV 45H,DPL
          MOV 46H,DPH
          PUSH 45H
          PUSH 46H
          SJMP DELAY

  SUB3:   DEC DPL
          MOV R0,DPL
          CJNE R0,#00H,SUB2
          LJMP STOP

;********************
;   PROGRAME DELAY
;********************
  DELAY:  MOV R7,05H
     L1:  DEC R7
          MOV A,R7
          JNZ DIS1
          LJMP SAVEDATA

;*****CHECK STOP SW. *****
   DIS1:  MOV 20H,P1
          JB 4,V1
          LJMP STOP
```

```
;*************************************************
;   PROGRAME SELEC DEVICE   AND READ DATA OF TEM.
;*************************************************
          V1:  MOV DPTR,#PORTCON
               MOV A,#94H
               MOVX @DPTR,A          ;SET CONTROL WORD
               MOV A,#00H
               MOV DPTR,#PORTB
               MOVX @DPTR,A          ;SELECE DEVICE
               MOVX @DPTR,A

     TEST1:    MOV DPTR,#PORTC
               MOVX A,@DPTR
               MOV 20H,A
               JNB 1,TEST1           ;TEST BIT /OBF OF 8255
               MOV R0,#05H
               MOV A,#08H
               MOV DPTR,#PORTB

        OE:    MOVX @DPTR,A          ;SEND "1" TO OE<o/p enabel>
               DJNZ R0,OE
               MOV DPTR,#PORTA       ;READ DATA OF TEM.
               MOVX A,@DPTR
               MOV B,#05H
               DIV AB
               LJMP BCD

           NEXT:  MOV 47H,A
                  MOV A,B
                  RL A
                  SWAP A
                  MOV 48H,A

;SET HIGH Impedance DATA O/P OF ADC
                  MOV DPTR,#PORTB
                  MOV A,#00H
                  MOVX @DPTR,A
```

```
;****CHECK STOP SW.******
                MOV 20H,P1
                JB 4,V2
                LJMP STOP

;*********************************************
;   FOR SHOW DATA TEM. AT DISPLAY 7-SEGMENT
;*********************************************
;SET PORT 8255
;PORT A,B AND C IS O/P PORT

          V2:   MOV DPTR,#PORTCON
                MOV A,#0F0H             ;set control wowd
                MOVX @DPTR,A
                MOV R2,#1CH             ;loop 2 sec for display

         SO1:   MOV R3,#0FFH
         SOO:   MOV A,48H
                MOV R6,A
                SWAP A
                ANL A,#00H
                ORL A,#0A4H             ;SET DISPLAY 0
                MOV DPTR,#PORTB
                MOV R0,#0EH

       SHOW1:   MOVX @DPTR,A
                DJNZ R0,SHOW1
                MOV A,R6
                ANL A,#0F0H
                ORL A,#03H              ;SET DISPLAY 1
                MOV R0,#0FH

       SHOW2:   MOVX @DPTR,A
                DJNZ R0,SHOW2
                MOV A,47H
                MOV R6,A
```

```
                SWAP A
                ANL A,#0F0H
                ORL A,#02H          ;SET DISPLAY 2
                MOV DPTR,#PORTB
                MOV R0,#0EH

          SHOW3: MOVX @DPTR,A
                DJNZ R0,SHOW3
                MOV A,R6
                ANL A,#0F0H
                ORL A,#01H          ;DISPLAY 3
                MOV R0,#0EH

          SHOW4: MOVX @DPTR,A
                DJNZ R0, SHOW4

;*****CHECK STOP SW.*****
                MOV 20H,P1
                JB 4,V3
                LJMP STOP

            V3: DJNZ R3,SOO
                DJNZ R2,SO1
                LJMP L1
;*******************************
; PROGRAME  SAVE DATATO RAM
;*******************************
   SAVEDATA:    POP 46H                ;DATA OF RAM
                POP 45H
                POP 44H
                POP 43H
                MOV DPL,43H            ;SET @DPL=00
                MOV DPH,44H            ;SET @DPH=23H
                MOV R0,DPL
                CJNE R0,#0FFH,ATT1
                INC DPH
```

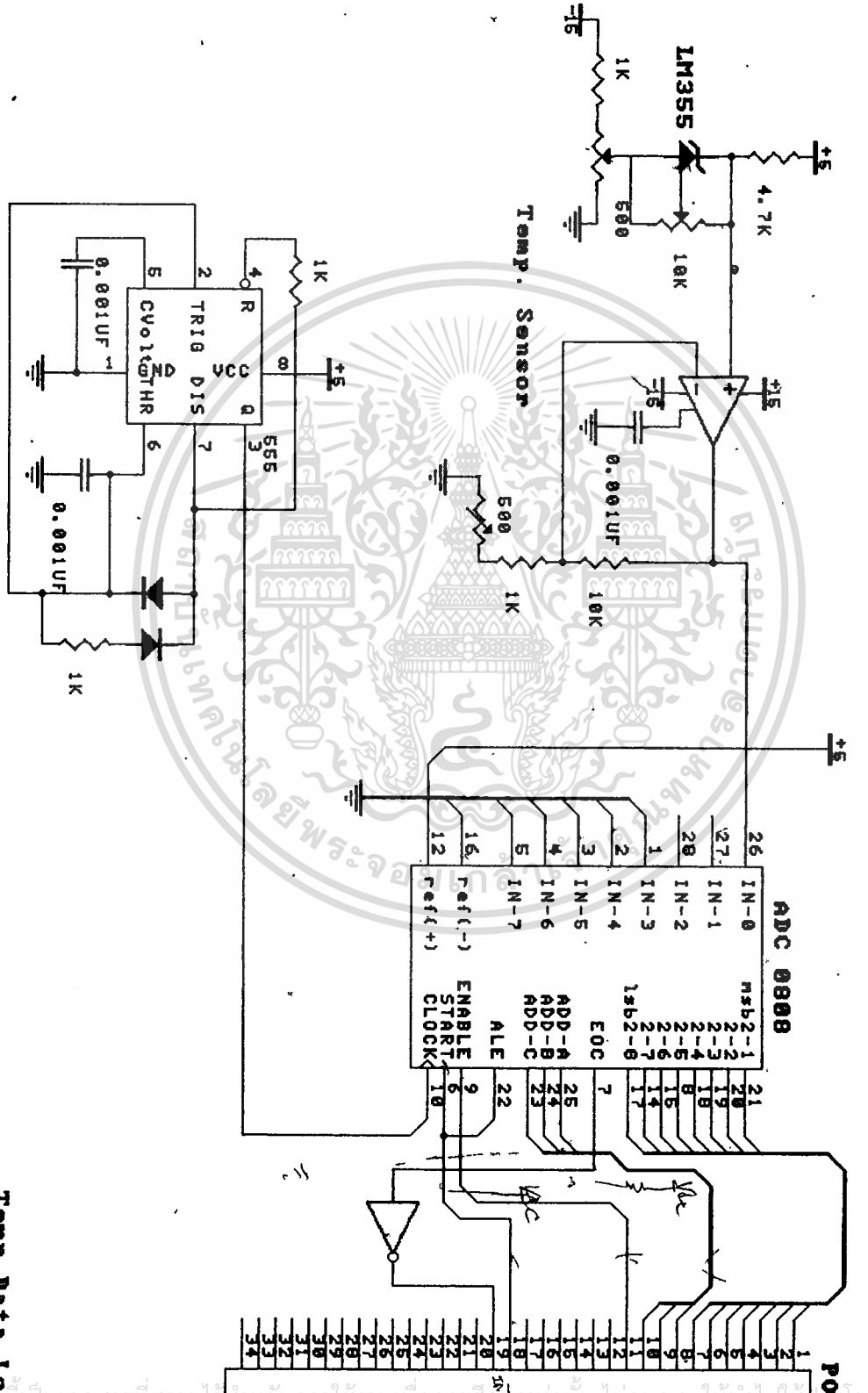
```
ATT1: INC DPL
      MOV A,47H
      MOVX @DPTR,A
      MOV R0,DPL
      CJNE R0,#0FFH,ATT2
      INC DPH

ATT2: INC DPL
      MOV A,48H
      MOVX @DPTR,A
      MOV 43H,DPL
      MOV 44H,DPH
      PUSH 43H
      PUSH 44H

      ;FINISHE SAVE DATA OF TEM.

      LJMP LOOP          ;REPEAT  UNTILL FULL RAM

;******************
;  BINARY TO BCD
;******************
BCD:  MOV R1,A
      ANL A,#0F0H
      CJNE A,#00H,W1
      MOV A,R1
      ANL A,#0FH
      MOV 21H,#0AH
      MOV R0,#06H

LP1:  CJNE A,21H,W2
      MOV A,R1
      ADD A,#06H
      SJMP RUN
```

```
W2:   INC 21H
      DJNZ R0,LP1
      SJMP RUN

W1:   CJNE A,#10H,W3
      SJMP WE1

W3:   CJNE A,#20H,W4
      SJMP WE2

W4:   CJNE A,#30H, RUN
      SJMP WE3

WE1:  MOV 21H,#0AH
      MOV R0,#06H
      MOV A,R1
      ANL A,#0FH

LP2:  CJNE A,21H,G1
      MOV A,R1
      ADD A,#06H
      SJMP CK1

G1:   INC 21H
      DJNZ R0,LP2
      MOV A,R1

CK1:  ADD A,#06H
      MOV 21H,#0AH
      MOV R0,#06H
      MOV 4AH,A
      ANL A,#0FH

LP3:  CJNE A,21H,G2
      MOV A,4AH
      ADD A,#06H
      SJMP RUN
```

```
G2:   INC 21H
      DJNZ R0,LP3
      MOV A,4AH
      SJMP RUN

WE2:  MOV 21H,#0EH
      MOV A,R1
      ANL A,#0FH
      MOV R0,#02H

LP4:  CJNE A,21H,CK2

WE3:  MOV A,R1
      ADD A,#06H
      SJMP G3

CK2:  INC 21H
      DJNZ R0,LP4
      MOV A,R1

G3:   ADD A,#06H
      ADD A,#06H
      SJMP CK1

RUN:  LJMP NEXT

;************************************
STOP: MOV DPL,43H
      MOV DPH,44H
      MOV R0,DPL
      CJNE R0,#0FFH,ATT3
      INC DPH

ATT3: INC DPL
      MOV A,#0FEH
      MOVX @DPTR,A
      MOV R0,DPL
```

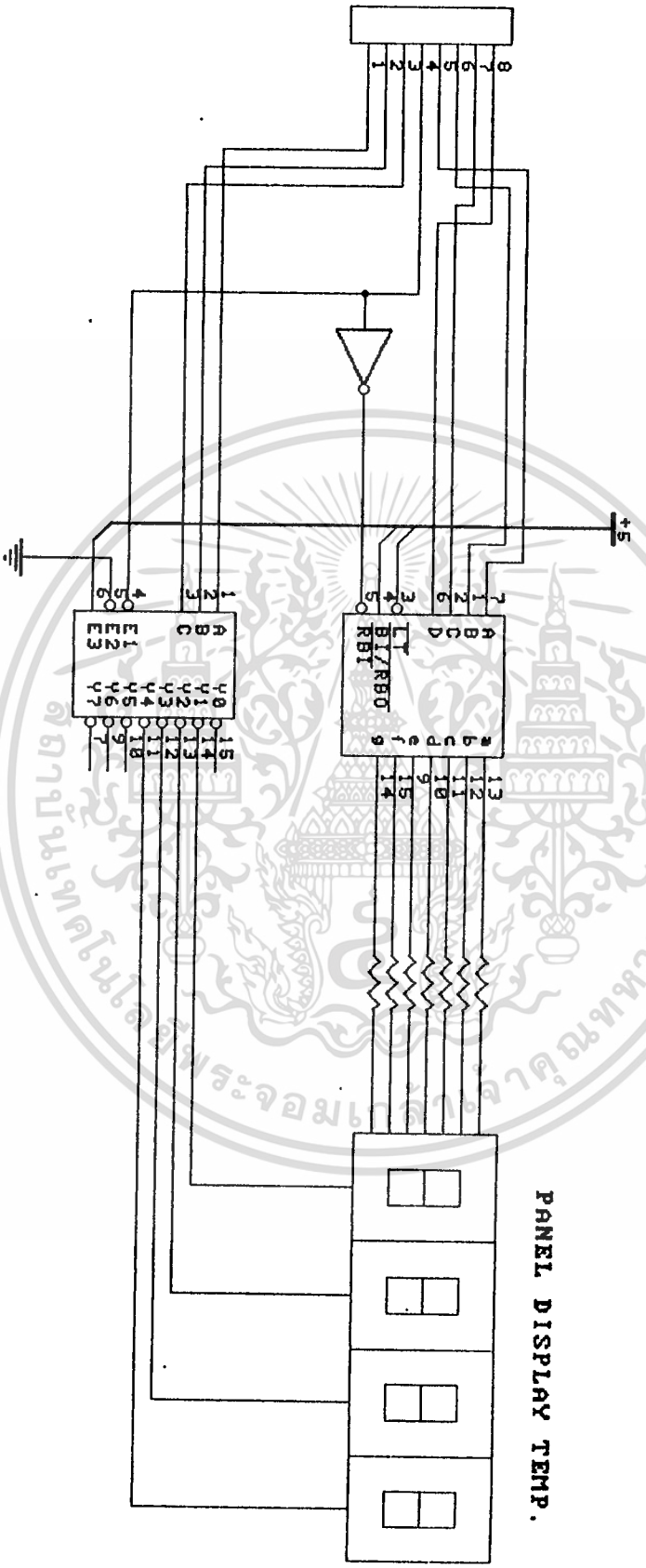
```
        CJNE R0,#0FFH,ATT4
        INC DPH

ATT4:   INC DPL
        MOV A,#0FFH
        MOVX @DPTR,A

LAST:   MOV DPTR,#PORTCON
        MOV A,#0F0H
        MOVX @DPTR,A
        MOV DPTR,#PORTB
        MOV A,#00H
        MOVX @DPTR,A
        SJMP FINISH

;*****************************
;       LINK PC
;*****************************
RX:     MOV DPTR,#PORTCON
        MOV A,#0F0H
        MOVX @DPTR,A
        MOV DPTR,#PORTB
        MOV R0,#0FH

RE1:    MOV A,#52H
        MOVX @DPTR,A
        DJNZ R0,RE1
        MOV TMOD,#20H       ; TIMER 1 MODE 2
        MOV SCON,#50H       ; SERIAL PORT MODE 1
        MOV PCON,#00H
        MOV IE,#00H
        MOV TH1,#0FDH       ;SET BAUD RATE 9600b/s
        SETB TR1

RX1:    JNB RI,RX1
        CLR RI
        MOV A,SBUF
```

```
                CJNE A,#0AH,RX1
                CLR REN

;****** TRANSFER DATA ****

                MOV DPL,#00H
                MOV DPH,#20H

          TX1:  MOVX A,@DPTR
                CJNE A,#0FFH,TX2
                LJMP LAST

          TX2:  MOV SBUF,A

          TX3:  JNB TI,TX3
                CLR TI
                MOV R0,DPL
                CJNE R0,#0FFH,TX4
                INC DPH

          TX4:  INC DPL
                SJMP TX1
       FINISH:  END
```



Temp Data Logger

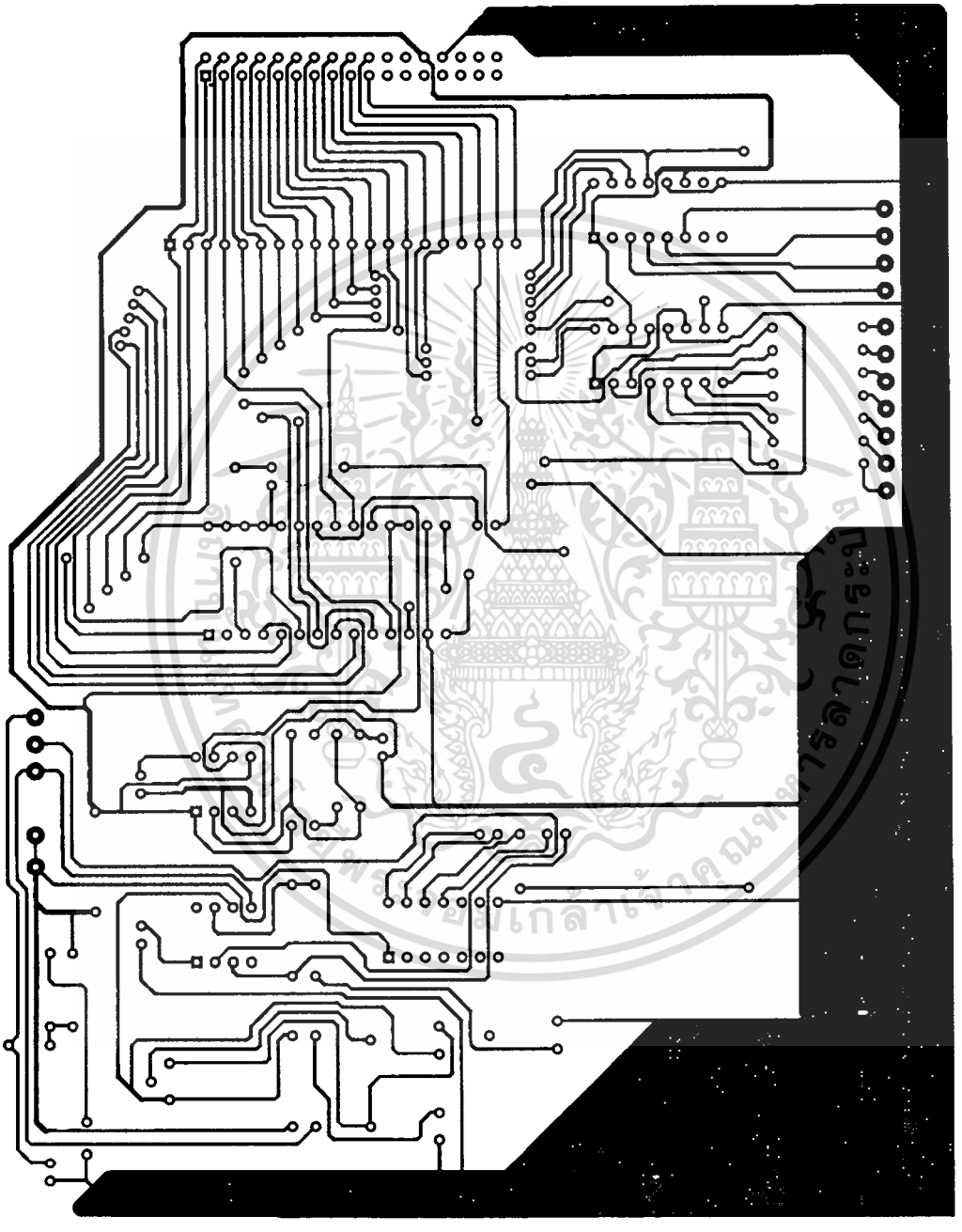
PAGE 1.



Temp Data Logge

PAGE 4

ลายวงจร

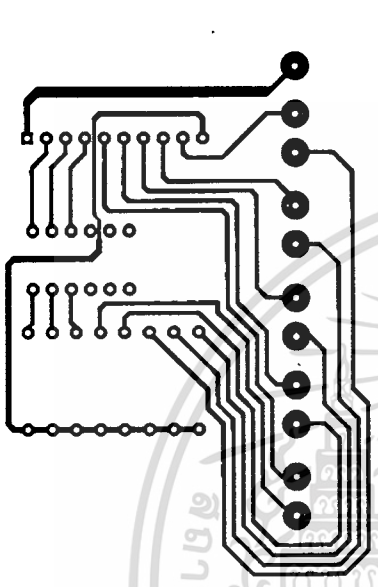


รูป ก. ลายวงจร ADC, CLOCK และ DISPLAY

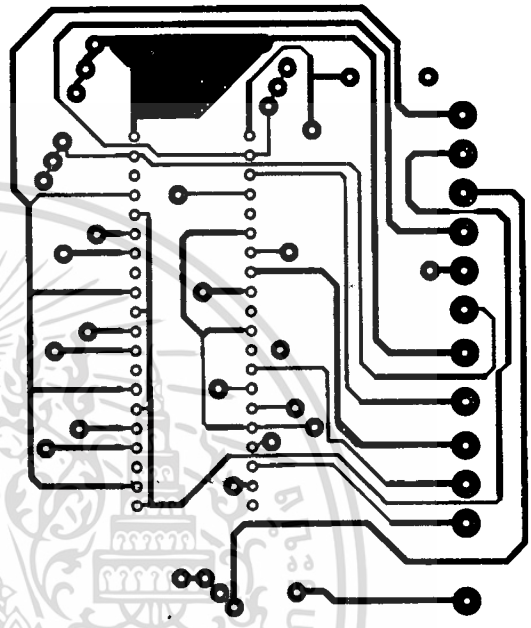
รูป ข. SWITCH CONTROL และ DIP SWITCH

รูป ค. 7- SEGMENT

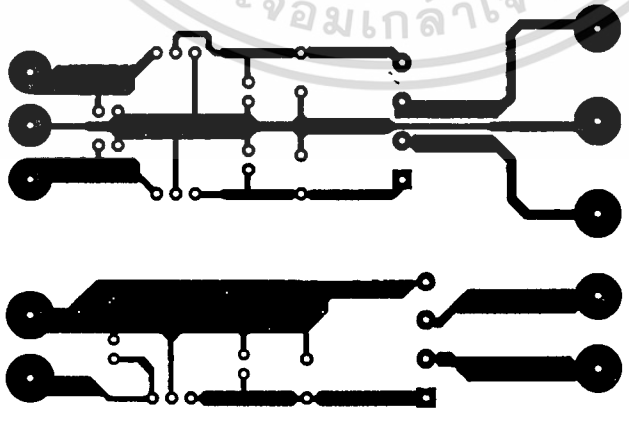
รูป ง. POWER SUPPLY ± 15V และ 5V



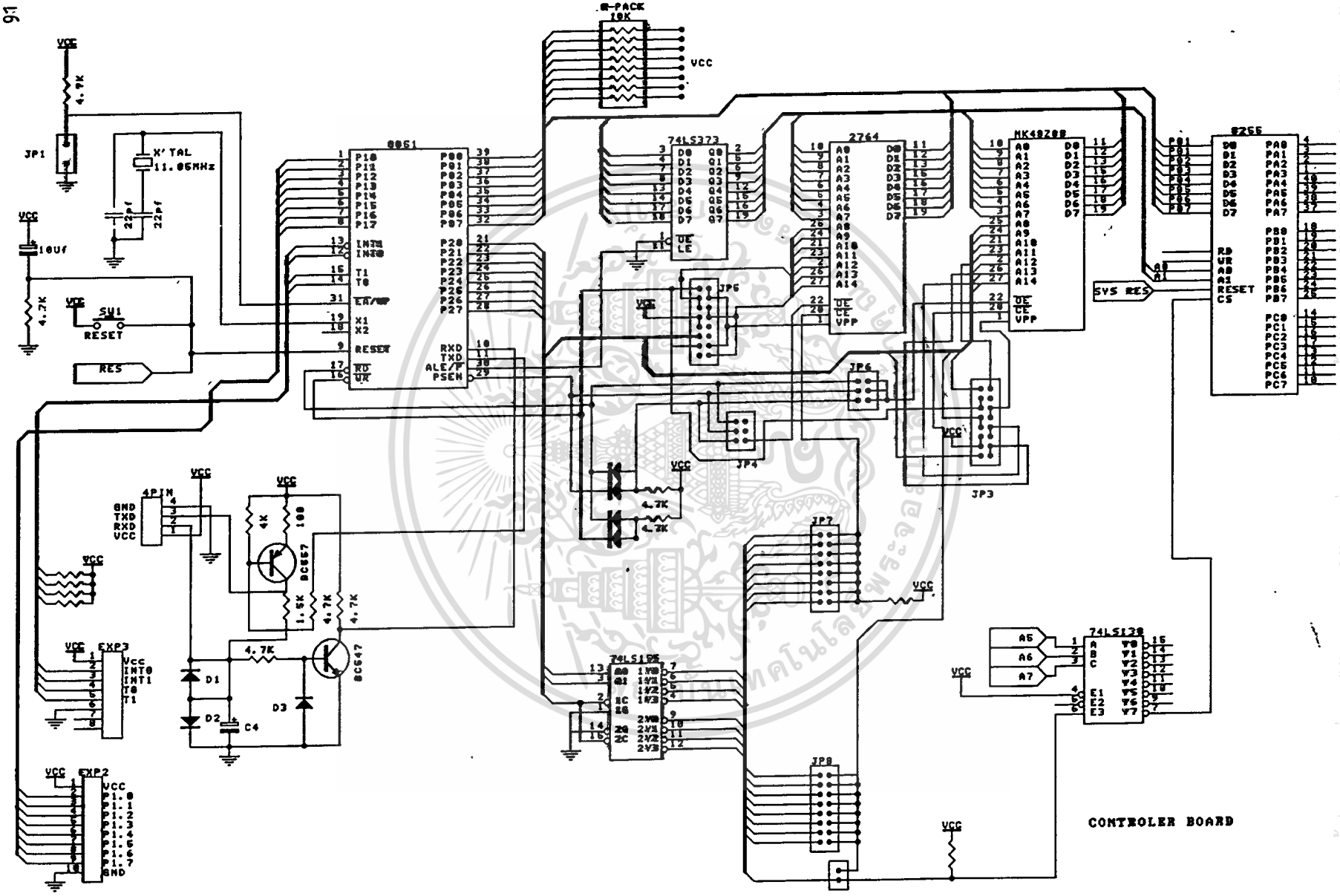
รูป ข.



รูป ค.



รูป ง.



ภาคผนวก

ขั้นตอนการใช้งานเครื่องบันทึกค่าอุณหภูมิ

1) เลือกการทำงาน

1. ต้องการบันทึกค่าอุณหภูมิ ให้ "ON" SW._ SEND
2. ต้องการส่งข้อมูล (ค่าอุณหภูมิ) เพื่อแสดงผลบน Computer "OFF" SW _ SEND

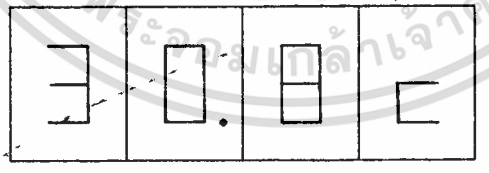
เมื่อเลือกการทำงานที่ 1

- "ON" SW.Power ที่ Display จะแสดงค่า 00.00
- Reset ระบบ
- ใส่ค่าเวลาเริ่มต้นการทำงาน โดย

  SW.HOUR ใส่ค่าเวลาเป็น ชั่วโมง

  SW MINUTE ใส่ค่าเวลาเป็น นาที
- กด Switch Start

  เมื่อกด SW.Start แล้วที่ Display จะแสดงค่าอุณหภูมิ โดยมีหน่วย °C โดยหลักสุดท้ายของ Display จะมีตัวอักษร C ปรากฏ แสดงว่าเป็นค่าอุณหภูมิ



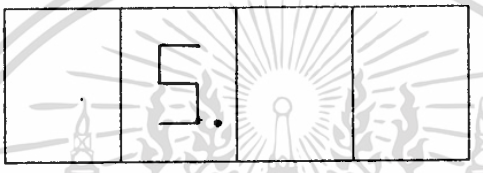
- เมื่อต้องการหยุดการทำงานของเครื่อง

  กด SW STOP เครื่องจะหยุดทำงานพร้อม Display จะดับ

เมื่อเลือกการทำงานที่ 2

เมื่อเลือกการทำงานที่ 2 นี้ หมายถึงว่า ผู้ใช้ต้องการติดต่อรับส่งข้อค่าอุณหภูมิบนเครื่องกับ Computer เพื่อแสดงค่าอุณหภูมิพร้อมเวลาบนจอ Computer โดยให้ต่อสายรับส่งข้อมูลแบบ Serial ระหว่าง Computer กับ เครื่องบันทึกค่าอุณหภูมิ

- เมื่อ "ON" SW. POWER เครื่องบันทึกค่าอุณหภูมิแล้ว ที่ Display จะแสดงตัวอักษร



ขณะนี้ หมายถึงว่าเครื่องพร้อมที่จะส่งข้อมูลแล้ว โดยรับคำสั่งจาก Computer และเมื่อได้รับคำสั่งจาก Computer ให้ส่งข้อมูลแล้ว เครื่องจะส่งข้อมูลค่าอุณหภูมิและเวลาจนหมด

เมื่อเครื่องส่งข้อมูลจนหมดแล้ว จะเห็นได้ว่า ที่ Display จะดับ นั้นหมายถึงว่า ข้อมูลถูกส่งไปยัง PC เรียบร้อยแล้ว

สรุปผลการทำงาน

เครื่อง Temperature Data Logger ที่ประกอบขึ้นนี้สามารถวัดและบันทึกค่าอุณหภูมิในหน่วยของศาเซลเซียส โดยความละเอียดทศนิยม 1 ตำแหน่งโดยการทำงานทั้งหมดจะถูกควบคุมโดย ไมโครคอนโทรเลอร์เบอร์ 8031 ซึ่งในเครื่อง Temperature Data Logger นี้ได้ใช้ Board Controler Sp-31 ของบริษัท ETT.

สำหรับ Sensor ที่ใช้ตรวจวัดค่าอุณหภูมิคือ LM 335 ซึ่งคุณสมบัติของ LM 335 นี้ค่าแรงดัน Output จะแปรผันกับค่าอุณหภูมิ องศาฟาเรนไฮท์ โดยแรงดันนี้จะถูกแปลงให้เป็นแรงดันในหน่วยของศาเซลเซียสพร้อมทั้งขยายค่าความละเอียดของอุณหภูมิ

สำหรับ Analog To Digital ได้ใช้ IC 0808 ที่มี Output 8 Bit และ Input Analog มีถึง 8 Input การทำงานของ ADC 0808 และ Display 7 Segment จะถูกควบคุมโดย Microcontroler 8031 จึงจะควบคุมผ่าน Port 8255

ค่าของอุณหภูมิที่ถูกบันทึกนั้นจะถูกเก็บไว้ที่ RAM ที่มี Back up Battery ภายในตัว การส่งข้อมูลระหว่าง PC กับเครื่อง Temperature Data Logger เพื่อแสดงค่าของเวลาและอุณหภูมินั้นจะเป็นการส่งข้อมูลแบบอนุกรม

การแสดงข้อมูลบน PC

การแสดงข้อมูลที่ถูกบันทึกไว้ทั้งหมดในเครื่อง Temperature Data Logger บนจอ Computor นั้น โดยเครื่อง Temperture Data Logger ต่อผ่าน Port อนุกรม COM1 หรือ COM2 ของ Computor กับเครื่องบันทึกและการส่งข้อมูลแบบแบบอนุกรมผ่าน RS 232 โดยข้อมูล 1 ข้อมูลมีขนาด 10 Bit

บิทที่ 1 เป็น Start Bit และตามด้วยข้อมูลที่เป็นค่าอุณหภูมิที่มีขนาด 8 Bit และตามด้วย Stop bit และตามด้วย Stop Bit 1 Bit

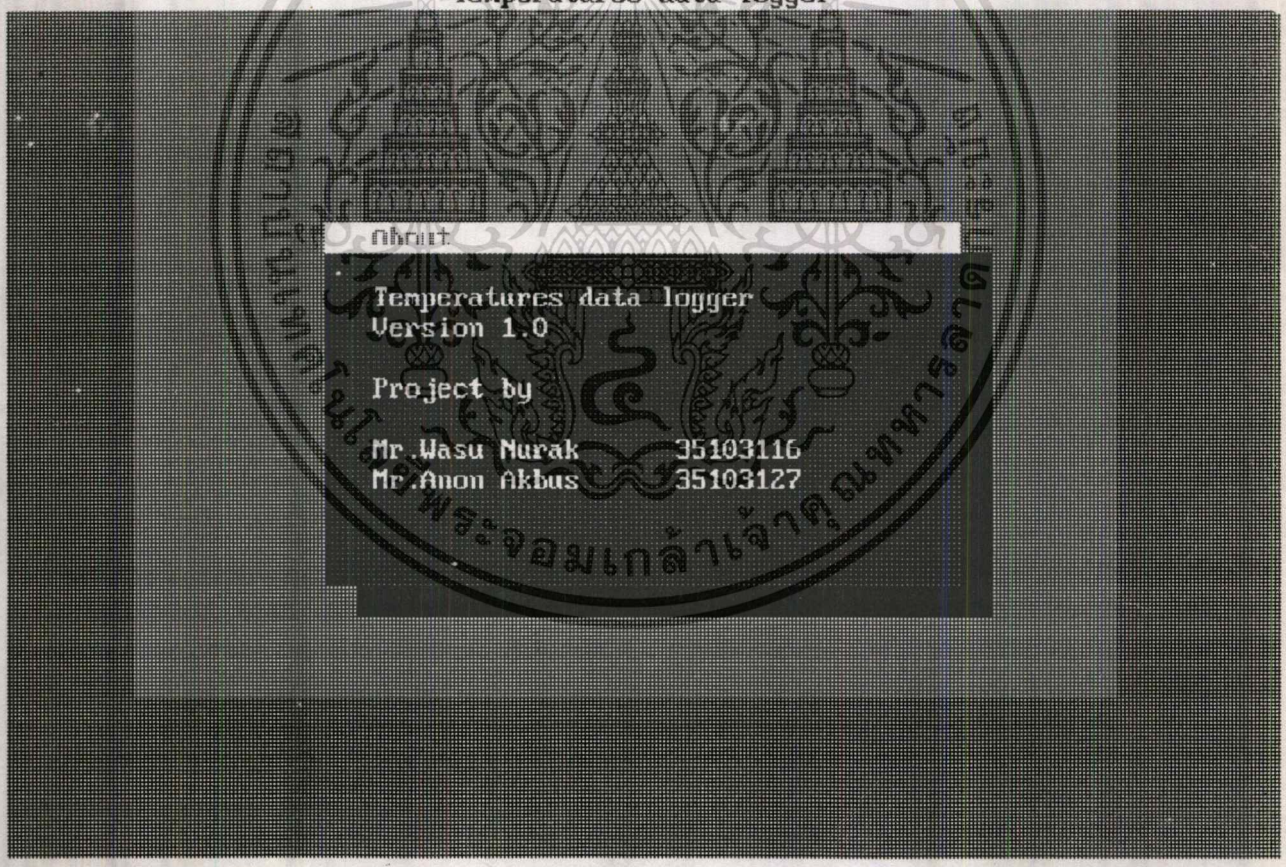
สำหรับโปรแกรมแสดงผลค่าอุณหภูมิในรูปแบบต่างๆ ที่จะกล่าวต่อไปนี้เป็นโปรแกรมที่เขียนจาก โปรแกรมภาษาซี

การแสดงข้อมูลอุณหภูมิสามารถแสดงได้ 3 รูปแบบ

1. Text Mode
2. Curve Graph
3. Bar Graph

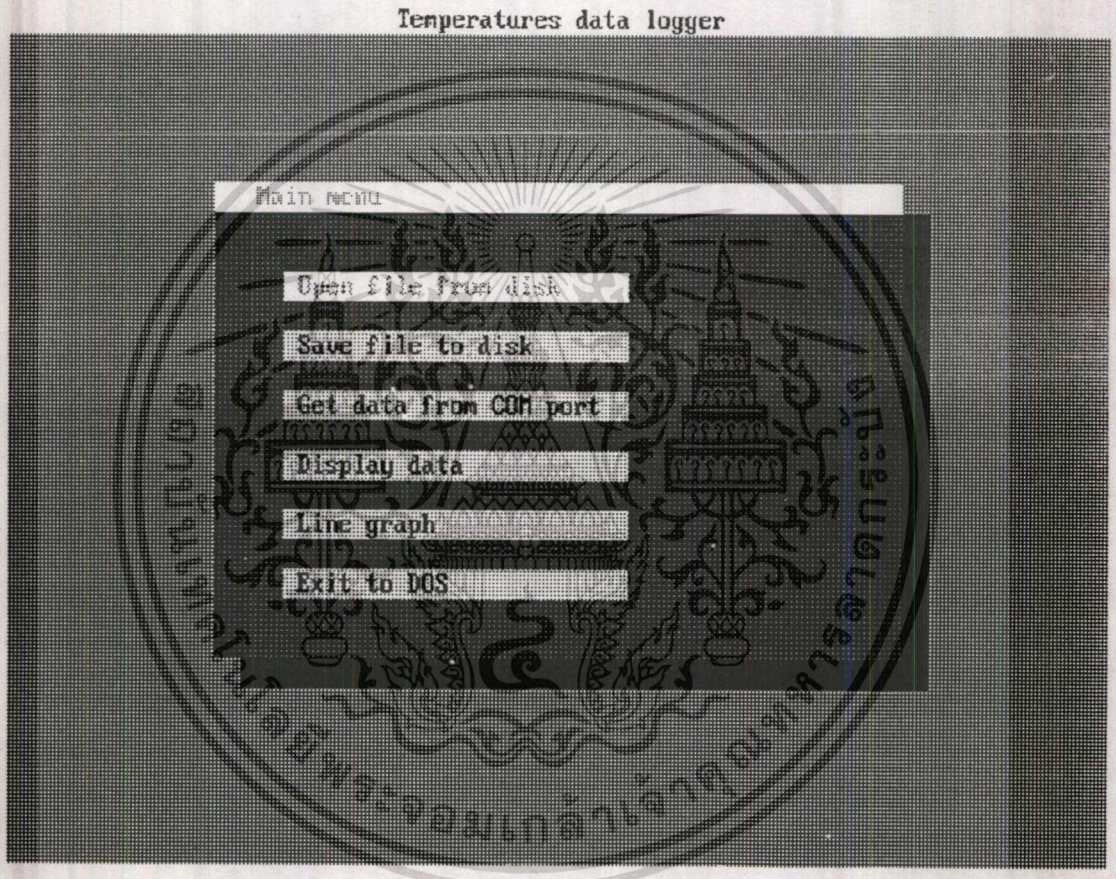
วิธีการใช้งานโปรแกรม DLOG.EXE

หลังจากเรียกไฟล์ DLOG.EXE ขึ้นมาใช้งานซึ่งได้แสดงดังรูปที่ 1 จากนั้นให้กดปุ่มใดๆ เพื่อทำงานต่อ



รูปที่ 1

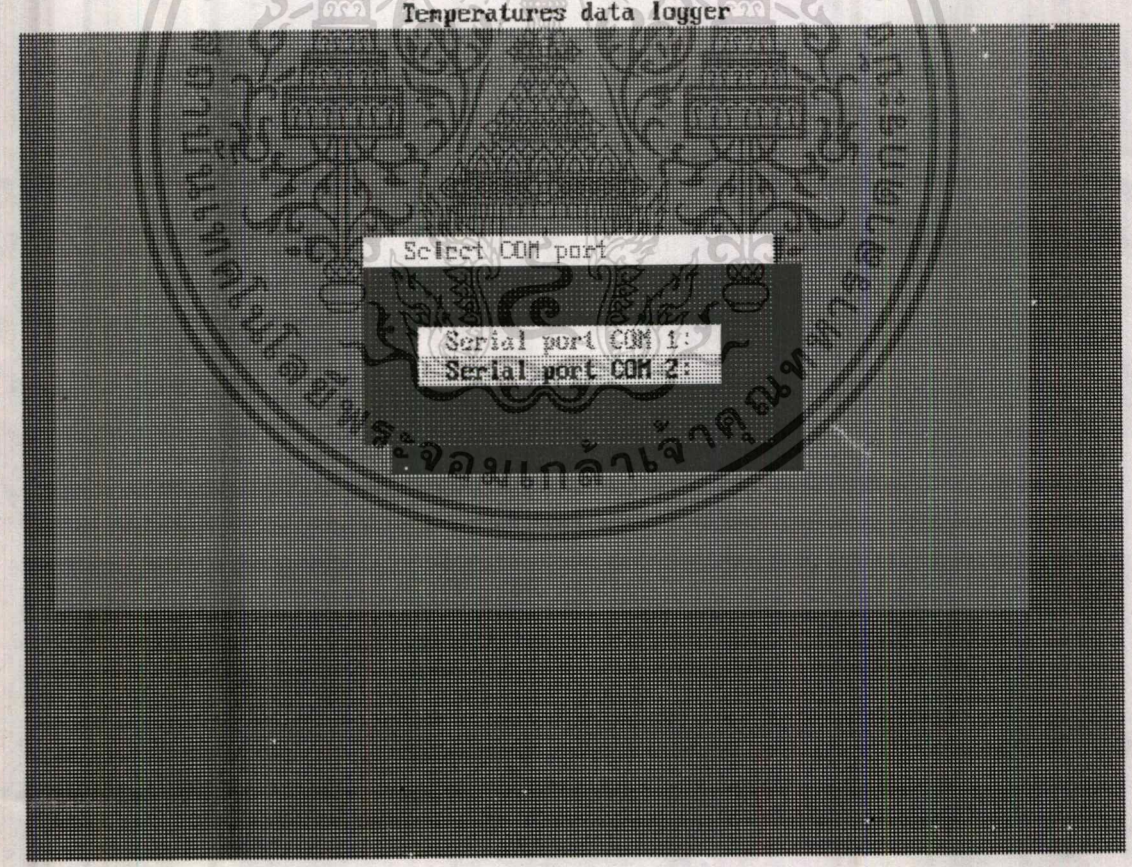
หลังจากที่ได้กดปุ่มใด ๆ แล้ว หน้าจอก็จะเปลี่ยนไปแสดง Main Menu แทน ดังแสดงไว้ในรูปที่ 2 ซึ่งสามารถทำงานใน Fuction ต่าง ๆ โดยการเลื่อนแถบสว่าง (High Light) มายัง Menu ที่ต้องการ



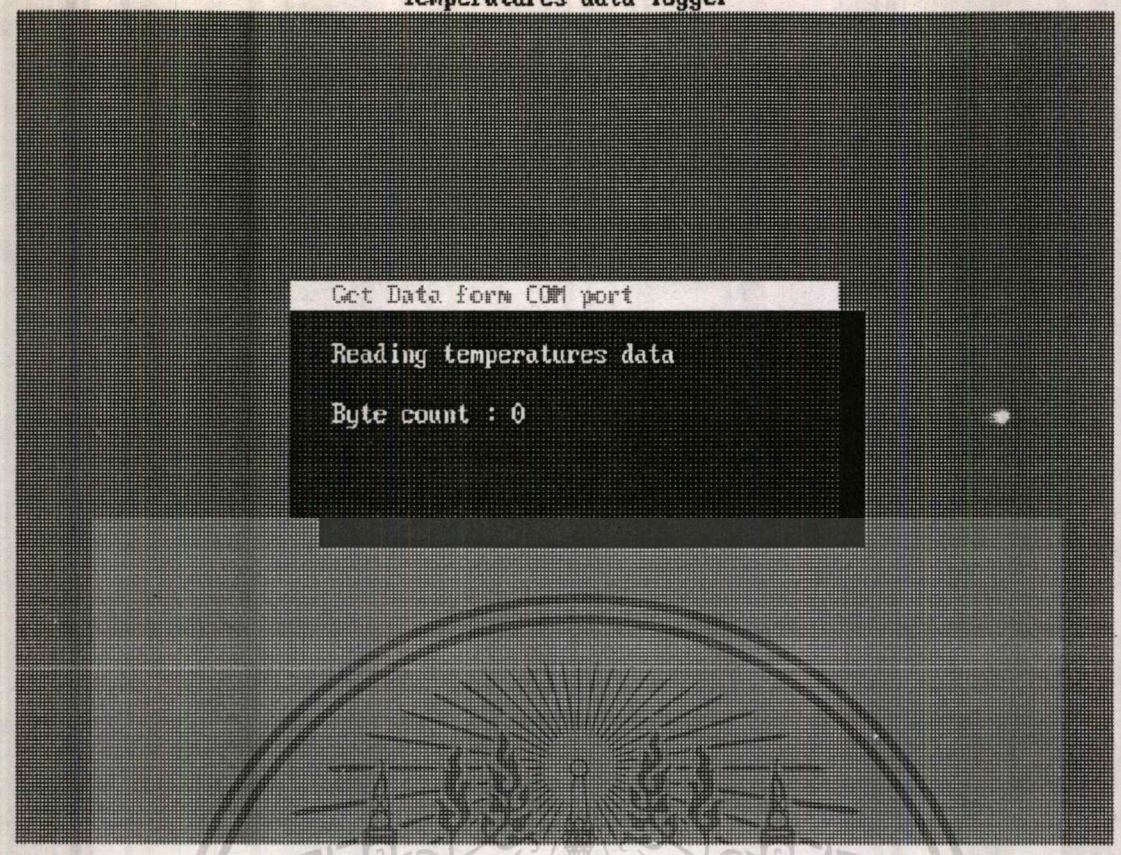
รูปที่ 2

ขั้นตอนการ Terminate DLOG กับ Data-logger Box

1. เลื่อนแถบสว่างมาที่ Get data from COM port แสดงในรูปที่ 3.
2. กำหนด Port ที่ใช้ในการ Terminate แสดงในรูปที่ 4.
3. หลังจากที่ทำการ Load Data เรียบร้อยแล้วก็จะกลับมาสู่ Main Menu อีกครั้ง เพื่อให้ User เลือกชนิดของการ แสดงผล
4. เลือกการแสดงผลใน Text Mode ก็จะสามารถดูค่าของอุณหภูมิกับเวลา ซึ่งแสดงไว้ในรูปที่ 5.
5. เลือกการแสดงผลแบบ Line Graph แสดงดังรูปที่ 6.
6. เลือกการแสดงผลแบบ Bar Graph แสดงดังรูปที่ 7.



รูปที่ 4



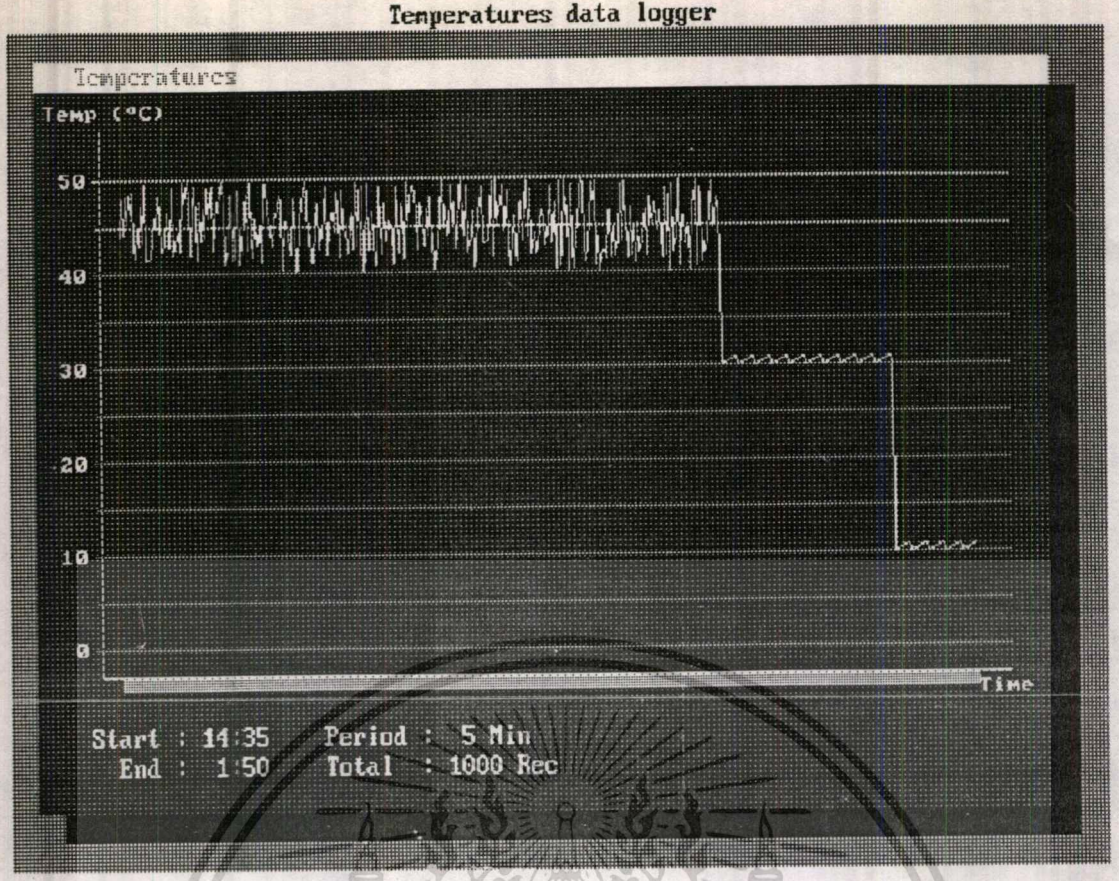
รูปที่ 5

Temperatures data logger

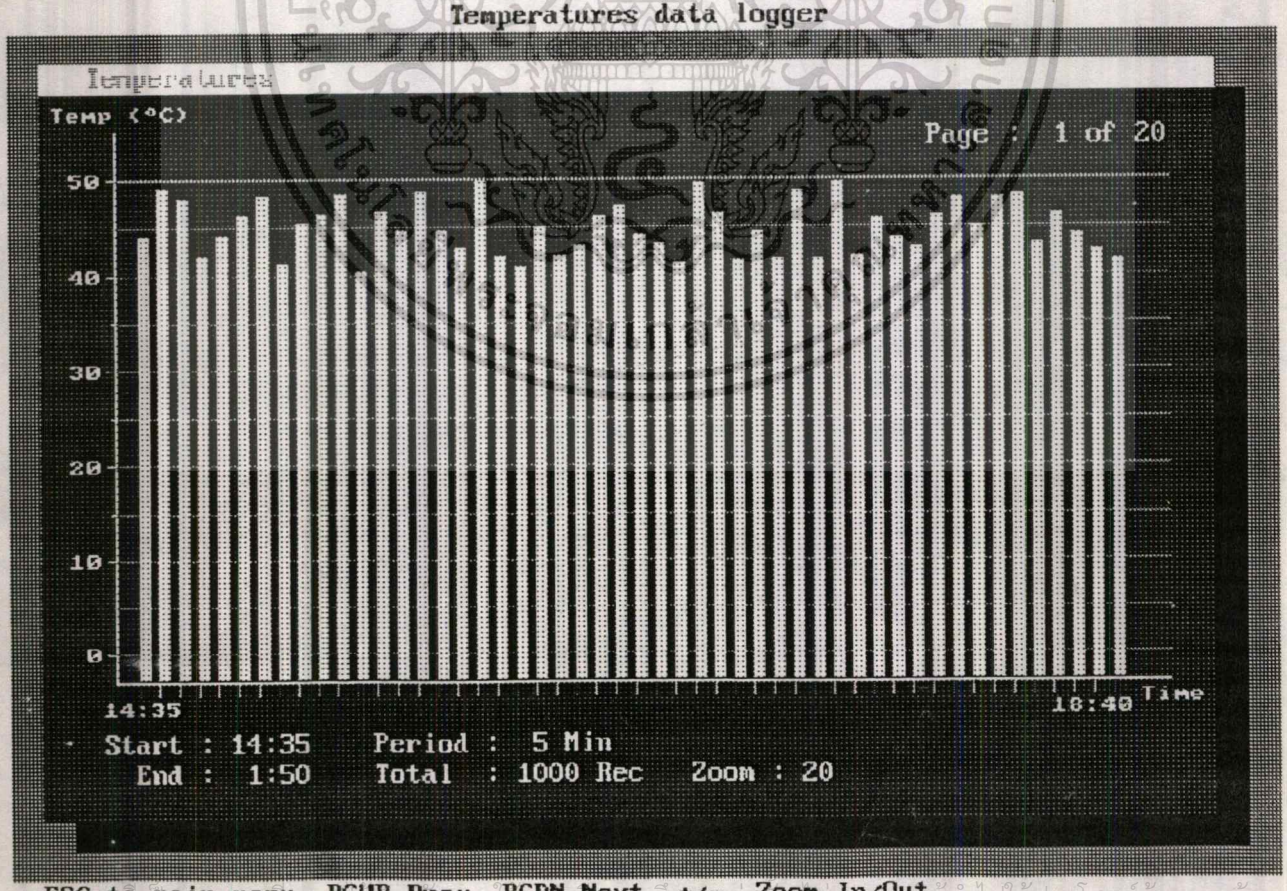
Temperatures

No.	Time	Temp °C
1	14:35	44.00
2	14:40	49.05
3	14:45	18.10
4	14:50	42.15
5	14:55	44.20
6	15:00	46.25
7	15:05	18.30
8	15:10	41.35
9	15:15	15.40
10	15:20	46.45
11	15:25	18.50
12	15:30	40.55
13	15:35	16.60
14	15:40	44.65
15	15:45	18.70

Start : 14:35 Period : 5 Min
End : 1:50 Total : 1000 Rec



รูปที่ 6



รูปที่ 7

Solfware File DLOG.C

```
typedef struct tagTEMPLOG
{
    BYTE    Hour;
    BYTE    Minute;
    BYTE    Period;
    TEMP    Data[MAX_DATA];
    UINT    Count;
} TEMPLOG;
typedef TEMPLOG     *PTEMPLOG;
typedef struct tagFILELOG
{
    UINT    Signature;
    TEMPLOG tmpLog;
} FILELOG;

UINT abs2hour ( UINT abs )
{
    return ( abs % 1440 ) / 60;
}

UINT abs2min ( UINT abs )
{
    return ( abs % 60 );
}

UINT bcd2bin ( UINT v )
{
    return (v / 16) * 10 + (v % 16);
}
```

```
void beep ( int freq, int dura )
{
    sound(freq);
    delay(dura);
    nosound();
}

void StatusLine ( char *sline )
{
    int     i;
    char    sbuf[81];
    Vputs( 0, _screen_hight - 1, COLOR_STATUS, sbuf );
    if ( sline != NULL )   Vputs( 2, _screen_hight - 1, COLOR_STATUS, sline )
}

void About ( void )
{
    PWINDOW pwin;
    RECT    rc = { 20, 8, 40, 12 };

    pwin = OpenWindow( rc, "About" );
    Wputs( pwin, 2, 1, COLOR_DLG_TEXT, "Temperatures data logger" );
    Wputs( pwin, 2, 2, COLOR_DLG_TEXT, "Version 1.0" );
    Wputs( pwin, 2, 4, COLOR_DLG_TEXT, "Project by" );
    Wputs( pwin, 2, 6, COLOR_DLG_TEXT, "Mr.Wasu Nurak      35103116");
    Wputs( pwin, 2, 7, COLOR_DLG_TEXT, "Mr.Anon Akbus      35103127");
    GetKey();
    CloseWindow( pwin );
    delay( 300 );
```

```
void beep ( int freq, int dura )
{
    sound(freq);
    delay(dura);
    nosound();
}

void StatusLine ( char *sline )
{
    int     i;
    char    sbuf[81];
    Vputs( 0, _screen_hight - 1, COLOR_STATUS, sbuf );
    if ( sline != NULL )   Vputs( 2, _screen_hight - 1, COLOR_STATUS, sline );
}

void About ( void )
{
    PWINDOW pwin;
    RECT    rc = { 20, 8, 40, 12 };

    pwin = OpenWindow( rc, "About" );
    Wputs( pwin, 2, 1, COLOR_DLG_TEXT, "Temperatures data logger" );
    Wputs( pwin, 2, 2, COLOR_DLG_TEXT, "Version 1.0" );
    Wputs( pwin, 2, 4, COLOR_DLG_TEXT, "Project by" );
    Wputs( pwin, 2, 6, COLOR_DLG_TEXT, "Mr.Wasu Nurak      35103116");
    Wputs( pwin, 2, 7, COLOR_DLG_TEXT, "Mr.Anon Akbus      35103127");
    GetKey();
    CloseWindow( pwin );
    delay( 300 );
```

```
int MainMenu ( int m )
{
    PWINDOW pwin;
    RECT    rc = { 15, 5, 50, 18 };
    static char  *sButton[] = { "Open file from disk",
                                "Save file to disk",
                                "Get data from COM port",
                                "Display data",
                                "Line graph",
                                "Bar graph",
                                "Exit to DOS" };

    pwin = OpenWindow( rc, "Main menu" );

    StatusLine( "ESC-to Exit" );
    do
    {
        for ( i = 0; i < 7; i++ )
        {
            Wprintf( pwin,  4, i * 2 + 2, ( sel == i ) ? COLOR_BTN_SEL : COLOR
        }

        switch ( GetScanKey())
        {
            case KEY_UP:    if ( --sel < 0 ) sel = 6;   break;
            case KEY_ESC:   result = CMD_EXIT;          break;
            case KEY_END:   sel = 6;                    break;
            case KEY_ENTER:
                if ( fHaveData == FALSE )
                if ( sel == CMD_SAVE      || sel == CMD_SHOWLIST ||
                    break;
```

```
                result = sel;
            }
        } while ( result == -1 );


    return result;
}


void InitScreen ( void )
{
    int     i;
    char    sbuf[81];

    memset( sbuf, '\x20', _screen_width );
    sbuf[81] = '\0';

    }

    Vputs( 0, 0, COLOR_CAPTION, sbuf );
    Vputs( 0, _screen_hight - 1, COLOR_STATUS, sbuf );

    Vputs( 28, 0, COLOR_CAPTION, "Temperatures data logger" );
}


BOOL GetData ( PTEMPLOG pTmplog )
{
    UINT    data;
    BYTE    *pTmp;
```

```
int     i, count = 0;
PWINDOW pwin;
RECT    rc = { 20, 10, 40, 8 };
BOOL    result = TRUE;
PTEMPLOG pTmplog2;

SerialEmpty();
delay( 10 );
SerialSend( START_CODE );

pTmp = (BYTE *)pTmplog2;


pwin = OpenWindow( rc, "Get Data form COM port" );
Wputs( pwin, 2, 3, COLOR_DLG_TEXT, "Byte count : 0" );

do
{
    if ( HitKey() == TRUE )
    {
        if ( GetScanKey() == KEY_ESC )
        {
            result = FALSE;
            break;
        }
    }

    if (( data = SerialRead()) != _SERIAL_EMPTY )
    {
      Wprintf( pwin, 15, 3, COLOR_DLG_TEXT, "%d", count );
    }
```

```
    if ( count > MAX_DATA )      break;
} while ( data != END_CODE );

pTmplog2->Count = (count - 3) / sizeof( TEMP );

CloseWindow( pwin );

if ( result == TRUE )
{  // Convert BCD value to BIN value
    pTmplog2->Hour   = bcd2bin( pTmplog2->Hour );
    pTmplog2->Minute = bcd2bin( pTmplog2->Minute );

    for ( i = 0; i < pTmplog2->Count; i++ )
    {
        pTmplog2->Data[i].Fix   = bcd2bin( pTmplog2->Data[i].Fix );
        pTmplog2->Data[i].Float = bcd2bin( pTmplog2->Data[i].Float );
    }

    fHaveData = TRUE;
    beep( 2000, 200 );
    delay( 60 );
    beep( 2000, 200 );
}
else
    beep( 2000, 200 );
MessageBox( "Tranfer", ( result == TRUE ) ? "Data tranferred complete."
                                          "Interrupted by user" );
free( pTmplog2 );
return result;
```

```
}
void ShowList ( PTEMPLOG pTmplog )
{
    PWINDOW pwin;
    RECT    rc = { 20, 3, 40, 24 };
    int     i, maxRow, hlist = 0;
    UINT    skey, absTime, startH, startM;

    maxRow = ( pTmplog->Count > MAX_LIST_ROW ) ? MAX_LIST_ROW : pTmplog->Cou

    pwin = OpenWindow( rc, "Temperatures" );

    StatusLine( "ESC-to main menu, /, Home, End, PgUp, PgDn" );

    Wputs( pwin,  5, 1, COLOR_DLG_TEXT, "No." );
    Wputs( pwin, 11, 1, COLOR_DLG_TEXT, "Time" );
    Wputs( pwin, 27, 1, COLOR_DLG_TEXT, "Temp  C" );

    Wprintf( pwin, 3, 18, COLOR_DLG_TEXT, "Start : %2d:%02d", startH, startM
    Wprintf( pwin, 3, 19, COLOR_DLG_TEXT, "  End : %2d:%02d",
             abs2hour(absTime + (pTmplog->Period * (pTmplog->Count - 1))),
             abs2min( absTime + (pTmplog->Period * (pTmplog->Count - 1))));

    Wprintf( pwin, 20, 19, COLOR_DLG_TEXT, "Total  : %2d Rec", pTmplog->Cou

    do
    {
        for ( i = 0; i < MAX_LIST_ROW; i++ )
```

```
            }
        } while ( skey != KEY_ESC && skey != KEY_ENTER );
        CloseWindow( pwin );
}

#define LG_START_X      30
#define LG_START_Y      20
#define LG_WIDTH        530
#define LG_HEIGHT       390

void LineGraph ( PTEMPLOG pTmplog )
{
    PWINDOW pwin;
    RECT    rc = { 2, 2, 74, 25 };
    int     i, j, x, y, ox = -1, oy = -1;
    UINT    stepOfValue, stepOfGrid, count;
    UINT    absTime, startH, startM;

    startH = pTmplog->Hour;
    startM = pTmplog->Minute;
    absTime = time2abs( startH, startM );

    pwin = OpenWindow( rc, "Temperatures" );

    StatusLine( "ESC-to main menu" );

    Wprintf( pwin, 3, 21, COLOR_DLG_TEXT, "Start : %2d:%02d", startH, startM );
    Wprintf( pwin, 3, 22, COLOR_DLG_TEXT, "  End : %2d:%02d",
            abs2hour(absTime + (pTmplog->Period * (pTmplog->Count - 1))),
            abs2min( absTime + (pTmplog->Period * (pTmplog->Count - 1))));
```

```
    Wprintf( pwin, 20, 21, COLOR_DLG_TEXT, "Period : %2d Min", pTmplog->Peri
    Wprintf( pwin, 20, 22, COLOR_DLG_TEXT, "Total  : %2d Rec", pTmplog->Coun

    WHline( pwin, LG_START_X, LG_END_X, LG_END_Y, YELLOW );        // draw x
    WVline( pwin, LG_START_X, LG_START_Y, LG_END_Y, YELLOW );      // draw y



    WCputs( pwin, LG_END_X - 0, LG_END_Y + 8, WHITE, "Time" );

    WCprintf( pwin, LG_START_X + 12, LG_END_Y + 14, YELLOW, "%2d:%02d",
            startH, startM );
            abs2min( absTime + (pTmplog->Period * (pTmplog->Count - 1))));

    for ( i = 1, j = 50; i < 12; i++ )
    {
        WHline( pwin, ( i % 2 == 1 ) ? LG_START_X - 4 : LG_START_X - 2, LG_S
        WHline( pwin, LG_START_X + 1, LG_END_X, LG_START_Y + i * (LG_GRID_Y/

        if ( i % 2 == 1 )
        {
            WCprintf( pwin, LG_START_X - 17, LG_START_Y + i * (LG_GRID_Y/2),
            j -= 10;
        }
    }

//    pTmplog->Count = 800;
    if ( pTmplog->Count > 500 )
    {
        stepOfValue = ((long)pTmplog->Count * 100L) / 500L;
```

```
        stepOfGrid  = 1;
        count = 500;
    }
    else
    {
        stepOfValue = 100;
        stepOfGrid  = 500 / pTmplog->Count;
        count = pTmplog->Count;
    }
    for ( i = 0; i < count; i++ )
    {
        x = LG_START_X + 12 + i * stepOfGrid;
        y = LG_START_Y + LG_GRID_Y / 2 + 250 -


        WVline( pwin, x, LG_END_Y + 1, LG_END_Y + 7, COLOR_LG_SCALE );

        if ( ox != -1 && oy != -1 )
        {
            Wline( pwin, ox, oy, x, y, COLOR_LG_LINE );
        }

        if ( pTmplog->Count < 200 )
        {
//          Wrectangle( pwin, x - 2, y - 2, x + 2, y + 2, COLOR_LG_MARK );
        }
        ox = x;
        oy = y;
```

```
    GetKey();
    CloseWindow( pwin );
}

#define BG_START_X      30
#define BG_START_Y      20
#define BG_END_Y        (BG_START_Y + BG_HEIGHT)
#define BG_GRID_Y       50
#define BG_MAX_COL      20
#define BG_MODE_FIT     1
#define BG_MODE_PAGE    2

void BarGraph ( PTEMPLOG pTmplog )
{
    PWINDOW pwin;
    RECT    rc = { 2, 2, 74, 25 };
    UINT    absTime, startH, startM, zoom = 0, skey;
    static UINT zoomTbl[] = { 1, 2, 5, 10, 20, 50, 100, 200 };

    startH = pTmplog->Hour;
    startM = pTmplog->Minute;


    pwin = OpenWindow( rc, "Temperatures" );

    StatusLine( "ESC-to main menu, PGUP-Prev, PGDN-Next, +/-  Zoom In/Out" );

    Wprintf( pwin, 3, 21, COLOR_DLG_TEXT, "Start : %2d:%02d", startH, startM );
    Wprintf( pwin, 3, 22, COLOR_DLG_TEXT, "  End : %2d:%02d",
            abs2hour(absTime + (pTmplog->Period * (pTmplog->Count - 1))),
```

```
            abs2min( absTime + (pTmplog->Period * (pTmplog->Count - 1))));
    Wprintf( pwin, 20, 21, COLOR_DLG_TEXT, "Period : %2d Min", pTmplog->Period
    Wprintf( pwin, 20, 22, COLOR_DLG_TEXT, "Total  : %2d Rec", pTmplog->Count )

    WHline( pwin, BG_START_X, BG_END_X, BG_END_Y, YELLOW );        // draw x
    WVline( pwin, BG_START_X, BG_START_Y, BG_END_Y, YELLOW );      // draw y

    WCputs( pwin, LG_START_X + 2, LG_START_Y - 10, WHITE, "Temp ( C)" );
    WCputs( pwin, LG_END_X - 0, LG_END_Y + 8, WHITE, "Time" );

    for ( i = 1, j = 50; i < 12; i++ )
    {
        WHline( pwin, ( i % 2 == 1 ) ? BG_START_X - 4 : BG_START_X - 2, BG_STAR
//      WHline( pwin, BG_START_X + 1, BG_END_X, BG_START_Y + i * (BG_GRID_Y/2),

        if ( i % 2 == 1 )
        {
            WCprintf( pwin, BG_START_X - 17, BG_START_Y + i * (BG_GRID_Y/2), WH
            j -= 10;
        }
    }

    do
    {
        Wbox( pwin, BG_START_X + 1, BG_START_Y + 1, BG_END_X - 1, BG_END_Y - 1,
        Wbox( pwin, BG_START_X - 10, BG_END_Y + 1, BG_END_X - 18, BG_END_Y + 25

        for ( i = 1, j = 50; i < 12; i++ )
        {
            WHline( pwin, BG_START_X + 1, BG_END_X, BG_START_Y + i * (BG_GRID_Y
```

```
    }
countPage = pTmplog->Count / zoomTbl[zoom];
numOfpage = zoomTbl[zoom] + ((( pTmplog->Count % zoomTbl[zoom] ) == 0 ) ? 0
startOfpage = countPage * page;

Wprintf( pwin, 55,  1, COLOR_DLG_TEXT, "Page : %2d of %-3d", page + 1, numO
Wprintf( pwin, 40, 22, COLOR_DLG_TEXT, "Zoom : %-3d", zoomTbl[zoom] );
WCprintf( pwin, BG_START_X + 12, BG_END_Y + 14, YELLOW, "%2d:%02d",
        abs2hour(absTime + (pTmplog->Period * (startOfpage))),
        abs2min( absTime + (pTmplog->Period * (startOfpage))));

WCprintf( pwin, BG_END_X - 40, BG_END_Y + 14, YELLOW, "%2d:%02d",
        abs2hour(absTime + (pTmplog->Period * (startOfpage + countPage - 1
        abs2min( absTime + (pTmplog->Period * (startOfpage + countPage - 1


if ( countPage > 500 )
{
    stepOfValue = ((long)pTmplog->Count * 100L) / 500L;
    stepOfGrid  = 1;
{
    stepOfValue = 100;
    stepOfGrid  = 500 / countPage;
    bgWidth = ( stepOfGrid <= 2 ) ? 0 : stepOfGrid / 2;
    count = countPage;
}

for ( i = 0; i < count; i++ )
{
    if ( startOfpage + i > pTmplog->Count - 1 )    break;
```

```
    x = BG_START_X + 12 + i * stepOfGrid;
    y = BG_START_Y + BG_GRID_Y / 2 + 250 -
        (pTmplog->Data[(((long)i + startOfpage) * (long)stepOfValue) / 100]
        (pTmplog->Data[(((long)i + startOfpage) * (long)stepOfValue) / 100]

    WVline( pwin, x, BG_END_Y + 1, BG_END_Y + 7, COLOR_BG_SCALE );

    if ( bgWidth != 0 )
        Wbox( pwin, x, BG_END_Y - 2, x + bgWidth, y, LIGHTCYAN );
    else
        Wline( pwin, x, BG_END_Y - 2, x, y, LIGHTCYAN );

}

    switch ( skey = GetScanKey())
    {
        case KEY_PGUP:  if ( page > 0 )
                            page--;
                        break;

        case KEY_PGDN:  if ( page < numOfpage - 1 )
                            page++;
                        break;

        case 0x4E:      if ( zoom < sizeof( zoomTbl ) / sizeof( UINT ) - 1)
                        {
                           if ( pTmplog->Count / zoomTbl[zoom] > 5 )
                              zoom++;
                           page = 0;
                        }
                        break;
```

```
            case 0x4A:      if ( zoom > 0 )
                            {
                               zoom--;
                               page = 0;
                            }
                            break;
        }
    } while ( skey != KEY_ESC && skey != KEY_ENTER );

    CloseWindow( pwin );
}
void SetCOM ( void )
{
    UINT    numOfCom;
    PWINDOW pwin;
    RECT    rc = { 25, 8, 30, 18 };
    int     i, l, sel = 0, result = -1;
    static char  *sCOM[] = { "  Serial port COM 1:  ",
                             "  Serial port COM 2:  ",
                             "  Serial port COM 3:  ",
                             "  Serial port COM 4:  ",
                             "  Serial port COM 5:  ",
                             "  Serial port COM 6:  ",
                             "  Serial port COM 7:  ",
                             "  Serial port COM 8:  " };
    if ( numOfCom == 1 )
    {
        comPort = 0;
        return;
    }
    rc.bottom = numOfCom + 5;
```

```
    pwin = OpenWindow( rc, "Select COM port" );
    do
    {
        for ( i = 0; i < numOfCom; i++ )
        {
            Wputs( pwin,  3, i + 2, ( sel == i ) ? COLOR_BTN_SEL : COLOR_BTN_NO
        }
        switch ( GetScanKey())
        {
            case KEY_UP:    if ( --sel < 0 ) sel = numOfCom - 1;   break;
            case KEY_DOWN:  if ( ++sel > numOfCom - 1) sel = 0;    break;
            case KEY_ENTER: result = sel;
        }
    } while ( result == -1 );

    comPort = result;

    CloseWindow( pwin );
}


    PWINDOW pwin;
    RECT    rc = { 20, 8, 40, 8 };
    char    s[256];
    UINT    key, skey, ckey;
    int     result = FALSE, len = 0;

    pwin = OpenWindow( rc, Title );
    len = strlen( string );
    strcpy( s, string );
    do
```

```
{
    Wprintf( pwin, 2, 2, COLOR_LST_TEXT, "%-31s", s );

    key = GetKey();
    skey = key >> 8;
    ckey = key & 0x00FF;

    switch ( skey )
    {
        case KEY_ENTER:
            s[len] = '\0';
            result = TRUE;
            break;

        case KEY_BACKSPACE:
            if ( len > 0 )
            {
                s[len] = '\0';
                s[--len] = '\xDB';
            }
            break;

        default:
            if ( len < 30 )
            {
                s[len] = ckey;
                s[++len] = '\xDB';
                s[len+1] = '\0';
            }
    }
} while ( skey != KEY_ESC );
```

```
void OpenFile ( PTEMPLOG pTmplog )
{
    PFILELOG    pfileLog;
    char        sfile[80] = { "\0" };
    int         hfile;
    char        sbuf[80];

    pfileLog = malloc( sizeof( FILELOG ));

    if ( GetString( "Open file", sfile ) == TRUE )
    {
        if (( hfile = open( sfile, O_RDONLY | O_BINARY )) == -1 )
        {
            sprintf( sbuf, "Can't open file '%s'", strupr( sfile ));
            MessageBox( "ERROR", sbuf );

            return;
        }
        read( hfile, pfileLog, sizeof( FILELOG ));

        if ( pfileLog->Signature != FILE_SIGNATURE )
        {
            sprintf( sbuf, "'%s' is not temperature log file.", strupr( sfile )
            MessageBox( "ERROR", sbuf );
            free( pfileLog );
            return;
        }

        *pTmplog = pfileLog->tmpLog;
        fHaveData = TRUE;
```

```
    }
}
void SaveFile ( PTEMPLOG pTmplog )
{
    PFILELOG    pfileLog;
    char        sfile[80] = { "\0" };
    int         hfile;
    pfileLog = malloc( sizeof( FILELOG ));
    if ( GetString( "Save data", sfile ) == TRUE )
    {
        if (( hfile = creat( sfile, S_IREAD | S_IWRITE )) == -1 )
        {
            char sbuf[80];

            sprintf( sbuf, "Can't create file '%s'", strupr( sfile ));
            MessageBox( "ERROR", sbuf );
            free( pfileLog );
            return;
        }
        close( hfile );
        free( pfileLog );
    }
}
void BarGraph ( PTEMPLOG pTmplog );
int main ( void )
{
    int         menu = 0;

    InitVedio( _MODE_AUTO_ );

    InitScreen();
```

```
    About();
    SetCOM();
    InitSerial( comPort, _BUAD_9600 );

    _fmode = O_BINARY;
    pTmplog = malloc( sizeof( TEMPLOG ));
    memset( pTmplog, 0, sizeof( TEMPLOG ));
/*
    {
        int i, j, l = 0;
        randomize();
        for ( i = 0; i < 1000; i++ )
        for ( j = 0; j < 3; j+=5, l++ )
        {
           pTmplog->Data[l].Float = j;
           pTmplog->Data[l].Fix = random( 10 ) + 40;
           if ( l > 700 )  pTmplog->Data[l].Fix = 30;
           if ( l > 900 )  pTmplog->Data[l].Fix = 10;
        }
        pTmplog->Hour   = 14;
        pTmplog->Minute = 35;
        pTmplog->Period = 5;
        pTmplog->Count  = l;
    }
*/
    do
    {
        switch ( menu = MainMenu( menu ))
        {
            case CMD_OPEN:      OpenFile( pTmplog );    break;
```

```
            case CMD_SAVE:       SaveFile( pTmplog );     break;
            case CMD_GETDATA:    GetData( pTmplog );      break;
            case CMD_SHOWLIST:   ShowList( pTmplog );     break;
            case CMD_LINEGRAPH:  LineGraph( pTmplog );    break;
            case CMD_BARGRAPH:   BarGraph( pTmplog );     break;
        }
    } while ( menu != CMD_EXIT );

    free( pTmplog );
    RestoreSerial();
    DoneVedio();
    return 0;
}
```

National Semiconductor

ADC0808, ADC0809 8-Bit μP Compatible A/D Converters With 8-Channel Multiplexer

General Description

The ADC0808, ADC0809 data acquisition component is a monolithic CMOS device with an 8-bit analog-to-digital converter, 8-channel multiplexer and microprocessor compatible control logic. The 8-bit A/D converter uses successive approximation as the conversion technique. The converter features a high impedance chopper stabilized comparator, a 256R voltage divider with analog switch tree and a successive approximation register. The 8-channel multiplexer can directly access any of 8-single-ended analog signals.

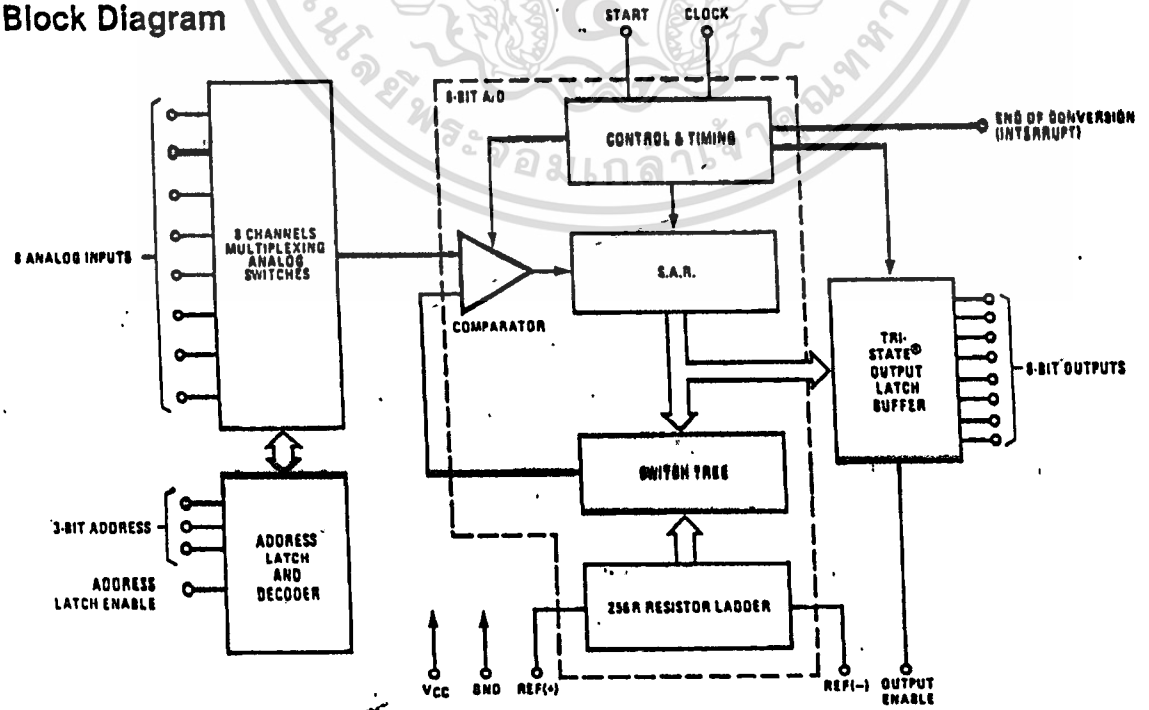
The device eliminates the need for external zero and full-scale adjustments. Easy interfacing to microprocessors is provided by the latched and decoded multiplexer address inputs and latched TTL TRI-STATE® outputs.

The design of the ADC0808, ADC0809 has been optimized by incorporating the most desirable aspects of several A/D conversion techniques. The ADC0808, ADC0809 offers high speed, high accuracy, minimal temperature dependence, excellent long-term accuracy and repeatability, and consumes minimal power. These features make this device ideally suited to applications from process and machine control to consumer and automotive applications. For 16-channel multiplexer with common output (sample/hold port) see ADC0816 data sheet. (See AN-247 for more information.

Features

- Resolution — 8-bits
- Total unadjusted error — ±1/2 LSB and ±1 LSB
- No missing codes
- Conversion time — 100 μs
- Single supply — 5 V_{DC}
- Operates ratiometrically or with 5 V_{DC} or analog span adjusted voltage reference
- 8-channel multiplexer with latched control logic
- Easy interface to all microprocessors, or operates "stand alone"
- Outputs meet T^2L voltage level specifications
- 0V to 5V analog input voltage range with single 5V supply
- No zero or full-scale adjust required
- Standard hermetic or molded 28-pin DIP package
- Temperature range −40°C to +85°C or −55°C to +125°C
- Low power consumption — 15 mW
- Latched TRI-STATE® output

Block Diagram



TRI-STATE® is a registered trademark of National Semiconductor Corp.

Absolute Maximum Ratings (Notes 1 and 2)

Supply Voltage (V_{CC}) (Note 3)	6.5V
Voltage at Any Pin Except Control Inputs	−0.3V to (V_{CC} + 0.3V)
Voltage at Control Inputs (START, OE, CLOCK, ALE, ADD A, ADD B, ADD C)	−0.3V to +15V
Storage Temperature Range	−65°C to +150°C
Package Dissipation at T_A = 25°C	875 mW
Lead Temperature (Soldering, 10 seconds)	300°C

Operating Ratings (Notes 1 and 2)

Temperature Range (Note 1)	$T_{MIN} \le T_A \le T_{MAX}$
ADC0808CJ	−55°C ≤ T_A ≤ +125°C
ADC0808CCJ, ADC0808CCN, ADC0809CCN	−40°C ≤ T_A ≤ +85°C
Range of V_{CC} (Note 1)	4.5 V_{DC} to 6.0 V_{DC}

Electrical Characteristics

Converter Specifications: V_{CC} = 5 V_{DC} = $V_{REF(+)}$, $V_{REF(-)}$ = GND, $T_{MIN} \le T_A \le T_{MAX}$ and f_{CLK} = 640 kHz unless otherwise stated.

	Parameter	Conditions	Min	Typ	Max	Units
	ADC0808					
	Total Unadjusted Error (Note 5)	25°C			±1/2	LSB
		T_{MIN} to T_{MAX}			±3/4	LSB
	ADC0809					
	Total Unadjusted Error (Note 6)	0°C to 70°C			±1	LSB
		T_{MIN} to T_{MAX}			±1 1/4	LSB
	Input Resistance	From Ref(+) to Ref(−)	1.0	2.5		kΩ
	Analog Input Voltage Range	(Note 4) V(+) or V(−)	GND−0.10		V_{CC}+0.10	V_{DC}
$V_{REF(+)}$	Voltage, Top of Ladder	Measured at Ref(+)		V_{CC}	V_{CC}+0.1	V
$\frac{V_{REF(+)}+V_{REF(-)}}{2}$	Voltage, Center of Ladder		V_{CC}/2−0.1	V_{CC}/2	V_{CC}/2+0.1	V
$V_{REF(-)}$	Voltage, Bottom of Ladder	Measured at Ref(−)	−0.1	0		V
	Comparator Input Current	f_c = 640 kHz, (Note 6)	−2	±0.5	2	μA

Electrical Characteristics

Digital Levels and DC Specifications: ADC0808CJ 4.5V ≤ V_{CC} ≤ 5.5V, −55°C ≤ T_A ≤ +125°C unless otherwise noted
ADC0808CCJ, ADC0808CCN, and ADC0809CCN 4.75 ≤ V_{CC} ≤ 5.25V, −40°C ≤ T_A ≤ +85°C unless otherwise noted

	Parameter	Conditions	Min	Typ	Max	Units
ANALOG MULTIPLEXER						
$I_{OFF(+)}$	OFF Channel Leakage Current	V_{CC} = 5V, V_{IN} = 5V,				
		T_A = 25°C		10	200	nA
		T_{MIN} to T_{MAX}			1.0	μA
$I_{OFF(-)}$	OFF Channel Leakage Current	V_{CC} = 5V, V_{IN} = 0,				
		T_A = 25°C	−200	−10		nA
		T_{MIN} to T_{MAX}	−1.0			μA
CONTROL INPUTS						
$V_{IN(1)}$	Logical "1" Input Voltage		V_{CC}−1.5			V
$V_{IN(0)}$	Logical "0" Input Voltage				1.5	V
$I_{IN(1)}$	Logical "1" Input Current (The Control Inputs)	V_{IN} = 15V			1.0	μA
$I_{IN(0)}$	Logical "0" Input Current (The Control Inputs)	V_{IN} = 0	−1.0			μA
I_{CC}	Supply Current	f_{CLK} = 640 kHz		0.3	3.0	mA

Functional Description

Multiplexer: The device contains an 8-channel single-ended analog signal multiplexer. A particular input channel is selected by using the address decoder. Table I shows the input states for the address lines to select any channel. The address is latched into the decoder on the low-to-high transition of the address latch enable signal.

TABLE I

SELECTED ANALOG CHANNEL	ADDRESS LINE		
	C	B	A
IN0	L	L	L
IN1	L	L	H
IN2	L	H	L
IN3	L	H	H
IN4	H	L	L
IN5	H	L	H
IN6	H	H	L
IN7	H	H	H

CONVERTER CHARACTERISTICS

The Converter

The heart of this single chip data acquisition system is its 8-bit analog-to-digital converter. The converter is designed to give fast, accurate, and repeatable conversions over a wide range of temperatures. The converter is partitioned into 3 major sections: the 256R ladder network, the successive approximation register, and the comparator. The converter's digital outputs are positive true.

The 256R ladder network approach *(Figure 1)* was chosen over the conventional R/2R ladder because of its inherent monotonicity, which guarantees no missing digital codes. Monotonicity is particularly important in closed loop feedback control systems. A non-monotonic relationship can cause oscillations that will be catastrophic for the system. Additionally, the 256R network does not cause load variations on the reference voltage.

The bottom resistor and the top resistor of the ladder network in *Figure 1* are not the same value as the remainder of the network. The difference in these resistors causes the output characteristic to be symmetrical with the zero and full-scale points of the transfer curve. The first output transition occurs when the analog signal has reached +1/2 LSB and succeeding output transitions occur every 1 LSB later up to full-scale.

The successive approximation register (SAR) performs 8 iterations to approximate the input voltage. For any SAR type converter, n-iterations are required for an n-bit converter. *Figure 2* shows a typical example of a 3-bit converter. In the ADC0808, ADC0809, the approximation technique is extended to 8 bits using the 256R network.

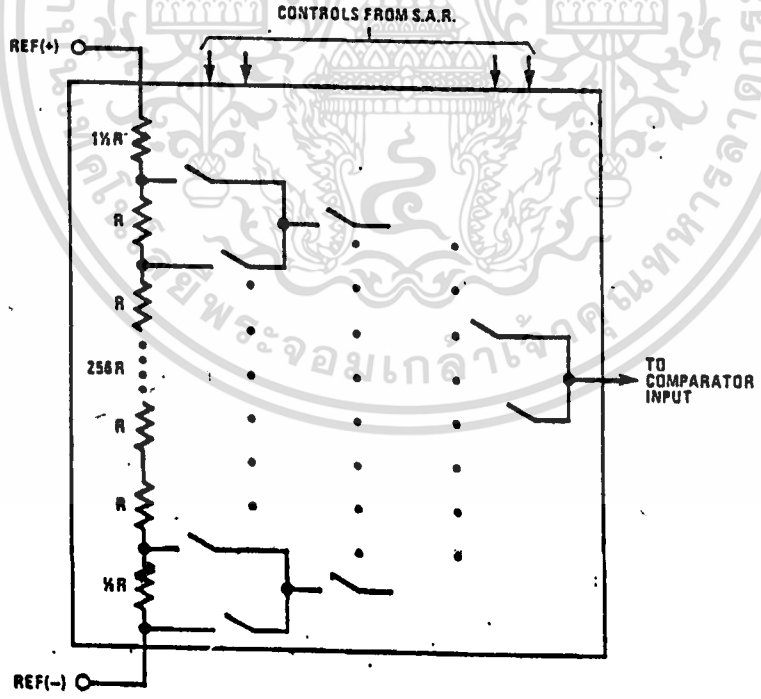
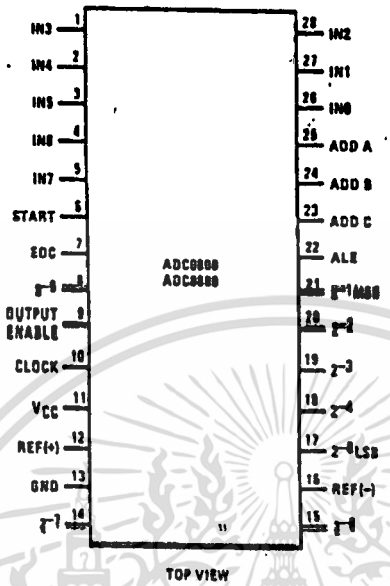


FIGURE 1. Resistor Ladder and Switch Tree

Connection Diagram

Dual-In-Line Package



TOP VIEW

Timing Diagram

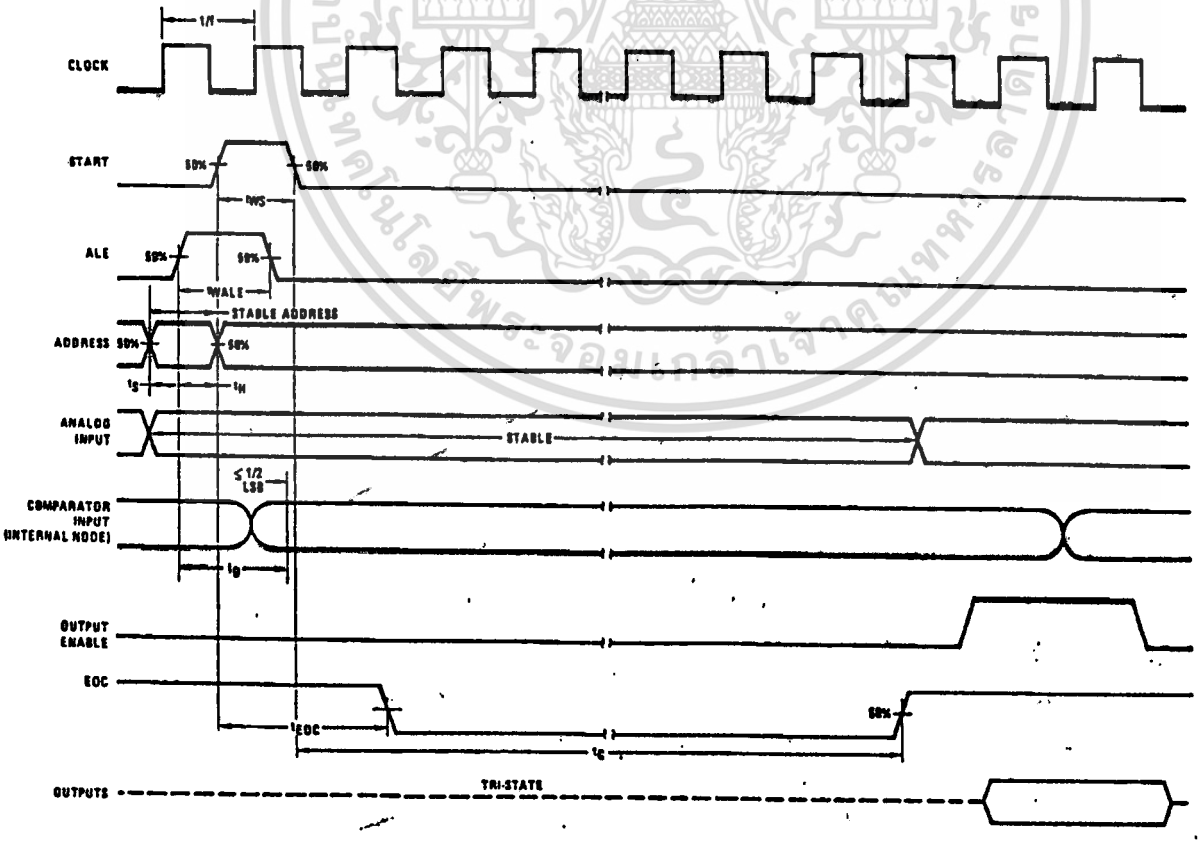


FIGURE 5

Electrical Characteristics (Continued)

Digital Levels and DC Specifications: ADC0808CJ $4.5V \le V_{CC} \le 5.5V$, $-55°C \le T_A \le +125°C$ unless otherwise noted
ADC0808CCJ, ADC0808CCN, and ADC0809CCN $4.75 \le V_{CC} \le 5.25V$, $-40°C \le T_A \le +85°C$ unless otherwise noted

Parameter		Conditions	Min	Typ	Max	Units
DATA OUTPUTS AND EOC (INTERRUPT)						
$V_{OUT(1)}$	Logical "1" Output Voltage	$I_O = -360\ \mu A$	$V_{CC}-0.4$			V
$V_{OUT(0)}$	Logical "0" Output Voltage	$I_O = 1.6$ mA			0.45	V
$V_{OUT(0)}$	Logical "0" Output Voltage EOC	$I_O = 1.2$ mA			0.45	V
I_{OUT}	TRI-STATE® Output Current	$V_O = 5V$			3	μA
		$V_O = 0$	-3			μA

Electrical Characteristics

Timing Specifications: $V_{CC} = V_{REF(+)} = 5V$, $V_{REF(-)} = GND$, $t_r = t_f = 20$ ns and $T_A = 25°C$ unless otherwise noted.

Symbol	Parameter	Conditions	Min	Typ	Max	Units
t_{WS}	Minimum Start Pulse Width	*(Figure 5)*		100	200	ns
t_{WALE}	Minimum ALE Pulse Width	*(Figure 5)*		100	200	ns
t_s	Minimum Address Set-Up Time	*(Figure 5)*		25	50	ns
t_H	Minimum Address Hold Time	*(Figure 5)*		25	50	ns
t_D	Analog MUX Delay Time From ALE	$R_S = 0\Omega$ *(Figure 5)*		1	2.5	μs
t_{H1}, t_{H0}	OE Control to Q Logic State	$C_L = 50$ pF, $R_L = 10$k *(Figure 8)*		125	250	ns
t_{1H}, t_{0H}	OE Control to Hi-Z	$C_L = 10$ pF, $R_L = 10$k *(Figure 8)*		125	250	ns
t_C	Conversion Time	$f_c = 640$ kHz, *(Figure 5)* (Note 7)	90	100	116	μs
f_c	Clock Frequency		10	640	1280	kHz
t_{EOC}	EOC Delay Time	*(Figure 5)*	0		8 + 2 μs	Clock Periods
C_{IN}	Input Capacitance	At Control Inputs		10	15	pF
C_{OUT}	TRI-STATE® Output Capacitance	At TRI-STATE® Outputs, (Note 12)		10	15	pF

Note 1: Absolute maximum ratings are those values beyond which the life of the device may be impaired.

Note 2: All voltages are measured with respect to GND, unless otherwise specified.

Note 3: A zener diode exists, internally, from V_{CC} to GND and has a typical breakdown voltage of 7 V_{DC}.

Note 4: Two on-chip diodes are tied to each analog input which will forward conduct for analog input voltages one diode drop below ground or one diode drop greater than the V_{CC} supply. The spec allows 100 mV forward bias of either diode. This means that as long as the analog V_{IN} does not exceed the supply voltage by more than 100 mV, the output code will be correct. To achieve an absolute 0 V_{DC} to 5 V_{DC} input voltage range will therefore require a minimum supply voltage of 4.900 V_{DC} over temperature variations, initial tolerance and loading.

Note 5: Total unadjusted error includes offset, full-scale, linearity, and multiplexer errors. See *Figure* 3. None of these A/Ds requires a zero or full-scale adjust. However, if an all zero code is desired for an analog input other than 0.0V, or if a narrow full-scale span exists (for example: 0.5V to 4.5V full-scale) the reference voltages can be adjusted to achieve this. See *Figure 13*.

Note 6: Comparator input current is a bias current into or out of the chopper stabilized comparator. The bias current varies directly with clock frequency and has little temperature dependence *(Figure 6)*. See paragraph 4.0.

Note 7: The outputs of the data register are updated one clock cycle before the rising edge of EOC.

Functional Description (Continued)

The A/D converter's successive approximation register (SAR) is reset on the positive edge of the start conversion (SC) pulse. The conversion is begun on the falling edge of the start conversion pulse. A conversion in process will be interrupted by receipt of a new start conversion pulse. Continuous conversion may be accomplished by tying the end-of-conversion (EOC) output to the SC input. If used in this mode, an external start conversion pulse should be applied after power up. End-of-conversion will go low between 0 and 8 clock pulses after the rising edge of start conversion.

The most important section of the A/D converter is the comparator. It is this section which is responsible for the ultimate accuracy of the entire converter. It is also the comparator drift which has the greatest influence on the repeatability of the device. A chopper-stabilized comparator provides the most effective method of satisfying all the converter requirements.

The chopper-stabilized comparator converts the DC input signal into an AC signal. This signal is then fed through a high gain AC amplifier and has the DC level restored. This technique limits the drift component of the amplifier since the drift is a DC component which is not passed by the AC amplifier. This makes the entire A/D converter extremely insensitive to temperature, long term drift and input offset errors.

Figure 4 shows a typical error curve for the ADC0808 as measured using the procedures outlined in AN-179.

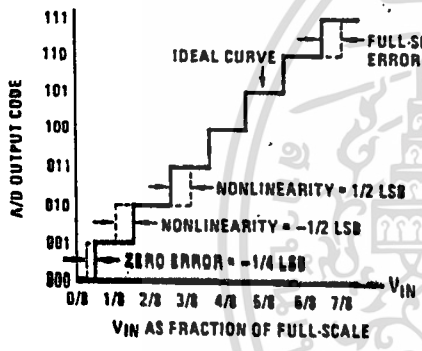


FIGURE 2. 3-Bit A/D Transfer Curve

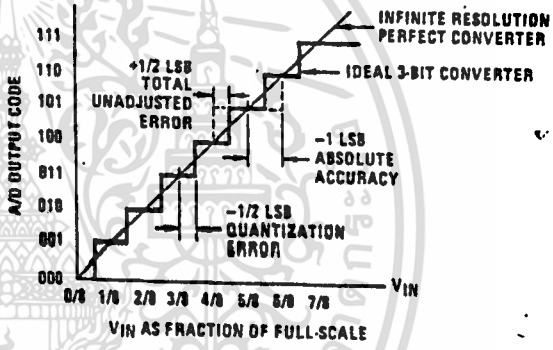


FIGURE 3. 3-Bit A/D Absolute Accuracy Curve

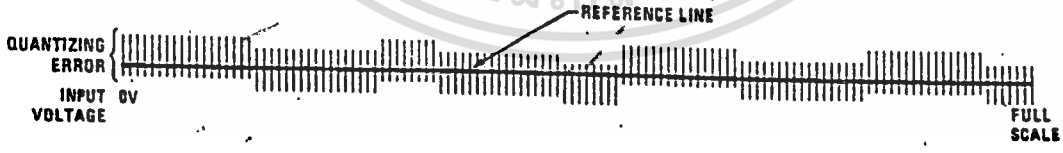


FIGURE 4. Typical Error Curve

Typical Performance Characteristics

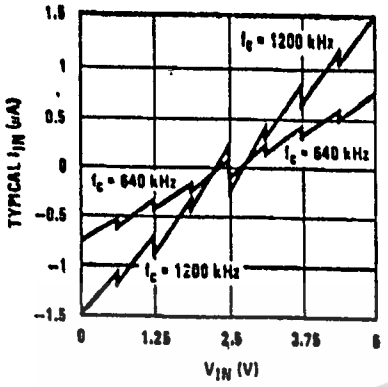


FIGURE 6. Comparator I_{IN} vs V_{IN} ($V_{CC} = V_{REF} = 5V$)

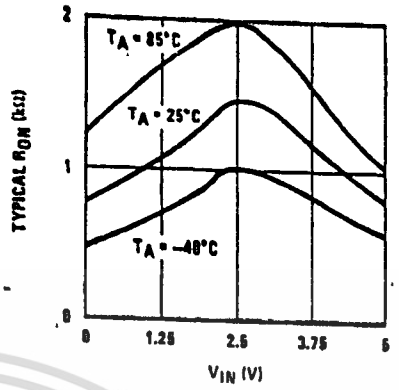


FIGURE 7. Multiplexer R_{ON} vs V_{IN} ($V_{CC} = V_{REF} = 5V$)

TRI-STATE® Test Circuits and Timing Diagrams

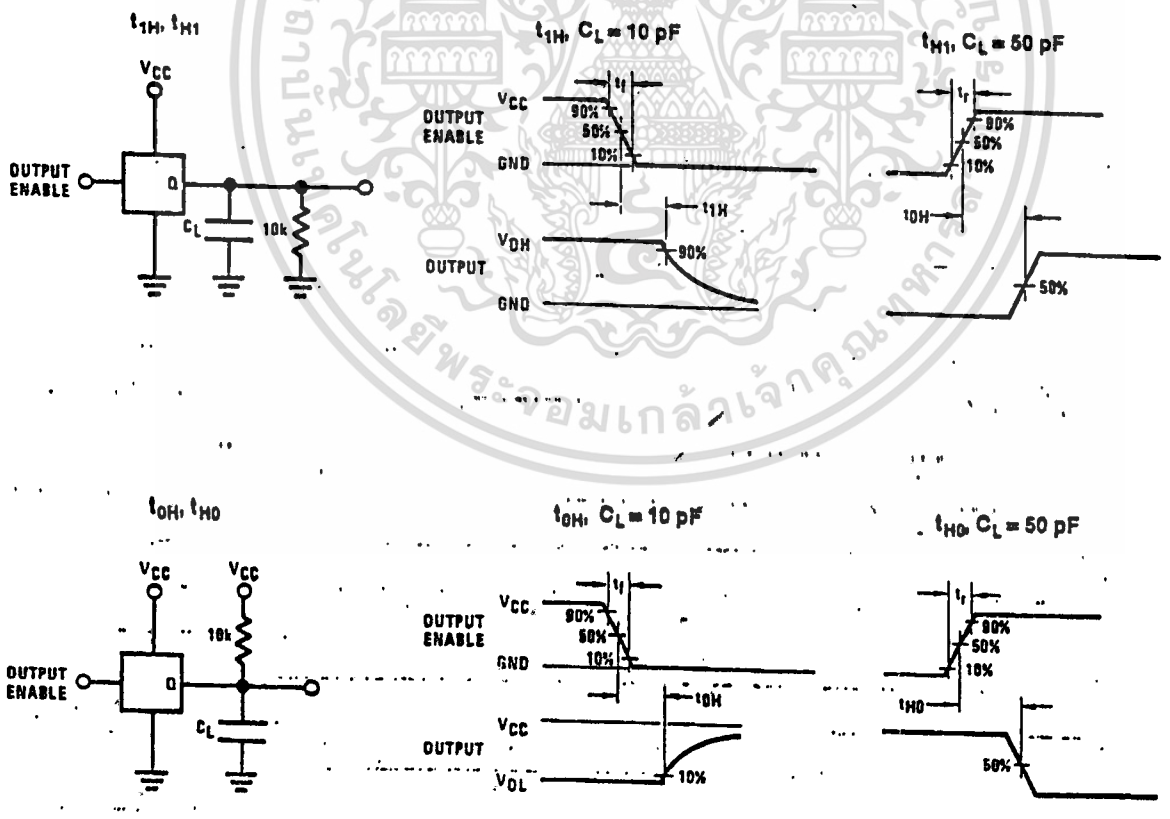


FIGURE 8

Applications Information

OPERATION

1.0 Ratiometric Conversion

The ADC0808, ADC0809 is designed as a complete Data Acquisition System (DAS) for ratiometric conversion systems. In ratiometric systems, the physical variable being measured is expressed as a percentage of full-scale which is not necessarily related to an absolute standard. The voltage input to the ADC0808 is expressed by the equation

$$\frac{V_{IN}}{V_{fs} - V_Z} = \frac{D_X}{D_{MAX} - D_{MIN}} \quad (1)$$

V_{IN} = Input voltage into the ADC0808
V_{fs} = Full-scale voltage
V_Z = Zero voltage
D_X = Data point being measured
D_{MAX} = Maximum data limit
D_{MIN} = Minimum data limit

A good example of a ratiometric transducer is a potentiometer used as a position sensor. The position of the wiper is directly proportional to the output voltage which is a ratio of the full-scale voltage across it. Since the data is represented as a proportion of full-scale, reference requirements are greatly reduced, eliminating a large source of error and cost for many applications. A major advantage of the ADC0808, ADC0809 is that the input voltage range is equal to the supply range so the transducers can be connected directly across the supply and their outputs connected directly into the multiplexer inputs, *(Figure 9)*.

Ratiometric transducers such as potentiometers, strain gauges, thermistor bridges, pressure transducers, etc., are suitable for measuring proportional relationships; however, many types of measurements must be referred to an absolute standard such as voltage or current. This means a system reference must be used which relates the full-scale voltage to the standard volt. For example, if $V_{CC} = V_{REF} = 5.12V$, then the full-scale range is divided into 256 standard steps. The smallest standard step is 1 LSB which is then 20 mV.

2.0 Resistor Ladder Limitations

The voltages from the resistor ladder are compared to the selected input 8 times in a conversion. These voltages are coupled to the comparator via an analog switch tree which is referenced to the supply. The voltages at the top, center and bottom of the ladder must be controlled to maintain proper operation.

The top of the ladder, Ref(+), should not be more positive than the supply, and the bottom of the ladder, Ref(−), should not be more negative than ground. The center of the ladder voltage must also be near the center of the supply because the analog switch tree changes from N-channel switches to P-channel switches. These limitations are automatically satisfied in ratiometric systems and can be easily met in ground referenced systems.

Figure 10 shows a ground referenced system with a separate supply and reference. In this system, the supply must be trimmed to match the reference voltage. For instance, if a 5.12V is used, the supply should be adjusted to the same voltage within 0.1V.

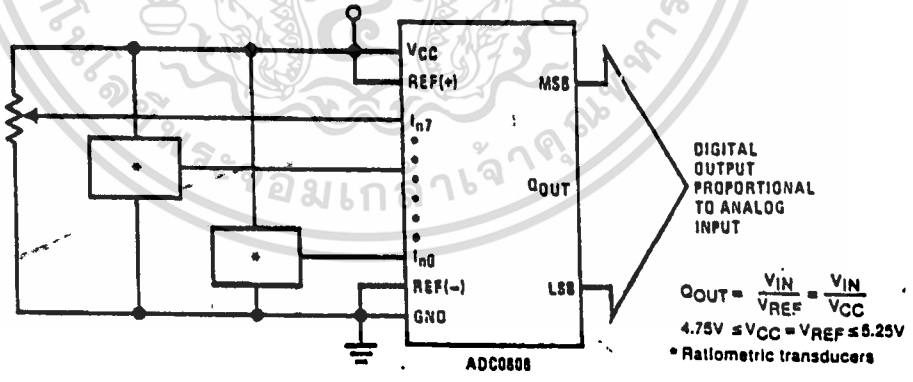


FIGURE 9. Ratiometric Conversion System

Applications Information (Continued)

The ADC0808 needs less than a milliamp of supply current so developing the supply from the reference is readily accomplished. In *Figure 11* a ground referenced system is shown which generates the supply from the reference. The buffer shown can be an op amp of sufficient drive to supply the milliamp of supply current and the desired bus drive, or if a capacitive bus is driven by the outputs a large capacitor will supply the transient supply current as seen in *Figure 12*. The LM301 is overcompensated to insure stability when loaded by the 10 μF output capacitor.

The top and bottom ladder voltages cannot exceed V_{CC} and ground, respectively, but they can be symmetrically less than V_{CC} and greater than ground. The center of the ladder voltage should always be near the center of the supply. The sensitivity of the converter can be increased, (i.e., size of the LSB steps decreased) by using a symmetrical reference system. In *Figure 13*, a 2.5V reference is symmetrically centered about $V_{CC}/2$ since the same current flows in identical resistors. This system with a 2.5V reference allows the LSB bit to be half the size of a 5V reference system.

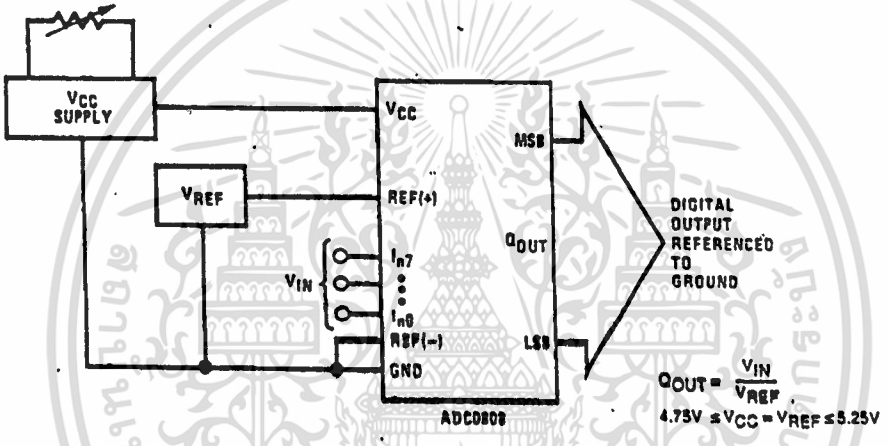


FIGURE 10. Ground Referenced Conversion System Using Trimmed Supply

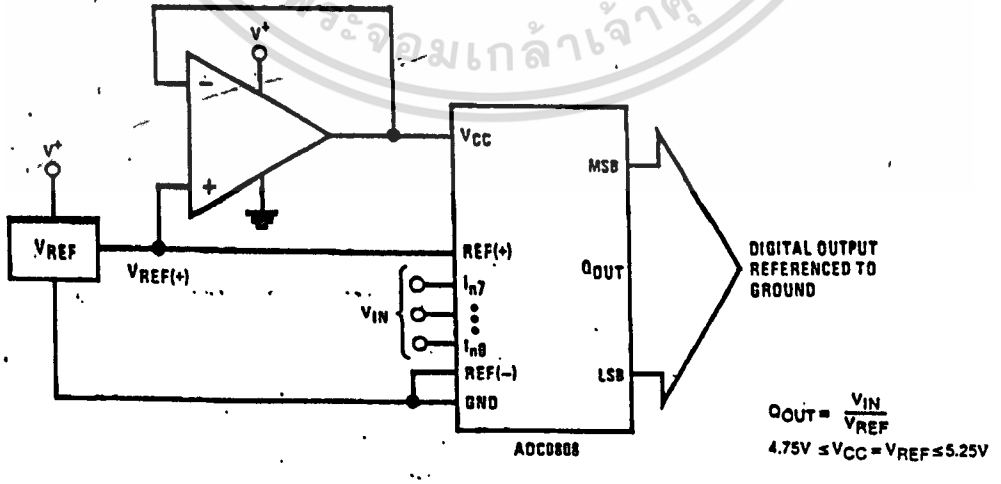


FIGURE 11. Ground Referenced Conversion System with Reference Generating V_{CC} Supply

Applications Information (Continued)

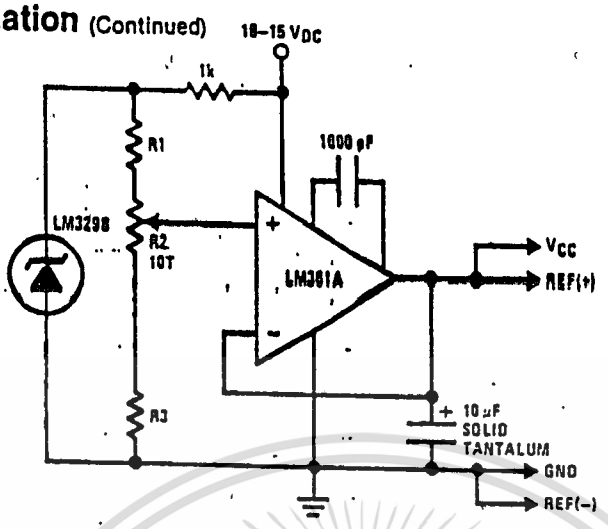


FIGURE 12. Typical Reference and Supply Circuit

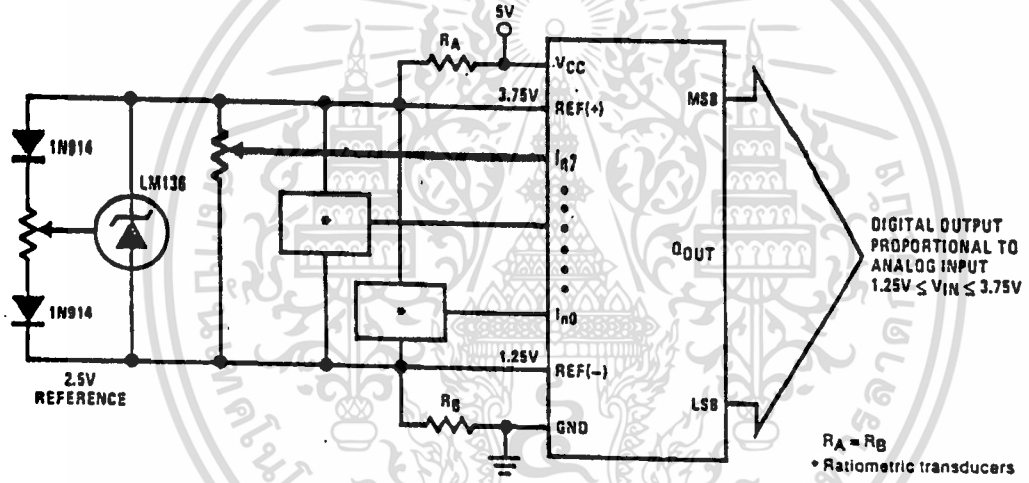


FIGURE 13. Symmetrically Centered Reference

3.0 Converter Equations

The transition between adjacent codes N and N + 1 is given by:

$$V_{IN} = \left\{ (V_{REF(+)} - V_{REF(-)}) \left[\frac{N}{256} + \frac{1}{512} \right] \pm V_{TUE} \right\} + V_{REF(-)} \quad (2)$$

The center of an output code N is given by:

$$V_{IN} = \left\{ (V_{REF(+)} - V_{REF(-)}) \left[\frac{N}{256} \right] \pm V_{TUE} \right\} + V_{REF(-)} \quad (3)$$

The output code N for an arbitrary input are the integers within the range:

$$N = \frac{V_{IN} - V_{REF(-)}}{V_{REF(+)} - V_{REF(-)}} \times 256 \pm \text{Absolute Accuracy} \quad (4)$$

where: V_{IN} = Voltage at comparator input
$V_{REF(+)}$ = Voltage at Ref(+)
$V_{REF(-)}$ = Voltage at Ref(−)
V_{TUE} = Total unadjusted error voltage (typically $V_{REF(+)} \div 512$)

4.0 Analog Comparator Inputs

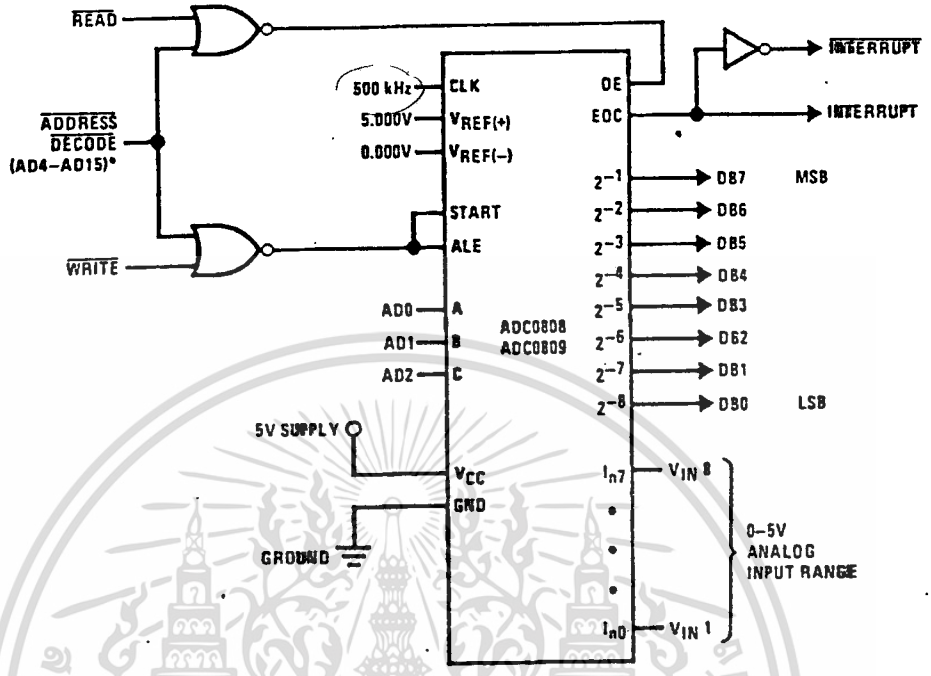
The dynamic comparator input current is caused by the periodic switching of on-chip stray capacitances. These are connected alternately to the output of the resistor ladder/switch tree network and to the comparator input as part of the operation of the chopper stabilized comparator.

The average value of the comparator input current varies directly with clock frequency and with V_{IN} as shown in *Figure 6*.

If no filter capacitors are used at the analog inputs and the signal source impedances are low, the comparator input current should not introduce converter errors, as the transient created by the capacitance discharge will die out before the comparator output is strobed.

If input filter capacitors are desired for noise reduction and signal conditioning they will tend to average out the dynamic comparator input current. It will then take on the characteristics of a DC bias current whose effect can be predicted conventionally.

Typical Application



* Address latches needed for 8085 and SC/MP interfacing the ADC0808 to a microprocessor

MICROPROCESSOR INTERFACE TABLE

PROCESSOR	READ	WRITE	INTERRUPT (COMMENT)
8080	MEMR	MEMW	INTR (Thru RST Circuit)
8085	RD	WR	INTR (Thru RST Circuit)
Z-80	RD	WR	INT (Thru RST Circuit, Mode 0)
SC/MP	NRDS	NWDS	SA (Thru Sense A)
6800	VMA•Φ2•R/W	VMA•Φ2•R/W	IRQA or IRQB (Thru PIA)

Ordering Information

TEMPERATURE RANGE		−40°C to +85°C		−55°C to +125°C
Error	± 1/2 Bit Unadjusted	ADC0808CCN	ADC0808CCJ	ADC0808CJ
	± 1 Bit Unadjusted	ADC0809CCN		
Package Outline		N28A Molded DIP	J28A Hermetic DIP	J28A Hermetic DIP

PRELIMINARY

AUGUST 1988

DATA SHEET 80C51-L / 80C31-L

CMOS SINGLE-CHIP 8 BIT 3V-MICROCONTROLLER

- **80C51-L - CMOS SINGLE-SHIP 8-BIT MICROCONTROLLER with factory mask-programmable ROM**
- **80C31-L - CMOS SINGLE-CHIP 8-BIT CONTROL-ORIENTED CPU with RAM and I/O**
- **80C51-L/C31-L: 0 TO 6 MHz, VCC = 2.7V TO 6V**

FEATURES

- **POWER CONTROL MODES**
- **128 x 8 BIT RAM**
- **32 PROGRAMMABLE I/O LINES**
- **TWO 16-BIT TIMER/COUNTERS**
- **64K PROGRAM MEMORY SPACE**
- **FULLY STATIC DESIGN**
- **HIGH PERFORMANCE SAJI VI CMOS PROCESS**
- **BOOLEAN PROCESSOR**
- **5 INTERRUPT SOURCES**
- **PROGRAMMABLE SERIAL PORT**
- **64K DATA MEMORY SPACE**
- **TEMPERATURE RANGE: 0 TO 70°C**

DESCRIPTION

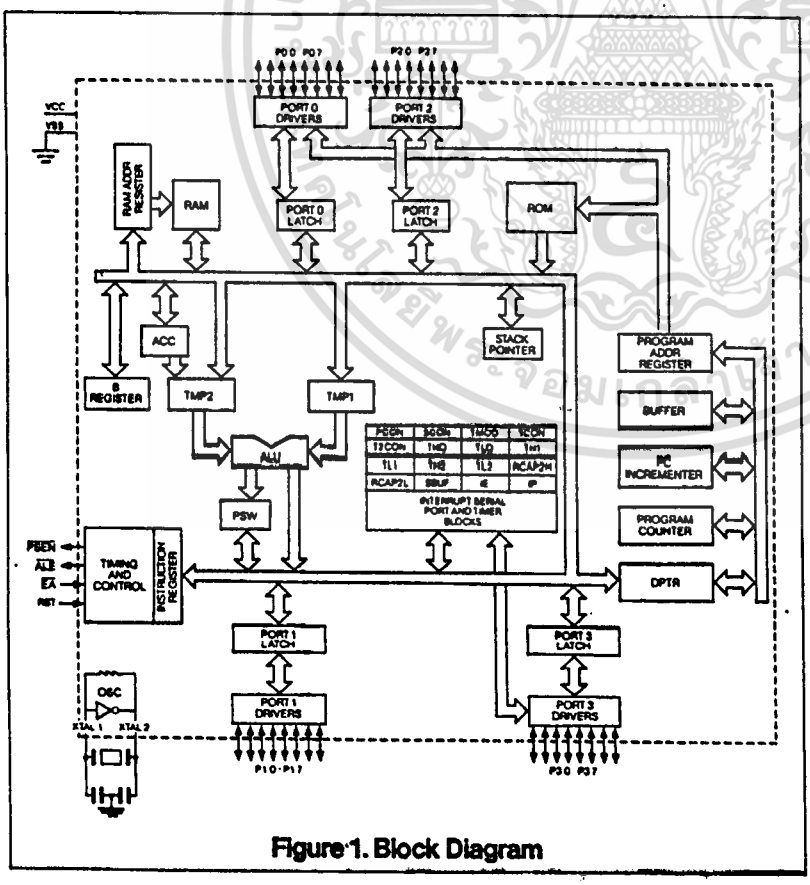


Figure 1. Block Diagram

MHS's 80C51 and 80C31 are high performance CMOS versions of the 8051/8031 NMOS single chip 8 bit μC and is manufactured using a self-aligned silicon gate CMOS process (SAJI VI).

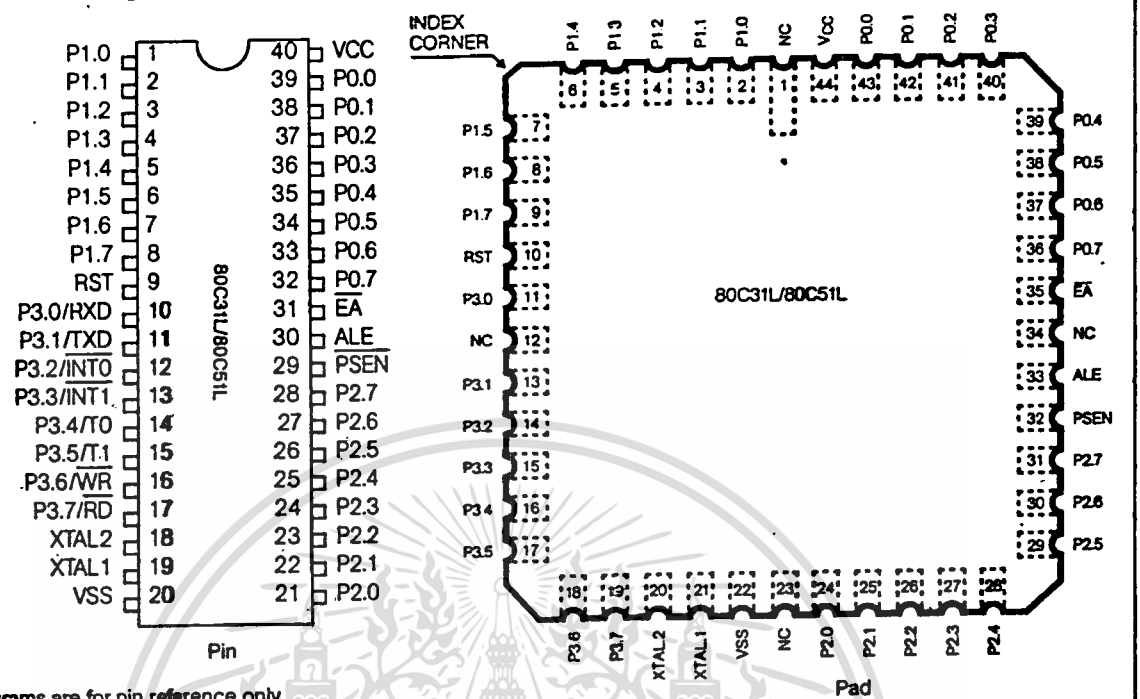
The fully static design of the MHS 80C51/80C31 allows to reduce system power consumption by bringing the clock frequency down to any value, even DC, without loss of data.

The 80C51 retains all the features of the 8051: 4K bytes of ROM; 128 bytes of RAM; 32 I/O lines; two 16 bit timers; a 5-source 2-level interrupt structure; a full duplex serial port; and on-chip oscillator and clock circuits.

In addition, the 80C51 has two software-selectable modes of reduced activity for further reduction in power consumption. In the Idle Mode the CPU is frozen while the RAM, the timers, the serial port, and the interrupt system continue to function. In the Power Down Mode the RAM is saved and all other functions are inoperative.

The 80C31 is identical to the 80C51 except that it has no on-chip ROM.

Figure 2. Configurations



IDLE AND POWER DOWN OPERATION

Figure 3 shows the internal Idle and Power Down clock configuration. As illustrated, Power Down operation stops the oscillator. Idle mode operation allows the interrupt, serial port, and timer blocks to continue to function while the clock to the CPU is gated off. These special modes are activated by software via the Special Function Register, Its hardware address is 87H. PCON is not bit addressable.

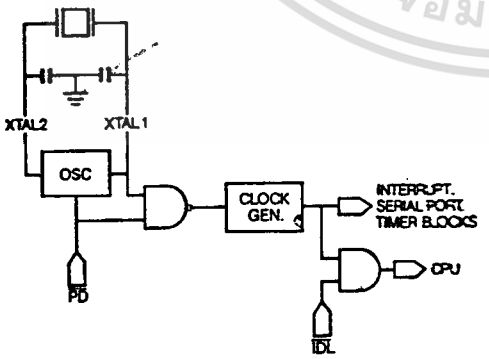


Figure 3. Idle and Power Down Hardware

PCON: Power Control Register

(MSB) (LSB)

SMOD	–	–	–	GF1	GF0	PD	IDL

Symbol	Position	Name and Function
SMOD	PCON.7	Double Baud rate bit. When set to a 1, the baud rate is doubled when the serial port is being used in either modes 1, 2 or 3.
–	PCON.6	(Reserved)
–	PCON.5	(Reserved)
–	PCON.4	(Reserved)
GF1	PCON.3	General-purpose flag bit.
GF0	PCON.2	General-purpose flag bit.
PD	PCON.1	Power Down bit. Setting this bit activates power down operation.
IDL	PCON.0	Idle mode bit. Setting this bit activates idle mode operation.

If 1's are written to PD and IDL at the same time, PD takes precedence. The reset value of PCON is (OXXX0000).

Table 1. Status of the external pins during Idle and Power Down modes

Mode	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Port Data	Port Data	Port Data	Port Data
Idle	External	1	1	Floating	Port Data	Address	Port Data
Power Down	Internal	0	0	Port Data	Port Data	Port Data	Port Data
Power Down	External	0	0	Floating	Port Data	Port Data	Port Data

IDLE MODE

The instruction that sets PCON.0 is the last instruction executed before the Idle mode is activated. Once in the Idle mode the CPU status is preserved in its entirety: the Stack Pointer, Program Counter, Program Status Word, Accumulator, RAM, and all other registers maintain their data during Idle. *Table 1* describes the status of the external pins during Idle mode.

There are two ways to terminate the Idle mode. Activation of any enabled interrupt will cause PCON.0 to be cleared by hardware, terminating Idle mode. The interrupt is serviced, and following RETI, the next instruction to be executed will be the one following theinstruction that wrote a 1 to PCON.0.

The flag bits GF0 and GF1 may be used to determine whether the interrupt was received during normal execution or during the Idle mode. For example, the instruction that writes to PCON.0 can also set or clear one or both flag bits. When Idle mode is terminated by an enabled interrupt, the service routine can examine the status of the flag bits.

The second way of terminating the Idle mode is with a hardware reset. Since the oscillator is still running, the hardware reset needs to be active for only 2 machine cycles (24 oscillator periods) to complete the reset operation.

POWER DOWN MODE

The instruction that sets PCON.1 is the last executed prior to entering power down. Once in power down, the oscillator is stopped. The contents of the onchip RAM and the Special Function Register is saved during power down mode. A hardware reset is the only way of exiting the power down mode. The hardware reset initiates the Special Function Register *(see Table 1)*.

In the Power Down mode. V_{CC} may be lowered to minimize circuit power consumption. Care must be taken to ensure the voltage is not reduced until the power down mode is entered, and that the voltage is restored before the hardware reset is applied which frees the oscillator. Reset should not be released until the oscillator has restarted and stabilized.

Table 1 describes the status of the external pins while in the power down mode. It should be noted that if the power down mode is activated while in external program memory, the port data that is held in the Special Function Register P2 is restored to Port 2. If the data is a 1, the port pin is held high during the power down mode by the strong pullup, T1, shown in *Figure 4*.

STOP CLOCK MODE

Due to static design, the MHS 80C31/C51 clock speed can be reduced until 0 MHz without any data loss in memory or registers. This mode allows step by step utilization, and permits to reduce system power consumption by bringing the clock frequency down to any value. At 0 MHz, the power consumption is the same as in the Power Down Mode.

80C51 I/O PORTS

The I/O port drive of the 80C51 is similar to the 8051. The I/O buffers for Ports 1, 2, and 3 are implemented as shown in *figure 4*.

When the port latch contains a 0, all pFETS in *figure 4* are off while the nFET is turned on. When the port latch makes a 0-to-1 transition, the nFET turns off. The strong pullup pFET, T1, turns on for two oscillator periods, pulling the output high very rapidly. As the output line is drawn high, pFET T3 turns on through the inverter to supply the I_{OH} source current. This inverter and T3 form a latch which holds the 1 and is supported by T2. When Port 2 is used as an address port, for access to external program of data memory, any address bit that contains a 1 will have his strong pullup turned on for the entire duration of the external memory access.

When an I/O pin on Ports 1, 2, or 3 is used as an input, the user should be aware that the external circuit must sink current during the logical 1-to-0 transition. The maximum sink current is specified as I_{TL} under the D.C. Specifications. When the input goes below approximately 2V, T3 turns off to save I_{CC} current. Note, when returning to a logical 1, T2 is the only internal pullup that is on. This will result in a slow rise time if the user's circuit does not force the input line high.

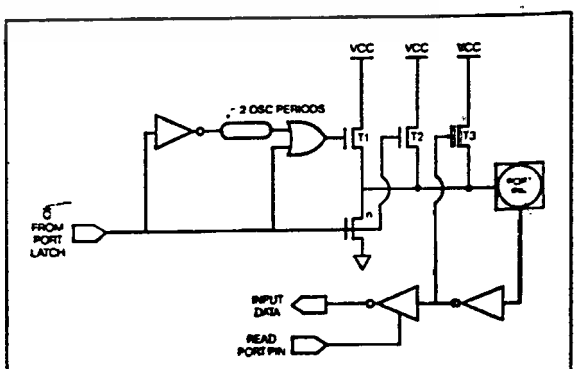


Fig. 4. I/O Buffers in the 80C51 (Ports 1, 2, 3)

80C51 PIN DESCRIPTIONS

VSS
Circuit ground potential

VCC
Supply voltage during normal, Idle, and Power Down operation.

Port 0
Port 0 is an 8-bit open drain bi-directional I/O port. Port 0 pins that have 1's written to them float, and in that state can be used as high-impedance inputs.

Port 0 is also the multiplexed low-order address and data bus during accesses to external Program and Data Memory. In this application it uses strong internal pullups when emitting 1's. Port 0 also outputs the code bytes during program verification in the 80C51. External pullups are required during program verification. Port 0 can sink eight LS TTL inputs.

Port 1
Port 1 is an 8-bit bi-directional I/O port with internal pullups. Port 1 pins that have 1's written to them are pulled high by the internal pullups, and in that state can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (IIL, on the data sheet) because of the internal pullups.

Port 1 also receives the low-order address bytes during program verification. In the 80C51, Port 1 can sink/source three LS TTL inputs. It can drive CMOS inputs without external pullups.

Port 2
Port 2 is an 8-bit bi-directional I/O port with internal pullups. Port 2 pins that have 1's written to them are pulled high by the internal pullups, and in that state can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (IIL, on the data sheet) because of the internal pullups. Port 2 emits the high-order address byte during fetches from external Program Memory and during accesses to external Data Memory that use 16-bit addresses (MOVX @ DPTR). In this application, it uses strong internal pullups when emitting 1's. During accesses to external Data Memory that uses 8-bit addresses (MOVX @ Ri), Port 2 emits the contents of the P2 Special Function Register.

It also receives the high-order address bits and control signals during progam verification in the 80C51. Port 2 can sink/source three LS TTL inputs. It can drive CMOS inputs without external pullups.

Port 3
Port 3 is an 8-bit bi-directional I/O port with internal pullups. Port 3 pins that have 1's written to them are pulled high by the internal pullups, and in that state can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (IIL, on the data sheet) because of the pullups. It also serves the functions of various special features of the MCS-51 Family, as listed below.

Port Pin	Alternate Function
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	INT0 (external interrupt 0)
P3.3	INT1 (external interrupt 1)
P3.4	T0 (Timer 0 external input)
P3.5	T1 (Timer 1 external input)
P3.6	$\overline{WR}$ (external Data Memory write strobe)
P3.7	$\overline{RD}$ (external Data Memory read strobe)

Port 3 can sink/source three LS TTL inputs. It can drive CMOS inputs without external pullups.

RST
A high level on this for two machine cycles while the oscillator is running resets the device. An internal pull-down resistor permits Power-On reset using only a capacitor connected to VCC.

ALE
Address Latch Enable output for latching the low byte of the address during accesses to external memory. ALE is activated as though for this purpose at a constant rate of 1/6 the oscillator frequency except during an external data memory access at which time one ALE pulse is skipped. ALE can sink/source 8 LS TTL inputs. It can drive CMOS inputs without an external pullup.

$\overline{PSEN}$
Program Store Enable output is the read strobe to external Program Memory. PSEN is activated twice each machine cycle during fetches from external Program Memory. (However, when executing out of external Program Memory, two activations of PSEN are skipped during each access to external Data Memory). PSEN is not activated during fetches from internal Program Memory. PSEN can sink/source 8 LS TTL inputs. It can drive CMOS inputs without an external pullup.

$\overline{EA}$
When EA is held high, the CPU executes out of internal Program Memory (unless the Program Counter exceeds OFFFH). When EA is held low, the CPU executes only out of external Program Memory. EA must not be floated.

XTAL1
Input to the inverting amplifier that forms the oscillator. Receives the external oscillator signal when an external oscillator is used.

XTAL2
Output of the inverting amplifier that forms the oscillator, and input to the internal clock generator. This pin should be floated when an external oscillator is used.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output respectively, of an inverting amplifier which is configured for use as an on-chip oscillator, as shown in figure 5. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected as shown in figure 6. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum high and low times specified on the Data Sheet must be observed.

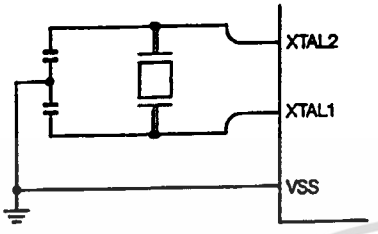


Figure 5. Crystal Oscillator

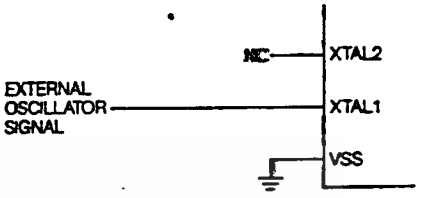


Figure 6. External Drive Configuration

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias:
- Commercial ..... 0°C to 70°C
- Industrial ..... −40°C to 85°C

Storage Temperature ..... −65°C to +150°C
Voltage on V_{CC} to V_{SS} ..... −0.5V to +7V
Voltage on Any Pin to V_{SS} ..... −0.5V to V_{CC} + 0.5V
Power Dissipation 1W*

*This value is based on the maximum allowable die temperature and the thermal resistance of the package.

***NOTICE:**

Stresses at or above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions may affect device reliability.

DC CHARACTERISTICS

TA = −40°C to 85°C; VCc = 2.7V to 6V; VSS = 0V; F = 0 to 6 MHz

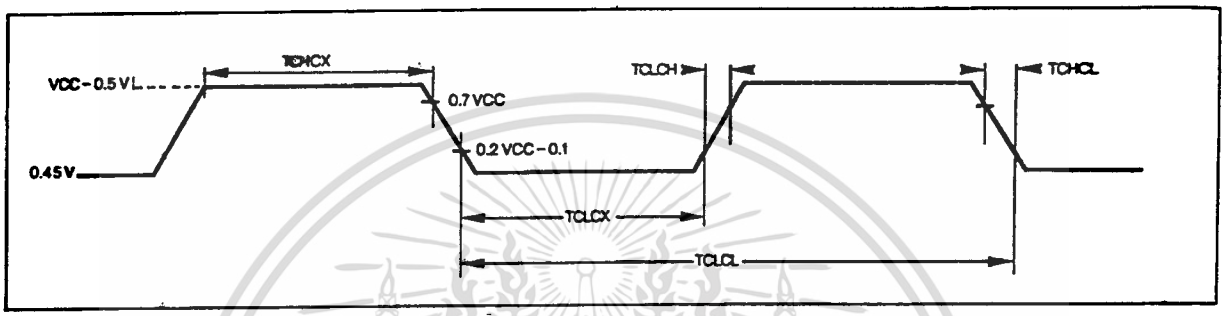
Symbol	Parameter	Min	Max	Unit	Test Conditions
VIL	Input Low Voltage	−0.5	0.2 V_{CC} −0.1	V	
VIH	Input High Voltage (Except XTALs and RST)	0.2 V_{CC} +0.9	V_{CC} +0.5	V	
VIH1	Input High Voltage to RST for Reset	0.7 V_{CC}	V_{CC} +0.5	V	
VIH2	Input High Voltage To XTAL 1	0.7 V_{CC}	V_{CC} +0.5	V	
VPD	Power Down Voltage To V_{CC} in PD Mode	2.0	6.0	V	
VOL	Output Low Voltage (Ports 1, 2, 3)		0.45	V	IOL = 1.6mA (note **1**)
VOL1	Output Low Voltage Port 0, ALE, PSEN		0.45	V	IOL = 3.2mA (note **1**)
VOH	Output High Voltage Ports 1, 2, 3	0.9 V_{CC}		V	IOH = −10μa
		2.4		V	IOH = −60μA V_{CC} = 5V ± 10%
VOH1	Output High Voltage (Port 0 in External in External Bus Mode), ALE, PSEN	0.9 V_{CC}		V	IOH = −40μA
		2.4		V	IOH = −400μA V_{CC} = 5V ± 10%
IIL	Logical 0 Input Current Ports 1,2,3		−50	μA	Vin = 0.45V
ILI	Input Leakage Current		± 10	μA	0.45 < Vin < V_{CC}
ITL	Logical 1 to 0 Transition Current (Ports 1, 2, 3)		−500	μA	Vin = 2.0V
ICCPD	Power Supply Current (Power Down Mode)	50	10	μA	V_{CC} = 2.0V to 5.5V (note 2)
RRST	RST Pulldown Resistor	50	150	kΩ	
CIO	Capacitance of I/O Buffer		10	pF	f_C = 1MHz, T_A = 25°C

Note 1:

Capacitive loading on Ports 0 and 2 may cause spurious noise pulses to be superimposed on the V_{OL}s of ALE and Ports 1 and 3. The noise is due to external bus capacitance discharging into the Port 0 and Port 2 pins when these pins make 1-to-0 transitions during bus operations. In the worst cases (capacitive loading 100 pF), the noise pulse on the ALE line may exceed 0.45V with maxi V_{OL} peak 0.6V. A Schmitt Trigger use is not necessary.

EXTERNAL CLOCK DRIVE CHARACTERISTICS (XTAL 1)

Symbol	Parameter	Variable Clock freq = 0 to 6 MHz		Unit
		Min	Max	
TCLCL	Oscillator Period	166		ns
TCHCX	High Time	20		ns
TCLCX	Low Time	20		ns
TCLCH	Rise Time		20	ns
TCHCL	Fall Time		20	ns



AC CHARACTERISTICS

(T_A = -40°C to 85°C, V_{CC} = 2.7V to 6V, V_{SS} = 0V)
(Load Capacitance for Port 0, ALE, and $\overline{PSEN}$ = 100pf; Load Capacitance for All Other Outputs = 80pf).

EXTERNAL PROGRAM MEMORY CHARACTERISTICS

Symbol	Parameter	Min	Max	Units
TLHLL	ALE Pulse Width	2TCLCL-40		ns
TAVLL	Address Valid to ALE	TCLCL-55		ns
TLLAX	Address Hold After ALE	TCLCL-35		ns
TLLIV	ALE to Valid instr In		4TCLCL-170	ns
TLLPL	ALE to $\overline{PSEN}$	TCLCL-25		ns
TPLPH	$\overline{PSEN}$ Pulse Width	3TCLCL-35		ns
TPLIV	$\overline{PSEN}$ to Valid Instr In		3TCLCL-220	ns
TPXIX	Input Instr Hold After $\overline{PSEN}$	0		ns
TPXIZ	Input Instr Float After $\overline{PSEN}$		TCLCL-20	ns
TPXAV	$\overline{PSEN}$ to Address Valid	TCLCL-8		ns
TAVIV	Address to Valid Instr In		5TCLCL-220	ns
TPLAZ	$\overline{PSEN}$ Low to Address Float		0	ns

See next page for External Data Memory Characteristics.

EXTERNAL DATA MEMORY CHARACTERISTICS

Symbol	Parameter	Min	Max	Units
TRLRH	$\overline{RD}$ Pulse Width	6TCLCL-100		ns
TWLWH	$\overline{WR}$ Pulse Width	6TCLCL-100		ns
TLLAX	Data Address Hold After ALE	TCLCL-35		ns
TRLDV	$\overline{RD}$ to Valid Data In		5TCLCL-165	ns
TRHDX	Data Hold After $\overline{RD}$	0		ns
TRHDZ	Data Float After $\overline{RD}$		2TCLCL-70	ns
TLLDV	ALE to Valid Data In		8TCLCL-150	ns
TAVDV	Address to Valid Data In		9TCLCL-165	ns
TLLWL	ALE to $\overline{WR}$ or $\overline{RD}$	3TCLCL-50	3TCLCL+50	ns
TAVWL	Address to $\overline{WR}$ or $\overline{RD}$	4TCLCL-130		ns
TQVWX	Data Valid to $\overline{WR}$ Transition	TCLCL-60		ns
TQVWH	Data Setup to $\overline{WR}$ High	7TCLCL-150		ns
TWHQX	Data Hold After $\overline{WR}$	TCLCL-50		ns
TRLAZ	$\overline{RD}$ Low to Address Float		0	ns
TWHLH	$\overline{RD}$ or $\overline{WR}$ High to ALE High	TCLCL-40	TCLCL+40	ns

MAXIMUM I_{CC} (mA)

	Operating (Note 3)			Idle (Note 4)		
Freq. V_{CC}	2.7V	5V	6V	2.7V	5V	6V
1 MHz	0.8 mA	1.5 mA	1.8 mA	400 µA	800 µA	1 mA
6 MHz	4 mA	8 mA	10 mA	1.2 mA	3.5 mA	3.8 mA

Note 2:
Power Down I_{CC} is measured will all output pins disconnected; EA = Port 0 = V_{CC}; XTAL2 N.C.; RST = V_{SS}

Note 3:
I_{CC} is measured with all output pins disconnected; XTAL1 driven with TCLCH, TCHCL = 5 ns, $V_{IL} = V_{SS}$ + 0.5V; $V_{IH} = V_{CC}$ − 0.5V; XTAL2 N.C.; EA = RST = Port 0 = V_{CC}. I_{CC} would be slightly higher if a crystal oscillator used.

Note 4:
Idle I_{CC} is measured with all output pins disconnected; XTAL1 driven TCLCH, TCHCL = 5 ns, $V_{IL} = V_{SS}$ + 0.5V; $V_{IH} = V_{CC}$ − 0.5V; XTAL2 N.C.; Port 0 = V_{CC}; EA = RST = V_{SS}.

EXPLANATION OF THE AC SYMBOLS

Each timing symbol has 5 characters. The first character is always a 'T' (stands for time). The other characters, depending on their positions, stand for the name of a signal or the logical status of that signal. The following is a list all the characters and what they stand for.

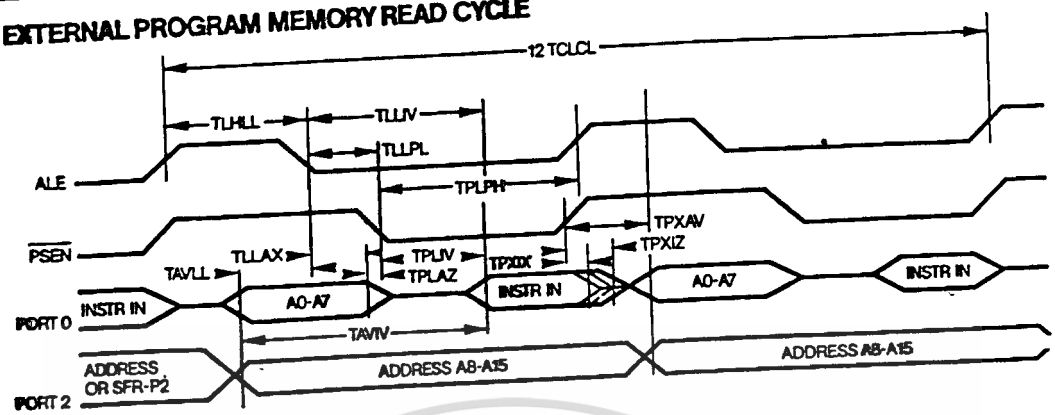
EXAMPLE:

TAVLL = Time for Address Valid to ALE low.
TLLPL = Time for ALE low to $\overline{PSEN}$ low.

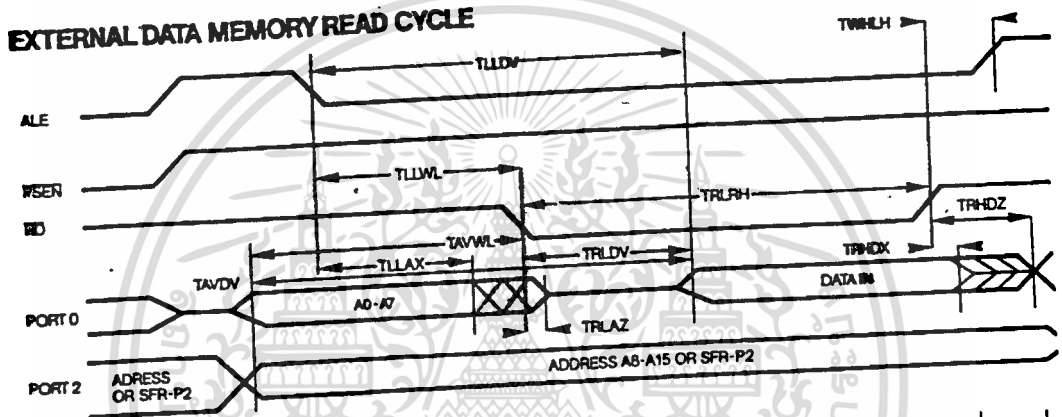
A: Address.
C: Clock.
D: Input data.
H: Logic level HIGH.
I: Instruction (program memory contents).
L: Logic level LOW, or ALE.
P: PSEN
Q: Output data.
R: READ signal.
T: Time.
V: Valid.
W: WRITE signal
X: No longer a valid logic level.
Z: Float.

AC TIMING DIAGRAMS

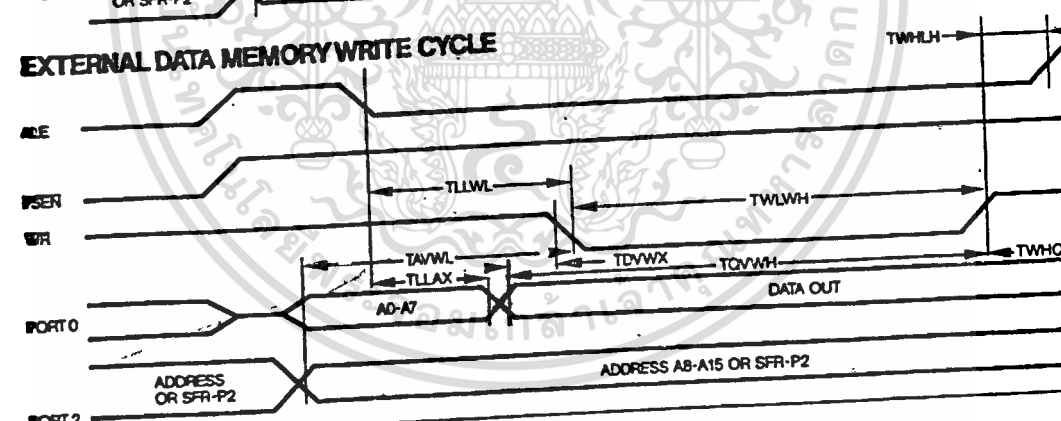
EXTERNAL PROGRAM MEMORY READ CYCLE



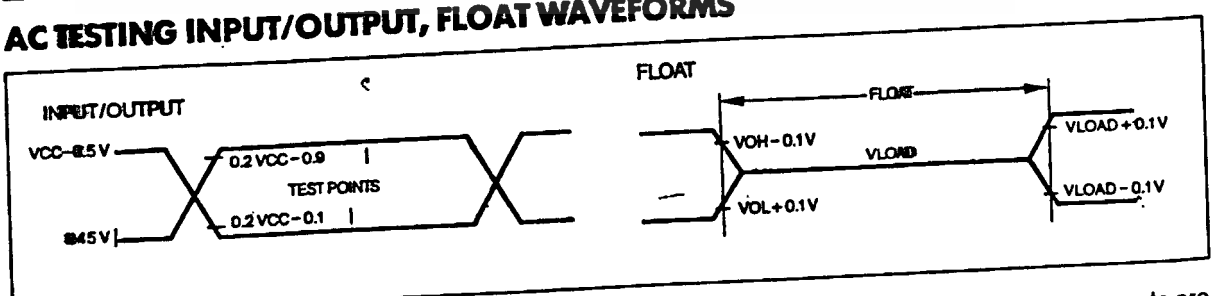
EXTERNAL DATA MEMORY READ CYCLE



EXTERNAL DATA MEMORY WRITE CYCLE



AC TESTING INPUT/OUTPUT, FLOAT WAVEFORMS



AC inputs during testing are driven at VCC – 0.5 for a logic "1" and 0.45V for a logic "0". Timing measurements are made at VIH min for a logic "1" and VIL max for a logic "0". For timing purposes a port pin is no longer floating when a 100 mV change from load voltage occurs and begins to float when a 100 mV change from the loaded VOH/VOL level occurs. IOL/IOH ≥ ± 20 Ma.

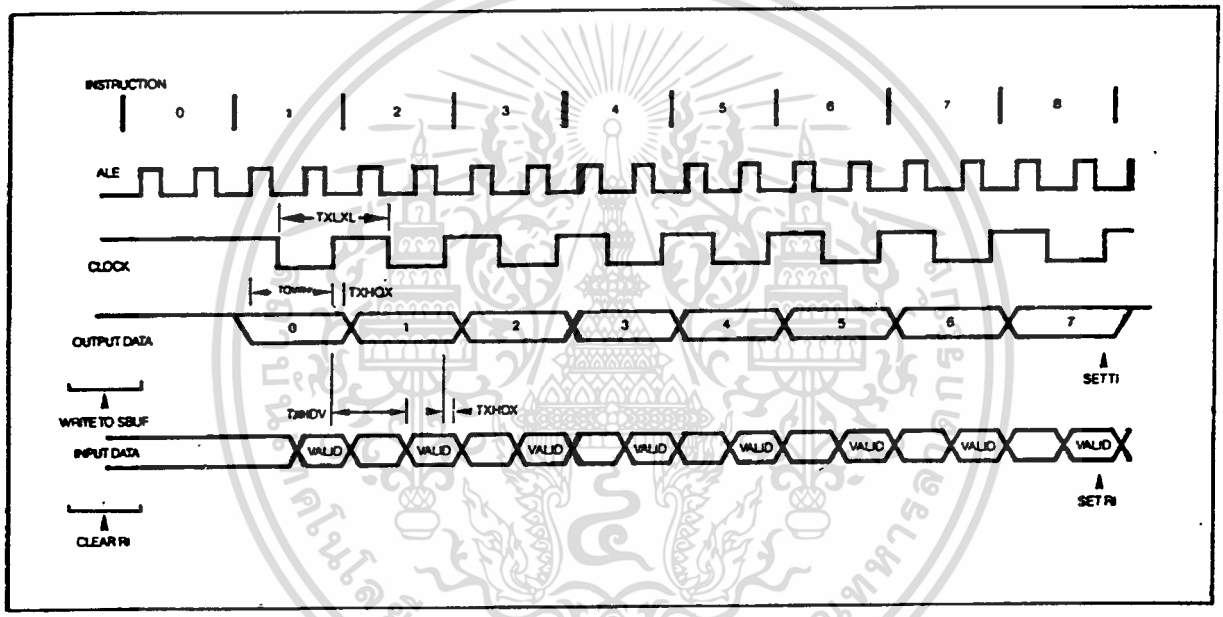
SERIAL PORT TIMING - SHIFT REGISTER MODE

A.C. CHARACTERISTICS:

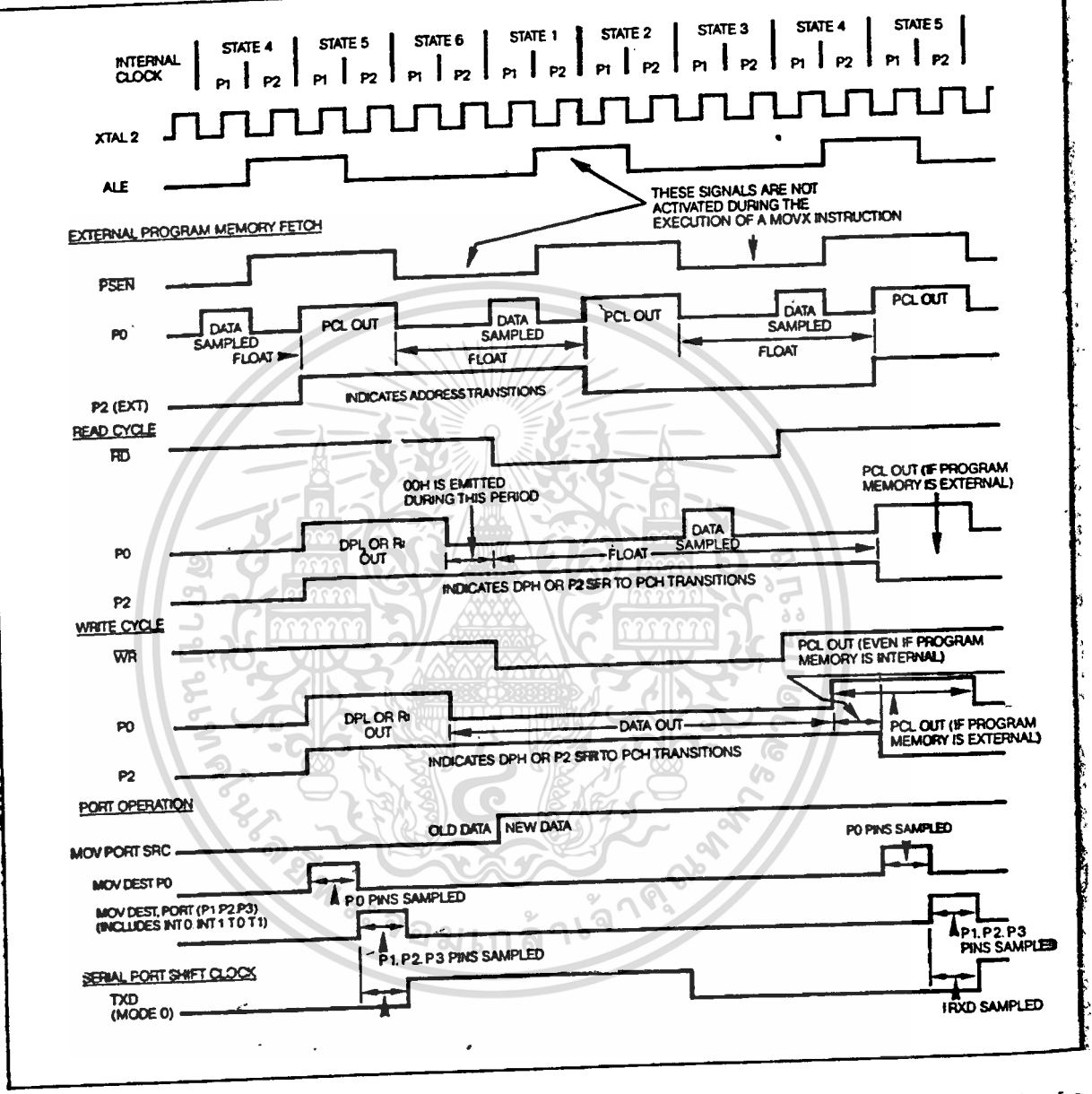
(T_A = 0°C to 70°C; V_{SS} = 0V; V_{CC} = 2.7V to 6V; Load Capacitance = 80 pF)

Symbol	Parameter	Min	Max	Units
TXLXL	Serial Port Clock Cycle Time	12TCLCL		μs
TQVXH	Output Data Setup to Clock Rising Edge	10TCLCL–133		ns
TXHQX	Output Data Hold After Clock Rising Edge	2TCLCL–117		ns
TXHDX	Input Data Hold After Clock Rising Edge	0		ns
TXHDV	Clock Rising Edge to Input Data Valid		10TLCL-133	ns

SHIFT REGISTER TIMING WAVEFORMS



CLOCK WAVEFORMS



This diagram indicates when signals are clocked internally. The time it takes the signals to propagate to the pins, however, ranges from 25 to 125 ns. This propagation delay is dependent on variables such as temperature and pin loading. Propagation also varies from output to output and component. Typically though ($T_A = 25°C$ fully loaded) RD and WR propagation delays are approximately 50 ns. The other signals are typically 85 ns. Propagation delays are incorporated in the AC specifications.

Table 1. (Cont.)

DATA TRANSFER

Mnemonic		Description	Byte	Cyc
MOV	A,Rn	Move register to Accumulator	1	1
MOV	A,direct	Move direct byte to Accumulator	2	1
MOV	A,@Ri	Move indirect RAM to Accumulator	1	1
MOV	A,#data	Move immediate data to Accumulator	2	1
MOV	Rn,A	Move Accumulator to register	1	1
MOV	Rn,direct	Move direct byte to register	2	2
MOV	Rn,#data	Move immediate data to register	2	1
MOV	direct,A	Move Accumulator to direct byte	2	1
MOV	direct,Rn	Move register to direct byte	2	2
MOV	direct,direct	Move direct byte to direct	3	2
MOV	direct,@Ri	Move indirect RAM to direct byte	2	2
MOV	direct,#data	Move immediate data to direct byte	3	2
MOV	@Ri,A	Move Accumulator to indirect RAM	1	1
MOV	@Ri,direct	Move direct byte to indirect RAM	2	2
MOV	@Ri,#data	Move immediate data to indirect RAM	2	1
MOV	DPTR,#data 16	Load Data Pointer with a 16-bit constant	3	2
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to A	1	2
MOVC	A,@A+PC	Move Code byte relative to PC to A	1	2
MOVX	A,@Ri	Move External RAM (8-bit addr) to A	1	2
MOVX	A,@DPTR	Move External RAM (16-bit addr) to A	1	2
MOVX	@Ri,A	Move A to External RAM (8-bit addr)	1	2
MOVX	@DPTR,A	Move A to External RAM (16-bit addr)	1	2
PUSH	direct	Push direct byte onto stack	2	2
POP	direct	Pop direct byte form stack	2	2
XCH	A,Rn	Exchange register with Accumulator	1	1
XCH	A,direct	Exchange direct byte with Accumulator	2	1
XCH	A,@Ri	Exchange indirect RAM with A	1	1
XCHD	A,@Ri	Exchange low-order nibble ind RAM with A	1	1

BOOLEAN VARIABLE MANIPULATION

Mnemonic		Description	Byte	Cyc
CLR	C	Clear Carry flag	1	1
CLR	bit	Clear direct bit	2	1
SETB	C	Set Carry flag	1	1
SETB	bit	Set direct Bit	2	1
CPL	C	Complement Carry flag	1	1
CPL	bit	Complement direct bit	2	1
ANL	C,bit	AND direct bit to Carry flag	2	2
ANL	C,/bit	AND complement of direct bit to Carry	2	2
ORL	C,bit	OR direct bit to Carry flag	2	2
ORL	C,/bit	OR complement of direct bit to Carry	2	2
MOV	C,bit	Move direct bit to Carry flag	2	1
MOV	bit,C	Move Carry flag to direct bit	2	2

PROGRAM AND MACHINE CONTROL

Mnemonic		Description	Byte	Cyc
ACALL	addr 11	Absolute Subroutine Call	2	2
LCALL	addr 16	Long Subroutine Call	3	2
RET		Return from subroutine	1	2
RETI		Return from interrupt	1	2
AJMP	addr 11	Absolute Jump	2	2
LJMP	addr 16	Long Jump	3	2
SJMP	rel	Short Jump (relative addr)	2	2
JMP	@A+DPTR	Jump indirect relative to the DPTR	1	2
JZ	rel	Jump if Accumulator is Zero	2	2
JNZ	rel	Jump if Accumulator is Not Zero	2	2
JC	rel	Jump if Carry flag is set	2	2
JNC	rel	Jump if No Carry flag	2	2

Table 1. MCS®-51 Instruction Set Description

ARITHMETIC OPERATIONS

Mnemonic		Description	Byte	Cyc
ADD	A,Rn	Add register to Accumulator	1	1
ADD	A,direct	Add direct byte to Accumulator	2	1
ADD	A,@Ri	Add indirect RAM to Accumulator	1	1
ADD	A,#data	Add immediate data to Accumulator	2	1
ADDC	A,Rn	Add register to Accumulator with Carry	1	1
ADDC	A,direct	Add direct byte to A with Carry flag	2	1
ADDC	A,@Ri	Add indirect RAM to A with Carry flag	1	1
ADDC	A,#data	Add immediate data to A with Carry flag	2	1
SUBB	A,Rn	Subtract register from A with Borrow	1	1
SUBB	A,direct	Subtract direct byte from A with Borrow	2	1
SUBB	A,@Ri	Subtract indirect RAM from A with Borrow	1	1
SUBB	A,#data	Subtract immed. data from A with Borrow	2	1
INC	A	Increment Accumulator	1	1
INC	Rn	Increment register	1	1
INC	direct	Increment direct byte	2	1
INC	@Ri	Increment indirect RAM	1	1
INC	DPTR	Increment Data Pointer	1	2
DEC	A	Decrement Accumulator	1	1
DEC	Rn	Decrement register	1	1
DEC	direct	Decrement direct byte	2	1
DEC	@Ri	Decrement indirect RAM	1	1
MUL	AB	Multiply A & B	1	4
DIV	AB	Divide A by B	1	4
DA	A	Decimal Adjust Accumulator	1	1

LOGICAL OPERATIONS

Mnemonic		Destination	Byte	Cyc
ANL	A,Rn	AND register to Accumulator	1	1
ANL	A,direct	AND direct byte to Accumulator	2	1
ANL	A,@Ri	AND indirect RAM to Accumulator	1	1
ANL	A,#data	AND immediate data to Accumulator	2	1
ANL	direct,A	AND Accumulator to direct byte	2	1
ANL	direct,#data	AND immediate data to direct byte	3	2
ORL	A,Rn	OR register to Accumulator	1	1
ORL	A,direct	OR direct byte to Accumulator	2	1
ORL	A,@Ri	OR indirect RAM to Accumulator	1	1
ORL	A,#data	OR immediate data to Accumulator	2	1
ORL	direct,A	OR Accumulator to direct byte	2	1
ORL	direct,#data	OR immediate data to direct byte	3	2
XRL	A,Rn	Exclusive-OR register to Accumulator	1	1
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	1
XRL	A,@Ri	Exclusive-OR indirect RAM to A	1	1
XRL	A,#data	Exclusive-OR immediate data to A	2	1
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	1
XRL	direct,#data	Exclusive-OR immediate data to direct	3	2
CLR	A	Clear Accumulator	1	1
CPL	A	Complement Accumulator	1	1
RL	A	Rotate Accumulator Left	1	1
RLC	A	Rotate A Left through the Carry flag	1	1
RR	A	Rotate Accumulator Right	1	1
RRC	A	Rotate A Right through Carry flag	1	1
SWAP	A	Swap nibbles within the Accumulator	1	1

Table 1. (Cont.)

PROGRAM AND MACHINE CONTROL (cont.)

Mnemonic		Description	Byte	Cyc
JB	bit,rel	Jump if direct Bit set	3	2
JNB	bit,rel	Jump if direct Bit Not set	3	2
JBC	bit,rel	Jump if direct Bit is set & Clear bit	3	2
CJNE	A,direct,rel	Compare direct to A & Jump if Not Equal	3	2
CJNE	A,#data,rel	Comp. immed. to A & Jump if Not Equal	3	2
CJNE	Rn,#data.rel	Comp. immed. to reg & Jump if Not Equal	3	2
CJNE	@Ri,#data. rel	Comp. immed. to ind. & Jump if Not Equal	3	2
DJNZ	Rn,rel	Decrement register & Jump if Not Zero	2	2
DJNZ	direct.rel	Decrement direct & Jump if Not Zero	3	2
NOP		No operation	1	1

Notes on data addressing modes:

Rn	– Working register R0-R7
direct	– 128 internal RAM locations, any I/O port, control or status register
@Ri	– Indirect internal RAM location addressed by register R0 or R1
#data	– 8-bit constant included in instruction
#data 16	– 16-bit constant included as bytes 2 & 3 of instruction
bit	– 128 software flags, any I/O pin, control or status bit

Notes on program addressing modes:

addr 16	– Destination address for LCALL & LJMP may be anywhere within the 64-k program memory address space
Addr 11	– Destination address for ACALL & AJMP will be within the same 2-k page of program memory as the first byte of the following instruction
rel	– SJMP and all conditional jumps include an 8-bit offset byte. Range is +127–128 bytes relative to first byte of the following instruction.

Table 2. Instruction Opcodes in Hexadecimal Order

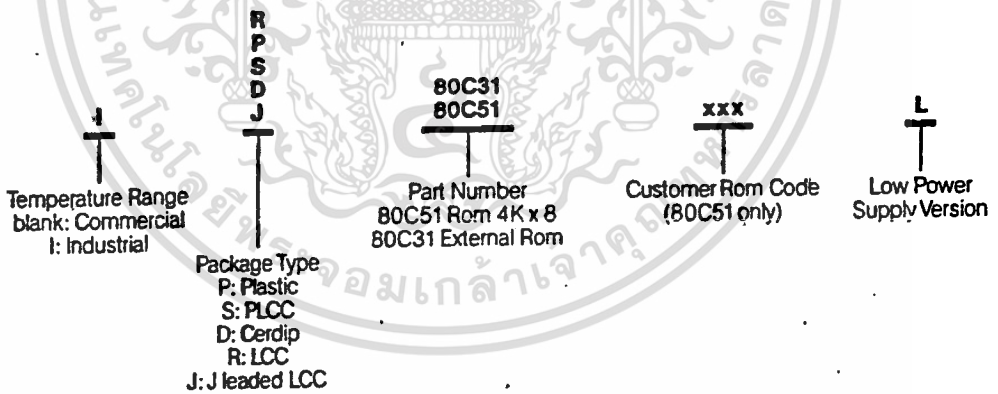
Hex Code	Number of Bytes	Mnemonic	Operands
00	1	NOP	
01	2	AJMP	code addr
02	3	LJMP	code addr
03	1	RR	A
04	1	INC	A
05	2	INC	data addr
06	1	INC	@R0
07	1	INC	@R1
08	1	INC	R0
09	1	INC	R1
0A	1	INC	R2
0B	1	INC	R3
0C	1	INC	R4
0D	1	INC	R5
0E	1	INC	R6
0F	1	INC	R7
10	3	JBC	bit addr,code addr
11	2	ACALL	code addr
12	3	LCALL	code addr
13	1	RRC	A
14	1	DEC	A
15	2	DEC	data addr
16	1	DEC	@R0
17	1	DEC	@R1
18	1	DEC	R0
19	1	DEC	R1
1A	1	DEC	R2
1B	1	DEC	R3
1C	1	DEC	R4
1D	1	DEC	R5
1E	1	DEC	R6
1F	1	DEC	R7
20	3	JB	bit addr,code addr
21	2	AJMP	code addr
22	1	RET	
23	1	RL	A
24	2	ADD	A,data
25	2	ADD	A,data addr
26	1	ADD	A,@R0
27	1	ADD	A,@R1
28	1	ADD	A,R0
29	1	ADD	A,R1
2A	1	ADD	A,R2
2B	1	ADD	A,R3
2C	1	ADD	A,R4
2D	1	ADD	A,R5
2E	1	ADD	A,R6
2F	1	ADD	A,R7
30	3	JNB	bit addr,code addr
31	2	ACALL	code addr
32	1	RETI	
33	1	RLC	A
34	2	ADDC	A,#data
35	2	ADDC	A,data addr
36	1	ADDC	A,@R0
37	1	ADDC	A,@R1
38	1	ADDC	A,R0
39	1	ADDC	A,R1
3A	1	ADDC	A,R2
3B	1	ADDC	A,R3
3C	1	ADDC	A,R4
3D	1	ADDC	A,R5
3E	1	ADDC	A,R6
3F	1	ADDC	A,R7
40	2	JC	code addr
41	2	AJMP	code addr
42	2	ORL	data addr,A
43	3	ORL	data addr,#data
44	2	ORL	A,#data
45	2	ORL	A,data addr
46	1	ORL	A,@R0
47	1	ORL	A,@R1
48	1	ORL	A,R0
49	1	ORL	A,R1
4A	1	ORL	A,R2
4B	1	ORL	A,R3
4C	1	ORL	A,R4
4D	1	ORL	A,R5
4E	1	ORL	A,R6
4F	1	ORL	A,R7
50	2	JNC	code addr
51	2	ACALL	code addr
52	2	ANL	data addr,A
53	3	ANL	data addr,#data
54	2	ANL	A,#data
55	2	ANL	A,data addr
56	1	ANL	A,@R0
57	1	ANL	A,@R1
58	1	ANL	A,R0
59	1	ANL	A,R1
5A	1	ANL	A,R2
5B	1	ANL	A,R3
5C	1	ANL	A,R4
5D	1	ANL	A,R5
5E	1	ANL	A,R6
5F	1	ANL	A,R7
60	2	JZ	code addr
61	2	AJMP	code addr
62	2	XRL	data addr A
63	3	XRL	data addr,#data
64	2	XRL	A,#data
65	2	XRL	A,data addr

Table 2. (Cont.)

Hex Code	Number of Bytes	Mnemonic	Operands
66	1	XRL	A,@R0
67	1	XRL	A,@R1
68	1	XRL	A,R0
69	1	XRL	A,R1
6A	1	XRL	A,R2
6B	1	XRL	A,R3
6C	1	XRL	A,R4
6D	1	XRL	A,R5
6E	1	XRL	A,R6
6F	1	XRL	A,R7
70	2	JNZ	code addr
71	2	ACALL	code addr
72	2	ORL	C,bit addr
73	1	JMP	@A+DPTR
74	2	MOV	A,#data
75	3	MOV	data addr,#data
76	2	MOV	@R0,#data
77	2	MOV	@R1,#data
78	2	MOV	R0,#data
79	2	MOV	R1,#data
7A	2	MOV	R2,#data
7B	2	MOV	R3,#data
7C	2	MOV	R4,#data
7D	2	MOV	R5,#data
7E	2	MOV	R6,#data
7F	2	MOV	R7,#data
80	2	SJMP	code addr
81	2	AJMP	code addr
82	2	ANL	C,bit addr
83	1	MOVC	A,@A+PC
84	1	DIV	AB
85	3	MOV	data addr,data addr
86	2	MOV	data addr,@R0
87	2	MOV	data addr,@R1
88	2	MOV	data addr,R0
89	2	MOV	data addr,R1
8A	2	MOV	data addr,R2
8B	2	MOV	data addr,R3
8C	2	MOV	data addr,R4
8D	2	MOV	data addr,R5
8E	2	MOV	data addr,R6
8F	2	MOV	data addr,R7
90	3	MOV	DPTR,#data
91	2	ACALL	code addr
92	2	MOV	bit addr,C
93	1	MOVC	A,@A+DPTR
94	2	SUBB	A,#data
95	2	SUBB	A,data addr
96	1	SUBB	A,@R0
97	1	SUBB	A,@R1
98	1	SUBB	A,R0
99	1	SUBB	A,R1
9A	1	SUBB	A,R2
9B	1	SUBB	A,R3
9C	1	SUBB	A,R4
9D	1	SUBB	A,R5
9E	1	SUBB	A,R6
9F	1	SUBB	A,R7
A0	2	ORL	C,bit addr
A1	2	AJMP	code addr
A2	2	MOV	C,bit addr
A3	1	INC	DPTR
A4	1	MUL	AB
A5		reserved	
A6	2	MOV	@R0,data addr
A7	2	MOV	@R1,data addr
A8	2	MOV	R0,data addr
A9	2	MOV	R1,data addr
AA	2	MOV	R2,data addr
AB	2	MOV	R3,data addr
AC	2	MOV	R4,data addr
AD	2	MOV	R5,data addr
AE	2	MOV	R6,data addr
AF	2	MOV	R7,data addr
B0	2	ANL	C,bit addr
B1	2	ACALL	code addr
B2	2	CPL	bit addr
B3	1	CPL	C
B4	3	CJNE	A,#data,code addr
B5	3	CJNE	A,data addr,code addr
B6	3	CJNE	@R0,#data,code addr
B7	3	CJNE	@R1,#data,code addr
B8	3	CJNE	R0,#data,code addr
B9	3	CJNE	R1,#data,code addr
BA	3	CJNE	R2,#data,code addr
BB	3	CJNE	R3,#data,code addr
BC	3	CJNE	R4,#data,code addr
BD	3	CJNE	R5,#data,code addr
BE	3	CJNE	R6,#data,code addr
BF	3	CJNE	R7,#data,code addr
C0	2	PUSH	data addr
C1	2	AJMP	code addr
C2	2	CLR	bit addr
C3	1	CLR	C
C4	1	SWAP	A
C5	2	XCH	A,data addr
C6	1	XCH	A,@R0
C7	1	XCH	A,@R1
C8	1	XCH	A,R0
C9	1	XCH	A,R1
CA	1	XCH	A,R2
CB	1	XCH	A,R3

Table 2. (Cont.)

Hex Code	Number of Bytes	Mnemonic	Operands
CC	1	XCH	A,R4
CD	1	XCH	A,R5
CE	1	XCH	A,R6
CF	1	XCH	A,R7
D0	2	POP	data addr
D1	2	ACALL	code addr
D2	2	SETB	bit addr
D3	1	SETB	C
D4	1	DA	A
D5	3	DJNZ	data addr,code addr
D6	1	XCHD	A,@R0
D7	1	XCHD	A,@R1
D8	2	DJNZ	R0,code addr
D9	2	DJNZ	R1,code addr
DA	2	DJNZ	R2,code addr
DB	2	DJNZ	R3,code addr
DC	2	DJNZ	R4,code addr
DD	2	DJNZ	R5,code addr
DE	2	DJNZ	R6,code addr
DF	2	DJNZ	R7,code addr
E0	1	MOVX	A,@DPTR
E1	2	AJMP	code addr
E2	1	MOVX	A,@R0
E3	1	MOVX	A,@R1
E4	1	CLR	A
E5	2	MOV	A,data addr
E6	1	MOV	A,@R0
E7	1	MOV	A,@R1
E8	1	MOV	A,R0
E9	1	MOV	A,R1
EA	1	MOV	A;R2
EB	1	MOV	A,R3
EC	1	MOV	A,R4
ED	1	MOV	A,R5
EE	1	MOV	A,R6
EF	1	MOV	A,R7
F0	1	MOVX	@DPTR,A
F1	2	ACALL	code addr
F2	1	MOVX	@R0,A
F3	1	MOVX	@R1,A
F4	1	CPL	A
F5	2	MOV	data addr,A
F6	1	MOV	@R0,A
F7	1	MOV	@R1,A
F8	1	MOV	R0,A
F9	1	MOV	R1,A
FA	1	MOV	R2,A
FB	1	MOV	R3,A
FC	1	MOV	R4,A
FD	1	MOV	R5,A
FE	1	MOV	R6,A
FF	1	MOV	R7,A



National Semiconductor

Industrial/Automotive/Functional Blocks/Telecommunications

LM135/LM235/LM335, LM135A/LM235A/LM335A Precision Temperature Sensors

General Description

The LM135 series are precision, easily-calibrated, integrated circuit temperature sensors. Operating as a 2-terminal zener, the LM135 has a breakdown voltage directly proportional to absolute temperature at +10 mV/°K. With less than 1Ω dynamic impedance the device operates over a current range of 400 μA to 5 mA with virtually no change in performance. When calibrated at 25°C the LM135 has typically less than 1°C error over a -100°C temperature range. Unlike other sensors the LM135 has a linear output.

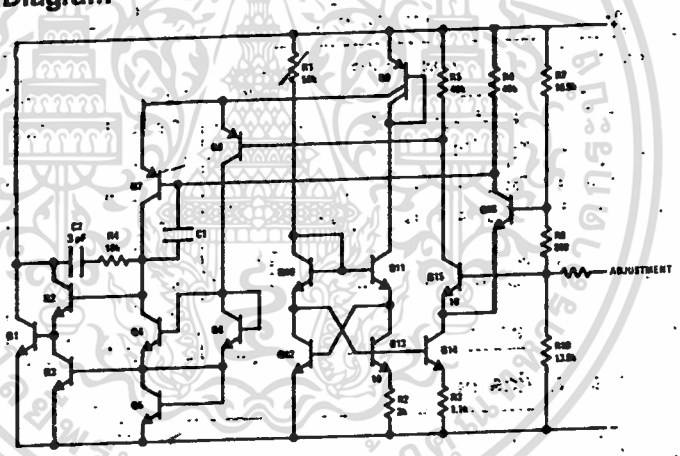
Applications for the LM135 include almost any type of temperature sensing over a −55°C to +150°C temperature range. The low impedance and linear output make interfacing to readout or control circuitry especially easy.

The LM135 operates over a −55°C to +150°C temperature range while the LM235 operates over a −40°C to +125°C temperature range. The LM335 operates from −10°C to +100°C. The LM135/LM235/LM335 are available packaged in hermetic TO-46 transistor packages while the LM235 and LM335 are also available in plastic TO-92 packages.

Features

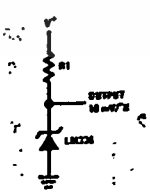
- Directly calibrated in °Kelvin
- 1°C initial accuracy available
- Operates from 400 μA to 5 mA
- Less than 1Ω dynamic impedance
- Easily calibrated
- Wide operating temperature range
- 200°C overrange
- Low cost

Schematic Diagram

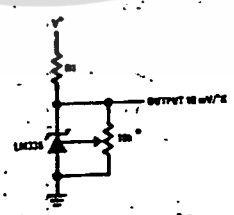


Typical Applications

Basic Temperature Sens

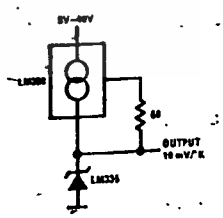


Calibrated Sensor



*Calibrate for 2.982V at 25°C

Wide Operating Supply



Absolute Maximum Ratings

Reverse Current	10 mA
Forward Current	10 mA
Storage Temperature	
TO-46 Package	−60°C to +180°C
TO-92 Package	−60°C to +150°C
Specified Operating Temperature Range	

	Continuous	Intermittent
LM135, LM135A	−55°C to +150°C	150°C to 200°C
LM235, LM235A	−40°C to +125°C	125°C to 150°C
LM335, LM335A	−10°C to +100°C	100°C to 125°C

Lead Temperature (Soldering, 10 seconds) 300°C

Temperature Accuracy LM135/LM235, LM135A/LM235A (Note 1)

PARAMETER	CONDITIONS	LM135A/LM235A			LM135/LM235		
		MIN	TYP	MAX	MIN	TYP	MAX
Operating Output Voltage	$T_C = 25°C$, $I_R = 1$ mA	2.97	2.98	2.99	2.95	2.98	3.01
Uncalibrated Temperature Error	$T_C = 25°C$, $I_R = 1$ mA		0.5	1		1	3
Uncalibrated Temperature Error	$T_{MIN} < T_C < T_{MAX}$, $I_R = 1$ mA		1.3	2.7		2	5
Temperature Error with 25°C Calibration	$T_{MIN} < T_C < T_{MAX}$, $I_R = 1$ mA		0.3	1		0.5	1.5
Calibrated Error at Extended Temperatures	$T_C = T_{MAX}$ (Intermittent)		2			2	
Non-Linearity	$I_R = 1$ mA		0.3	0.5		0.3	1

Temperature Accuracy LM335, LM335A (Note 1)

PARAMETER	CONDITIONS	LM335A			LM335		
		MIN	TYP	MAX	MIN	TYP	MAX
Operating Output Voltage	$T_C = 25°C$, $I_R = 1$ mA	2.95	2.98	3.01	2.92	2.98	3.04
Uncalibrated Temperature Error	$T_C = 25°C$, $I_R = 1$ mA		1	3		2	6
Uncalibrated Temperature Error	$T_{MIN} < T_C < T_{MAX}$, $I_R = 1$ mA		2	5		4	9
Temperature Error with 25°C Calibration	$T_{MIN} < T_C < T_{MAX}$, $I_R = 1$ mA		0.5	1		1	2
Calibrated Error at Extended Temperatures	$T_C = T_{MAX}$ (Intermittent)		2			2	
Non-Linearity	$I_R = 1$ mA		0.3	1.5		0.3	1.5

Electrical Characteristics (Note 1)

PARAMETER	CONDITIONS	LM135/LM235 LM135A/LM235A			LM335 LM335A		
		MIN	TYP	MAX	MIN	TYP	MAX
Operating Output Voltage Change with Current	400 μA < I_R < 5 mA At Constant Temperature		2.5	10		3	14
Dynamic Impedance	$I_R = 1$ mA		0.5			0.6	
Output Voltage Temperature Drift			+10			+10	
Time Constant	Still Air		80			80	
	100 ft/Min Air		10			10	
	Stirred Oil		1			1	
Time Stability	$T_C = 125°C$		0.2			0.2	

Note 1: Accuracy measurements are made in a well-stirred oil bath. For other conditions, self heating must be considered.

Application Hints

CALIBRATING THE LM135

Included on the LM135 chip is an easy method of calibrating the device for higher accuracies. A pot connected across the LM135 with the arm tied to the adjustment terminal allows a 1-point calibration of the sensor that corrects for inaccuracy over the full temperature range.

This single point calibration works because the output of the LM135 is proportional to absolute temperature with the extrapolated output of sensor going to 0V output at 0°K (−273.15°C). Errors in output voltage versus temperature are only slope (or scale factor) so a slope calibration at one temperature corrects at all temperatures.

The output of the device (calibrated or uncalibrated) can be expressed as:

$$V_{OUT_T} = V_{OUT_{T_O}} \times \frac{T}{T_O}$$

where T is the unknown temperature and T_O is a reference temperature, both expressed in degrees Kelvin. By calibrating the output to read correctly at one temperature the output at all temperatures is correct. Nominally the output is calibrated at 10 mV/°K.

To insure good sensing accuracy several precautions must be taken. Like any temperature sensing device, self heating can reduce accuracy. The LM135 should be operated at the lowest current suitable for the application. Sufficient current, of course, must be available to drive both the sensor and the calibration pot at the maximum operating temperature.

If the sensor is used in an ambient where the thermal resistance is constant, self heating errors can be calibrated out. This is possible if the device is run with a temperature stable current. Heating will then be proportional to zener voltage and therefore temperature. This makes the self heating error proportional to absolute temperature the same as scale factor errors.

Typical Applications (Continued)

Minimum Temperature Sensing

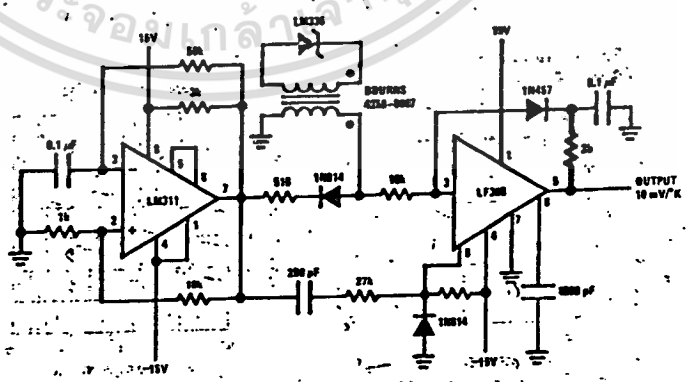
Average Temperature Sensing

Remote Temperature Sensing

Wire length for 1°C error due to wire drop

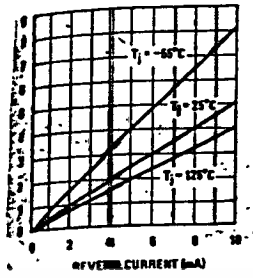
AWG	I_R = 1 mA FEET	I_R = 0.5 mA FEET
14	4000	8000
16	2500	5000
18	1600	3200
20	1000	2000
22	625	1250
24	400	800

Isolated Temperature Sensor

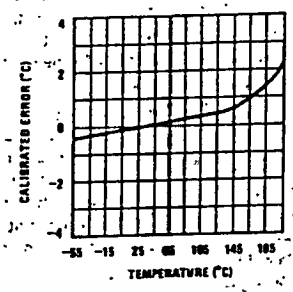


Performance Characteristics

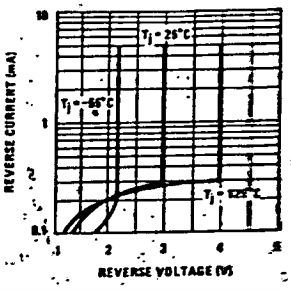
Reverse Voltage Change



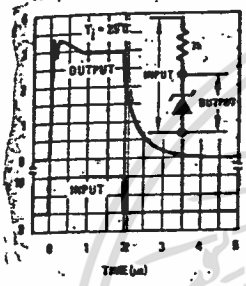
Calibrated Error



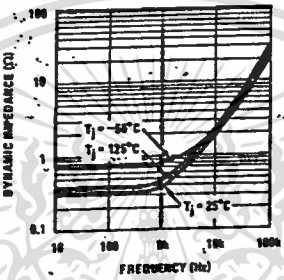
Reverse Characteristics



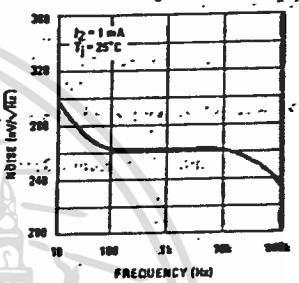
Response Time



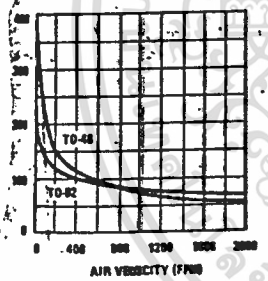
Dynamic Impedance



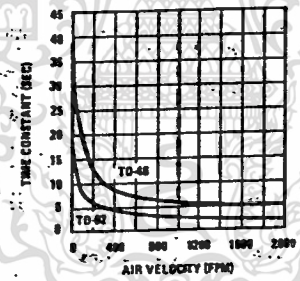
Noise Voltage



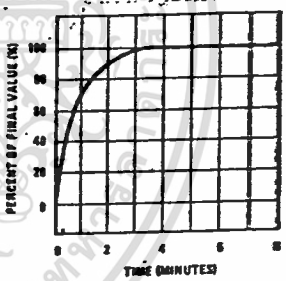
Thermal Resistance Junction to Air



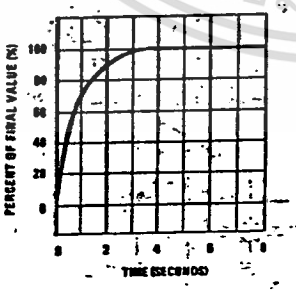
Thermal Time Constant



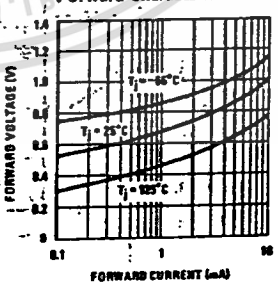
Thermal Response in Still Air



Thermal Response in Stirred Oil Bath



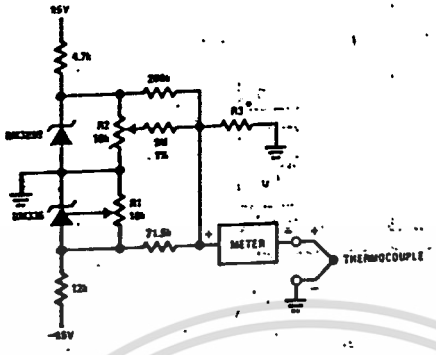
Forward Characteristics



Typical Applications (Continued)

THERMOCOUPLE COLD JUNCTION COMPENSATION

Compensation for Grounded Thermocouple



*Select R3 for proper thermocouple type

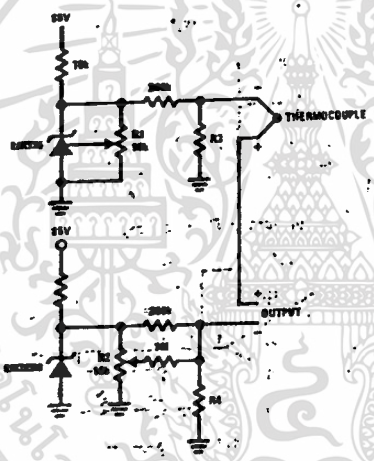
THERMOCOUPLE	R3	SEEBECK COEFFICIENT
J	377Ω	52.3 μV/°C
T	308Ω	42.8 μV/°C
K	293Ω	40.8 μV/°C
S	45.8Ω	6.4 μV/°C

Adjustments: Compensates for both sensor and resistor tolerances

1. Short LM329B
2. Adjust R1 for Seebeck Coefficient times ambient temperature (in degrees K) across R3
3. Short LM335 and adjust R2 for voltage across R3 corresponding to thermocouple type

J	14.32 mV	K	11.17 mV
T	11.79 mV	S	1.768 mV

Single Power Supply Cold Junction Compensation



*Select R3 and R4 for thermocouple type

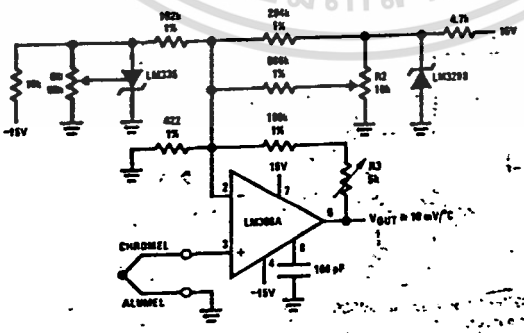
THERMOCOUPLE	R3	R4	SEEBECK COEFFICIENT
J	1.05K	385Ω	52.3 μV/°C
T	856Ω	315Ω	42.8 μV/°C
K	816Ω	300Ω	40.8 μV/°C
S	128Ω	46.3Ω	6.4 μV/°C

Adjustments:

1. Adjust R1 for the voltage across R3 equal to the Seebeck Coefficient times ambient temperature in degrees Kelvin.
2. Adjust R2 for voltage across R4 corresponding to thermocouple

J	14.32 mV	K	11.17 mV
T	11.79 mV	S	1.768 mV

Centigrade Calibrated Thermocouple Thermometer



Terminate thermocouple reference junction in close proximity to LM335.

Adjustments:

1. Apply signal in place of thermocouple and adjust R3 for a gain of 245.7.
2. Short non-inverting input of LM308A and output of LM329B to ground.
3. Adjust R1 so that V_{OUT} = 2.982V @ 25°C.
4. Remove short across LM329B and adjust R2 so that V_{OUT} = 246 mV @ 25°C.
5. Remove short across thermocouple.

Definition of Terms

Operating Output Voltage: The voltage appearing across the positive and negative terminals of the device at specified conditions of operating temperature and current.

Uncalibrated Temperature Error: The error between the operating output voltage at 10 mV/°K and case temperature at specified conditions of current and case temperature.

Calibrated Temperature Error: The error between operating output voltage and case temperature at 10 mV/°K over a temperature range at a specified operating current with the 25°C error adjusted to zero.

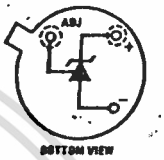
Connection Diagrams

TO-92
Plastic Package

BOTTOM VIEW

Order Number LM235Z, LM335Z
or LM335AZ
See NS Package Z03A

TO-46
Metal Can Package*



BOTTOM VIEW

*Case is connected to negative pin

Order Number LM135H, LM235H,
LM335H, LM135AH, LM235AH
or LM335AH
See NS Package H03H