สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

เครื่องวัดสัญญาณอนาลอกผ่านโครงข่ายอินเตอร์เน็ต

นายชูพงศ์ นพพลกรัง
นายทวีชัย แสงชัยทิพย์
นายสามารถ ไชยมงค์

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต
ภาควิชาฟิสิกส์ประยุกต์
คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2543

เลขหมู่
เลขทะเบียน 40214
วัน, เดือน, ปี 20 ส.ค. 2544

b 11091903
i

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Internet Telemetering Device

Mr. CHUPONG NOPPOLKRANG

Mr. TAWEECHAI SAENGCHAITHIP

Mr. SAMART CHAIYAMONG

A Special Project Submitted in Partial Fulfillment of the

Requirement for the Degree of Bachelor of Science

Department of Applied Physics

Faculty of Science

King Mongkut's Institute of Technology Ladkrabang

2000

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อโครงงานพิเศษ	เครื่องวัดสัญญาณอนาลอกผ่านโครงข่ายอินเตอร์เน็ต
โดย	นายชูพงศ์ นพพลกรัง
	นายทวีชัย แสงชัยทิพย์
	นายสามารถ ไชยมงค์
ภาควิชา	ฟิสิกส์ประยุกต์
อาจารย์ที่ปรึกษา	ผศ. วิชิต ศิริโชติ

ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้นับโครงงานพิเศษฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

.....
(ผศ. วิชาญ เตชิตธีระ)
คณะกรรมการโครงการพิเศษ

หัวหน้าภาควิชา

.....
(ผศ. วิชิต ศิริโชติ)

ประธานกรรมการ

.....
(อ. ภัทรียา กิตติเดชาญ)

กรรมการ

.....
(อ. บดินทร์ ดำรงศักดิ์)

กรรมการ

.....
(อ. ภูมินทร์ จินดาจิธาวัฒน์)

กรรมการ

ลิขสิทธิ์ของภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อโครงงานพิเศษ เครื่องวัดสัญญาณอนาลอกผ่านโครงข่ายอินเตอร์เน็ต

โดย นายชูพงศ์ นพพลกรัง

นายทวีชัย แสงชัยทิพย์

นายสามารถ ไชยมงค์

อาจารย์ที่ปรึกษา ผศ.วิชิต ศิริโชติ

ภาควิชา ฟิสิกส์ประยุกต์

ปีการศึกษา 2543

บทคัดย่อ

โครงงานพิเศษนี้ทำการสร้างเครื่องมือการตรวจวัดอุณหภูมิและแรงดันไฟฟ้าโดยใช้ไมโครคอนโทรลเลอร์ ตระกูล AVR เบอร์ 90S8535 ซึ่งอาศัยวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอลที่อยู่ในตัวไมโครคอนโทรลเลอร์รับค่าสัญญาณและแรงดันที่ได้จากวงจรวัดแรงดันและวงจรเซนเซอร์อุณหภูมิตามลำดับ และเครื่องมือนี้ได้ทำการเชื่อมต่อกับ PC Server ผ่านทางคอมพอร์ต ซึ่ง PC Server จะมีโปรแกรมที่สร้างขึ้นโดยการเขียนโปรแกรม Visual Basic ที่ใช้คอมโพแน็นต์ TCP/IP มาช่วยในการสื่อสารข้อมูลระยะไกลผ่านระบบเครือข่ายอินเตอร์เน็ตไปยังเครื่อง PC Client ซึ่งผู้ใช้งานสามารถทราบข้อมูลของการตรวจวัดผ่านทางโปรแกรม Telnet ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Special Project Title	Internet Telemetering Device
Name	Mr. Chupong Noppolgrang
	Mr. Taweechai Saengchaithip
	Mr. Samart Shaiyamong
Special Project Advisor	Asst. Prof. Wichit Sirichote
Department	Applied Physics
Academic Year	2000

Abstract

A device used for measuring temperature and AC voltage has been built. The control circuit has the AVR 90S8535 microcontroller with 8-channel 10-bit analog-to-digital converter, and sensor circuitry. This device can be connected to PC server through COM port. The PC server runs a program written by Visual basic using TCP/IP components thus providing remote access via Internet. User can access the data converted by ADC circuit through remote PC Client by using Telnet program.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

โครงงานนี้สำเร็จลุล่วงได้ด้วยดี เนื่องด้วยความอนุเคราะห์จากหลายๆฝ่ายด้วยกัน ดังนี้

บิดาและมารดา	ผู้ให้กำเนิด คอยอุปการะเลี้ยงดู เป็นผู้ให้กำลังใจ และอยู่เคียงข้างเราตลอดมา
ผศ. วิชิต ศิริโชติ	ผู้ให้โอกาส ให้คำปรึกษาและชี้แนะแนวทางในการทำโครงงาน
ผศ. อนุพงศ์ สรงประภา	ผู้ที่เอื้อเฟื้ออุปกรณ์ในการทดลอง
นายอนันตชัย นัยจิตร	ผู้ที่ช่วยให้คำปรึกษาในโครงงาน และให้ความอนุเคราะห์ในการใช้เครื่องมือ
นายชวัช เพลินประภาพร	ผู้ที่เอื้อเฟื้ออุปกรณ์ที่ใช้ในการตรวจวัดอุณหภูมิ
นายนิรันดร์ ลัดนากุล	ผู้ให้คำปรึกษาในการเขียนโปรแกรม Visual Basic
นายเบญจพล ตันฮู้	ผู้ให้คำปรึกษาในการเขียนโปรแกรม Visual Basic
เพื่อนๆ รุ่น 16	ที่คอยให้คำปรึกษา และกำลังใจในการทำโครงงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

		หน้า
บทคัดย่อภาษาไทย		ก
บทคัดย่อภาษาอังกฤษ		ข
กิตติกรรมประกาศ		ค
สารบัญเรื่อง		ง
สารบัญตาราง		ฉ
สารบัญรูปภาพ		ช
สารบัญกราฟ		ซ
บทที่ 1	การสื่อสารระยะไกลผ่านอินเตอร์เน็ต	1
	1.1 ที่มาของโครงงาน	1
	1.2 จุดประสงค์โครงงาน	2
	1.3 วิธีการดำเนินงาน	3
	1.4 ประโยชน์ที่ได้รับ	3
บทที่ 2	ทฤษฎีและหลักการ	4
	2.1 TCP/IP	4
	2.2 AVR AT90S8535 8 บิต	6
	2.3 Com Port	17
	2.4 Parallel Port	23
	2.5 เซนเซอร์อุณหภูมิ	24
	2.6 ตัวแปลงสัญญาณอนาลอกเป็นดิจิตอล	24
	2.7 การจัดสัญญาณคลื่นด้วยไดโอด	26
	2.8 ออปแอมป์	28
	2.9 Visual Basic	28
	2.10 Telnet	29
บทที่ 3	การดำเนินการวิจัย	31
	3.1 การออกแบบวงจรในส่วนของฮาร์ดแวร์ (Hardware)	31
	3.2 การออกแบบวงจรส่วนของซอฟต์แวร์(Software)ที่ใช้กับไมโครคอนโทรลเลอร์	35

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

3.3 การออกแบบในส่วนที่ติดต่อสื่อสารผ่านระบบเครือข่ายอินเตอร์เน็ต 37
บทที่ 4 ผลการทดลอง 42
4.1 ผลการวัดแรงดันไฟฟ้า 42
4.2 ผลการวัดอุณหภูมิ 44
บทที่ 5 สรุปผลการทดลอง และแนวทางการพัฒนา 50

ภาคผนวก ก โปรแกรมการทำงาน
ภาคผนวก ข รายละเอียดอุปกรณ์
เอกสารอ้างอิง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

หน้า

1. ตารางที่ 1 แสดงการวัดค่าแรงดันไฟฟ้า (โวลท์) 41
2. ตารางที่ 2 แสดงการวัดค่าอุณหภูมิ (องศาเซลเซียส) 42

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

		หน้า
1. รูปภาพที่ 1.1	รูปแบบการสื่อสาร	2
2. รูปภาพที่ 2.1	TCP/IP Stack และ OSI Model	5
3. รูปภาพที่ 2.2	บล็อกไดอะแกรมของAVR	7
4. รูปภาพที่ 2.3	บล็อกไดอะแกรมของหน่วยความจำ	9
5. รูปภาพที่ 2.4	โครงสร้างของรีจิสเตอร์ใช้งานทั่วไป	10
6. รูปภาพที่ 2.5	แสดงรีจิสเตอร์ X, Y และ Z	10
7. รูปภาพที่ 2.6	โครงสร้างวงจรรีเซ็ต	11
8. รูปภาพที่ 2.7	โครงสร้างของ SPI	13
9. รูปภาพที่ 2.8	บล็อกไดอะแกรมของวงจรแปลงสัญญาณ ADC	14
10. รูปภาพที่ 2.9	โครงสร้างของไทม์เมอร์/เคาน์เตอร์ 0	15
11. รูปภาพที่ 2.10	โครงสร้างของไทม์เมอร์/เคาน์เตอร์ 1	16
12. รูปภาพที่ 2.11	ลักษณะของSerial Port	17
13. รูปภาพที่ 2.12	ลักษณะของParallel Port	23
14. รูปภาพที่ 2.13	กราฟแสดงคุณสมบัติของวงจรเอดีซีขนาด 3 บิต	25
15. รูปภาพที่ 2.14	แสดงการทำงานของวงจรเอดีซีแบบประมาณค่าต่อเนื่อง	26
16. รูปภาพที่ 2.15	แสดงลักษณะของวงจรแคลมป์	27
17. รูปภาพที่ 2.16	แสดงลักษณะของวงจรแคลมป์ที่มีการไบอัส	27
18. รูปภาพที่ 3.1	แสดงการออกแบบวงจรที่ใช้ในการทดลอง	33
19. รูปภาพที่ 3.2	แสดงบอร์ดทดลอง	34
20. รูปภาพที่ 3.3	แสดงการต่อสายสัญญาณ ISP	35
21. รูปภาพที่ 3.4	แสดงโฟว์ชาร์ทของโปรแกรมภาษา C	36
22. รูปภาพที่ 3.5	แสดงหน้าจอ Terminal	38
23. รูปภาพที่ 3.6	แสดง From Properties	39
24. รูปภาพที่ 3.7	แสดง From Login	40
25. รูปภาพที่ 3.8	แสดง From Command	40
26. รูปภาพที่ 3.9	แสดงโฟว์ชาร์ทของโปรแกรม Visual Basic	41
27. รูปภาพที่ 4.1	การทดลองวัดแรงดันไฟฟ้า	42
28. รูปภาพที่ 4.2	การทดลองวัดอุณหภูมิ	44

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญกราฟ

หน้า

1. กราฟที่ 1 แสดงการเปลี่ยนแปลงของอุณหภูมิ 49

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่1
บทนำ

1.1 ที่มาของโครงงาน

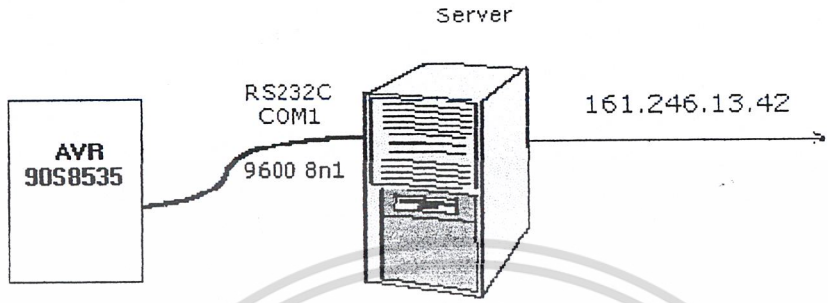
ในการศึกษาการประยุกต์ใช้อุปกรณ์ทางอิเล็กทรอนิกส์มาช่วยเพิ่มความสะดวกสบายในการทำงาน หรือในชีวิตประจำวันเริ่มมีมากขึ้น โดยมีการประดิษฐ์อุปกรณ์ทางอิเล็กทรอนิกส์ที่สามารถควบคุมได้ในหลายๆลักษณะ ซึ่งนิยมใช้ไมโครคอนโทรลเลอร์เข้ามามีส่วนช่วยในการควบคุม ซึ่งเราสามารถที่จะเขียนโปรแกรมไว้ภายในไมโครคอนโทรลเลอร์นี้ แล้วนำมาใช้ให้เข้ากับงานที่เราต้องการได้ ซึ่งในที่นี้เราได้นำไมโครคอนโทรลเลอร์มาใช้ในการตรวจวัดอุณหภูมิและแรงดันไฟฟ้า เป็นการยกตัวอย่างซึ่งเราอาจจะนำไมโครคอนโทรลเลอร์ไปประยุกต์ใช้งานได้กว้างกว่านี้อีกมาก

เนื่องจากว่าถึงแม้เราจะสามารถนำเอาไมโครคอนโทรลเลอร์มาประยุกต์ใช้กับงานได้ในหลายๆด้านก็ตาม แต่ไมโครคอนโทรลเลอร์เองก็มีข้อจำกัดในหลายๆด้าน ซึ่งทำให้ในการใช้งานอาจจะไม่สะดวกนัก เช่น ไมโครคอนโทรลเลอร์ที่เราทำขึ้นมาอาจจะมีขนาดใหญ่ ซึ่งจะทำให้ไม่สะดวกในการพกพาไปตามสถานที่ต่างๆได้ อีกอย่างข้อจำกัดในการควบคุมเราอาจจะต้องมาควบคุมที่ไมโครคอนโทรลเลอร์โดยตรงซึ่งบางครั้งเราอาจจะไม่สะดวกมาควบคุมจากสถานที่ไกลๆ

จากข้อจำกัดบางอย่างที่กล่าวในข้างต้นทำให้คณะผู้จัดทำทำการเชื่อมต่อ (Interface) บอร์ดที่มีไมโครคอนโทรลเลอร์เข้ากับ Computer ทาง Com Port ตามมาตราฐาน RS-232 เพื่อที่จะเพิ่มความสะดวกในการควบคุมการทำงานของไมโครคอนโทรลเลอร์ และทำการเขียนโปรแกรมเพื่อทำการ Interface ผ่านระบบเครือข่ายอินเตอร์เน็ตเพื่อเพิ่มขีดความสามารถในการรับ-ส่งข้อมูลหรือควบคุมไมโครคอนโทรลเลอร์ได้สะดวกยิ่งขึ้น ซึ่งในปัจจุบันระบบเครือข่ายอินเตอร์เน็ตมีบทบาทมากในปัจจุบัน ทางคณะผู้จัดทำจึงนำมาประยุกต์ใช้กับไมโครคอนโทรลเลอร์ ซึ่งในที่นี้จะเป็นการควบคุม และรับ-ส่งข้อมูลในการวัดอุณหภูมิ และแรงดัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงงานชิ้นนี้จะอาศัยโครงข่ายอินเตอร์เน็ตเป็นตัวกลางในการสื่อสาร ดังรูปที่ 1.1



รูปที่1.1 รูปแบบการสื่อสาร

จากรูป เราให้เครื่องคอมพิวเตอร์เป็น เซิร์ฟเวอร์ (SERVER) ทำการ Interface กับไมโครคอนโทรลเลอร์แล้วทำการเชื่อมต่อผ่าน อินเตอร์เน็ต เกตเวย์ (internet gateway) ไมโครคอนโทรลเลอร์ทำการรับและส่งข้อมูลหรือคำสั่งผ่านทางพอร์ตอนุกรม (serial port com1)จากเครื่องคอมพิวเตอร์ที่มีเลขหมายไอพี แอดเดรส(IP Address)ประจำเครื่อง โดยวิธีนี้เราจะสามารถทำการติดต่อเครื่องคอมพิวเตอร์ได้ทุกที่โดยผ่านโครงข่ายอินเตอร์เน็ตด้วยการใช้โปรแกรมเทลเน็ต (telnet)ใส่ค่าหมายเลขไอพีที่ทราบ ซึ่งวิธีการนี้เป็นความคิดเริ่มต้นที่นำไปสู่การพัฒนาวงจรต่างๆ เช่น Telemetering , Remote control ,Alert signal เป็นต้น

1.2 จุดประสงค์ของโครงงาน

- เพื่อทำศึกษาการทำงานและโครงสร้างของไมโครคอนโทรลเลอร์ ชนิด AVR 90S8535
- เพื่อทำการศึกษาการสื่อสารรับและส่งผ่านข้อมูลโดยผ่านโครงข่ายอินเตอร์เน็ต
- เพื่อเสริมสร้างทักษะในการสร้างอุปกรณ์ที่ใช้ในการสื่อสาร และควบคุม
- เพื่อเป็นการศึกษาลักษณะการเขียนโปรแกรมที่ใช้ในการควบคุมไมโครคอนโทรลเลอร์ชนิด AVR 90S8535 ซึ่งเป็นการใช้ภาษา C และการเขียนโปรแกรม
- ใช้ในการ Interface ที่สามารถมาควบคุมไมโครคอนโทรลเลอร์ ซึ่งในที่นี้คือโปรแกรม Visual Basic

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.3 วิธีการดำเนินงาน

- ศึกษาไมโครคอนโทรลเลอร์ชนิด AVR 90S8535 โดยเฉพาะส่วน ADC
- ทำการต่อวงจรอย่างง่ายๆ โดยการใช้ไมโครคอนโทรลเลอร์ชนิด AVR 90S8535 เป็นตัวควบคุมการทำงาน
- ศึกษาโปรแกรมที่นำมาใช้ในการดำเนินงาน ได้แก่ โปรแกรมภาษา C และโปรแกรม Visual Basic
- ศึกษา Sensor ที่เรานำมาใช้ในการวัดอุณหภูมิคือ LM 35D
- ทำการออกแบบวงจรบอร์ดวัดอุณหภูมิ และแรงดันไฟฟ้า
- ทำการเขียนโปรแกรมภาษา C ที่ใช้ในการควบคุมการทำงานผ่านในตัวไมโครคอนโทรลเลอร์ชนิด AVR 90S8535
- ทดลองใช้บอร์ดในการวัดอุณหภูมิ และแรงดันไฟฟ้า
- ศึกษาโปรแกรม Visual Basic เพื่อที่ทำการ Interface ผ่านคอมพิวเตอร์ในการทำงานร่วมกับบอร์ดไมโครคอนโทรลเลอร์ และส่งที่ทำการสื่อสารรับ-ส่งข้อมูลผ่านระบบเครือข่ายอินเตอร์เน็ต ซึ่งในที่นี้จะเป็นการอาศัยผ่านโปรแกรมพื้นฐาน Telnet
- เขียนโปรแกรม Visual Basic ที่ใช้ในการควบคุมบอร์ดไมโครคอนโทรลเลอร์ชนิด AVR 90S8535 ที่เชื่อมต่อกับคอมพิวเตอร์ และสามารถรับ-ส่งข้อมูลผ่านระบบเครือข่ายอินเตอร์เน็ตทาง Telnet ได้
- ทำการทดลองวัดอุณหภูมิและแรงดันไฟฟ้าโดยบอร์ดไมโครคอนโทรลเลอร์ชนิด AVR 90S8535 โดยโปรแกรมที่ทำหน้าที่เป็น Server และรับ-ส่งไปยังเครื่องคอมพิวเตอร์อีกเครื่องที่ทำหน้าที่เป็นเครื่อง Client ผ่านทาง Telnet

1.4 ประโยชน์ที่ได้รับ

- เป็นหลักการพื้นฐานในการประยุกต์ใช้ไมโครคอนโทรลเลอร์มาช่วยทำงาน หรือเพิ่มความสะดวกสบายในชีวิตประจำวัน
- เราสามารถนำบอร์ดไปใช้ในการวัดค่าอุณหภูมิ และแรงดันไฟฟ้า ณ ตำแหน่งต่างๆ และยังสามารถที่จะดูข้อมูลได้โดยผ่านคอมพิวเตอร์ และโปรแกรม Telnet จากที่ใดก็ได้ที่มีการเชื่อมต่อผ่านอินเตอร์เน็ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่2
ทฤษฎีที่เกี่ยวข้อง

2.1 TCP/IP

TCP/IP เป็นมาตรฐานของการรับส่งข้อมูลระหว่างคอมพิวเตอร์ 2 ระบบ ที่มีจุดเริ่มต้นราว 30 ปีมาแล้ว โดยกระทรวงกลาโหมสหรัฐฯ ซึ่งเรียกว่าโครงการARPANET

การรับส่งข้อมูลจะแบ่งข้อมูลออกเป็นส่วนย่อยๆ เรียกว่า Packet ข้อมูลแต่ละส่วนนี้จะถูกส่งไปให้คอมพิวเตอร์ผู้รับที่ปลายทางผ่านสายส่งข้อมูล โดยแต่ละส่วนอาจใช้เส้นทางสำหรับส่งข้อมูลคนละทางก็ได้ คอมพิวเตอร์ปลายทางจะนำข้อมูลที่ได้รับมาต่อรวมกันตามลำดับจนครบ หากเส้นทางที่ส่งข้อมูลเสียหาย หรือเครื่องคอมพิวเตอร์บางส่วนในเครือข่ายเสียหายข้อมูลก็จะถูกส่งไปใหม่โดยใช้เส้นทางอื่นแทนโดยอัตโนมัติ

Software ที่ใช้ควบคุมการรับส่งข้อมูลของ ARPANET ประกอบด้วยส่วนหลักๆ 2 ส่วน คือ Transmission Control Protocol หรือ TCP และ Internet Protocol หรือ IP ซึ่ง TCP มีหน้าที่ตรวจสอบการรับส่งข้อมูลระหว่างคอมพิวเตอร์ผู้รับและผู้ส่งให้ได้รับข้อมูลถูกต้องครบถ้วน หากข้อมูลสูญหายก็จะแจ้งให้ต้นทางส่งข้อมูลมาใหม่ ส่วน IP จะมีหน้าที่เลือกเส้นทางที่ใช้รับส่งข้อมูลผ่านระบบเครือข่าย และตรวจสอบ address ของผู้รับ โดยใช้ข้อมูลขนาด 4 byte หรือ 32 bit เป็นตัวกำหนด address ของผู้รับเรียกว่า IP address

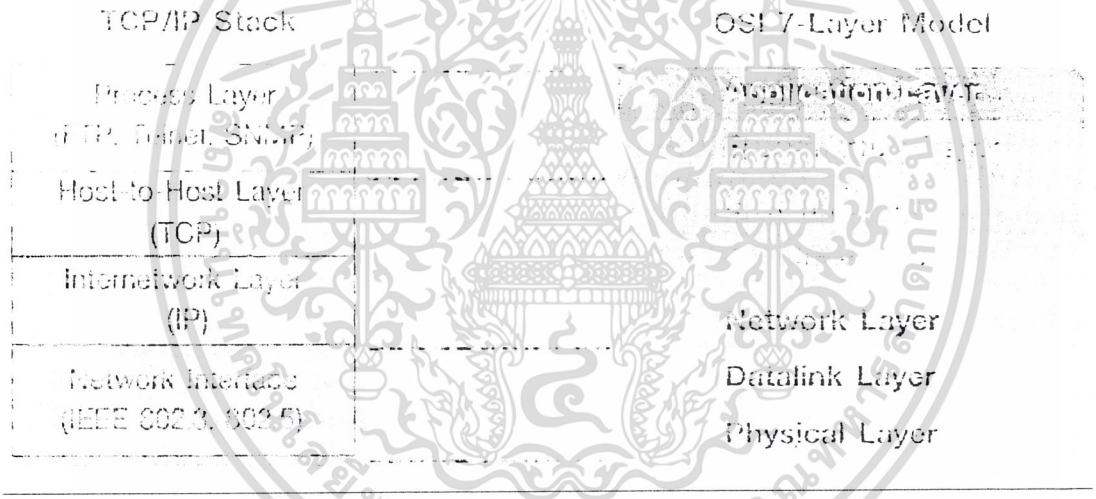
ในการเชื่อมต่อคอมพิวเตอร์หลายๆเครื่องเข้ากันเป็นระบบเครือข่ายนั้นปัญหาที่พบก็คือ การที่คอมพิวเตอร์เป็นคนละยี่ห้อหรือคนละระบบ ทำให้ขาดมาตรฐานส่วนกลางที่จำเป็นต้องใช้ในการรับส่งข้อมูลและเป็นขีดจำกัดในการเชื่อมต่อคอมพิวเตอร์คนละชนิดไม่ให้รับส่งข้อมูลกันได้ ระบบคอมพิวเตอร์ในยุคนั้นจึงเป็นระบบปิด(closed system)

ปัญหานี้ทำให้หน่วยงานกำหนดมาตรฐานสากล คือ ISO จัดการกำหนดโครงสร้างทั้งหมดที่จำเป็นต้องใช้ในการสื่อสารข้อมูลระหว่างคอมพิวเตอร์คนละระบบ ซึ่งเป็นการทำงานแบบที่เรียกว่า ระบบเปิด (open system) เราเรียกโครงสร้างของมาตรฐานการรับส่งข้อมูลนี้ว่า Open System Interconnection หรือ OSI

โดย OSI กำหนดให้การสื่อสารข้อมูลระหว่างระบบแบ่งออกเป็น 7 ขั้นตอนย่อยๆ เรียกว่า OSI 7-Layer Reference Model แต่เนื่องจาก TCP/IP เป็นมาตรฐานที่เกิดขึ้นก่อน OSI 7 – Layer

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กล่าวโดยสรุปก็คือ โปรโตคอล TCP/IP ทำงานโดยแบ่งเป็นชั้นเทียบกับ OSI Model ได้ กลไกในการทำงานของโปรโตคอล TCP/IP มี 4 ชั้น ซึ่งในชั้นแรกคือ Process Layer ทำหน้าที่ติดต่อกับแอพพลิเคชั่นและโปรโตคอลที่แอพพลิเคชั่นนั้นๆใช้งาน และส่งต่อมาให้ชั้น Host – to – Host Layer เพื่อติดต่อกันระหว่างเครื่องเซิร์ฟเวอร์ให้บริการกับเครื่องผู้ขอใช้บริการ ในชั้นนี้จะมีการสร้าง session หรือการเชื่อมต่อระหว่างระบบขึ้นตามแต่ละโปรโตคอลที่ต้องการ ต่อมาเป็นการผนึกข้อมูลไปเป็น IP datagram ที่ชั้น Internetwork Layer โดยอาศัยโปรโตคอล IP เพื่อให้สามารถติดต่อส่งข้อมูลข้ามเครือข่ายไปยังเครือข่ายและเครื่องที่ถูกต้องได้ และสุดท้ายการส่งข้อมูลออกสู่โลกภายนอกต้องอาศัยกลไกในชั้น Network Interface Layer เพื่อแปลงข้อมูลใหม่ เพิ่มข้อมูลที่จำเป็นในการอ้างอิงตำแหน่งและแปลงข้อมูลเป็นสัญญาณไฟฟ้าส่งออกไปยังเครือข่าย และอาจจะออกไปยัง Gateway หรือ Router เพื่อข้ามเครือข่ายออกไปยังเส้นทางที่กำหนดไว้ในอินเตอร์เน็ตต่อไป



รูปที่ 2.1 TCP/IP Stack และ OSI Model

โปรโตคอลที่มีบทบาทสำคัญในการทำงานในเครือข่ายอินเตอร์เน็ตคือ Internet Protocol (โปรโตคอล IP) เนื่องจาก เมื่อโปรโตคอลอื่นๆต้องการส่งผ่านข้อมูลข้ามเครือข่ายในอินเตอร์เน็ตนั้นจะต้องอาศัยการผนึกข้อมูล (Encapsulation) ไปกับโปรโตคอล IP ที่มีกลไกการระบุเส้นทาง (Route Service) ผ่าน Gateway หรือ Router เพื่อนำข้อมูลไปยังเครือข่ายและเครื่องปลายทางที่ถูกต้อง

เมื่อมีการเชื่อมต่อกันแล้วก็จำเป็นต้องมีการกำหนดหรือระบุเลขหมายของอุปกรณ์ทุกชิ้นทุกชนิดในเครือข่าย เพื่อให้อ้างอิงได้โดยไม่ซ้ำกัน เพราะถ้าซ้ำกันแล้วการรับส่งข้อมูลอาจจะไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถึงมือผู้รับปลายทางได้อย่างถูกต้อง เลขหมายดังกล่าวจะเรียกว่า Address หรือหมายเลขประจำตัวที่มีข้อกำหนดเป็นมาตรฐาน ซึ่งในการใช้งานโปรโตคอล TCP/IP ที่เชื่อมโยงเครือข่ายอินเตอร์เน็ตนี้ เลขหมายที่ใช้อ้างถึงกันจะใช้เป็นตัวเลขที่เรียกว่า IP address (Internet Protocol address) หรือ " หมายเลข IP "

IP address

หมายเลข IP address ถูกกำหนดขึ้นมาให้เป็นหมายเลขอ้างอิงประจำตัวของอุปกรณ์ต่างๆที่เชื่อมต่ออยู่ในเครือข่ายอินเตอร์เน็ต โดยการกำหนดหมายเลข IP address ให้แต่ละเครื่องหรือแต่ละอุปกรณ์นี้จะต้องไม่ซ้ำกัน ซึ่งหมายเลข IP address นี้จะไม่ถูกผูกติดกับตัวฮาร์ดแวร์ แต่อย่างใดจึงสามารถกำหนดใหม่หรือแก้ไขเปลี่ยนแปลงได้เมื่อมีการเปลี่ยนตัวฮาร์ดแวร์

การทำงานของโปรโตคอล IP จำเป็นต้องอาศัยหมายเลข IP address นี้เพื่อระบุและอ้างถึงอุปกรณ์ต่างๆที่ต่ออยู่ในเครือข่ายไม่ว่าจะเป็น เว็บเซิร์ฟเวอร์,เมล์เซิร์ฟเวอร์,อุปกรณ์ Router ฯลฯ หมายเลข IP address จะเป็นค่าตัวเลขขนาด 32 บิต ถูกแบ่งออกเป็นส่วนละ 8 บิต รวมเป็น 4 ส่วนและคั่นแต่ละส่วนด้วยเครื่องหมายจุด (.) ดังนั้นค่าตัวเลขในแต่ละส่วนจะมีได้ตั้งแต่ 0 ถึง 255 ตัวอย่างของ IP address เช่น 161.246.13.45 เป็นต้น

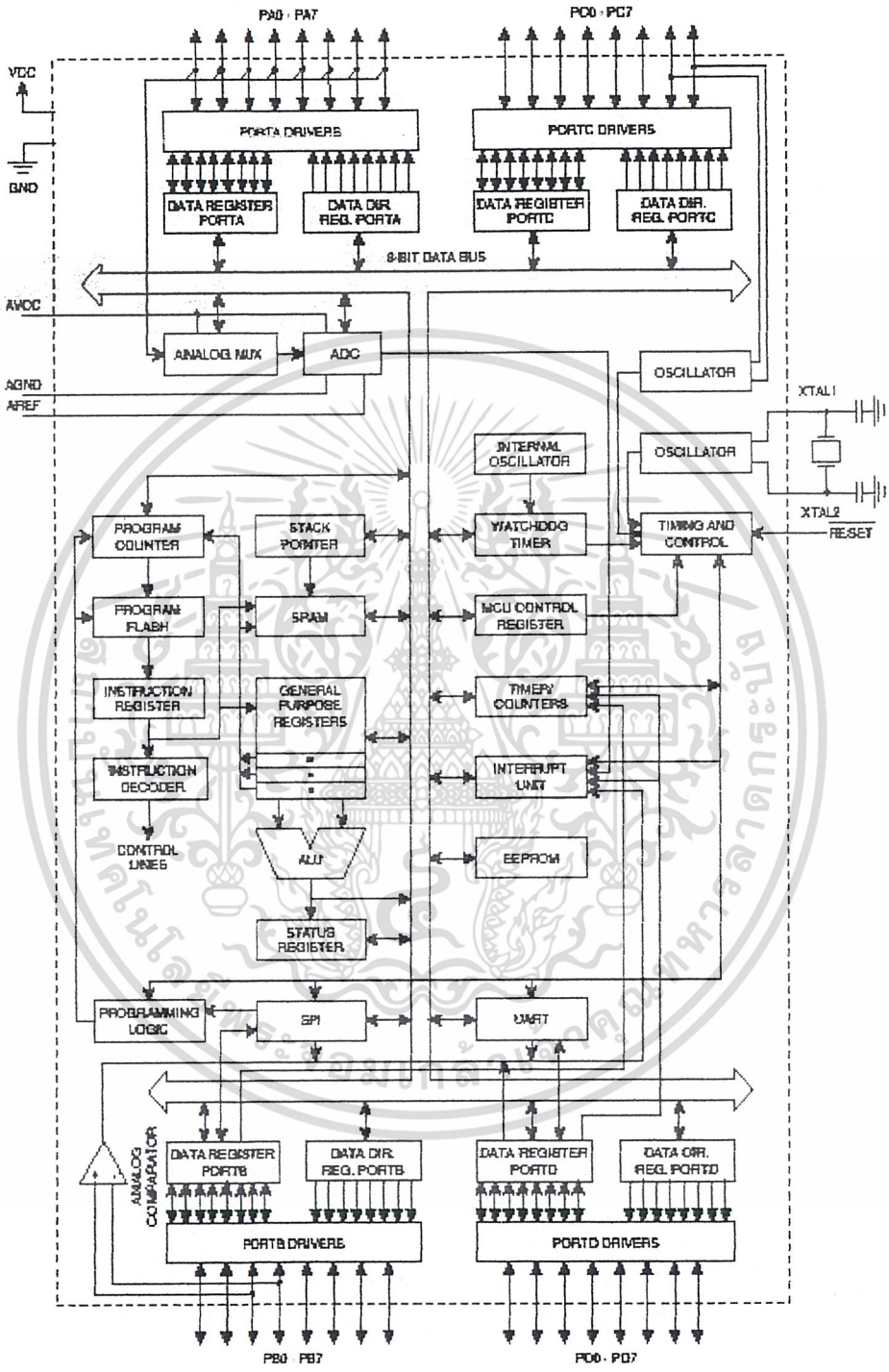
ในการติดต่อระหว่างคอมพิวเตอร์ 2 ระบบด้วย TCP/IP นั้น การเชื่อมต่อคอมพิวเตอร์จากระยะไกลผ่านโมเด็มและสายโทรศัพท์จะใช้ Telnet เป็นโปรแกรมติดต่อกับคอมพิวเตอร์ปลายทาง ซึ่งจะเป็นการใช้งานในลักษณะ Remote Terminal คือคำสั่งต่างๆที่พิมพ์จากแป้นพิมพ์ของเราจะถูกส่งไปให้คอมพิวเตอร์ปลายทางทำงานแทน และส่งข้อความแสดงผลบนจอภาพกลับมาให้เราเท่านั้น ดังนั้นการทำงานต่างๆ จึงเกิดขึ้นที่เครื่องปลายทางทั้งสิ้น

2.2 AVR AT90S8535 8 บิต

ไมโครคอนโทรลเลอร์ พร้อมหน่วยความจำ 4 K/8K แฟลช

AVR AR90S8535 เป็นไอซีไมโครคอนโทรลเลอร์ขนาด 8 บิต ตระกูลใหม่จาก ATMEL ที่ออกมา โดยรวมข้อดีจาก CPU ของ 68HC11 และ 8051 พร้อมรูปแบบหน่วยความจำแบบแฟลช รูปแบบโครงสร้าง CPU แบบ RISC ในรูปที่2.2 แสดงโครงสร้างบล็อกไดอะแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.2 บล็อกไดอะแกรมของAVR

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณสมบัติเด่นของไมโครคอนโทรลเลอร์AT90S8535

ด้านสถาปัตยกรรม

- สถาปัตยกรรมภายในถูกออกแบบให้ใช้สถาปัตยกรรมแบบ RISC (Reduce Instruction Set Computer)
- มีคำสั่งในการควบคุมการทำงานของไมโครคอนโทรลเลอร์จำนวน 118 คำสั่ง
- กลุ่มรีจิสเตอร์ใช้งานทั่วไปขนาด 8 บิต จำนวน 32 ตัว
- การประมวลคำสั่งเข้าใกล้ 1 MIPS per MHz
- watchdogที่ทำการโปรแกรมเองได้
- มีวงจรPower on Reset
- ความถี่สัญญาณนาฬิกา 0-8 เมกะเฮิรตซ์
- มี 3 SLEEP MODE: IDEL, POWER SAVE และ POWER DOWN

ด้านหน่วยความจำและการเก็บข้อมูล

- หน่วยความจำแบบแฟลชสำหรับบันทึกโปรแกรมขนาด 8 กิโลไบต์ แบบ ISP
- หน่วยความจำแบบ EEPROM สำหรับบันทึกข้อมูลขนาด 512 ไบต์
- หน่วยความจำแบบ RAM ขนาด 512 ไบต์

ด้านการต่อกับอุปกรณ์ภายนอก

- ตัวแปลงสัญญาณอนาลอกเป็นดิจิตอลขนาด10บิต 8 ช่อง
- ระบบการสื่อสารแบบฟลูดูเพล็กซ์(UART)
- ระบบการสื่อสารข้อมูลดิจิตอลแบบซิงโครนัส (SPI) 1 แชนแนล
- มีไทม์เมอร์/เคาน์เตอร์ ขนาด 16 บิต 1 แชนแนล
- มีไทม์เมอร์/เคาน์เตอร์ ขนาด 8 บิต 2 แชนแนล
- ระบบการเปรียบเทียบระดับสัญญาณอนาลอก (Analog Comparator)
- แหล่งกำเนิดความถี่แบบ PWM3 แชนแนล
- พอร์ตอินพุตและเอาต์พุตขนาด 8 บิต จำนวน 4 พอร์ต (Hi, Lo, Z)

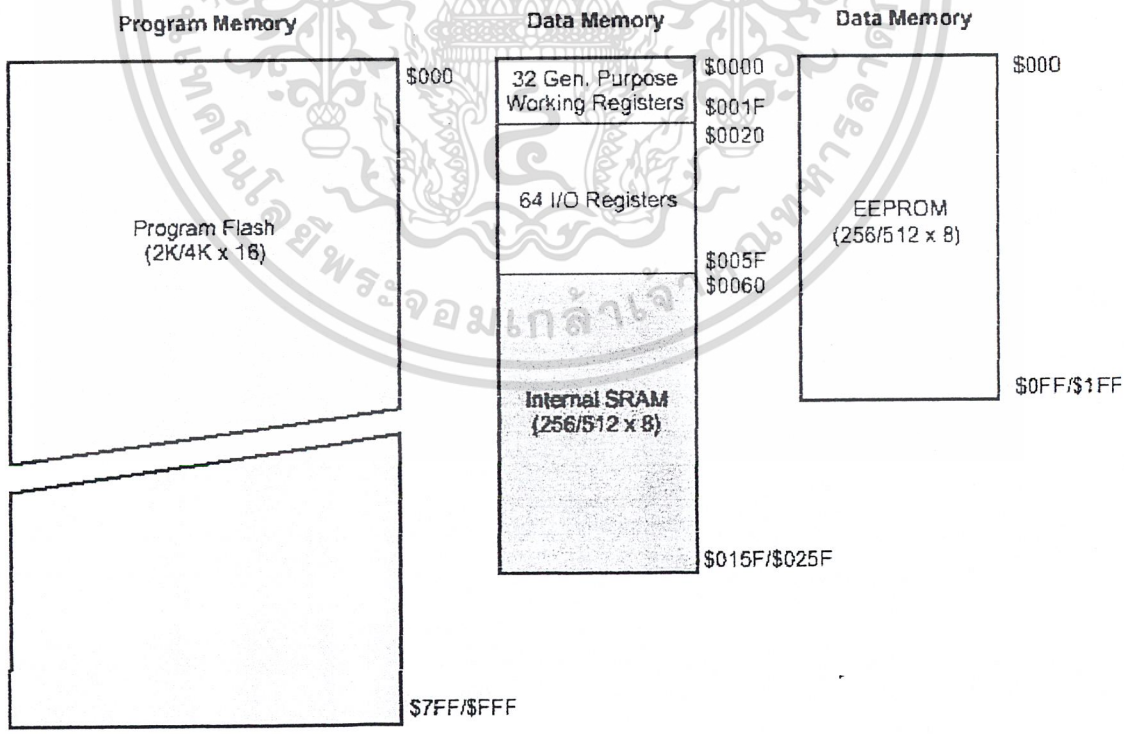
ลักษณะทั่วไป

AT90S8535 เป็นไมโครคอนโทรลเลอร์ขนาด 8 บิต ที่มีสถาปัตยกรรมแบบ RISC โดยระบบการทำงานของไมโครคอนโทรลเลอร์จะใช้หลักการออกแบบของฮาวาร์ด ด้วยการแยกระบบบัสของโปรแกรมและดาต้าออกจากกัน ทำให้ CPU สามารถ Fetch และ Execute คำสั่งได้ภาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

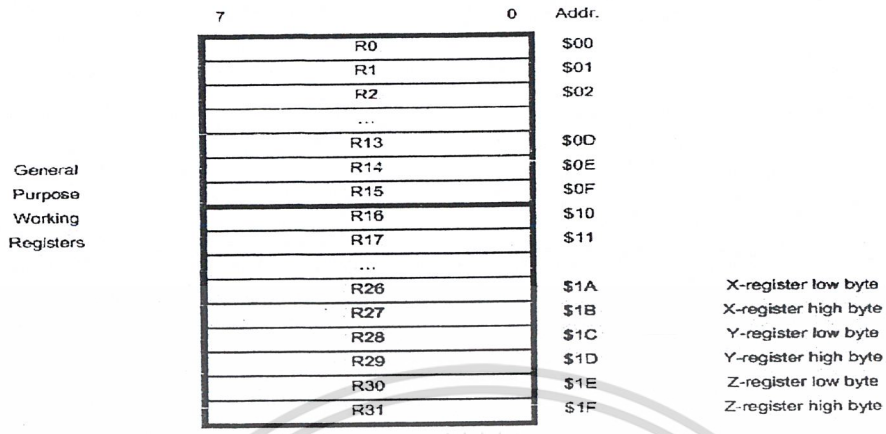
ใน 1 คาบเวลาซึ่งทำให้สามารถประมวลคำสั่งได้ 1 MIPS/MHz หน่วยความจำแบบFlashออกแบบให้โปรแกรมในระบบ(In-system Programming) ซึ่งทำการโปรมแกรมได้โดยไม่ต้องต่อวงจรมามายนัก ภายในประกอบด้วยรีจิสเตอร์ใช้งานทั่วไปขนาด 8 บิต จำนวน 32 ตัว ซึ่งแต่ละตัวจะต่อเข้ากับ ALU โดยตรง และมีพอร์ตที่สามารถทำงานได้ 2 ทิศทาง จำนวน 32 เส้นสัญญาณ และระบบไทม์เมอร์ / เคาน์เตอร์ จำนวน 3 ชุด ที่มีโหมดการทำงานเสริมในส่วนของการสร้างสัญญาณ PWM และส่วนของการตรวจจับสัญญาณอินพุตแคปเจอร์ (Input capture) มีอุปกรณ์สื่อสารข้อมูลอนุกรมแบบ UART และ SPI และยังเสริมด้วยระบบการแปลงสัญญาณอนาลอกเป็นดิจิตอล ขนาด 10 บิต จำนวน 8 แชนแนล ที่มาพร้อมกับ MCU มีส่วนการทำงานของวอตช์ด็อกไทม์เมอร์ เพื่อตรวจสอบการทำงานของไมโครคอนโทรลเลอร์ และมีระบบการประหยัดพลังงานอีก 3 ระบบ

ALU จะสนับสนุนการกระทางคณิตศาสตร์และลอจิกระหว่าง รีจิสเตอร์กับรีจิสเตอร์ หรือระหว่างรีจิสเตอร์กับค่าคงที่ ซึ่งการเรียกใช้รีจิสเตอร์สามารถกระทำได้โดยการอ้างหน่วยความจำภายในที่ตำแหน่ง $00-$1F จำนวน 32 ตำแหน่ง และใน MCU ยังได้จัดแบ่งให้มีรีจิสเตอร์ที่ใช้ควบคุมการทำงานของหน่วยอินพุตและเอาต์พุตต่างๆ อีก 64 ตำแหน่ง โดยสามารถเรียกใช้งานได้โดยการอ้างตำแหน่งหน่วย ความจำที่ตำแหน่ง $20-$5F ดังรูปที่2.3



รูปที่ 2.3 : บล็อกไดอะแกรมของหน่วยความจำ

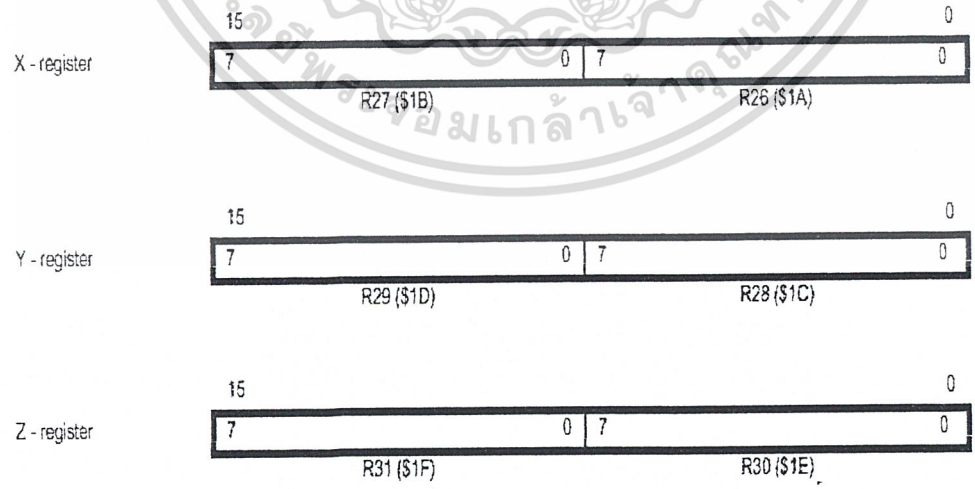
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.4 : โครงสร้างของรีจิสเตอร์ใช้งานทั่วไป

รีจิสเตอร์ใช้งานทั่วไป

ใน รูปที่ 2.4 แสดงการจัดวางตำแหน่งของรีจิสเตอร์ใช้งานทั่วไปทั้งหมด โดยมีรีจิสเตอร์ที่สามารถนำมาใช้งานเป็นรีจิสเตอร์คู่เพื่อทำเป็นตัวชี้ข้อมูลที่อยู่ในหน่วยความจำ ซึ่งรีจิสเตอร์ในกลุ่มนี้จะใช้ชื่อว่า X, Y และ Z ดังแสดงในรูปที่ 2.5 โดยในรีจิสเตอร์ R26...R31 นี้สามารถนำมาต่อกันเพื่อทำเป็นรีจิสเตอร์คู่เพื่อใช้งานเป็นตัวชี้ข้อมูลในชื่อของรีจิสเตอร์X,YและZ



รูปที่ 2.5 : แสดงรีจิสเตอร์ X, Y และ Z

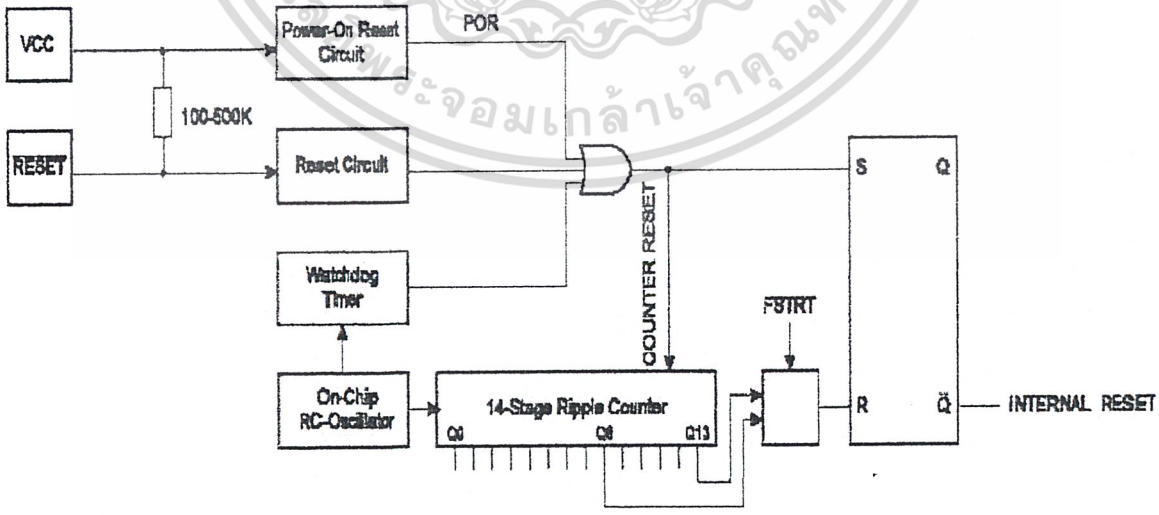
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน่วยความจำ SRAM

หน่วยความจำภายใน MCU จัดให้มีหน่วยความจำข้อมูล โดยหน่วยความจำทั้งหมดจะถูกแบ่งออกเป็นพื้นที่ของรีจิสเตอร์ใช้งานทั่วไป รีจิสเตอร์ใช้งานอินพุต/เอาต์พุต และหน่วยความจำภายใน SRAM โดยใน 96 ตำแหน่งแรกจะถูกแบ่งออกเป็นส่วนของรีจิสเตอร์ และอีก 512 ตำแหน่งจะถูกจัดไว้เป็นส่วนของหน่วยความจำภายใน SRAM สำหรับการเข้าถึงข้อมูลจะถูกแบ่งออกเป็น 5 ส่วนคือ Direct, Indirect with Displacement, Indirect with Pre-Decrement และ Indirect with Post-Increment ดังแสดงโครงสร้างของหน่วยความจำไว้ในรูปที่2.3

การรีเซ็ตและการอินเตอร์รัปต์

อินเตอร์รัปต์เวกเตอร์ของแต่ละอินเตอร์รัปต์ออกจากกัน ในการควบคุมการตอบสนองของอินเตอร์รัปต์แต่ละแหล่งสามารถแยกการควบคุมได้จากบิตอีนาเบิลของไมโครคอนโทรลเลอร์ AT90S8535 มีระบบการตอบสนองสัญญาณอินเตอร์รัปต์จาก 16 แหล่งสัญญาณ โดยได้แยกอินอินเตอร์รัปต์ทั้งหมดโดยตำแหน่งแรก ๆ ในหน่วยความจำโปรแกรมจะเป็นตำแหน่งที่ถูกใช้เป็นพื้นที่ของอินเตอร์รัปต์เวกเตอร์ โดยเริ่มจากอินเตอร์รัปต์ที่มีระดับความสำคัญสูงสุด คือ รีเซ็ต จนถึงอินเตอร์รัปต์ที่มีระดับความสำคัญต่ำสุด รูปที่ 2.6 แสดงโครงสร้างของวงจรรีเซ็ต รีจิสเตอร์ทั้งหมดจะถูกกำหนดให้อยู่ที่ค่าเริ่มต้นและโปรแกรมจะเริ่มทำงานที่ตำแหน่ง $0000 โดยคำสั่งที่ตำแหน่ง $0000 เมื่อถูกทำการรีเซ็ต



รูปที่ 2.6 โครงสร้างของวงจรรีเซ็ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญญาณรีเซ็ต ไมโครคอนโทรลเลอร์ AT90S8535 มีสัญญาณรีเซ็ต 3 แหล่ง คือ

1. วงจรของ Power On Reset (POR) ถูกสร้างขึ้นเพื่อให้แน่ใจว่า MCU จะถูกรีเซ็ต เมื่อระดับของแรงดันไฟเลี้ยงถึงระดับที่จะทำให้ระบบการทำงานภายในของ MCU ทำงานได้อย่างถูกต้อง

2. External Reset การรีเซ็ตจากภายนอกจะเกิดขึ้นโดยการให้ลอจิก LOW ที่ขา RESET อย่างน้อย 50นาโนวินาที ซึ่งเมื่อสัญญาณที่ขอบขาขึ้นของขารีเซ็ตขึ้นถึงระดับแรงดันเทรสโฮล รีเซ็ต (Reset Threshold Voltage) จะทำให้วงจรหน่วงเวลา (Time Delay) เริ่มทำงาน ซึ่ง CPU จะถูกรีเซ็ต

3. Watchdog Reset เมื่อ WTD TIME-OUT เริ่มกำเนิดสัญญาณพัลส์ที่มีคาบเวลาเท่ากับคาบเวลาของสัญญาณนาฬิกา โดยในช่วงขอบขาลงของพัลส์จะทำให้ระบบการหน่วงเวลาเริ่มทำงานจนหมดช่วงเวลาของTIME-OUTจึงจะทำให้CPUรีเซ็ต

การสื่อสารข้อมูลแบบอนุกรม

ภายในไมโครคอนโทรลเลอร์ AT90S8535 มีระบบการสื่อสารแบบอนุกรม 2 แบบ คือ Serial Peripheral Interface (SPI) ซึ่งเป็นระบบการสื่อสารที่ใช้การสื่อสารแบบซิงโครนัส และอีกระบบหนึ่งคือระบบAsynchronous Receiver and Transmitter (UART) ซึ่งเป็นระบบการสื่อสารที่ใช้การสื่อสารแบบอะซิงโครนัส ดังแสดงโครงสร้างของ SPI ไว้ในรูปที่ 2.7

Serial Peripheral Interface (SPI) เป็นการอินเตอร์เฟสสัญญาณแบบอนุกรมมีคุณลักษณะดังนี้คือ

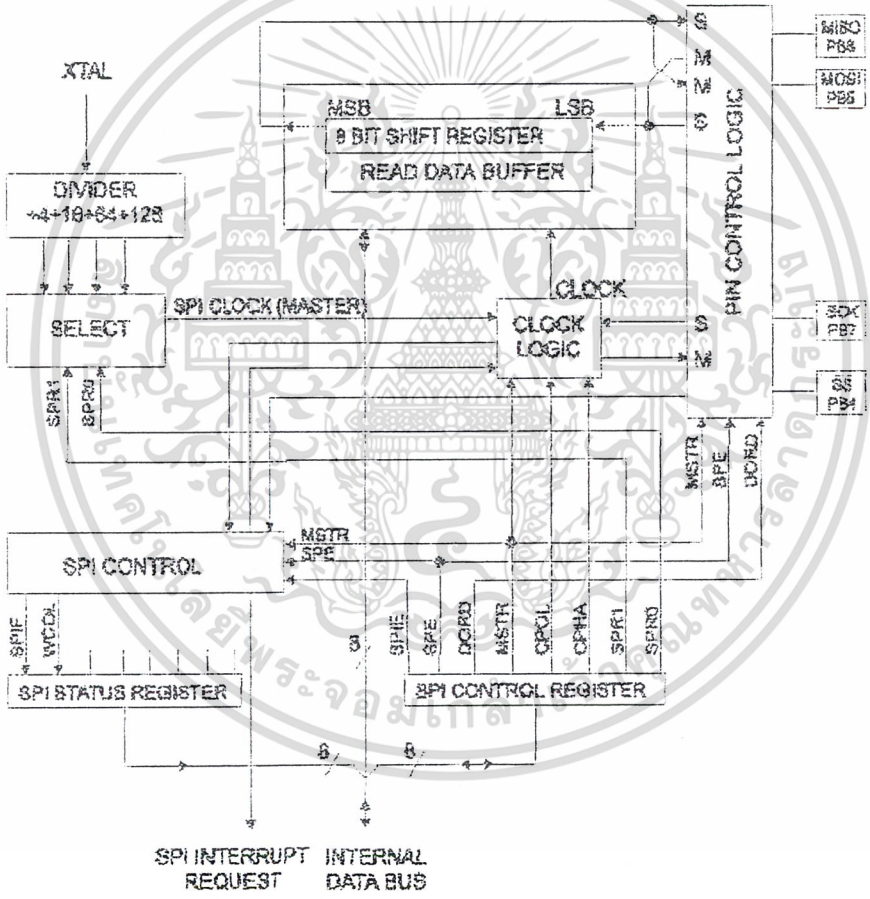
- เป็นการสื่อสารแบบฟูลดูเพล็กซ์ที่ใช้สายสัญญาณ 3 เส้นในการส่งถ่ายข้อมูล
- สามารถทำงานเป็นได้ทั้งตัวส่งข้อมูลและตัวรับข้อมูล
- สามารถเลือกได้ว่าจะให้ส่งบิต MSB หรือ LSB ออกก่อน
- มี FLAG แสดงสถานะการส่งข้อมูลสิ้นสุด

Asynchronous Receiver and Transmitter (UART) เป็นการอินเตอร์เฟสสัญญาณแบบอะซิงโครนัสมีคุณลักษณะดังนี้คือ

- สามารถเปลี่ยนแปลง BAUD RATE ของการสื่อสารได้หลาย BAUD RATE
- สามารถสื่อสารได้ในอัตรา BAUD RATE ที่สูงในขณะที่ความถี่คริสตอลต่ำ
- สื่อสารข้อมูลได้ทั้ง 8 บิตและ 9 บิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- มีส่วนของการกำจัดสัญญาณรบกวน (Noise filtering)
- ตรวจจับความผิดพลาดของกรอบข้อมูลผิดพลาด
- ตรวจจับความผิดพลาดของบิตเริ่มต้น
- จัดให้มีการแยกระบบอินเตอร์รัปต์ของการสื่อสารออกเป็น 3 ส่วนคือ อินเตอร์รับต์ที่เกิดจากการส่งข้อมูล อินเตอร์รัปต์ที่เกิดจากการรับข้อมูล และอินเตอร์รัปต์ที่เกิดจากการไม่มีข้อมูลในรีจิสเตอร์
- มีบัฟเฟอร์ในการเก็บข้อมูลที่จะส่งออกและข้อมูลที่รับเข้า



รูปที่ 2.7 โครงสร้างของ SPI

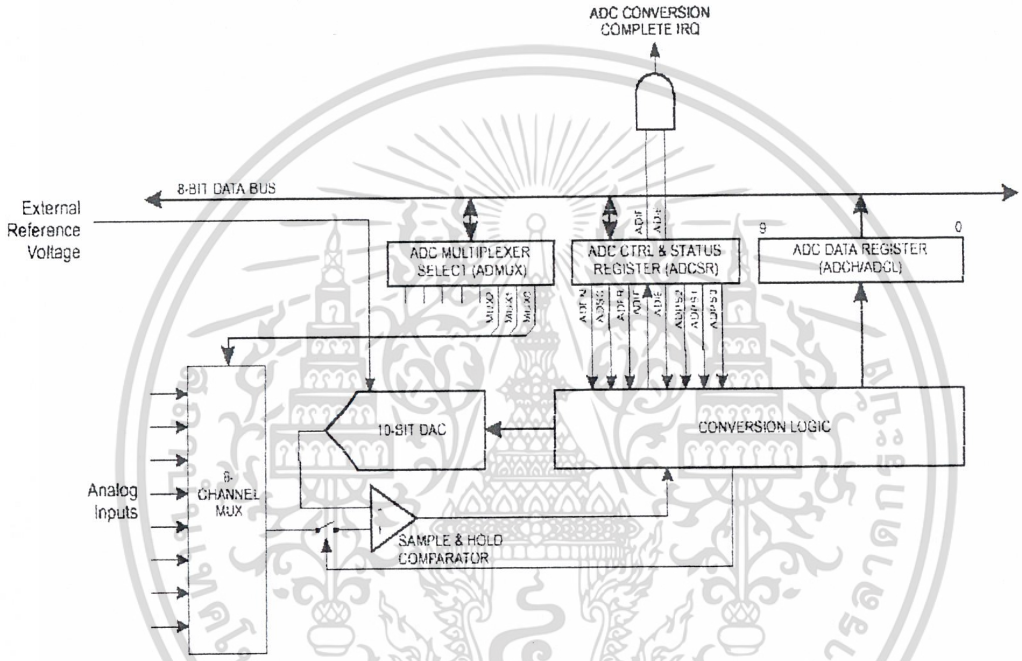
ระบบการแปลงสัญญาณอนาลอกเป็นดิจิตอล

ในส่วนของวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอลนั้นมีคุณสมบัติเด่นดังต่อไปนี้

- ความละเอียดขนาด 10 บิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ความถูกต้อง ± 1/2 LSB
- ค่าเวลาการแปลงสัญญาณ 65-260 ไมโครวินาที
- แชนเนลอินพุตเป็นแบบมัลติเพล็กซ์
- มีโหมดการแปลงสัญญาณแบบซิงเกิลหรือรันอิสระ
- มีการอินเตอร์รัปต์บนวงจรแปลงอย่างสมบูรณ์
- ตัดสัญญาณรบกวนเมื่อเข้าสู่โหมด sleep mode



รูปที่ 2.8 บล็อกไดอะแกรมของวงจรแปลงสัญญาณ ADC

ในรูปที่ 2.8 แสดงบล็อกไดอะแกรมของวงจรแปลงสัญญาณ ADC ซึ่ง AT90S8535 ได้จัดให้มีวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอลขนาด 10 บิตจำนวน 8 แชนเนล ซึ่งแต่ละแชนเนลจะรับสัญญาณเข้ามาทางขาของพอร์ต A ในแต่ละขาโดยในระบบจะมีวงจรแซมเปิลและโฮลเพื่อช่วยให้สัญญาณอนาลอกที่รับเข้ามาแปลงเป็นสัญญาณดิจิตอลมีระดับสัญญาณคงที่ โดยวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอลนี้จะมีแหล่งจ่ายไปและกราวด์แยกกันต่างหากจากแหล่งจ่ายไฟของระบบดิจิตอล ซึ่งในการใช้งานจริงไม่ควรให้ความแตกต่างกันเกิน ± 0.3 โวลท์ ซึ่งในการใช้งานจะต้องจ่ายแรงดันไฟอ้างอิงและกราวด์ที่ขา AREF ในช่วงของระดับแรงดัน AVCC ถึง AGND

การทำงานในส่วนของการแปลงสัญญาณอนาลอกเป็นดิจิตอลสามารถทำงานได้ 2 โหมดคือ Single Conversion Mode และ Free Running Mode โดยการทำงานแบบ Single Conversion Modeผู้ใช้ต้องเป็นผู้กำหนดการใช้งานขึ้นเอง แต่ในส่วนของ Free Running Modeวงจร

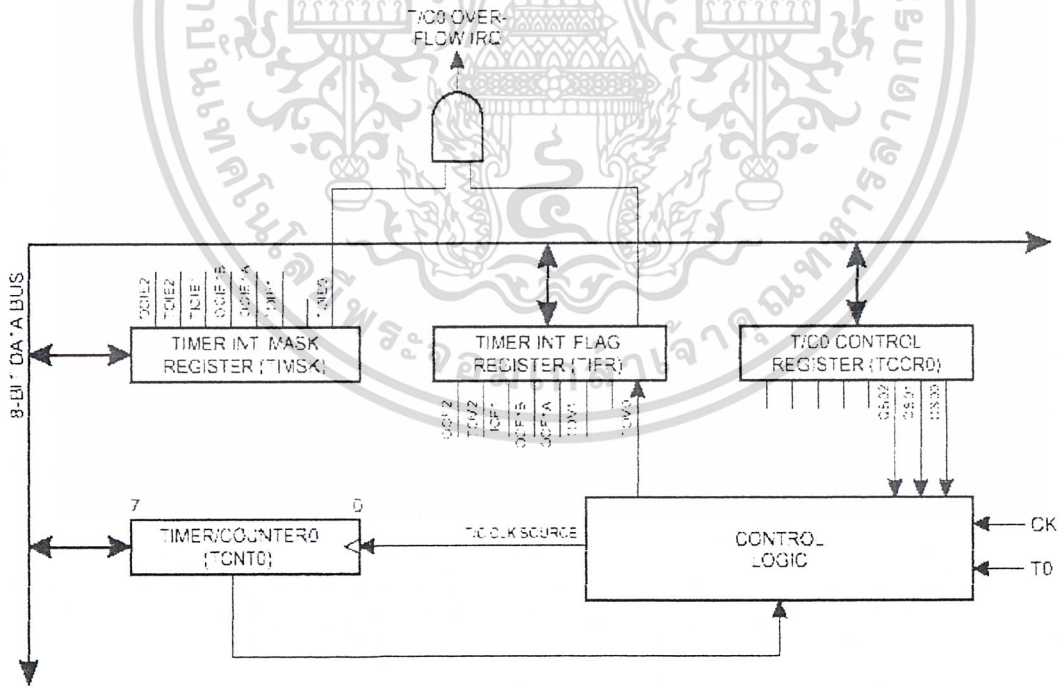
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แปลงสัญญาณ ADC จะเป็นตัวจัดการอ่านข้อมูลและเก็บในดาต้ารีจิสเตอร์ ADC ซึ่งบิตADCHและ ADCL ในรีจิสเตอร์ ADCSR จะเป็นบิตที่ใช้เลือกโหมดการใช้ของวงจรแปลงADC

ระบบไทม์เมอร์และเคาน์เตอร์

ภายใน AT90S8535 จัดให้มีวงจรไทม์เมอร์/เคาน์เตอร์ 3 ชุด โดยจัดเป็นไทม์เมอร์/เคาน์เตอร์ ขนาด 8 บิต 2 ชุดและไทม์เมอร์/เคาน์เตอร์ ขนาด 16 บิต 1 ชุด โดยไทม์เมอร์/เคาน์เตอร์ 2 สามารถรับสัญญาณนาฬิกาจากภายนอกได้ ซึ่งเป็นส่วนเพิ่มเติมที่จะใช้ไทม์เมอร์/เคาน์เตอร์ 2 มาทำเป็น RTC โดยใช้คริสตอลที่มีค่าความถี่เท่ากับ 32.768 กิโลเฮิร์ตซ มาเป็นฐานเวลา ไทม์เมอร์/เคาน์เตอร์ 0 และไทม์เมอร์/เคาน์เตอร์ 1 จะใช้วงจรปรีสเกลขนาด 10 บิตซึ่งเป็นส่วนที่เลือกความถี่ของสัญญาณที่ให้กับไทม์เมอร์/เคาน์เตอร์ร่วมกัน ส่วนไทม์เมอร์/เคาน์เตอร์ 2 นั้นจะใช้วงจรปรีสเกลแยกออกต่างหาก ในรูปที่ 2.9 รูปที่ 2.10 แสดงบล็อกไดอะแกรมของไทม์เมอร์/เคาน์เตอร์ 0, 1

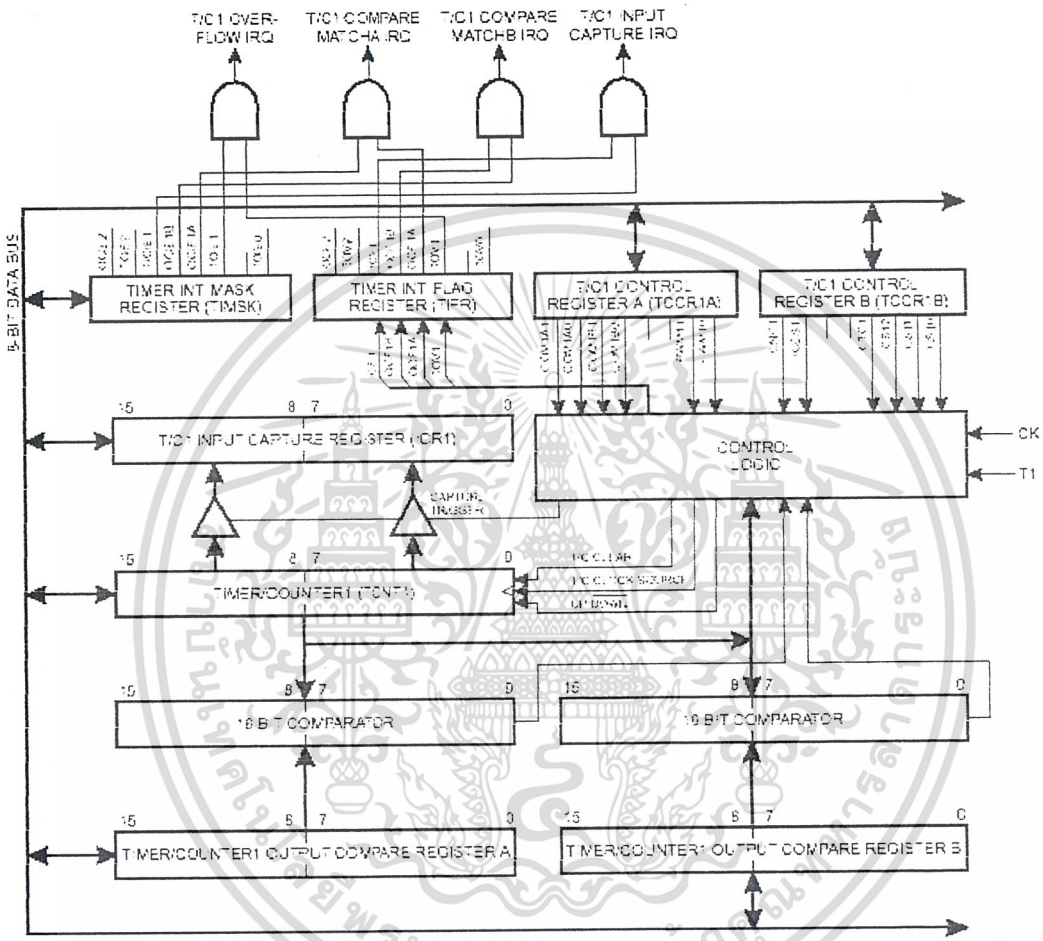
ไทม์เมอร์/เคาน์เตอร์ 0 จะมีขนาด 8 บิต ซึ่งสามารถเลือกสัญญาณนาฬิกาได้จาก CK (สัญญาณนาฬิกาของระบบ) ถูกหาร (Prescaling) หรือ สัญญาณนาฬิกาจากภายนอก



รูปที่ 2.9 โครงสร้างของไทม์เมอร์/เคาน์เตอร์ 0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไทม์เมอร์/เคาน์เตอร์ 1 จะมีขนาด 16 บิตโดยสามารถเลือกสัญญาณนาฬิกาได้จาก CK หรือ สัญญาณที่ได้รับการหารจาก CK (Prescaling) ส่วนการใช้งานไทม์เมอร์/เคาน์เตอร์ 1 ในโหมด PWM นั้นจะสามารถเลือกใช้งานได้ขนาด 8, 9 หรือ 10 บิต



รูปที่ 2.10 โครงสร้างของไทม์เมอร์/เคาน์เตอร์ 1

ไทม์เมอร์/เคาน์เตอร์ 2 มีขนาด 8 บิต ที่มีส่วนของ Input Capture, Output Compare Match และ PWM เหมือนกับไทม์เมอร์/เคาน์เตอร์ 1 แต่เพิ่มส่วนของการทำงานแบบซิงโครนัสเคาน์เตอร์เข้ามา

พอรต์ I/O

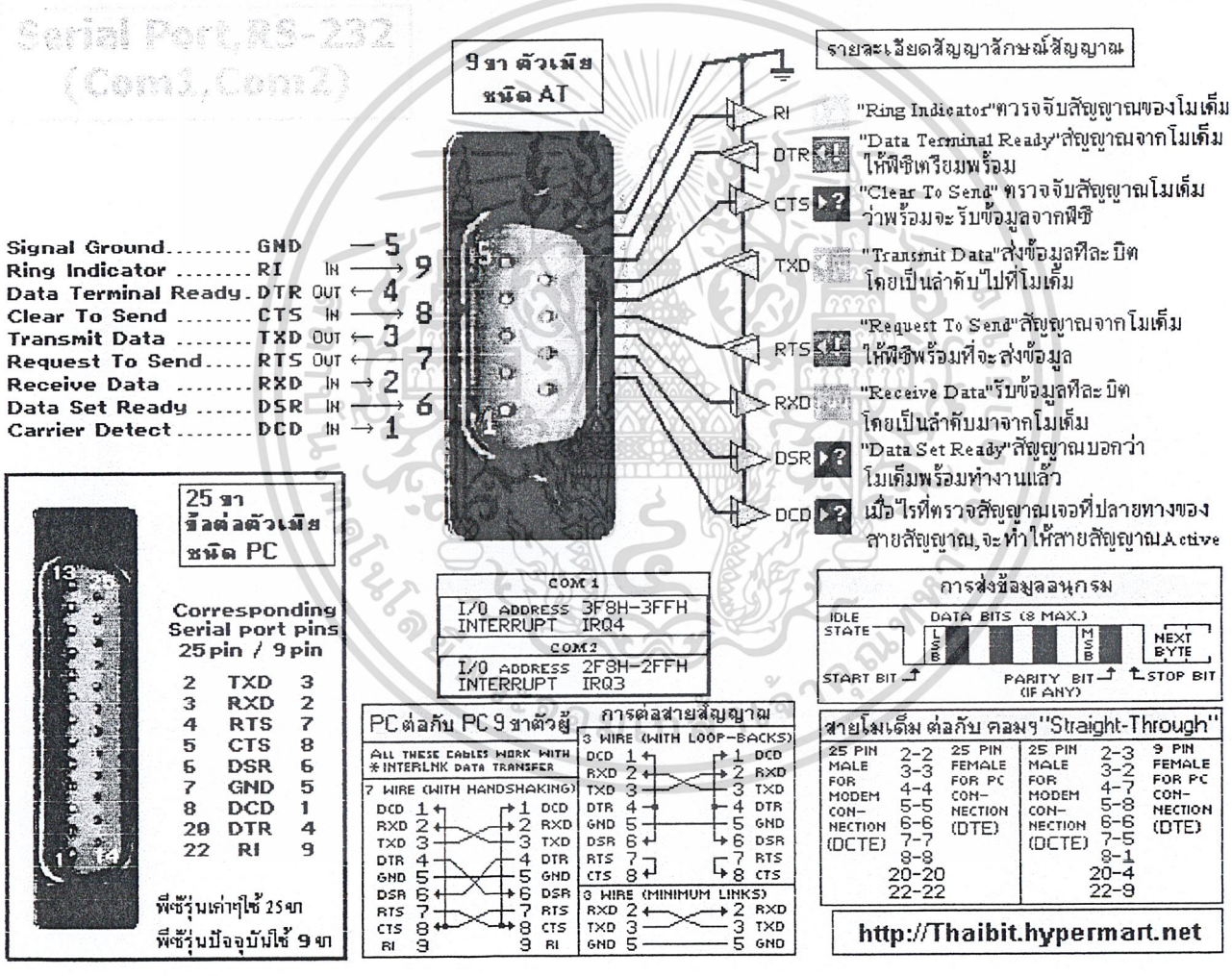
พอร์ตอินพุตและเอาต์พุตของAT90S8535 เป็นพอร์ตขนาด 8 บิตสองทิศทาง มีความต้านทานพลูอัพภายใน ได้แก่พอร์ตA, พอร์ตB, พอร์ตC และพอร์ตD การเข้าถึงพอร์ตสามารถเขียนและอ่านผ่านทางไอโอแอดแดรสคือ PORTX , DDRX และ PINX โดยPINX เป็นแอดแดรสที่อ่าน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

เพียงอย่างเดียว ส่วนPORTX DDRXทำการอ่านและเขียนข้อมูลได้ โดยXเป็นหมายเลขพอร์ต รายละเอียดและหน้าที่การทำงานของแต่ละพอร์ตดูได้จากดาต้าชีทของAT90S8535

2.3 COM PORT



รูปที่2.11 แสดงลักษณะของSerial Port

การที่อุปกรณ์สองตัวจะสื่อสารสารกันได้นั้น ต้องมีการเชื่อมต่อด้วยวิธีใดวิธีหนึ่ง เพื่อให้สัญญาณไฟฟ้าที่ถูกส่งโดยฝ่ายหนึ่งสามารถถูกรับโดยอีกฝ่ายหนึ่งสามารถรับสัญญาณไฟฟ้าที่ส่งมาจากอีกฝ่ายหนึ่งได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

40214

การสื่อสารอาจเกิดขึ้นโดยตรงด้วยการเชื่อมต่ออุปกรณ์สองตัวด้วยสายสัญญาณ หรือโดยอ้อมด้วยสื่อกลางที่สอดแทรกเข้ามา สื่อกลางนี้มักจะเป็นระบบโทรศัพท์สาธารณะ ซึ่งในกรณีนี้โมเด็ม (Modem) เพื่อแปลงสัญญาณที่ปลายด้านหนึ่งให้เป็นสัญญาณที่เหมาะสมกับการส่ง ผ่านสายโทรศัพท์ และเพื่อแปลงมันกลับที่ปลายอีกด้านหนึ่ง สื่อชนิดอื่น เช่น เส้นใยนำแสง และการส่งผ่านคลื่นวิทยุ สามารถนำมาใช้ได้เช่นกัน อุปกรณ์การสื่อสารจะทำให้คอมพิวเตอร์สามารถสื่อสารโดยใช้สื่อเหล่านี้ เสมือนถูกเชื่อมต่อด้วยวิธีเดียวกับอุปกรณ์อนุกรมปกติ ดังนั้นหลักการที่กล่าวไว้สำหรับการสื่อสารโดยตรง จะประยุกต์ใช้กับการสื่อสารโดยอ้อมได้เช่นกัน

ปลั๊ก และซ็อกเกต

ปลั๊กและซ็อกเกตสำหรับการเชื่อมต่อสายสัญญาณเข้ากับอุปกรณ์อนุกรมมีอยู่มากมายหลายชนิด แต่นิยมใช้หัวต่อ D-type ชนิด 9 และ 25 ขา (บางครั้งเรียกว่า DB-25 และ DB-9) กันมากที่สุด นอกจากนี้ยังมีหัวต่อชนิดอื่นที่ใช้กัน เช่น หัวต่อแบบ DIN ที่ใช้ในแอปเปิลคอมพิวเตอร์ (Apple Computer)

หัวต่อ D-type (ที่ถูกตั้งชื่อแบบนี้เพราะว่า เมื่อมองจากด้านหน้าของหัวต่อจะมีรูปร่างคล้ายตัว D) ประกอบด้วยขา หรือรูจำนวนหนึ่ง แบบที่มีขาเป็นหัวต่อตัวผู้ แบบที่มีรูเป็นหัวต่อตัวเมีย แต่ละขาหรือรูจะมีหมายเลขซึ่งโดยทั่วไปจะถูกพิมพ์ไว้ข้างๆ

มาตรฐาน RS-232-C

เพื่อที่จะทำให้อุปกรณ์จากผู้ผลิตต่างกันทำงานร่วมกันได้ มาตราฐานหลายชนิดจึงได้รับการออกแบบขึ้น มาตราฐานที่ใช้กันกว้างขวางที่สุดคือ RS-232-C ถูกประกาศในปี 1969 โดย Electronic Industries Association มาตรฐาน RS-232-C ที่ร่างขึ้นในตอนเริ่มแรกสำหรับกำหนดการเชื่อมต่อระหว่างเทอร์มินัล (terminal) และโมเด็มระบุคุณลักษณะทางไฟฟ้าของวงจรระหว่างอุปกรณ์สองตัว และกำหนดชื่อ และหมายเลขแก่สายที่จำเป็นสำหรับการเชื่อมต่อวงจร ชื่อวงจรตามมาตราฐาน RS-232-C(AA,AB เป็นต้น) จำได้ยากในทางปฏิบัติจึงใช้ชื่อย่อแทน

ตัวอย่างเช่น สายเส้นที่ 2 ทีชื่อ BA แต่ใช้กันทั่วไปว่า TXD (Transmitted Data) ตามมาตราฐาน RS-232-C สายเส้นที่ 2 นำข้อมูลจากเทอร์มินัลไปสู่โมเด็ม เพื่อให้การทำงานถูกต้องเทอร์มินัลต้องส่งเอาต์พุตออกที่สายเส้นที่ 2 และโมเด็มต้องรับข้อมูลบนสายเส้นที่ 2 เพราะฉะนั้นสายเส้นที่ 2 เป็นสายส่งข้อมูลสำหรับอุปกรณ์บางอย่าง และเป็นสายรับข้อมูลสำหรับอุปกรณ์อย่างอื่น การเชื่อมต่อโดยตรงจากสายเส้นที่ 2 บนอุปกรณ์หนึ่งเข้ากับสายเส้นที่ 2 บนอุปกรณ์อีกตัวหนึ่งสามารถทำได้ต่อเมื่ออุปกรณ์ส่งข้อมูลบนสายเส้นที่ 2 และอีกตัวหนึ่งรับข้อมูลบนสายเส้นที่ 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพื่อป้องกันไม่ให้อุปกรณ์ส่งข้อมูลบนสายเส้นเดียวกัน อุปกรณ์จึงถูกแบ่งออกเป็นสองชนิด อุปกรณ์อย่างเช่นเทอร์มินัล ซึ่งใช้สายเส้นที่ 2 สำหรับเอาต์พุต เรียกว่า DTE (Data Terminal Equipment) อุปกรณ์อย่างเช่น โมเด็มซึ่งใช้สายเส้นที่ 2 สำหรับอินพุต เรียกว่า DCE (Data Communication Equipment)

อุปกรณ์ DTE และ DCE

ตามมาตราฐาน RS-232-C อุปกรณ์ DTE ควรใช้หัวต่อตัวผู้ และอุปกรณ์ DCE ควรใช้หัวตัวเมีย อย่างไรก็ตามผู้ผลิตไม่ได้ปฏิบัติตามกฎนี้เสมอ ดังนั้นจึงไม่อาจแยกแยะอุปกรณ์ DTE และ DCE โดยการมองผ่านๆ ได้เสมอไป

เมื่อทราบว่าอุปกรณ์หนึ่งเป็น DTE และอีกตัวหนึ่งเป็น DCE ในทางทฤษฎีแล้วสามารถเชื่อมต่อได้อย่างง่ายดาย โดยการเชื่อมต่อสายที่มีหมายเลขตรงกัน เช่น เส้นที่ 2 กับ 2, 3 กับ 3 เป็นต้น เรียกว่าการเชื่อมต่อแบบตรงไปตรงมา แต่มีผู้ผลิตบางรายที่ไม่ได้ทำตามมาตราฐาน และทำให้เกิดปัญหาหลายอย่าง ปัญหาเหล่านี้จะได้รับการกล่าวถึงในภายหลัง เช่นเดียวกับวิธีจัดการกับสถานการณ์ที่อุปกรณ์ทั้งสองเป็น DTE หรือ DCE เหมือนกัน ในตอนนี้ให้ถือว่าอุปกรณ์หนึ่งเป็น DTE และอีกตัวหนึ่งเป็น DCE และแต่ละฝ่ายส่งสัญญาณที่อีกฝ่ายต้องการบนสายที่ตรงกัน

การสื่อสารทางเดียว

วงจรหลักที่ถูกใช้สำหรับการสื่อสารมีอยู่สามวงจร คือสายเส้นที่ 2 สำหรับข้อมูลจาก DTE ไปยัง DCE สายเส้นที่ 3 สำหรับข้อมูลจาก DCE ไปยัง DTE และสายเส้นที่ 7 สำหรับซิกแนล-กราวด์ (signal ground) ซึ่งเป็นจุดอ้างอิงร่วมสำหรับขั้ว และแรงดันไฟฟ้าของสายอื่น ในกรณีที่ง่ายที่สุดซึ่งมีเพียงอุปกรณ์หนึ่งส่งและอีกตัวรับใช้สายเพียงสองเส้นก็เพียงพอ คือสายเส้นที่ 2 หรือ 3 และสายเส้นที่ 7

ฮาร์ดแวร์แฮนด์เช็คกิ้ง

ในหลายๆกรณี อุปกรณ์ฝ่ายส่งจำเป็นต้องรู้ว่าอุปกรณ์ฝ่ายรับพร้อมที่จะรับข้อมูลหรือไม่ ตัวอย่างเช่น การส่งข้อมูลไปที่เครื่องพิมพ์ ความเร็วของการสื่อสารอาจเร็วกว่าความเร็วของเครื่องพิมพ์ เครื่องพิมพ์ต้องระงับการส่งข้อมูลของคอมพิวเตอร์จนกว่ามันพร้อมที่จะรับข้อมูล ในทำนองเดียวกันกับการส่งข้อมูลจากคอมพิวเตอร์เครื่องหนึ่งไปอีกเครื่องหนึ่ง และคอมพิวเตอร์ตัวที่สองไม่สามารถประมวลผลข้อมูลได้เร็วเท่ากับอัตราที่ข้อมูลเข้ามา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทั้งสองกรณี ต้องมีข่าวสารถูกส่งกลับจากอุปกรณ์ฝ่ายรับไปยังอุปกรณ์ฝ่ายส่ง เพื่อแจ้งว่ามันพร้อมหรือไม่ ข่าวสารนี้เรียกว่า *โฟลว์คอนโทรล (Flow control)* หรือ *แฮนด์เช็คกิ้ง (hand chaking)* แฮนด์เช็คกิ้งมีสองแบบคือ ฮาร์ดแวร์ และซอฟต์แวร์ ทั้งคู่เกี่ยวข้องกับสัญญาณที่ส่งกลับมาจากอุปกรณ์ฝ่ายรับไปยังอุปกรณ์ฝ่ายส่ง ในฮาร์ดแวร์แฮนด์เช็คกิ้งอุปกรณ์ฝ่ายรับส่งแรงดันไฟฟ้าบวกไปตามวงจรแฮนด์เช็คกิ้งเมื่อมันพร้อมที่จะรับข้อมูล เมื่อคอมพิวเตอร์ฝ่ายส่งได้รับแรงดันลบ มันจะรู้ว่าต้องหยุดส่งข้อมูล ในซอฟต์แวร์แฮนด์เช็คกิ้งสัญญาณแฮนด์เช็คกิ้งเป็นอักขระพิเศษที่ถูกส่งไปตามวงจรข้อมูลแทนแฮนด์เช็คกิ้ง

การใช้ฮาร์ดแวร์แฮนด์เช็คกิ้งอย่างน้อยต้องมีการเชื่อมต่อเพิ่มเติมหนึ่งเส้นเพื่อนำพาสัญญาณทำให้จำนวนของสายทั้งหมดเป็นสามเส้นคือ สายข้อมูลซิกแนลกราวด์ และแฮนด์เช็คกิ้ง

จาก DTE ไปสู่ DCE

เมื่ออุปกรณ์ DTE ส่งข้อมูลไปที่อุปกรณ์ DCE ข้อมูลถูกส่งไปตามสายเส้นที่ 2 และใช้สายเส้นที่ 7 ซิกแนลกราวด์ ตามปกติอุปกรณ์ DCE ควบคุมการส่งแฮนด์เช็คกิ้งจากอุปกรณ์ DTE บนสายเส้นที่ 6 ชื่อว่า DSR (Data Set Ready) ถ้าเครื่องพิมพ์เป็น DCE และคอมพิวเตอร์เป็น DTE สายเส้นที่ 6 บนคอมพิวเตอร์ควรถูกเชื่อมต่อเข้ากับสายเส้นที่ 6 บนเครื่องพิมพ์ และเครื่องพิมพ์จะรักษาแรงดันไฟฟ้าบวกบนสายเส้นที่ 6 ตราบเท่าที่มันสามารถรับข้อมูล เมื่อมันต้องการที่จะบอกคอมพิวเตอร์ให้หยุดการส่งข้อมูล มันจะลดแรงดันไฟฟ้าบนสายเส้นที่ 6 ให้เป็นสถานะลบ

บ่อยครั้งที่วงจรแฮนด์เช็คกิ้งชุดที่สอง คือสายเส้นที่ 5 ถูกใช้โดยอุปกรณ์ DCE เพื่อควบคุมการส่งจากอุปกรณ์ DTE วงจรนี้มีชื่อว่า CTS (Clear To Send) เมื่อสายแฮนด์เช็คกิ้งทั้งสองเส้นถูกใช้ อุปกรณ์ DTE ต้องได้รับการออกแบบให้ส่งข้อมูลก็ต่อเมื่อสายทั้งสองเป็นไฮ (high) หรือแรงดันไฟฟ้าบวก บางครั้งสายนั้นอาจมีความหมายต่างไป เช่นเส้นหนึ่งอาจบอกอุปกรณ์ฝ่ายส่งให้หยุดการพิมพ์จนกระทั่งข้อมูลถูกพิมพ์ไปได้จำนวนหนึ่ง และเส้นที่เหลืออาจจะแจ้งว่ากระดาษของเครื่องพิมพ์หมด อย่างไรก็ตามความหมายเหล่านี้ไม่ได้มีมาตราฐาน เนื่องจากคอมพิวเตอร์หลายชนิดถูกโปรแกรมไม่ให้ส่งข้อมูล ถ้าสายแฮนด์เช็คกิ้งทั้งสองไม่เป็นไฮแม้แต่กับเครื่องพิมพ์ที่ไม่ได้กำหนดความหมายพิเศษกับสายชุดที่สองอย่างน้อยก็ควรรักษาแรงดันไฟฟ้าบวกไว้ อย่างไรก็ตามบางครั้งสัญญาณชุดที่สองต้องถูกสร้างหลอกขึ้นมาโดยการต่อมันเข้ากับชุดแรก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DCE ไปยัง DTE

เมื่อให้อุปกรณ์ DCE จะสื่อสารกับอุปกรณ์ DTE สายเส้นที่ 3 ต้องถูกใช้สำหรับการส่งผ่านข้อมูล และถ้าต้องการแฮนด์เช็คกิ้ง จะใช้สายเส้นที่ 20 เพื่อส่งแฮนด์เช็คกิ้งจากอุปกรณ์ DTE ไปยังอุปกรณ์ DCE สายเส้นที่ 20 มีชื่อว่า DTR (Data Terminal Ready) สายแฮนด์เช็คกิ้งชุดที่สองคือ สายเส้นที่ 4 เรียกว่า Request To Send (RQS หรือ RT1S4) ไม่ได้ถูกนำมาใช้เสมอไป

การสื่อสารสองทาง

ในหลายกรณีที่ข้อมูลถูกส่งผ่านในสองทิศทาง โดยเฉพาะเมื่อคอมพิวเตอร์สองตัวสื่อสารกัน รวมทั้งในกรณีที่ใช้ซอฟต์แวร์แฮนด์เช็คกิ้งด้วยเช่นกัน จำนวนสายที่น้อยที่สุดที่จำเป็นในการสื่อสารสองทางคือ สามเส้น ได้แก่ สายข้อมูลในแต่ละทิศทาง และซิกแนลกราวด์ การเพิ่มเติมสายแฮนด์เช็คกิ้งในแต่ละทิศทาง ทำให้จำนวนสายรวมเป็นห้าเส้น เมื่อสายแฮนด์เช็คกิ้งชุดที่สองถูกนำมาใช้เพิ่มเติมลงในแต่ละทิศทาง สายทั้งหมดที่ใช้คือเจ็ดเส้น บางครั้งอาจมีการเพิ่มสายอีกสองเส้น เพื่อทำให้โมเด็มสามารถให้ข้อมูลมากขึ้นแก่คอมพิวเตอร์หรือเทอร์มินัล ได้แก่ CD (Carrier Detect) ถูกเชื่อมต่อเข้ากับขา 8 เพื่อแจ้งการคงอยู่ของสัญญาณพาหะ และ RI (Ring Indicator) ถูกเชื่อมต่อกับขา 22 เพื่อแสดงว่าโมเด็มกำลังถูกเรียกโดยอุปกรณ์ระยะไกล ซึ่งก็คือการตรวจสอบสัญญาณกริ่งของโทรศัพท์นั่นเอง จำนวนวงจรทั้งหมดจะกลายเป็นเก้า

แม้ว่ายังมีวงจรอื่นอีกหลายวงจรถูกกำหนดโดย RS-232-C แต่ทั้งเก้าวงจรนี้ถูกใช้กันมากที่สุด และเป็นเพียงชุดเดียวที่โดยปกติถูกเชื่อมต่อกับไมโครคอมพิวเตอร์ ซึ่งเป็นเหตุผลที่ไมโครคอมพิวเตอร์ใช้หัวต่อ 9 ขาแทน 25 ขา สำหรับนำพาสัญญาณที่จำเป็นทั้งหมดของวงจร RS-232-C

นัลโมเด็ม

ดังที่ได้กล่าวมาก่อนหน้านี้แล้วว่าในตอนแรก RS-232-C นำไปใช้สำหรับกำหนดการเชื่อมต่อระหว่างการเชื่อมต่อระหว่างเทอร์มินัลซึ่งเป็น DTE กับโมเด็มซึ่งเป็น DCE ต่อมาได้มีการนำไปประยุกต์ใช้กับการเชื่อมต่อระหว่างอุปกรณ์อื่นอีกหลายชนิดที่ไม่ได้ถูกกำหนดให้เป็น DTE หรือ DCE อย่างเป็นกิจลักษณะ เช่น ไมโครคอมพิวเตอร์ หรือเครื่องพิมพ์

เนื่องจากไม่มีมาตราฐานที่ชี้ชัดว่าอุปกรณ์ควรเป็น DTE หรือ DCE บ่อยครั้งจึงจำเป็นต้องเชื่อมต่ออุปกรณ์ DTE สองตัว หรืออุปกรณ์ DCE สองตัวเข้าด้วยกัน ในลักษณะเช่นนี้ต้องทำการเชื่อมต่อสาย 2 บนอุปกรณ์ตัวแรกกับสาย 3 บนอุปกรณ์ตัวที่สองและสาย 3 บนอุปกรณ์ตัวแรกกับสาย 2 บนตัวที่สอง สายแฮนด์เช็คกิ้งก็ต้องถูกไขว้ในทำนองเดียวกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การไขว้สายอาจทำได้โดยการต่ออุปกรณ์ด้วยสายสัญญาณซึ่งถูกไขว้ไว้แล้ว หรือโดยการซื้อหัวต่อพิเศษที่เชื่อมต่ออุปกรณ์ทั้งสอง และทำการไขว้สายที่จำเป็นไว้ภายใน ไม่ว่ากรณีใดสายที่อยู่ตรงกลางหรือหัวต่อจะถูกเรียกว่า *นัลโมเด็ม (null modem)* มันทำให้อุปกรณ์ DTE สองตัวคุยกันได้โดยไม่ต้องมีอุปกรณ์ DCE เป็นทางผ่าน หรือในทางกลับกันคือระหว่าง DCE กับ DCE

สัญญาณทางไฟฟ้า

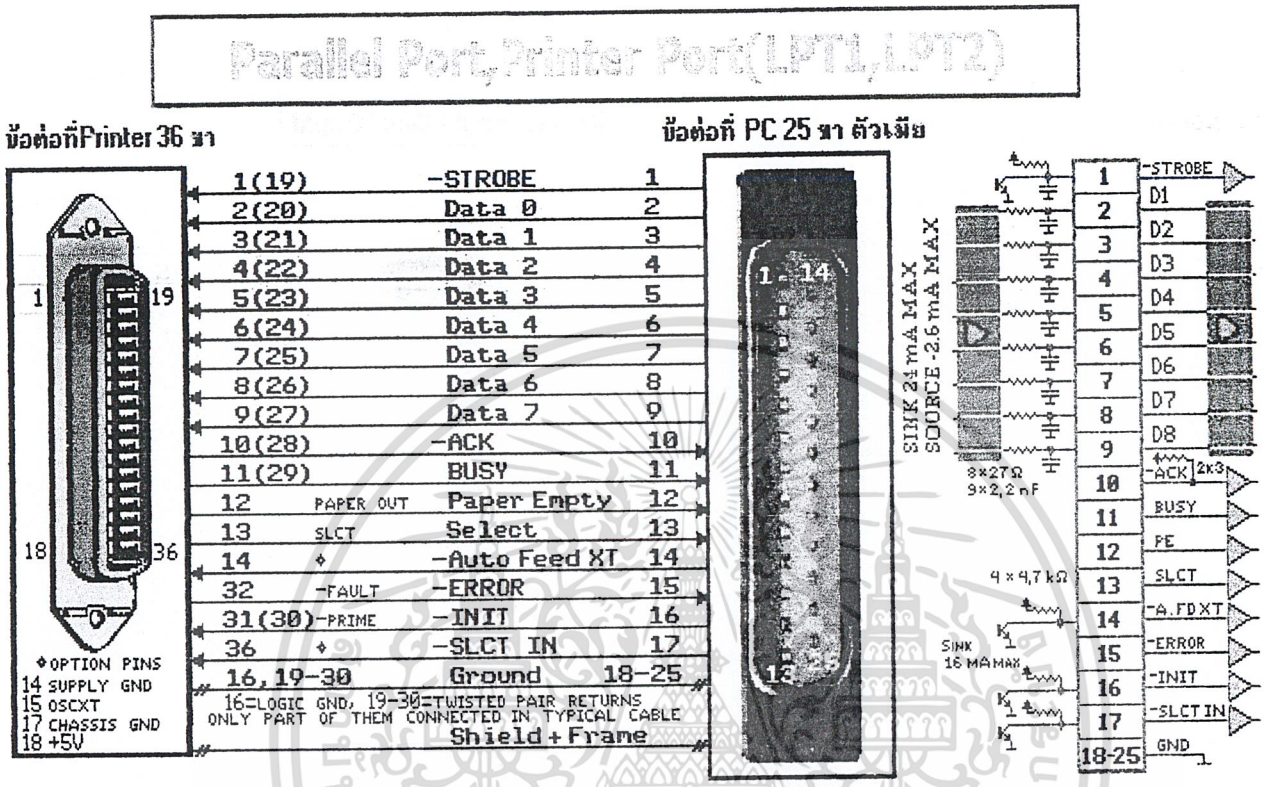
มาตรฐาน RS-232-C กำหนดคุณลักษณะของสัญญาณทางไฟฟ้าที่ถูกใช้ในการเชื่อมต่ออนุกรมโดยตรง มีเพียงสองลักษณะคือ SPACE แสดงถึงไบนารี 0 หรือแรงดันไฟฟ้าบอกและ MARK แสดงถึงไบนารี 1 หรือแรงดันไฟฟ้าลบ

บนสายข้อมูล (เช่นสาย 2 และ 3) แรงดันไฟฟ้าบวกแสดงถึงค่าลอจิก (logic) 0 และแรงดันไฟฟ้าลบแสดงถึงค่าลอจิก 1 บนสายแฮนด์เช็คกิ้ง (เช่น DTR และ DSR) แรงดันไฟฟ้าบวกแสดงว่าส่งข้อมูลได้ ส่วนแรงดันไฟฟ้าลบหมายถึงหยุดส่งข้อมูล

แรงดันไฟฟ้าบวก (สถานะ SPACE) อยู่ระหว่าง +5 ถึง +15 โวลต์สำหรับเอาต์พุต และระหว่าง +3 ถึง +15 โวลต์สำหรับอินพุต ความแตกต่างมีไว้เผื่อกรณีที่แรงดันไฟฟ้าสูญหายเนื่องจากความยาวของสายสัญญาณ ในทำนองเดียวกัน แรงดันไฟฟ้าลบ (สถานะ MARK) ถูกกำหนดไว้ระหว่าง –5 ถึง –15 โวลต์สำหรับเอาต์พุต และ –3 ถึง –15 โวลต์สำหรับอินพุต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4 PARALELL PORT



ตำแหน่งบิตและหน้าที่ของการทำงาน

Controls : X7A,X7E

Bit7,Bit6,Bit5 --> ไม่ใช้ครับ
Bit4 +IRQ Enable --> "1" อนุญาตให้ทำอินเตอร์รัพ
แต่ถ้าเกิด -->ACK จะเปลี่ยนจาก "1" เป็น "0"
Bit3 +SLCT IN -->"1" เลือกเครื่องพิมพ์
Bit2 -INIT -->50 ไมโครวินาที "0" สัญญาณรีเซทเครื่องพิมพ์
Bit1 +AUTO FD XT -->"1"กรณีเครื่องพิมพ์ทำ"Line-Feed"
Bit0 +Strobe --> 0.5 ไมโครวินาที ส่ง"1"ข้อมูลสัญญาณนาฬิกา
ไปที่เครื่องพิมพ์

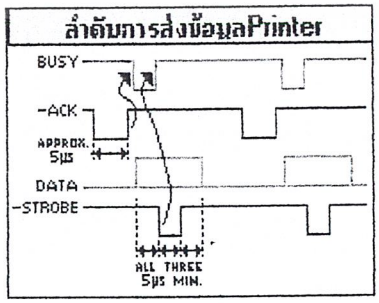
Data : X78,X7C Bits 7..0

Status : X79,X7D

Bit7 -Busy -->เมื่อเป็น"1"เครื่องพิมพ์ไม่รับข้อมูล
Bit6 -ACK -->"0" คือเครื่องพิมพ์พร้อมทำงาน
Bit5 +PE -->"1" คือเครื่องพิมพ์ไม่มีกระดาษ
Bit4 +SLCT -->"1" คือเลือกเครื่องพิมพ์
Bit3 -ERROR -->"0" คือเครื่องพิมพ์มีเงื่อนไขผิดผลาด
Bit2,Bit1,Bit0 --> ไม่ใช้ครับ

LPT1
I/O Address 378H-37FH Interrupt IRQ7
LPT2
I/O Adress 278H-27FH Interrupt IRQ5

PC ต่อกับ PC 25 ขา ขั้วต่อตัวเมีย				
2-15	3-13	4-12	5-10	6-11
15-2	3-13	12-4	10-5	11-6
25-25				



SINK คือ การต่อโหลดการแหล่งจ่ายโดยที่สัญญาณเป็นไฟบวก
SOURCE คือ การต่อโหลดการแหล่งจ่ายโดยที่สัญญาณเป็นไฟลบ

รูปที่ 2.12 แสดงลักษณะของParallel

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 เซนเซอร์อุณหภูมิ

LM35Dเป็นวงจรรวมที่ทำเป็นเซนเซอร์อุณหภูมิตรวจจับอุณหภูมิที่เป็นองศาเซลเซียส บรรจุอยู่ในตัวถังพลาสติกแบบ TO-92 โดยให้แรงดันเอาต์พุต มีความเป็นเส้นตรง(linear)ที่เป็นสัดส่วนกับอุณหภูมิองศาเซลเซียส ซึ่งมีข้อดีกว่าเซนเซอร์ที่ตรวจจับอุณหภูมิองศาเคลวินโดยผู้ใช้งานไม่ต้องทำการปรับแต่งและคำนวณเพื่อให้ได้ค่าองศาเซลเซียสออกมา ซึ่งมีคุณสมบัติดังนี้

- ให้ค่าอุณหภูมิองศาเซลเซียสโดยตรง
- มีความเป็นลิเนียร์ที่ +10.0 mV/ $^{\circ}$C
- มีความถูกต้องที่ $\pm$0.6 องศาเซลเซียสที่อุณหภูมิห้อง
- วัดค่าอุณหภูมิที่ช่วง – 55 ถึง +150 $^{\circ}$C
- ทำงานที่ย่านแรงดัน 4 ถึง 30 โวลท์
- เกิดความร้อนในตัวเองต่ำ 0.08 $^{\circ}$C
- ค่าอิมพีแดนซ์เอาต์พุตต่ำ 1 โอห์มที่โหลด 1 เมกะโอห์ม

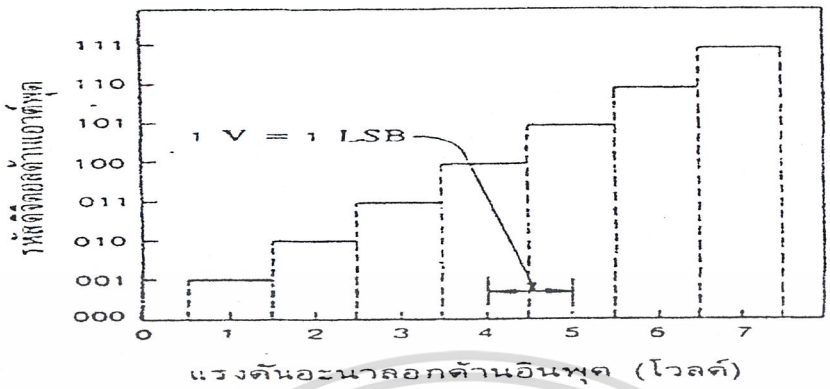
2.6 ตัวแปลงสัญญาณอนาลอกเป็นดิจิตอล

สัญญาณอนาลอกเป็นสัญญาณที่ไม่สามารถนำไปใช้กับคอมพิวเตอร์โดยตรงได้จึงจำเป็นต้องมีตัวแปลงสัญญาณอนาลอกให้เป็นสัญญาณทางดิจิตอล วงจรที่ทำหน้าที่ดังกล่าวจะเรียกว่า เอดีซี

หลักการของวงจรเอดีซี

หากนำวงจรเอดีซีขนาด 3 บิต มาเขียนกราฟระหว่างสัญญาณอินพุตและเอาต์พุต โดยสมมติว่าแรงดันอินพุตเปลี่ยนค่าจาก 0 –7 โวลท์ และได้สัญญาณเอาต์พุตที่เป็นสัญญาณดิจิตอลจาก 000-111 ดังรูปที่ 2.13

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.13 กราฟแสดงคุณสมบัติของวงจรเอดีซีขนาด 3 บิต

ค่าความละเอียดของเอดีซีที่ได้จากการเปลี่ยนแปลงแรงดันอินพุตแล้วทำให้ได้สัญญาณดิจิตอลเปลี่ยนค่าบิตนัยสำคัญต่ำสุดไป

ความละเอียด = ค่าแรงดันอินพุตต่อหนึ่งบิต = ค่าเต็มสเกลหารด้วย 2^n-1

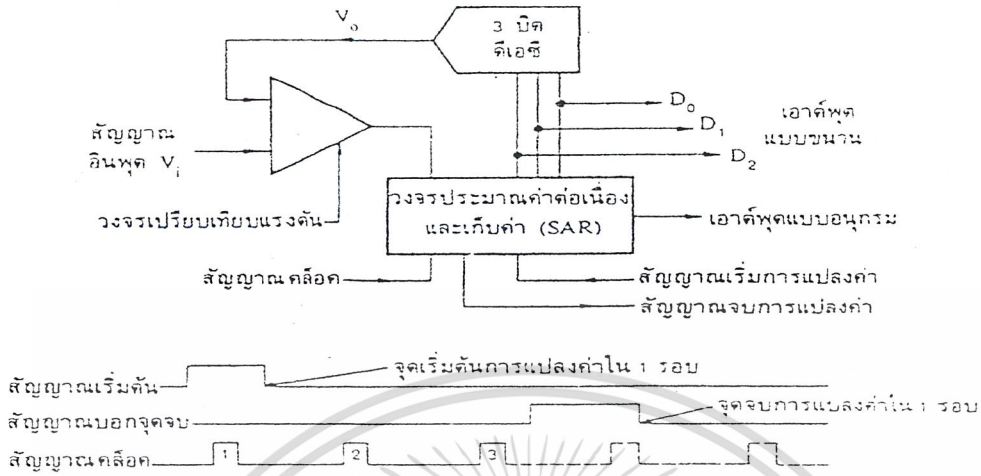
n คือ จำนวนบิตของวงจรเอดีซี

จากรูปที่2.13 จะเห็นได้ว่าขณะเอาต์พุตเป็น001 แรงดันอินพุตมีค่าเท่ากับ 1 โวลท์ ซึ่งค่านี้เกิดจากแรงดันค่าเฉลี่ยของ 0.5 โวลท์ กับ 1.5 โวลท์ ซึ่งค่าที่อยู่ระหว่างสองแรงดันนี้จะมีค่าเท่ากับ เอาต์พุตเท่ากับ 001 ดังนั้นหากต้องการให้ค่านี้ให้มีค่าผิดพลาดน้อยลงจำเป็นต้องเพิ่มค่าจำนวนบิตให้สูงขึ้น

เอดีซีแบบประมาณค่าต่อเนื่อง

เอดีซีแบบนี้ประกอบไปด้วยวงจรเอดีซี วงจรเปรียบเทียบแรงดันและวงจรรีจิสเตอร์เก็บค่าที่ได้หลังจากการประมาณค่าสัญญาณอินพุตที่เข้ามา โดยวงจรรีจิสเตอร์มีขาอินพุต 1 ขา และขาควบคุมอีก 3 ขา คือ ขาแรกเป็นขาสัญญาณบอกจุดเริ่มกระบวนการการแปลงค่า ขาที่ 2 เป็นขาบอกจุดจบกระบวนการแปลงค่า และขาสุดท้ายเป็นอินพุตรับสัญญาณคล็อกสำหรับควบคุมกระบวนการแปลงค่าแต่ละรอบ ดังรูปที่ 2.14

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่2.14 แสดงการทำงานของวงจรเอดีซีแบบประมาณค่าต่อเนื่อง

หลักการทำงาน

การแปลงค่าในแต่ละรอบจะเริ่มขึ้นเมื่อวงจรได้รับสัญญาณเริ่มต้น วงจรรีจีสเตอร์ส่งค่าดิจิตอลที่ได้ประมาณค่าแล้วออกไปวงจรเอดีซี เพื่อทำการแปลงค่าเป็นสัญญาณอนาลอกส่งกลับไปวงจรเปรียบเทียบแรงดันเทียบกับแรงดันอินพุตที่เข้ามาใหม่ว่าค่าใดมีค่ามากกว่ากัน เพื่อนำไปปรับค่าสัญญาณดิจิตอลแต่ละบิตให้ถูกต้องกับค่าที่ป้อนเข้ามายังอินพุต การเปรียบเทียบเริ่มจากบิตสูงไปยังบิตต่ำเสมอ เมื่อครบทุกบิตวงจร SAR จะส่งสัญญาณสิ้นสุดกระบวนการออกไป และจะได้นับสัญญาณดิจิตอลเอาต์พุตที่สัมพันธ์กับค่าแรงดันทางอินพุต

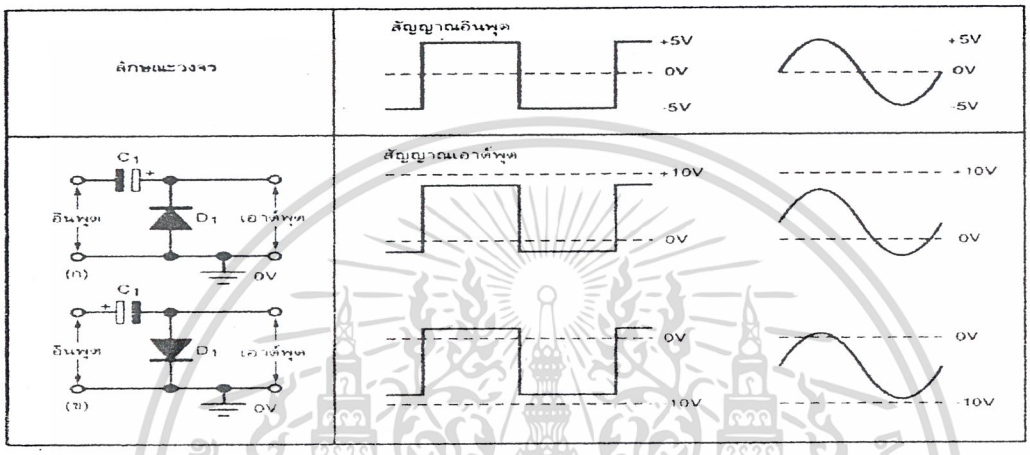
2.7 การจัดสัญญาณคลื่นด้วยไดโอด

คุณสมบัติของไดโอดในการเป็นสวิตซ์ออนและออฟ ก็จะอาศัยการไบอัสตรงหรือการไบอัสกลับนั่นเอง ไดโอดจะเริ่มนำกระแสด้วยค่าแรงดันไบอัสตรงค่าหนึ่งที่เรียกว่า ค่าแรงดันคัทอิน โดยปกติแรงดันนี้จะมีค่าประมาณ 0.6-0.7 โวลท์ สำหรับไดโอดชนิดซิลิกอน ไดโอดจะเสมือนลัดวงจรให้กระแสไหลผ่านหรืออยู่ในสภาวะออน ส่วนไบอัสกลับตัวไดโอดจะเสมือนอยู่ในสภาวะออฟ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

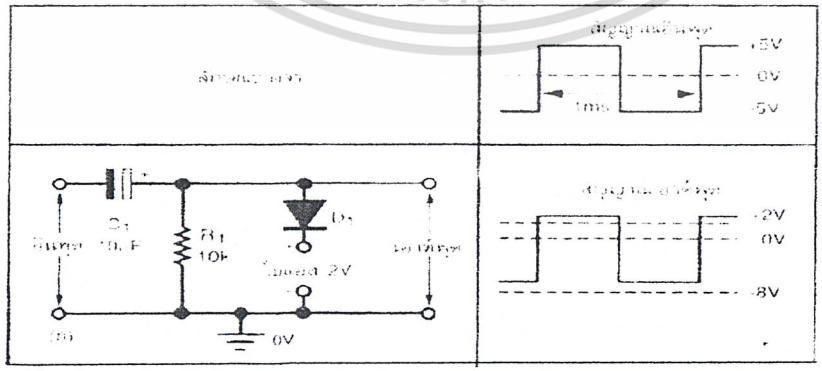
การใช้ไดโอดในวงจรแคลมป์

วงจรแคลมป์เป็นวงจรที่ทำการปรับระดับสัญญาณโดยพยายามคงรูปลักษณะสัญญาณเอาไว้ แต่ระดับแรงดันสูงสุดของสัญญาณเอาต์พุตจะถูกแคลมป์ไปอยู่ที่ระดับแรงดันอ้างอิงที่ศูนย์โวลท์ วงจรแคลมป์ดังรูปที่ 2.15



รูปที่ 2.15 แสดงลักษณะวงจรแคลมป์

จากรูปจะเห็นได้ว่าค่าแรงดันสูงสุดจะมีค่าเท่ากับแรงดันจากยอดถึงยอดของสัญญาณอินพุต ดังนั้นถ้าลักษณะของสัญญาณอินพุตที่มีช่วงสวิงทั้งช่วงบนและช่วงล่างอย่างสมมาตร ค่าแรงดันสูงสุดของสัญญาณเอาต์พุตจะมีค่าเป็นสองเท่าของแรงดันอินพุต ถ้าต้องการให้วงจรแคลมป์ไปยังระดับอ้างอิงอื่นๆนอกจากศูนย์ สามารถทำได้โดยการให้แรงดันไบอัสที่เหมาะสมแก่ไดโอด ดังรูปที่ 2.16



รูปที่ 2.16 แสดงวงจรแคลมป์ที่มีการไบอัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.8 ออปแอมป์

ออปแอมป์เป็นวงจรรวมที่ทำจากสารกึ่งตัวนำ สามารถนำไปสร้างวงจรต่างๆได้มากมาย มีคุณสมบัติและพารามิเตอร์ทั่วไปดังนี้

- อัตราการขยายแรงดันสูง
- อินพุตอิมพีแดนซ์สูง
- เอาท์พุทอิมพีแดนซ์ต่ำ
- การไบอัสทางอินพุตจะใช้กระแสต่ำ
- แรงดันออฟเซ็ททางอินพุตมีค่าเป็นศูนย์
- การตอบสนองความถี่ขึ้นอยู่กับออปแอมป์แต่ละตัวสามารถดูในคาต้าชีท
- อัตราสลูว์เป็นค่าที่บอกความไวต่อการเปลี่ยนแปลงระดับสัญญาณของแอปแอมป์

จากคุณสมบัติของออปแอมป์นำมาออกแบบเป็นวงจรแรงดันตามหรือเรียกว่าบัฟเฟอร์ ซึ่งเป็นวงจรที่มีอิมพีแดนซ์ทางอินพุตที่สูงและใช้กระแสที่ไหลเข้าทางอินพุตต่ำ แรงดันทางด้านอินพุตมีค่าเท่ากับแรงดันทางด้านเอาท์พุต มีอัตราการขยายเท่ากับหนึ่ง

2.9 Visual Basic

ประวัติความเป็นมา

Visual Basic เป็นภาษาคอมพิวเตอร์ที่พัฒนาโดยบริษัทไมโครซอฟท์ โดยที่ตัวภาษาเองมีรากฐานมาจากภาษา Basic (Beginner 's All purpose Symbolic Instruction) ไมโครซอฟท์ได้พัฒนาโปรแกรมภาษา Basic มานานนับสิบปี ตั้งแต่ภาษา MBASIC (Microsoft Basic) , BASICA (Basic Advanced) , GWBASIC และ Quick Basic ซึ่งได้ติดตั้งมาพร้อมกับระบบปฏิบัติการ MS DOS ในที่สุดโดยใช้ชื่อว่า QBASIC

Visual Basic เวอร์ชันแรกคือ เวอร์ชัน 1.0 ออกสู่สายตาประชาชนตั้งแต่ปี 1991 โดยในช่วงแรกนั้นยังไม่มีความสามารถต่างจากภาษา QBASIC มากนัก แต่จะเน้นเรื่องเครื่องมือที่ช่วยในการเขียนโปรแกรมบนวินโดวส์

สำหรับ Visual Basic ในปัจจุบันคือ เวอร์ชัน 6.0 ซึ่งออกมาในปี 1998 ได้เพิ่มความสามารถในการเขียนโปรแกรมติดต่อกับเครือข่ายอินเตอร์เน็ต การเชื่อมต่อกับระบบฐานข้อมูล รวมทั้งปรับปรุงเครื่องมือและการเขียนโปรแกรมเชิงวัตถุ (Object Oriented Programming) ให้สมบูรณ์ยิ่งขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทำไมจึงต้อง Visual Basic

เหตุผลที่เลือก Visual Basic เป็นภาษาในการเขียนโปรแกรม เนื่องจาก Visual Basic มีข้อดีหลายประการ คือ ง่ายต่อการเรียนรู้เหมาะสำหรับผู้เริ่มต้น ทั้งในเรื่องไวยากรณ์ ของภาษาเอง และเครื่องมือการใช้งานความนิยมของตัวภาษา โดยอาจกล่าวได้ว่า ภาษา Basic นั้น เป็นภาษาที่มีคนเรียนรู้และใช้งานมากที่สุดการพัฒนาอย่างต่อเนื่อง การปรับปรุงประสิทธิภาพในด้านของตัวภาษาและความเร็วของการประมวลผล และในเรื่องความสามารถใหม่ๆ เช่น การติดต่อกับระบบฐานข้อมูล การเชื่อมต่อกับเครือข่ายอินเตอร์เน็ต ผู้พัฒนาสำคัญของ Visual Basic คือบริษัทไมโครซอฟท์ ซึ่งจัดว่าเป็นยักษ์ใหญ่ของวงการคอมพิวเตอร์ในปัจจุบัน จึงมั่นใจได้ว่า Visual Basic จะยังมีการพัฒนาปรับปรุงและคงอยู่ไปอีกนาน

มีอะไรใหม่ใน Visual Basic 6.0

ความแตกต่างของ Visual Basic เวอร์ชัน 5.0 และ 6.0 ที่เห็นได้อย่างชัดเจนคือ ส่วนที่ใช้ในการพัฒนาโปรแกรมที่เกี่ยวข้องกับงานด้านฐานข้อมูล และโปรแกรมที่พัฒนาเพื่อให้ทำงานบนอินเตอร์เน็ต เช่น Dynamic HTML ส่วนในด้านอื่นๆก็เพิ่มความสามารถในการคอมไพล์โปรแกรมเป็น Native Code ซึ่ง Visual Basic 5.0 ไม่สามารถทำได้

ขั้นตอนการออกแบบและเขียนโปรแกรม

สามารถแบ่งการพัฒนาและเขียนโปรแกรมออกเป็นขั้นตอนหลักๆได้ดังนี้ คือ

ขั้นตอนที่ 1 ศึกษาความต้องการของผู้ใช้ และกำหนดวัตถุประสงค์ของโปรแกรม

ขั้นตอนที่ 2 ออกแบบหน้าจอของโปรแกรมที่เราต้องการ พร้อมกำหนดคุณสมบัติและส่วนประกอบต่างๆ ของโปรแกรมโดยละเอียด

ขั้นตอนที่ 3 เริ่มเขียนโปรแกรม

ขั้นตอนที่ 4 รวบรวมโปรแกรม แปลงให้อยู่ในรูปแบบที่พร้อมใช้งานและทดสอบการทำงานของโปรแกรม

ขั้นตอนที่ 5 แจกจ่ายโปรแกรมของเราสู่มือของผู้ใช้งาน

2.10 Telnet

การติดต่อใช้งานระหว่างคอมพิวเตอร์สองระบบด้วย TCP/IP นั้น การเชื่อมต่อคอมพิวเตอร์จากระยะไกลผ่านโมเด็มและสายโทรศัพท์จะใช้ Telnet เป็นโปรแกรมติดต่อกับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คอมพิวเตอร์ปลายทาง โดย Telnet จะเป็นการใช้งานในลักษณะ Remote Terminal คือคำสั่งต่างๆที่พิมพ์จากแป้นพิมพ์ของเราจะถูกส่งไปให้คอมพิวเตอร์ปลายทางทำงานแทน และส่งข้อความแสดงผลบนจอภาพกลับมาให้เราเท่านั้น ดังนั้นการทำงานต่างๆจึงเกิดขึ้นที่เครื่องปลายทางทั้งสิ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การดำเนินการวิจัย

การทำงานในโครงงานพิเศษนี้จะแบ่งได้เป็น 3 ส่วนใหญ่ โดยส่วนแรกจะเป็นส่วนของฮาร์ดแวร์ (Hardware) ซึ่งจะประกอบไปด้วยส่วนของอุปกรณ์ในการตรวจวัดอุณหภูมิ และส่วนตรวจวัดแรงดันไฟฟ้า ส่วนที่สองจะเป็นในส่วนของซอฟต์แวร์ (Software) ซึ่งจะเป็นการเขียนโปรแกรมเพื่อควบคุมการตรวจวัดทั้งสองแบบ คือ การตรวจวัดอุณหภูมิ และแรงดันไฟฟ้า โดยอาศัยโปรแกรมภาษา C และส่วนที่สามจะเป็นการติดต่อสื่อสารข้อมูลผ่านระบบเครือข่ายอินเตอร์เน็ตโดยอาศัยการใช้โปรแกรม Visual Basic

3.1 การออกแบบวงจรในส่วนของฮาร์ดแวร์ (Hardware)

3.1.1 ไมโครคอนโทรลเลอร์

การต่อวงจรการใช้งานไมโครคอนโทรลเลอร์ AVR วงจรเซนเซอร์อุณหภูมิและวงจรวัดแรงดันไฟฟ้าดังรูปที่ 3.1 วงจรไมโครคอนโทรลเลอร์ใช้คริสตอลที่ความถี่ 8 เมกะเฮิร์ตซ ต่อแหล่งจ่ายไฟ5โวลท์ให้กับขา VCC และ AVCC ซึ่งเป็นแรงดันไฟฟ้าที่ใช้กับวงจรเอดีซีที่อยู่ภายในไมโครคอนโทรลเลอร์ สามารถติดต่อกับคอมพิวเตอร์ผ่านทางพอร์ตอนุกรม RS-232 ที่BAUD RATE 9600 และทำการโปรแกรมข้อมูลลงหน่วยความจำแบบFLASHและEEPROMผ่านสายสัญญาณ MOSI, MISO, SCK และ RESET ซึ่งจะเป็นการโปรแกรมแบบอนุกรม ใช้พอร์ต A ของไมโครคอนโทรลเลอร์ที่ออกแบบเป็นวงจรเอดีซีแบบ Successive Approximation รับค่าสัญญาณอนาลอกโดยมีการสุ่มของสัญญาณที่ความเร็วประมาณ 118 ไมโครวินาที

3.1.2 วงจรตรวจวัดอุณหภูมิ

วงจรเซนเซอร์อุณหภูมิใช้วงจรรวมเบอร์LM35Dที่เป็นเซนเซอร์ที่แปลงค่าอุณหภูมิเป็นแรงดันไฟฟ้าที่ 10 มิลลิโวลท์ต่อองศาเซลเซียส ต่อเข้าไมโครคอนโทรลเลอร์ที่พอร์ตA(ADC0) ซึ่งทำการวัดแรงดันได้ที่ช่วงอุณหภูมิ 0 – 100 องศาเซลเซียส

3.1.3 วงจรวัดแรงดันไฟฟ้า

วงจรวัดแรงดันไฟฟ้าใช้หม้อแปลงเป็นตัวลดแรงดันให้มีค่าแรงดันสัญญาณที่เป็นแบบ Sinusoidal ที่มีแอมพลิจูดยอดถึงยอดไม่เกิน5โวลท์เพื่อให้ค่าแรงดันนี้ไม่เกินค่าแรงดันอินพุตของวงจรเอดีซีที่อยู่ในไมโครคอนโทรลเลอร์ ใช้ออปแอมป์ทำหน้าที่เป็นวงจรแรงดันตามให้อัตราขยายเท่ากับหนึ่งป้องกันกระแสที่ได้จากหม้อแปลงกรณีที่หม้อแปลงเกิดเสียหาย และทำการยก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระดับแรงดันโดยใช้วงจรแคลมป์ให้สัญญาณแรงดันไฟฟ้าสลับเข้าวงจรเอดีซีมีทั้งแรงดันทางด้านบวกและลบที่มีความสูงของแรงดันเท่ากัน การวัดค่าแรงดันยังผลหรือ Vrms จะวัดได้โดยใช้สมการ 1และ2ซึ่งเป็นสมการที่วัดแรงดันยังผลของไฟฟ้าที่จ่ายให้กับตามบ้านเรือนต่างๆ ค่าแรงดันนี้จะอยู่ในช่วงที่ไม่เกิน 210 ถึง 230 โวลท์ในเวลาปกติ

$$V_{rms} = \sqrt{\frac{1}{T}\int_{t1}^{t2} v^2 dt} \qquad 1)$$

$$\int_{t1}^{t2} v^2 dt = \sum_{n=0}^{169} v^2 \Delta t \qquad 2)$$

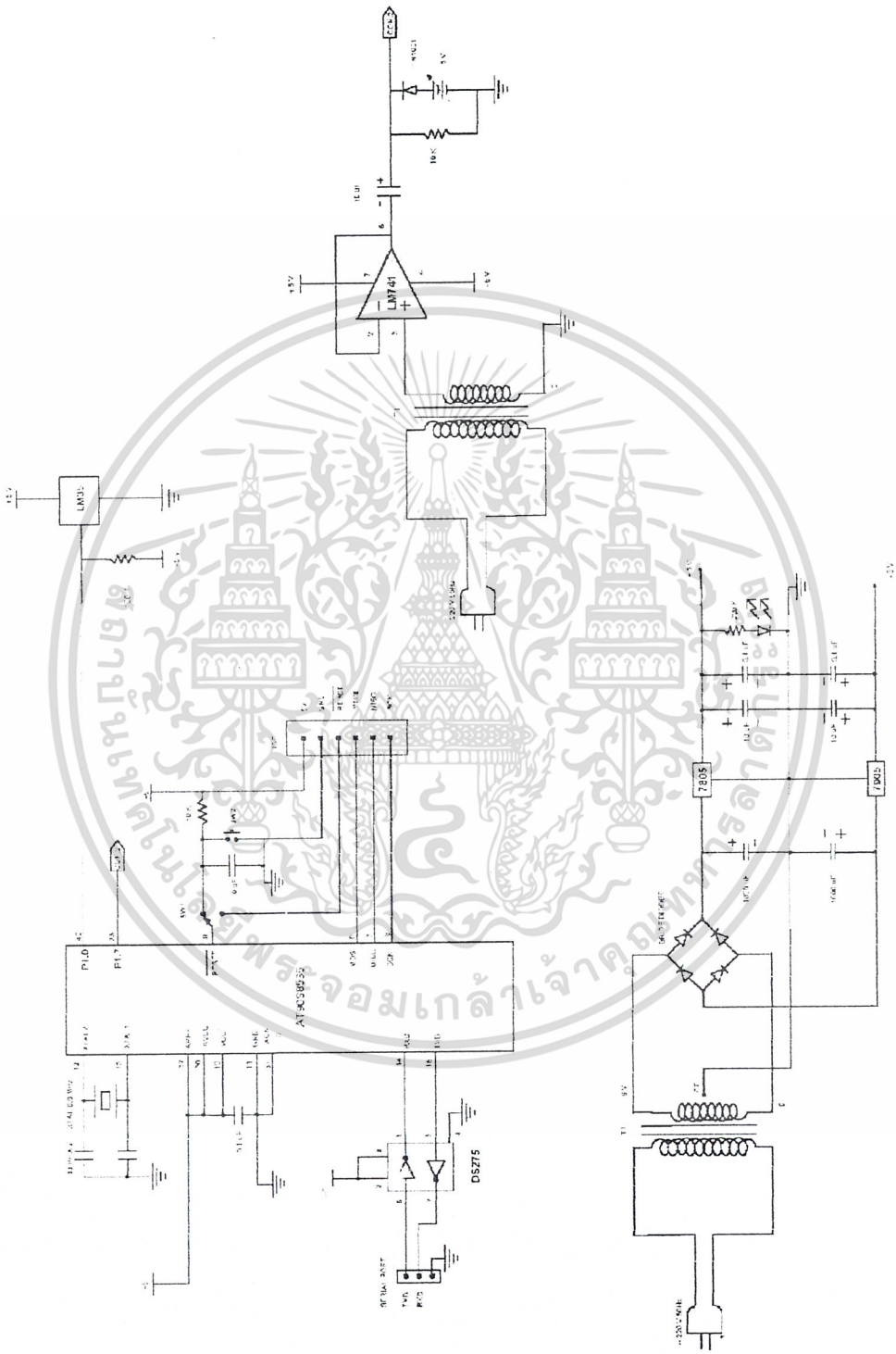
v^2 คือค่าแรงดันที่วัดได้จากวงจรเอดีซียกกำลังสอง

Δt คือค่าช่วงเวลาของการในการSampling สัญญาณของวงจรเอดีซีหนึ่งค่า มีค่าประมาณเท่ากับ 117.964 ไมโครวินาที

T คือคาบเวลาของสัญญาณคลื่นไซน์ในแรงดันไฟฟ้าที่ความถี่ 50 Hz มีค่าเท่ากับ 20 มิลลิวินาที

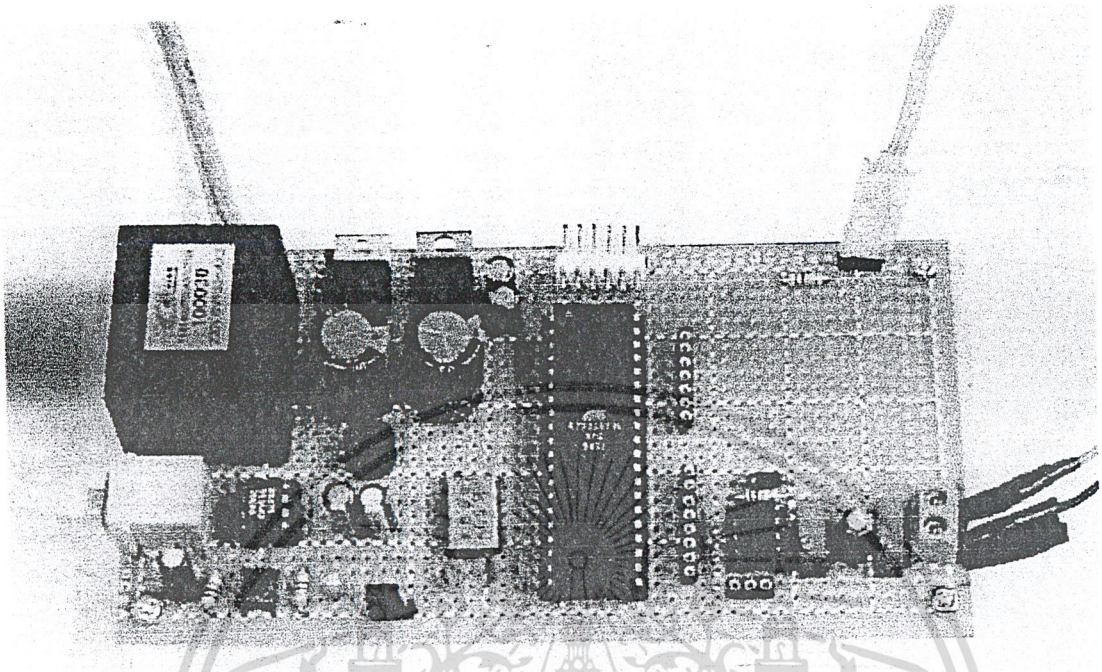
ในสมการที่ 2 เป็นสมการที่หาพื้นที่ใต้เส้นโค้งกราฟของสัญญาณไฟฟ้าคลื่นไซน์ที่ช่วงหนึ่งคาบเวลา ซึ่งวงจรเอดีซีในไมโครคอนโทรลเลอร์ทำการSampling สัญญาณได้ทั้งหมด 169 ค่าค่าที่ได้นี้จะเป็นค่าผลรวมทั้งหมดของพื้นที่ใต้กราฟซึ่งจะนำไปแทนในสมการที่ 1 จะทำให้สามารถหาค่าแรงดันยังผลได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่3.1 แสดงการออกแบบที่ใช้ในการทดลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

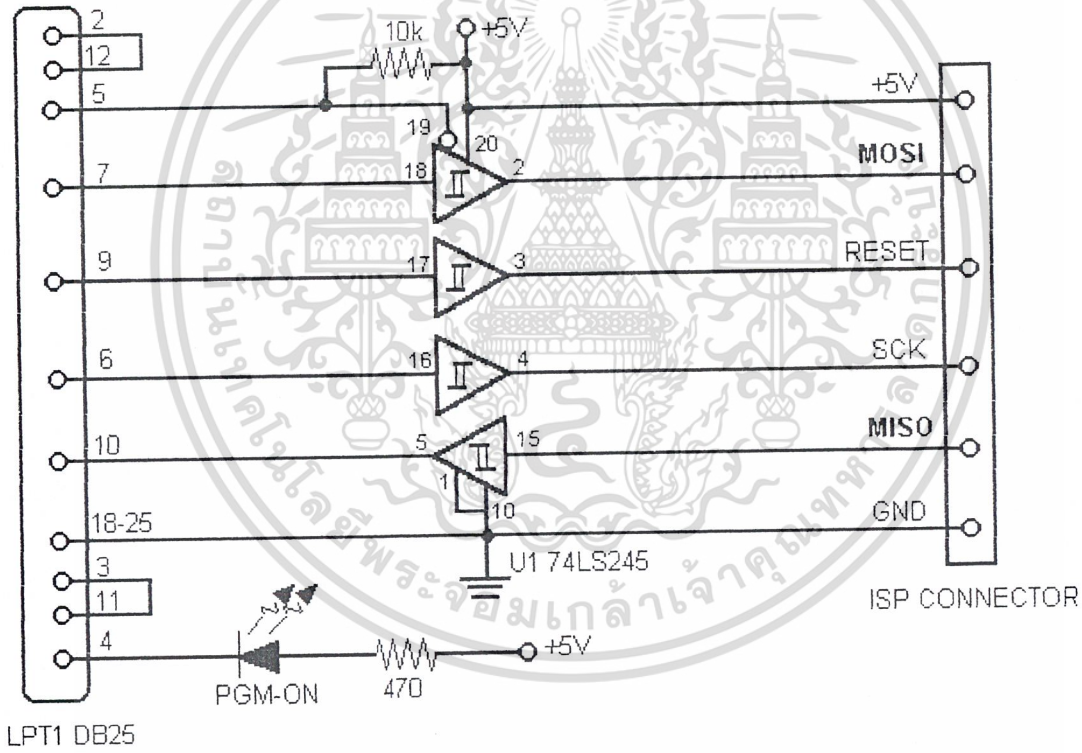


รูปที่ 3.2 แสดงบอร์ดทดลอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

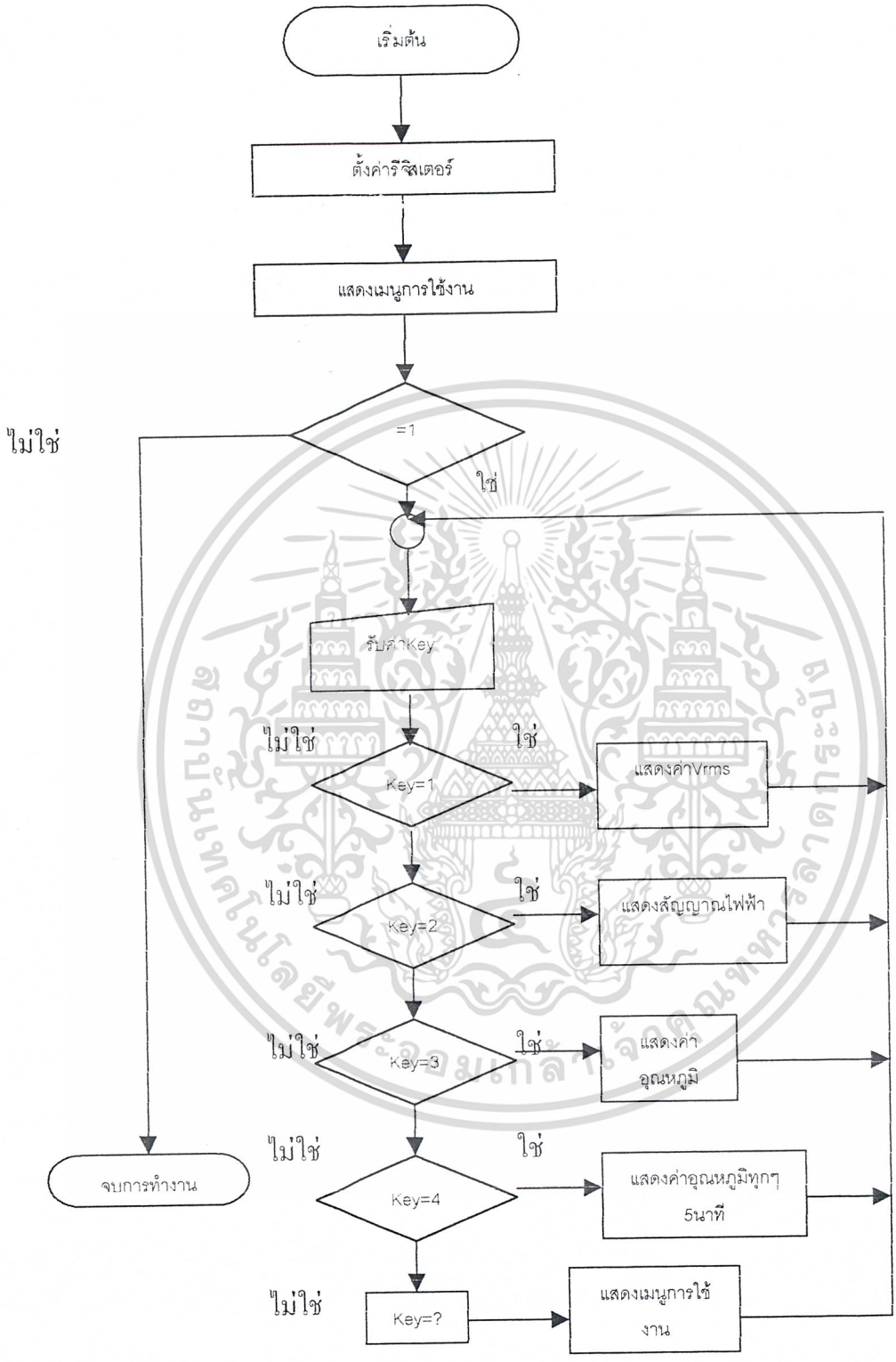
3.2 การออกแบบวงจรในส่วนของซอฟต์แวร์(Software)ที่ใช้กับไมโครคอนโทรลเลอร์

การเขียนโปรแกรมได้ใช้โปรแกรมโค้ดวิชั่น ทำการเขียนโปรแกรมภาษาซีใช้กับไมโครคอนโทรลเลอร์ซึ่งโปรแกรมนี้จะทำการแปลงภาษาซีเป็นภาษาแอสเซมบลีที่ใช้กับไมโครคอนโทรลเลอร์ตระกูลAVR และการอัดโปรแกรมลงหน่วยความจำของไมโครคอนโทรลเลอร์ก็สามารถทำได้โดยผ่านโปรแกรมนี้ผ่านทางโปรแกรมนี้ แต่ต้องทำการต่อสายดังรูปที่ 3.3 ซึ่งจะนำไปต่อเข้ากับพอร์ตขนานของเครื่องคอมพิวเตอร์และสายสายสัญญาณ MOSI, MISO, SCK และ RESET ของไมโครคอนโทรลเลอร์ ในการทำงานของโปรแกรมจะมีขั้นตอนดังนี้รูปที่ 3.4



รูปที่ 3.3 แสดงการต่อสายสัญญาณ ISP

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่3.4 แสดงโฟว์ชาร์ทของโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 การออกแบบในส่วนที่ติดต่อสื่อสารผ่านระบบเครือข่ายอินเตอร์เน็ต

การติดต่อสื่อสารเป็นสิ่งจำเป็น เพราะเป็นส่วนที่ทำหน้าที่เพื่อรับ-ส่งข้อมูลต่างๆที่ได้ทำการตรวจวัดไปให้ผู้ใช้ปลายทางทราบค่าที่ต้องการ โดยในที่นี้เป็นการสื่อสารที่อาศัยการส่งข้อมูลผ่านระบบเครือข่ายอินเตอร์เน็ต ซึ่งจะเป็นการอาศัยโมเด็มสำหรับเครื่องปลายทางที่อยู่นอกองค์กร หรืออาศัยการ์ด LAN สำหรับเครื่องปลายทางที่อยู่ภายในองค์กรเดียวกัน ซึ่งในการสื่อสารประเภทนี้จะประกอบไปด้วยเครื่องที่ทำหน้าที่เป็น Server ซึ่งเป็นเครื่องหลักที่ต่ออยู่กับอุปกรณ์ตรวจวัดโดยที่เครื่อง Server นี้จะมีการเขียนโปรแกรม Terminal โดยใช้ Visual Basic และเครื่องที่ทำหน้าที่เป็น Client ซึ่งเป็นเครื่องที่สามารถเข้ามาทำงานกับเครื่อง Server ได้โดยผ่านระบบเครือข่ายทางอินเตอร์เน็ต โดยที่เครื่อง Client นี้ไม่จำเป็นต้องติดตั้งโปรแกรมใดๆลงไปเพื่อความสะดวก เราจึงใช้โปรแกรมพื้นฐานที่มีอยู่ใน OS หลักๆทั่วๆไปที่ใช้ในปัจจุบันนี้ ในที่นี้คือโปรแกรม Telnet

ในส่วนนี้จะกล่าวถึงการเขียนโปรแกรม Visual Basic เป็นหลัก ในการทำการอินเตอร์เฟส (Interface) ระหว่างบอร์ดไมโครคอนโทรลเลอร์ (Microcontroller) ที่ทำหน้าที่ในการตรวจวัดกับคอมพิวเตอร์ (Computer)

การออกแบบซอฟต์แวร์ในส่วนของ Server

การทำงานในส่วนของ Server จะเป็นการทำหน้าที่รับ – ส่งข้อมูลกับบอร์ดไมโครคอนโทรลเลอร์ที่ทำหน้าที่ในการตรวจวัดอุณหภูมิกับตรวจวัดแรงดันไฟฟ้า ซึ่งในการออกแบบซอฟต์แวร์ในส่วนนี้จะเป็นการออกแบบโดยอาศัยโปรแกรม Visual Basic ซึ่งมีข้อดีคือ เป็นโปรแกรมที่สะดวกแก่การใช้งานในลักษณะ User Interface (การติดต่อกับผู้ใช้) เนื่องจากมีลักษณะหน้าตาคล้ายๆกับ Window ทั่วไป

ส่วนประกอบต่างๆในการออกแบบในส่วนของ Server จะแบ่งออกเป็น 4 ส่วนย่อยๆ ดังนี้

1. Form ส่วนที่มีหน้าที่ในการติดต่อกับบอร์ดไมโครคอนโทรลเลอร์ ในที่นี้มีชื่อว่า frmTerminal ซึ่งใน Form นี้จะประกอบไปด้วย
   - 1.1 ส่วนที่ติดต่อกับบอร์ดไมโครคอนโทรลเลอร์โดยผ่านทาง Port Com (ตามมาตราฐาน RS-232-C) ซึ่งใน Visual Basic จะใช้ Components ที่เรียกว่า Ms Comm Control
   - 1.2 ส่วนที่ทำหน้าที่ในการนับค่าเวลาต่างๆ ไม่ว่าจะเป็นระยะเวลาในการ Connect หรือเวลาในการทำงานร่วมกับบอร์ดไมโครคอนโทรลเลอร์ ซึ่งจะเรียกส่วนนี้ว่า Timer
   - 1.3 ส่วนที่ทำหน้าที่ในการแสดงผลของค่าที่วัดได้ออกมาในรูปของกราฟแสดงผล ซึ่งจะเป็นส่วนที่ทำหน้าที่ในการประมวลผลของข้อมูลส่วนที่เราต้องการนำมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Plot Graph โดยจะนำข้อมูลที่รับเข้ามาเก็บอยู่ในรูปของ Array แล้วนำค่ามาเรียงแล้วแสดงออกเป็นกราฟ ซึ่งจะเรียกส่วนนี้ว่า Ms Chart Control

1.4 ส่วนที่ทำหน้าที่ในการสร้าง Dialog (เป็นเมนูย่อยๆอีกประเภท) ซึ่งจะช่วยในการเพิ่มความสะดวกในการทำงาน ซึ่งเรียกว่า Ms Common Dialog

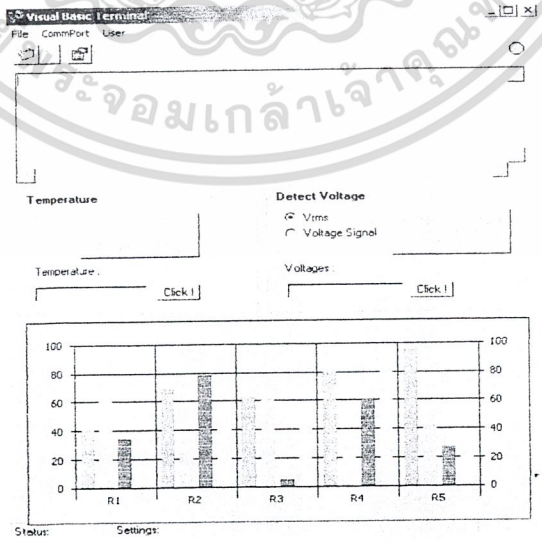
1.5 ส่วนที่ทำหน้าที่ในการรับ-ส่งข้อมูลผ่านระบบเครือข่ายอินเตอร์เน็ต เป็นส่วนที่จะทำการ Link ข้อมูลไปยังเครื่องปลายทาง (Client) ได้ ซึ่งในการทำงานต้องทำการเช็ค Port ที่รับส่งข้อมูลให้ตรงกับ Port Telnet ของมาตราฐาน OS ทั่วๆไป ซึ่งจะเรียกว่า Ms Winsock Control

ส่วนประกอบทั้ง 5 ส่วนที่ได้กล่าวไปนั้นเป็นส่วนของโปรแกรมอย่างเดียวไม่ได้ทำหน้าที่เป็น User Interface

1.6 Frame ที่มีชื่อว่า Temperature ซึ่งจะทำหน้าที่ในการตรวจวัดอุณหภูมิซึ่งใน frame นี้ยังไป Link กับ From อื่นที่เรียกว่า Form Command ที่เป็น Option ของมันได้อีกด้วย

1.7 Frame ที่มีชื่อว่า Detect Voltage ซึ่งทำหน้าที่ในการตรวจวัดแรงดันไฟฟ้าโดยสามารถเลือกที่จะวัดในส่วนที่เป็น Vrms และสามารถดูส่วนที่เป็นสัญญาณของแรงดันไฟฟ้าได้ และยังสามารถนำมา Plot Graph มาดูได้ด้วย

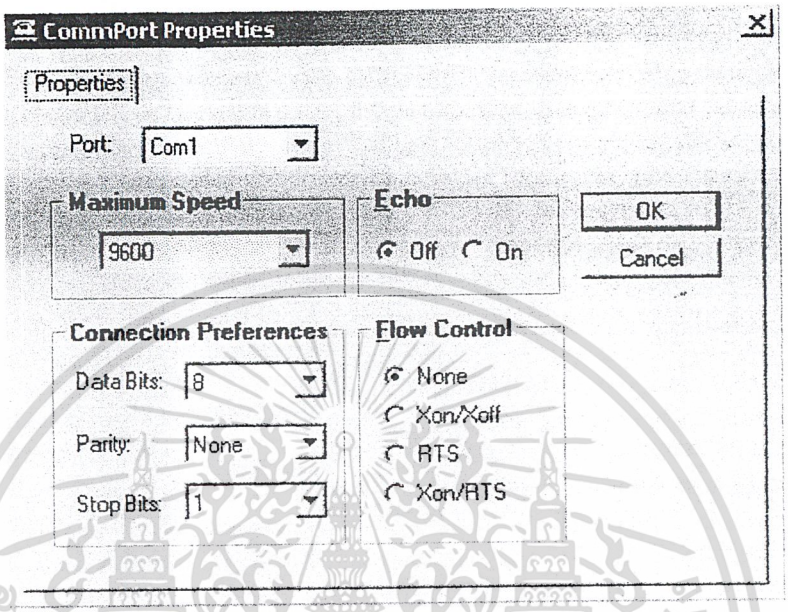
1.8 ส่วนที่เป็น Textbox ทำหน้าที่ในการแสดงค่าต่างๆ หรือผลของการติดต่อกับบอร์ดไมโครคอนโทรลเลอร์ และยังแสดงค่าวัดต่างๆได้เช่นเดียวกัน



รูป 3.5 แสดงหน้าจอ Terminal

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. Form ส่วนที่ทำหน้าที่เป็น Properties ในการเลือก Option ต่างๆในการติดต่อกับบอร์ดไมโครคอนโทรลเลอร์ ในที่นี้มีชื่อว่า frmProperties ซึ่งจะประกอบไปด้วย



รูปที่ 3.6 แสดง From Properties

2.1 ส่วนที่เลือก Port ต่างๆของ Com ได้ตั้งแต่ Com1-Com16 แล้วแต่การใช้งาน ซึ่งมีชื่อว่า cboPort

2.2 Frame ที่มีชื่อว่า Maximum Speed ซึ่งเป็นส่วนที่ทำหน้าที่ในการเลือกความเร็วในการรับส่งข้อมูลซึ่งความเร็วนี้จะต้อง Match กับอุปกรณ์ที่เรานำไปต่อจึงจะสามารถรับส่งข้อมูลกันได้ ซึ่งสามารถเลือกความเร็วได้ตั้งแต่ 110 – 256000 ซึ่งเรียกว่า cboSpeed

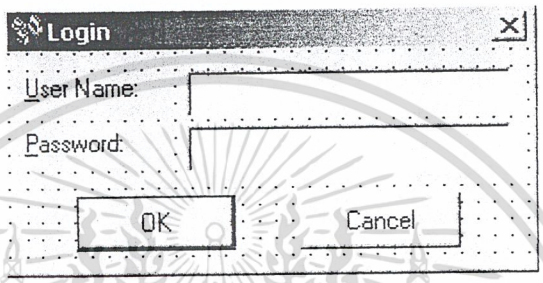
2.3 Frame ที่มีชื่อว่า Connection Preferences ซึ่งเป็นส่วนกำหนดค่าพื้นฐานต่างๆของการส่งข้อมูลซึ่งการจะเซ็ตค่าต่างๆจะต้องเป็นไปตามบอร์ดไมโครคอนโทรลเลอร์ที่ต่อด้วยซึ่งจะเซ็ตค่าการส่งข้อมูลให้ตรงกัน โดยจะประกอบไปด้วย Data Bits, Parity, Stop Bits

2.4 Frame ที่มีชื่อว่า Echo ซึ่งเป็นส่วนทำหน้าที่ในการกำหนดการสะท้อนการรับ-ส่งข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 Frame ที่มีชื่อว่า Flow Control ซึ่งเป็นส่วนที่เช็คแฮนด์เช็คกิ้งเป็นวิธีที่อุปกรณ์ฝ่ายรับใช้ควบคุมการไหลของข้อมูลจากอุปกรณ์ฝ่ายส่ง ซึ่งบางครั้งการแฮนด์เช็คกิ้งนี้อาจจะมาจากอุปกรณ์เสริมได้

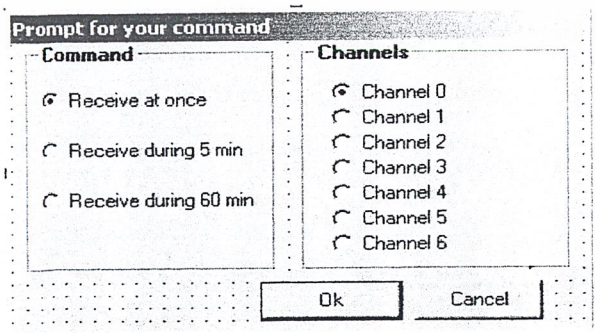
3. Form ส่วนที่ทำหน้าที่ในการขานรับการอนุญาติขอใช้ทำงาน หรือส่วนที่เป็นการล็อคการเข้ามาทำงานในโปรแกรม ซึ่งมีชื่อเรียกว่า frmLogin



รูปที่ 3.7 แสดง From Login

ซึ่งจะประกอบไปด้วย txtUseName ที่ทำหน้าที่รับ Login และส่วนที่เป็น txtPassword ที่ทำหน้าที่รับส่วนที่เป็น password แล้ว โดยการใช้ DBCombo Box Control ซึ่งเป็น Control ที่ Link กับฐานข้อมูลที่เก็บ username กับ password ไว้ โดยฐานข้อมูลที่ใช้คือ Microsoft Access 97 นั่นคือเราก็จะสามารถที่จะเปลี่ยน Password ได้จากฐานข้อมูลนี้

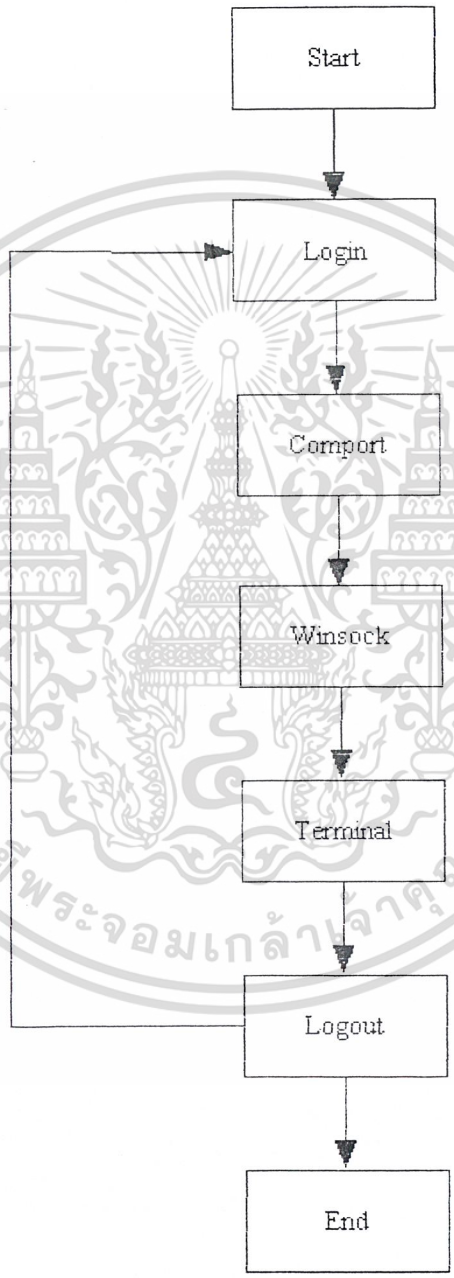
4. From ส่วนที่ทำหน้าที่ในการเลือกฟังก์ชั่นต่างๆของการวัดอุณหภูมิ ซึ่งมีชื่อว่า frmCommand



รูปที่ 3.8 แสดง From Command

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในส่วนนี้จะเป็นการเลือกรายละเอียดในการวัดอุณหภูมิซึ่งเราจะสามารถวัดค่าอุณหภูมิจาก channel ต่างๆจากบอร์ดได้ และยังสามารถที่จะเลือกวัดอุณหภูมิแบบต้องครั้งที่กดเพื่อต้องการทราบค่า แบบวัดทุกๆ 5 นาที หรือแบบวัดทุกชั่วโมงได้



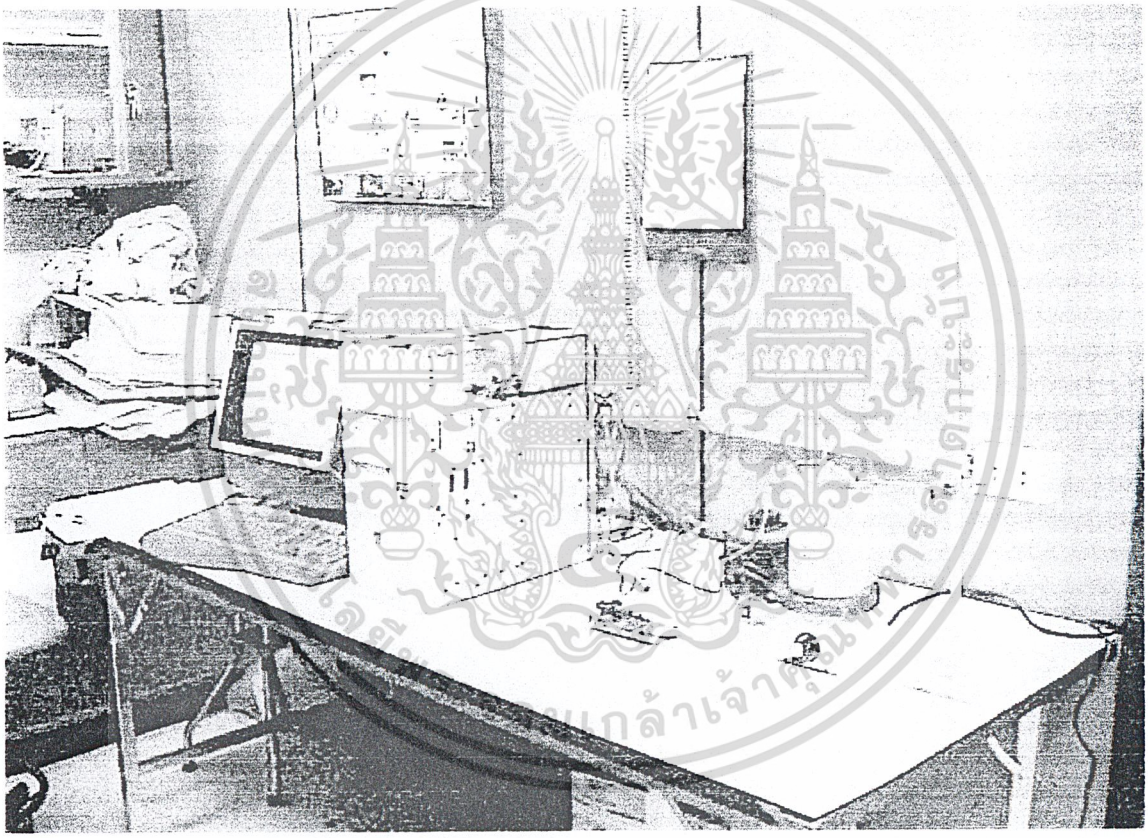
รูปที่3.9 แสดงโฟว์ชาร์ทของโปรแกรม Visual Basic

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

ผลการทดลอง

4.1 ผลการวัดแรงดันไฟฟ้า



รูปที่ 4.1 การทดลองวัดแรงดันไฟฟ้า

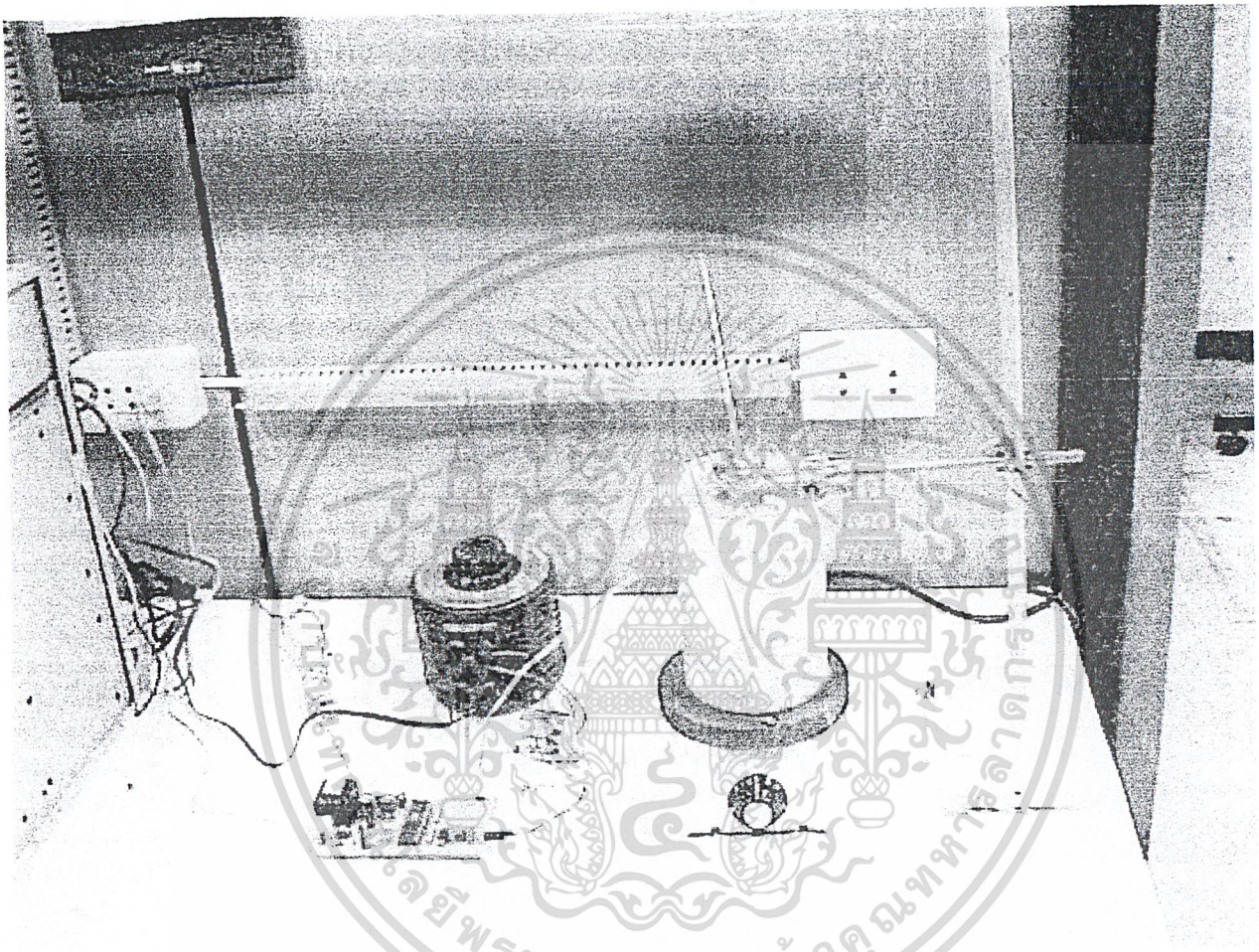
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 1 แสดงผลการวัดค่าแรงดันไฟฟ้า (โวลท์)

ค่าแรงดันไฟฟ้า (โวลท์)		
มัลติมิเตอร์	เครื่องมือวัดที่ออกแบบ	
FLUKE 45	Server	Client
230.78	230.50	230.50
228.17	228.20	228.20
226.16	226.00	226.00
224.21	224.40	224.40
222.10	221.90	221.90
220.13	220.20	220.20
218.45	218.20	218.20
216.63	216.20	216.20
214.89	214.70	214.70
212.56	212.30	212.30
210.53	210.20	210.20

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 ผลการวัดอุณหภูมิ



รูปที่ 4.2 การทดลองวัดอุณหภูมิ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2 แสดงการวัดค่าอุณหภูมิ (องศาเซลเซียส)

อุณหภูมิ(องศาเซลเซียส)	
เทอร์โมมิเตอร์ปรอท	เซนเซอร์อุณหภูมิ (ที่serverและ client)
0	0
1	2
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	12
12	13
13	14
14	15
15	16
16	17
17	18
18	19
19	20
20	21
21	22
22	23
23	24
24	25
25	26
26	27

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2 (ต่อ) แสดงการวัดค่าอุณหภูมิ (องศาเซลเซียส)

อุณหภูมิ(องศาเซลเซียส)	
เทอร์โมมิเตอร์ปรอท	เซนเซอร์อุณหภูมิ (ที่serverและ client)
27	28
28	29
29	30
30	31
31	32
32	33
33	34
34	35
35	35
36	36
37	38
38	39
39	40
40	41
41	42
42	43
43	44
44	45
45	46
46	47
47	48
48	49
49	50
50	50
51	51
52	52
53	53
54	54

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2 (ต่อ) แสดงการวัดค่าอุณหภูมิ (องศาเซลเซียส)

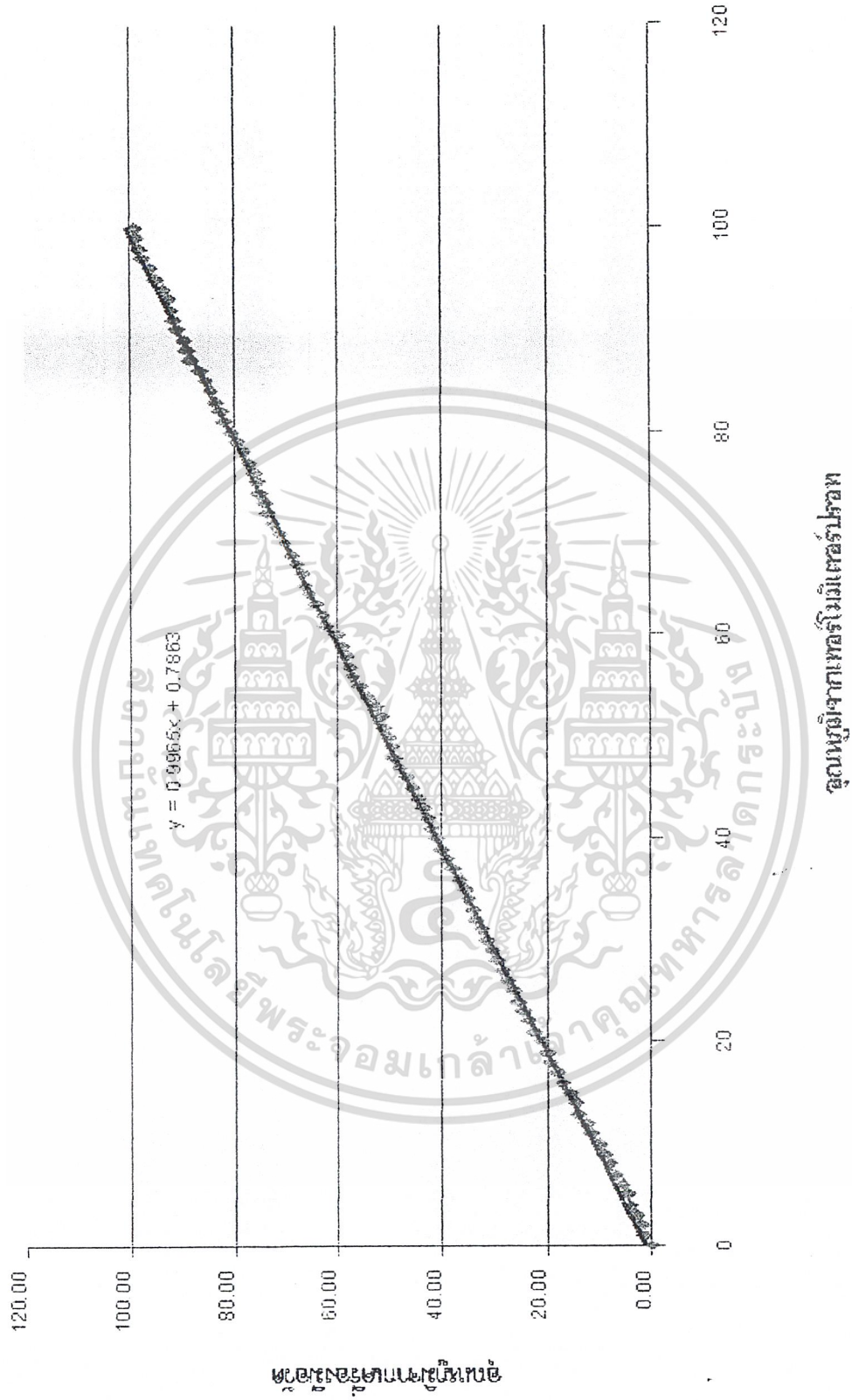
อุณหภูมิ(องศาเซลเซียส)	
เทอร์โมมิเตอร์ปรอท	เซนเซอร์อุณหภูมิ (ที่serverและ client)
55	56
56	57
57	58
58	58
59	59
60	60
61	62
62	63
63	64
64	65
65	66
66	66
67	67
68	68
69	70
70	71
71	72
72	73
73	74
74	75
75	75
76	76
77	77
78	78
79	79
80	80
81	82
82	82

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2 (ต่อ) แสดงการวัดค่าอุณหภูมิ (องศาเซลเซียส)

อุณหภูมิ(องศาเซลเซียส)	
เทอร์โมมิเตอร์ปรอท	เซนเซอร์อุณหภูมิ (ที่serverและ client)
83	84
84	85
85	86
86	87
87	89
88	89
89	90
90	91
91	92
92	92
93	93
94	94
95	95
96	96
97	98
98	98
99	99
100	99

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



กราฟที่1 แสดงการเปลี่ยนแปลงของอุณหภูมิ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุปผลการทดลองและแนวทางการพัฒนา

ผลการทดลองที่ได้เมื่อทำการเปรียบเทียบค่ามาตราฐานที่นำมาเป็นเกณฑ์อ้างอิงความถูกต้อง ค่าการวัดอุณหภูมิสามารถวัดได้ในช่วง 0 ถึง 100 องศาเซลเซียสเปรียบเทียบกับเทอร์โมมิเตอร์ปรอท ค่าที่ได้จะไม่มีความคงที่ที่อุณหภูมิหนึ่ง มีความคลาดเคลื่อนอยู่ในช่วง $\pm$ 1 ถึง 2 องศาเซลเซียส คิดเป็น 0.02 เปอร์เซ็นต์ ส่วนทางเครื่องมือวัดแรงดันทำการออกแบบให้สามารถทำการวัดแรงดันได้ที่ช่วง 0 ถึง 230 โวลท์โดยเปรียบเทียบกับมัลติมิเตอร์ ค่าที่ได้มีความคลาดเคลื่อนที่ $\pm$ 1 โวลท์ ความคลาดเคลื่อนนี้ทั้งหมดเป็นผลมาจากวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอลที่อยู่ในตัวไมโครคอนโทรลเลอร์เป็นส่วนใหญ่ที่แยกความแตกต่างของแรงดันไม่คงที่นั่นเอง

สำหรับโครงงานนี้เป็นโครงงานที่ใช้ประโยชน์ทางโครงข่ายอินเตอร์เน็ตมาประยุกต์เข้ากับไมโครคอนโทรลเลอร์เพื่อที่ทำการวัด,ควบคุมหรือตรวจสอบซึ่งก่อให้เกิดความสะดวกในการใช้งาน และการใช้ไมโครคอนโทรลเลอร์จะสามารถทำให้การควบคุมมีขอบเขตที่กว้าง ใช้งานได้หลากหลายรูปแบบ ซึ่งทางคณะผู้จัดทำโครงการพิเศษนี้ได้ในการใช้ไมโครคอนโทรลเลอร์มาออกแบบสร้างเครื่องมือวัดอุณหภูมิและแรงดันไฟฟ้า โดยอาศัยหลักการทำงานของ ADC ที่มีอยู่ในไมโครคอนโทรลเลอร์ชนิด AVR 90S8535 ซึ่งมีขนาด 10 บิทอยู่แล้วมาใช้ให้เป็นประโยชน์ และได้ทำการเขียนโปรแกรมที่ใช้ในการ Interface กับ Computer และยังสามารถติดต่อสื่อสารรับ-ส่งข้อมูลผ่านระบบเครือข่ายอินเตอร์เน็ตทางโปรแกรม Telnet ได้อีกด้วย ซึ่งในที่นี้คือโปรแกรม Visual Basic ซึ่งจะเห็นได้ว่าเป็นการประยุกต์ใช้ระบบเครือข่ายอินเตอร์เน็ตซึ่งเป็นที่นิยมกันในปัจจุบันมาช่วยในการควบคุม และรับ-ส่งข้อมูลของบอร์ดไมโครคอนโทรลเลอร์ชนิด AVR 90S8535 ได้อีกด้วย

ข้อเสนอแนะ และแนวทางการพัฒนาเพิ่มเติมของโครงการนี้

1. พัฒนาการนำไมโครคอนโทรลเลอร์ไปประยุกต์ในการควบคุมหรือตรวจวัดเพิ่มเติม
2. พัฒนาโปรแกรมที่ใช้ในไมโครคอนโทรลเลอร์ในการทำงาน และโปรแกรมบนคอมพิวเตอร์ที่ใช้ในการอินเตอร์เฟส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. ออกแบบ และพัฒนาส่วนที่ทำหน้าที่ในการวัดค่าอุณหภูมิ และแรงดันไฟฟ้าให้มีประสิทธิภาพเพิ่มมากขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก.
การเขียนโปรแกรมใช้งาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเขียนโปรแกรมภาษา C

```
#include <90s8535.h>
#include <math.h>
#include <stdio.h>
#define xtal 8000000L

// Read the ADC conversion result
unsigned int read_adc(unsigned char adc_input)
{
#ifdef BANDGAP
ADMUX=adc_input|0x40;
#else
ADMUX=adc_input;
#endif
ADCSR.6=1;
while (ADCSR.4==0);
ADCSR.4=1;
return ADCW;
}

// Declare your global variables here
int x[168];
static unsigned char ch;
static unsigned int sec,mins,hour,temp3;
flash float  constant1=4.8828125;
flash float  constant3=0.963644;
flash unsigned int  a=63529;

void menu(void)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 printf("Command\n\r");
 printf("Press 1 for read Voltage in Root Mean Square (Vrms) on channel 7\n\r");
 printf("Press 2 for read Voltage signals on channel 7\n\r");
 printf("Press 3 for read Temperature in degree Celsius \n\r");
 printf("Press 4 for read Temperature at every 5 minites\n\r");
 printf("\n\r");
}

unsigned int temp_now(unsigned char channel)
{
  unsigned int temp,temp1,temp4;
  float temp2;
    temp=read_adc(channel);
          temp1=temp;
          temp2=(float)temp1*constant1*10;
          temp2=temp2+(temp2-(temp2*constant3));
          temp4=(unsigned)temp2/1;
          printf("Temperature at channel %c is %3d.%-d degree
Celsius\n\r",channel,temp4/100,temp4%100);
    return 0;
}

void task1(void)
{
                        sec++;
        if(sec>=60)
         {
          sec=0;
          mins++;
           if(mins>=60)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            {
             mins=0;
             hour++;
              if(hour>=24)
                 {
                  hour=0;
                 }
            }
        }
}

void five(void)
{
                    task1();
                if(mins>=1)
                {
                    if((mins%5)==0)
                    {
                      if (mins!=temp3)
                      {
                            temp_now(ch);
                            temp3=mins;
                                }
                    }
            }
}

void rms(void)
{
    unsigned int n,solution;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
unsigned long int sum1,sum2;
float multi,sum;
long int temp,j;
int h,m,i;
sum=0;
    for(n=0;n<168;n++)              //receive ADC data
                        {
          x[n]=read_adc(7);
                        }
  for(n=0;n<168;n++)              // Compensation to 1024
                        {
                        i=x[n];
                              if(i==1023)
                                    {
                                          i+=1;
                                    x[n]=i;
                                    i=0;
                                    }
                        i=0;
                        }
      for(n=0;n<168;n++)          // Compare & Converse to real voltage
                        {                   // Range 0-230 Vrms
                              m=x[n];
                              h=m-512;
                              j=(long int)h*a;
                              j=j/1000;
                              x[n]=(int)j;
                              m=0;
                              h=0;
                              j=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                }
                for(n=0;n<168;n++)
        {
                temp=(long int)x[n];
                temp*=temp;             //voltage power 2
                multi=(float)temp;
                multi*=0.00117964;              // multiple by delta t
                sum+=multi;             // sum
                multi=0;
                        }
            sum=sum/20;
            sum=sum/1;
        sum1=(unsigned long int)sum;
        sum2=lsqrt(sum1);
            solution=(unsigned int)sum2;
            printf("Root mean Square (Vrms) = %d.%-d\n\r",solution/10,solution%10) ;

    sum=0;
    sum1=0;
    sum2=0;
    solution=0;
    multi=0;
    temp=0;
        j=0;
        h=0;
        i=0;
}

void voltage_signal(void)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
unsigned int n;
int h,m,k,i;
long int j;
for(n=0;n<168;n++)
{
 x[n]=read_adc(7);
 }
for(n=0;n<168;n++)
{
        i=x[n];
        {
                if(i==1023)
                {
                 i+=1;
                 x[n]=i;
                }
        }
}
 for(n=0;n<168;n++)
  {
        m=x[n];
        h=m-512;
        j=(long int)h*a;
        j=j/1000;
        k=(int)j;
        printf("  %i.%-i \r\n",k/100,k%100);
  }
  k=0;
  j=0;
  h=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    m=0;
    for(n=0;n<168;n++)
     {
                    x[n]=0;
          }
}


// TIMER1 overflow interrupt service routine
// occurs every 1 seconds
//Interupt for timer
interrupt [TIM1_OVF] void timer1_overflow(void)
{
// preset again TIMER1
TCNT1=0x10000-0x1e84;
five();
}
//Timer0 overflow interrupt
//occur every 30ms
//Interupt for watchdog
interrupt [10] void timer0_overflow(void)
{
//preset again TIMER0
#asm
          wdr
#endasm
TCNT0=0x100-0x15;
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void main()
{
// Declare your local variables here
unsigned char command1;

// Input/Output Ports initialization
// Port A
DDRA=0x00;          //input
PORTA=0x00;         //output

// Port B
DDRB=0x00;
PORTB=0x00;

// Port C
DDRC=0x00;
PORTC=0x00;

// Port D
DDRD=0x00;
PORTD=0x00;

// Timer/Counter 0 initialization
// Clock source:
// Clock value: Timer 0 Stopped
// Mode: Output Compare
// OC0 output: Disconnected
TCCR0 = 0x05;       //clock source Osc/1024
TIMSK = 0x01;       // Timer 0 overflow enable
TIFR=0;                                    //Clear Timer 0 overflow flag
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
// UART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// UART Receiver: On
// UART Transmitter: On
UCR=0x18;
UBRR=0x33;     // UART Baud rate: 9600

// ADC initialization
// ADC Clock frequency: 125.000 kHz
// Converstion time : 117.964 uS
ADCSR=0x86;



printf("\n\r");
printf("Telemetering With AVR Controller\n\r");
printf("\n\r");
menu();
printf("Type ' ? '  for Menu\n\r");
#asm
        sei                     // global enable interrupts
   wdr                          // Reset watchdog timer
#endasm
WDTCR=0x0b;       //watchdog timer 0.12s
TCNT0=0x100-0x15;  //preset Timer 0
while(1)
{
   command1=getchar();
        putchar(command1);
        printf("\n\r");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
if(command1)
{
   switch(command1)
    {
        case '1':          TIMSK=0x01;
                           rms();
                           command1=0;
                           break;

        case '2':          TIMSK=0x01;
                           voltage_signal();
                           command1=0;
                           break;

        case '3':          TIMSK=0x01;
                           printf("Which channel (0-6)\n\r");
                           ch=getchar();
                           temp_now(ch);
                           command1=0;
                           ch=0;
                           break;

        case '4':          printf("Which channel (0-6)\n\r");
                           ch=getchar();
                           temp_now(ch);
                           TCCR1A=0;
                           TCCR1B=5;  // TIMER1 clock is xtal/1024
                           sec=0;
                           mins=0;
                           hour=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                temp3=0;
                TCNT1=0x10000-0x1e84; // preset TIMER1
                TIFR=0;       // clear TIMER1 interrupts flags
                TIMSK=0x05;   // enable TIMER1 overflow interrupt
                GIMSK=0;      // all other interrupt sources are disabled
                command1=0;
                break;

        case '?':       TIMSK=0x01;
                menu();
                break;

        case 13:        TIMSK=0x01;
                printf("Internet Remote Control\n\r");
                command1=0;
                break;
                }
    }
}
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเขียนโปรแกรม Visual Basic

ส่วน Terminal

```
Option Explicit
Dim Ret As Integer
Dim Temp As String
Dim hLogFile As Integer
Dim StartTime As Date
Dim blnStartTemp As Boolean
Dim blnStartGetVoltInfo As Boolean
Dim strVoltInfo As String
Dim strtemp As String
Dim strTemperInfo As String
Private Sub cmdShowGraph_Click()
Dim strTmp As String
On Error GoTo ErrorHandler
   blnStartGetVoltInfo = False
   strVoltInfo = replace(strVoltInfo, ".", ",")
   strTmp = GenListString(strVoltInfo, 0)
   DisplayGraph strTmp, MSChart1
Exit Sub
ErrorHandler:
   MsgBox Err.Description, vbCritical, "Error : " & Err.Source
End Sub
Private Sub cmdShowTemperature_Click()
Dim strTmp As String
Dim i As Integer
   For i = 1 To 3
     strTmp = DecriptString(strTemperInfo, Chr(10))
   Next
   If strTmp = "" Then
   Else
     If Len(strTmp) < 25 Then
     Else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        strTmp = replace(strTmp, Chr(13), "")
        strTmp = Right(strTmp, Len(strTmp) - 28)
        strTmp = Left(strTmp, Len(strTmp) - 15)
        Label3.Caption = strTmp
        End If
    End If
End Sub
Private Sub cmdShowVoltage_Click()
Dim strTmp As String
Dim i As Integer
    blnStartGetVoltInfo = False
    For i = 1 To 2
      strTmp = DecriptString(strVoltInfo, Chr(10))
    Next
    Debug.Print strTmp
     If (strTmp = "") Then
     Else
          If Len(strTmp) < 25 Then
          Else
          strTmp = replace(strTmp, Chr(13), "")
          strTmp = Right(strTmp, Len(strTmp) - 26)
          Label4.Caption = strTmp
          End If
     End If
End Sub
Private Sub cmdStart_Click()
On Error GoTo ErrorHandler
    frmCommand.fraChannel.Visible = True
    frmCommand.fraChannel.Left = 2520
    frmCommand.fraCommand.Visible = True
    frmCommand.Show vbModal
   If Not frmCommand.blnCancel Then
       cmdStart.Enabled = False
       cmdStop.Enabled = True
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        blnStartTemp = True
        If strCommand <> "" Then
            txtTerm_KeyPress Asc(strCommand)
            txtTerm_KeyPress Asc(strChannel)
        End If
        ' ปิด Winsock
        WinsockServer.Close
        End If
Exit Sub
ErrorHandler:
    MsgBox Err.Description
End Sub
Private Sub cmdStop_Click()
    cmdStart.Enabled = True
    cmdStop.Enabled = False
    blnStartTemp = False
    lngDuration = 0
    TmrCommand.Enabled = False
End Sub
Private Sub cmdVolt_Click()
        cmdShowGraph.Enabled = True
On Error GoTo ErrorLine
        blnStartGetVoltInfo = True
        strVoltInfo = ""
        If Not frmCommand.blnCancel Then
            Select Case True
                Case optVolt(0).Value
                    frmCommand.fraChannel.Visible = True
                    frmCommand.fraCommand.Visible = False
                    frmCommand.fraChannel.Left = frmCommand.fraCommand.Left
                    txtTerm_KeyPress Asc("1")
                    cmdShowGraph.Enabled = False
                Case optVolt(1).Value
                    txtTerm_KeyPress Asc("2")
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        End Select
      End If
Exit Sub
ErrorLine:
  MsgBox Err.Description, vbCritical, "Error : " & Err.Source
End Sub
Private Sub Form_Load()
  Dim CommPort As String, Handshaking As String, Settings As String
  On Error Resume Next
  frmLogin.Show vbModal, Me
  If frmLogin.LoginSucceeded Then
    txtTerm.SelLength = Len(txtTerm)
    txtTerm.SelText = ""
    txtTerm.ForeColor = vbBlue
    App.Title = "Visual Basic Terminal"
    imgNotConnected.ZOrder
    frmTerminal.Move (Screen.Width - Width) / 2, (Screen.Height - Height) / 2
    Settings = GetSetting(App.Title, "Properties", "Settings", "")
    If Settings <> "" Then
      MSComm1.Settings = Settings
      If Err Then
        MsgBox Error$, 48
        Exit Sub
      End If
    End If
    CommPort = GetSetting(App.Title, "Properties", "CommPort", "")
    If CommPort <> "" Then MSComm1.CommPort = CommPort
    Handshaking = GetSetting(App.Title, "Properties", "Handshaking", "")
    If Handshaking <> "" Then
      MSComm1.Handshaking = Handshaking
      If Err Then
        MsgBox Error$, 48
        Exit Sub
      End If
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        End If
        Echo = GetSetting(App.Title, "Properties", "Echo", "")
    Else
    Unload Me
    End If
End Sub
Private Sub Form_Unload(Cancel As Integer)
    Dim Counter As Long
    If MSComm1.PortOpen Then
      Counter = Timer + 10
      Do While MSComm1.OutBufferCount
        Ret = DoEvents()
        If Timer > Counter Then
          Select Case MsgBox("Data cannot be sent", 34)
            Case 3
              Cancel = True
              Exit Sub
            Case 4
              Counter = Timer + 10
            Case 5
              Exit Do
          End Select
        End If
      Loop
      MSComm1.PortOpen = 0
    End If
    If hLogFile Then mnuCloseLog_Click
    End
End Sub
Private Sub imgConnected_Click()
    Call mnuOpen_Click
End Sub
Private Sub imgNotConnected_Click()
    ' เปิด Log File
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    Call mnuOpen_Click
End Sub
Private Sub mnuchang_Click()
    frmrepass.Show
End Sub
Private Sub mnuCloseLog_Click()
    ' ปิด Log File
    Close hLogFile
    hLogFile = 0
    mnuOpenLog.Enabled = True
    tbrToolBar.Buttons("OpenLogFile").Enabled = True
    mnuCloseLog.Enabled = False
    tbrToolBar.Buttons("CloseLogFile").Enabled = False
    frmTerminal.Caption = "Visual Basic Terminal"
End Sub
Private Sub mnuDial_Click()
    On Local Error Resume Next
    Static Num As String
    Num = "1-206-936-6735"
    Num = InputBox$("Enter Phone Number:", "Dial Number", Num)
    If Num = "" Then Exit Sub
    ' เปิด Port Com1
    If Not MSComm1.PortOpen Then
      mnuOpen_Click
      If Err Then Exit Sub
    End If
    mnuHangUp.Enabled = True
    tbrToolBar.Buttons("HangUpPhone").Enabled = True
    MSComm1.Output = "ATDT" & Num & vbCrLf
    StartTiming
End Sub
Private Sub mnuDTREnable_Click()
    MSComm1.DTREnable = Not MSComm1.DTREnable
    mnuDTREnable.Checked = MSComm1.DTREnable
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้ในการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
End Sub
Private Sub mnuFileExit_Click()
   ' เปิดหน้าต่าง Exit
   frmExit.Show
End Sub
Private Sub mnuHangup_Click()
   On Error Resume Next
   MSComm1.Output = "ATH"
   Ret = MSComm1.DTREnable
   MSComm1.DTREnable = True
   MSComm1.DTREnable = False
   MSComm1.DTREnable = Ret
   mnuHangUp.Enabled = False
   tbrToolBar.Buttons("HangUpPhone").Enabled = False
   If MSComm1.PortOpen Then MSComm1.PortOpen = False
   If Err Then MsgBox Error$, 48
   mnuSendText.Enabled = False
   tbrToolBar.Buttons("TransmitTextFile").Enabled = False
   mnuHangUp.Enabled = False
   tbrToolBar.Buttons("HangUpPhone").Enabled = False
   mnuDial.Enabled = True
   tbrToolBar.Buttons("DialPhoneNumber").Enabled = True
   sbrStatus.Panels("Settings").Text = "Settings: "
   mnuOpen.Checked = False
   imgNotConnected.ZOrder
   StopTiming
   sbrStatus.Panels("Status").Text = "Status: "
   On Error GoTo 0
End Sub
Private Sub mnuHCD_Click()
   If MSComm1.CDHolding Then
      Temp = "True"
   Else
      Temp = "False"
```

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    End If
    MsgBox "CDHolding = " + Temp
End Sub
Private Sub mnuHCTS_Click()
    If MSComm1.CTSHolding Then
        Temp = "True"
    Else
        Temp = "False"
    End If
    MsgBox "CTSHolding = " + Temp
End Sub
Private Sub mnuHDSR_Click()
    If MSComm1.DSRHolding Then
        Temp = "True"
    Else
        Temp = "False"
    End If
    MsgBox "DSRHolding = " + Temp
End Sub
Private Sub mnuInputLen_Click()
    On Error Resume Next
    Temp = InputBox$("Enter New InputLen:", "InputLen", Str$(MSComm1.InputLen))
    If Len(Temp) Then
        MSComm1.InputLen = Val(Temp)
        If Err Then MsgBox Error$, 48
    End If
End Sub
Private Sub mnulogout_Click()
frmTerminal.Hide
WinsockListen.Close
WinsockServer.Close
frmLogin.txtUserName = ""
frmLogin.txtPassword = ""
frmLogin.Show
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
frmLogin.LoginSucceeded = False
mnuOpen.Checked = False
 If MSComm1.PortOpen Then
    ' -- ปิด โปรแกรมหลัก
        cmdStart.Enabled = False
        cmdStop.Enabled = False
    mnuDial.Enabled = True
    tbrToolBar.Buttons("DialPhoneNumber").Enabled = True
    mnuHangUp.Enabled = False
    tbrToolBar.Buttons("HangUpPhone").Enabled = False
    imgNotConnected.ZOrder
    sbrStatus.Panels("Settings").Text = "Settings: "
    StopTiming
 End If
Label3.Caption = ""
txtTerm.Text = ""
End Sub
Private Sub mnuProperties_Click()
 ' แสดงหน้าต่าง Properties
 frmProperties.Show vbModal
End Sub
Private Sub mnuOpen_Click()
   On Error Resume Next
   Dim OpenFlag
   MSComm1.PortOpen = Not MSComm1.PortOpen
   If Err Then MsgBox Error$, 48
   OpenFlag = MSComm1.PortOpen
   mnuOpen.Checked = OpenFlag
   mnuSendText.Enabled = OpenFlag
   tbrToolBar.Buttons("TransmitTextFile").Enabled = OpenFlag
   If MSComm1.PortOpen Then
   ' เริ่มการทำงานของโปรแกรมหลัก
      cmdStart.Enabled = True
      cmdStop.Enabled = False
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        cmdVolt.Enabled = True
        mnuDial.Enabled = True
        tbrToolBar.Buttons("DialPhoneNumber").Enabled = True
        mnuHangUp.Enabled = True
        tbrToolBar.Buttons("HangUpPhone").Enabled = True
        imgConnected.ZOrder
        sbrStatus.Panels("Settings").Text = "Settings: " & MSComm1.Settings
        StartTiming
        ' เปิดโปรแกรมส่วน Winsock
    ' --- รายละเอียดโปรแกรมส่วนติดต่อ Telnet
        With WinsockListen
            .LocalPort = 23
            .Listen
        End With
    Else
    ' -- ปิดโปรแกรมหลัก
            cmdStart.Enabled = False
            cmdStop.Enabled = False
        mnuDial.Enabled = True
        tbrToolBar.Buttons("DialPhoneNumber").Enabled = True
        mnuHangUp.Enabled = False
        tbrToolBar.Buttons("HangUpPhone").Enabled = False
        imgNotConnected.ZOrder
        sbrStatus.Panels("Settings").Text = "Settings: "
        StopTiming
    End If
End Sub
Private Sub mnuOpenLog_Click()
  Dim replace
  On Error Resume Next
  OpenLog.Flags = cdlOFNHideReadOnly Or cdlOFNExplorer
  OpenLog.CancelError = True
  Do
    If Err = cdlCancel Then Exit Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        Temp = App.Path & "\All.log"
        Ret = Len(Dir$(Temp))
      If Err Then
        MsgBox Error$, 48
        Exit Sub
      End If
      If Ret Then
        replace = MsgBox("Replace existing file - " + Temp + "?", 35)
      Else
        replace = 0
      End If
    Loop While replace = 2
    If replace = 6 Then
      Kill Temp
      If Err Then
        MsgBox Error$, 48
        Exit Sub
      End If
    End If
    hLogFile = FreeFile
    Open Temp For Binary Access Write As hLogFile
    If Err Then
      MsgBox Error$, 48
      Close hLogFile
      hLogFile = 0
      Exit Sub
    Else
      Seek hLogFile, LOF(hLogFile) + 1
    End If
    ' เขียนวันเวลาให้กับ Log File
     Put hLogFile, , vbCrLf & vbCrLf & "<<<<<< Date = " & Format(Date, "dd  MMMM  yyyy") & " , Time
= " & Time & ">>>>>" & vbCrLf
    frmTerminal.Caption = "Visual Basic Terminal - " + OpenLog.FileTitle
    mnuOpenLog.Enabled = False
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    tbrToolBar.Buttons("OpenLogFile").Enabled = False
    mnuCloseLog.Enabled = True
    tbrToolBar.Buttons("CloseLogFile").Enabled = True
End Sub
Private Sub mnuParRep_Click()
    On Error Resume Next
    Temp = InputBox$("Enter Replace Character", "ParityReplace",
frmTerminal.MSComm1.ParityReplace)
    frmTerminal.MSComm1.ParityReplace = Left$(Temp, 1)
    If Err Then MsgBox Error$, 48
End Sub
Private Sub mnuRThreshold_Click()
    On Error Resume Next
    Temp = InputBox$("Enter New RThreshold:", "RThreshold", Str$(MSComm1.RThreshold))
    If Len(Temp) Then
        MSComm1.RThreshold = Val(Temp)
        If Err Then MsgBox Error$, 48
    End If
End Sub
Private Sub MSChart1_OLEStartDrag(Data As MSChart20Lib.DataObject, AllowedEffects As Long)
MSChart1.Visible = False
End Sub
Private Static Sub MSComm1_OnComm()
    Dim EVMsg$
    Dim ERMsg$
    Select Case MSComm1.CommEvent
        Case comEvReceive
            Dim Buffer As Variant
            Buffer = MSComm1.Input
            ShowData txtTerm, (StrConv(Buffer, vbUnicode))
        ' --- แสดงผลออกทาง Telnet
        If WinsockServer.State = sckConnected Then WinsockServer.SendData StrConv(Buffer,
vbUnicode)
    ' -------ตรวจวัดอุณหภูมิ
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        If blnStartTemp Then
            strTemperInfo = strTemperInfo & StrConv(Buffer, vbUnicode)
            ' -- ปิดโปรแกรมส่วน Winsock
            WinsockServer.Close
        End If
' -----ตรวจวัดแรงดันไฟฟ้า
    If blnStartGetVoltInfo Then
            strVoltInfo = strVoltInfo & StrConv(Buffer, vbUnicode)
    End If
    Case comEvSend
    Case comEvCTS
        EVMsg$ = "Change in CTS Detected"
    Case comEvDSR
        EVMsg$ = "Change in DSR Detected"
    Case comEvCD
        EVMsg$ = "Change in CD Detected"
    Case comEvRing
        EVMsg$ = "The Phone is Ringing"
    Case comEvEOF
        EVMsg$ = "End of File Detected"
    Case comBreak
        ERMsg$ = "Break Received"
    Case comCDTO
        ERMsg$ = "Carrier Detect Timeout"
    Case comCTSTO
        ERMsg$ = "CTS Timeout"
    Case comDCB
        ERMsg$ = "Error retrieving DCB"
    Case comDSRTO
        ERMsg$ = "DSR Timeout"
    Case comFrame
        ERMsg$ = "Framing Error"
    Case comOverrun
        ERMsg$ = "Overrun Error"
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        Case comRxOver
            ERMsg$ = "Receive Buffer Overflow"
        Case comRxParity
            ERMsg$ = "Parity Error"
        Case comTxFull
            ERMsg$ = "Transmit Buffer Full"
        Case Else
            ERMsg$ = "Unknown error or event"
    End Select
    If Len(EVMsg$) Then
        sbrStatus.Panels("Status").Text = "Status: " & EVMsg$
        Timer2.Enabled = True
    ElseIf Len(ERMsg$) Then
        sbrStatus.Panels("Status").Text = "Status: " & ERMsg$
        Beep
        Ret = MsgBox(ERMsg$, 1, "Click Cancel to quit, OK to ignore.")
        If Ret = 2 Then
            MSComm1.PortOpen = False
        End If
        Timer2.Enabled = True
    End If
End Sub
Private Sub mnuSendText_Click()
    Dim hSend, BSize, LF&
    On Error Resume Next
    mnuSendText.Enabled = False
    tbrToolBar.Buttons("TransmitTextFile").Enabled = False
    OpenLog.DialogTitle = "Send Text File"
    OpenLog.Filter = "Text Files (*.TXT)|*.txt|All Files (*.*)|*.*"
    Do
        OpenLog.CancelError = True
        OpenLog.FileName = ""
        OpenLog.ShowOpen
        If Err = cdlCancel Then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    mnuSendText.Enabled = True
    tbrToolBar.Buttons("TransmitTextFile").Enabled = True
    Exit Sub
  End If
  Temp = OpenLog.FileName
  Ret = Len(Dir$(Temp))
  If Err Then
    MsgBox Error$, 48
    mnuSendText.Enabled = True
    tbrToolBar.Buttons("TransmitTextFile").Enabled = True
    Exit Sub
  End If
  If Ret Then
    Exit Do
  Else
    MsgBox Temp + " not found!", 48
  End If
Loop
hSend = FreeFile
Open Temp For Binary Access Read As hSend
If Err Then
  MsgBox Error$, 48
Else
  CancelSend = False
  BSize = MSComm1.OutBufferSize
  LF& = LOF(hSend)
  Do Until EOF(hSend) Or CancelSend
    If LF& - Loc(hSend) <= BSize Then
      BSize = LF& - Loc(hSend) + 1
    End If
    Temp = Space$(BSize)
    Get hSend, , Temp
    MSComm1.Output = Temp
    If Err Then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            MsgBox Error$, 48
            Exit Do
          End If
          Do
            Ret = DoEvents()
          Loop Until MSComm1.OutBufferCount = 0 Or CancelSend
        Loop
      End If
      Close hSend
      mnuSendText.Enabled = True
      tbrToolBar.Buttons("TransmitTextFile").Enabled = True
      CancelSend = True
    End Sub
    Private Sub mnuSThreshold_Click()
      On Error Resume Next
      Temp = InputBox$("Enter New SThreshold Value", "SThreshold", Str$(MSComm1.SThreshold))
      If Len(Temp) Then
        MSComm1.SThreshold = Val(Temp)
        If Err Then MsgBox Error$, 48
      End If
    End Sub
    Private Static Sub ShowData(Term As Control, Data As String)
      On Error GoTo Handler
      Const MAXTERMSIZE = 16000
      Dim TermSize As Long, i
      TermSize = Len(Term.Text)
      If TermSize > MAXTERMSIZE Then
        Term.Text = Mid$(Term.Text, 4097)
        TermSize = Len(Term.Text)
      End If
      Term.SelStart = TermSize
      Do
        i = InStr(Data, Chr$(8))
        If i Then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    If i = 1 Then
      Term.SelStart = TermSize - 1
      Term.SelLength = 1
      Data = Mid$(Data, i + 1)
    Else
      Data = Left$(Data, i - 2) & Mid$(Data, i + 1)
    End If
  End If
Loop While i
Do
  i = InStr(Data, Chr$(10))
  If i Then
    Data = Left$(Data, i - 1) & Mid$(Data, i + 1)
  End If
Loop While i
i = 1
Do
  i = InStr(i, Data, Chr$(13))
  If i Then
    Data = Left$(Data, i) & Chr$(10) & Mid$(Data, i + 1)
    i = i + 1
  End If
Loop While i
Term.SelText = Data
If hLogFile Then
  i = 2
  Do
    Err = 0
    Put hLogFile, , Data
    If Err Then
      i = MsgBox(Error$, 21)
      If i = 2 Then
        mnuCloseLog_Click
      End If
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        End If
      Loop While i <> 2
    End If
    Term.SelStart = Len(Term.Text)
Exit Sub
Handler:
    MsgBox Error$
    Resume Next
End Sub
Private Sub optVolt_Click(Index As Integer)
    Select Case Index
        Case 0
            cmdVolt.Enabled = True
            cmdShowGraph.Enabled = False
            cmdShowVoltage.Enabled = True
        Case 1
            cmdVolt.Enabled = True
            cmdShowGraph.Enabled = False
            cmdShowVoltage.Enabled = False
        End Select
End Sub
Private Sub Timer2_Timer()
sbrStatus.Panels("Status").Text = "Status: "
Timer2.Enabled = False
End Sub
Private Sub TmrCommand_Timer()
    lngDuration = lngDuration + 1
    If intCommandType = 2 Then
    ' -- 5 min
        If lngDuration = 5 Then
            cmdStop = True
        End If
    End If
    If intCommandType = 3 Then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    ' --- 60 min
        If lngDuration = 60 Then
            cmdStop = True
        End If
    End If
End Sub
Private Sub txtTerm_KeyPress(KeyAscii As Integer)
    If MSComm1.PortOpen Then
        MSComm1.Output = Chr$(KeyAscii)
        If Not Echo Then
            txtTerm.SelStart = Len(txtTerm)
            KeyAscii = 0
        End If
    End If
End Sub
Private Sub tbrToolBar_ButtonClick(ByVal Button As MSComctlLib.Button)
Select Case Button.Key
Case "OpenLogFile"
    Call mnuOpenLog_Click
Case "CloseLogFile"
    Call mnuCloseLog_Click
Case "DialPhoneNumber"
    Call mnuDial_Click
Case "HangUpPhone"
    Call mnuHangup_Click
Case "Properties"
    Call mnuProperties_Click
Case "TransmitTextFile"
    Call mnuSendText_Click
End Select
End Sub
Private Sub Timer1_Timer()
    ' แสดงเวลาที่ต่อกับ Com Port
    sbrStatus.Panels("ConnectTime").Text = Format(Now - StartTime, "hh:nn:ss") & ""
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
End Sub
Private Sub StartTiming()
   StartTime = Now
   Timer1.Enabled = True
End Sub
Private Sub StopTiming()
   Timer1.Enabled = False
   sbrStatus.Panels("ConnectTime").Text = ""
End Sub
Private Sub WinsockListen_ConnectionRequest(ByVal requestID As Long)
Dim strWelcome As String
   ' -- If server in process can not connect by telnet
   If Not blnStartTemp Then
      If WinsockServer.State <> sckClosed Then _
      WinsockServer.Close
      ' Accept the request with the requestID
      ' parameter.
      WinsockServer.Accept requestID

      strWelcome = "Press command : " & vbCrLf & _
                   " press 1 : Detect Voltage in Root Mean Square (Vrms). " & vbCrLf & _
                   " press 2 : Detect Voltages Sinal on Channel 7. " & vbCrLf & _
                   " press 3 : Detect temperature for once. " & vbCrLf & _
                   " press 4 : Detect temperature for 5 min. " & vbCrLf & _
                   " press ? : Menu. " & vbCrLf & _
                   " press f : Open Log File.  " & vbCrLf
      WinsockServer.SendData strWelcome
   End If
End Sub
Private Sub WinsockServer_DataArrival(ByVal bytesTotal As Long)
Dim strData As String
   WinsockServer.GetData strData
   Select Case True
      Case LCase(strData) = "f"
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        ' -- List logfile name "All.log" to telnet
            WinsockServer.SendData "Content of log file is : " & vbCrLf & ReadTXTFile(App.Path &
"\All.log")
        Case Else
            txtTerm_KeyPress Asc(strData)
            txtTerm.Text = txtTerm.Text & strData
            ' -- Echo
            WinsockServer.SendData "<Receipt Command> :" & strData & vbCrLf
    End Select
End Sub
```

ส่วน Properties

```
Private iFlow As Integer, iTempEcho As Boolean
Sub LoadPropertySettings()
Dim i As Integer, Settings As String, Offset As Integer
' Load Port Settings
For i = 1 To 16
    cboPort.AddItem "Com" & Trim$(Str$(i))
Next i
' Load Speed Settings
cboSpeed.AddItem "110"
cboSpeed.AddItem "300"
cboSpeed.AddItem "600"
cboSpeed.AddItem "1200"
cboSpeed.AddItem "2400"
cboSpeed.AddItem "4800"
cboSpeed.AddItem "9600"
cboSpeed.AddItem "14400"
cboSpeed.AddItem "19200"
cboSpeed.AddItem "28800"
cboSpeed.AddItem "38400"
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
cboSpeed.AddItem "56000"
cboSpeed.AddItem "57600"
cboSpeed.AddItem "115200"
cboSpeed.AddItem "128000"
cboSpeed.AddItem "256000"
' Load Data Bit Settings
cboDataBits.AddItem "4"
cboDataBits.AddItem "5"
cboDataBits.AddItem "6"
cboDataBits.AddItem "7"
cboDataBits.AddItem "8"
' Load Parity Settings
cboParity.AddItem "Even"
cboParity.AddItem "Odd"
cboParity.AddItem "None"
cboParity.AddItem "Mark"
cboParity.AddItem "Space"
' Load Stop Bit Settings
cboStopBits.AddItem "1"
cboStopBits.AddItem "1.5"
cboStopBits.AddItem "2"
' Set Default Settings
Settings = frmTerminal.MSComm1.Settings
' In all cases the right most part of Settings will be 1 character
' except when there are 1.5 stop bits.
If InStr(Settings, ".") > 0 Then
   Offset = 2
Else
   Offset = 0
End If
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
cboSpeed.Text = Left$(Settings, Len(Settings) - 6 - Offset)
Select Case Mid$(Settings, Len(Settings) - 4 - Offset, 1)
Case "e"
   cboParity.ListIndex = 0
Case "m"
   cboParity.ListIndex = 1
Case "n"
   cboParity.ListIndex = 2
Case "o"
   cboParity.ListIndex = 3
Case "s"
   cboParity.ListIndex = 4
End Select
cboDataBits.Text = Mid$(Settings, Len(Settings) - 2 - Offset, 1)
cboStopBits.Text = Right$(Settings, 1 + Offset)
cboPort.ListIndex = frmTerminal.MSComm1.CommPort - 1
optFlow(frmTerminal.MSComm1.Handshaking).Value = True
If Echo Then
   optEcho(1).Value = True
Else
   optEcho(0).Value = True
End If
End Sub
Private Sub cmdCancel_Click()
Unload Me
End Sub
Private Sub cmdOK_Click()
Dim OldPort As Integer, ReOpen As Boolean
On Error Resume Next
Echo = iTempEcho
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
OldPort = frmTerminal.MSComm1.CommPort
NewPort = cboPort.ListIndex + 1
If NewPort <> OldPort Then              ' If the port number changes, close the old port.
   If frmTerminal.MSComm1.PortOpen Then
      frmTerminal.MSComm1.PortOpen = False
      ReOpen = True
   End If
   frmTerminal.MSComm1.CommPort = NewPort       ' Set the new port number.
   If Err = 0 Then
      If ReOpen Then
         frmTerminal.MSComm1.PortOpen = True
         frmTerminal.mnuOpen.Checked = frmTerminal.MSComm1.PortOpen
         frmTerminal.mnuSendText.Enabled = frmTerminal.MSComm1.PortOpen
         frmTerminal.tbrToolBar.Buttons("TransmitTextFile").Enabled                =
frmTerminal.MSComm1.PortOpen
      End If
   End If
   If Err Then
      MsgBox Error$, 48
      frmTerminal.MSComm1.CommPort = OldPort
      Exit Sub
   End If
End If
frmTerminal.MSComm1.Settings = Trim$(cboSpeed.Text) & "," & Left$(cboParity.Text, 1) _
   & "," & Trim$(cboDataBits.Text) & "," & Trim$(cboStopBits.Text)
If Err Then
   MsgBox Error$, 48
   Exit Sub
End If
frmTerminal.MSComm1.Handshaking = iFlow
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
If Err Then
   MsgBox Error$, 48
   Exit Sub
End If
SaveSetting App.Title, "Properties", "Settings", frmTerminal.MSComm1.Settings
SaveSetting App.Title, "Properties", "CommPort", frmTerminal.MSComm1.CommPort
SaveSetting App.Title, "Properties", "Handshaking", frmTerminal.MSComm1.Handshaking
SaveSetting App.Title, "Properties", "Echo", Echo
Unload Me
End Sub
Private Sub Form_Load()
' Set the form's size
Me.Left = (Screen.Width - Me.Width) / 2
Me.Top = (Screen.Height - Me.Height) / 2
' Size the frame to fit in the tabstrip control
fraSettings.Move tabSettings.ClientLeft, tabSettings.ClientTop
' Make sure the frame is the top most control
fraSettings.ZOrder
' Load current property settings
LoadPropertySettings
End Sub
Private Sub optEcho_Click(Index As Integer)
If Index = 1 Then
   iTempEcho = True
Else
   iTempEcho = False
End If
End Sub
Private Sub optFlow_Click(Index As Integer)
iFlow = Index
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

End Sub

ส่วน Repass

Public orDerdb2 As Database

Public orDerWs2 As Workspace

Public namedb2 As Recordset

Public passdb2 As Recordset

Private Sub cmdCancel_Click()

txtlogin.Text = ""

txtold.Text = ""

txtnew1.Text = ""

txtnew2.Text = ""

Me.Hide

End Sub

Private Sub cmdOK_Click()

passdb2.MoveFirst

Do While Not passdb2.EOF

If ((passdb2.Fields("Name")) = txtlogin.Text) And ((passdb2.Fields("Password")) = txtold.Text) Then

If txtnew1.Text = txtnew2.Text Then

passdb2.Edit

passdb2.Fields("password") = txtnew2.Text

passdb2.Update

passdb2.Requery

txtlogin.Text = ""

txtold.Text = ""

txtnew1.Text = ""

txtnew2.Text = ""

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        frmrepass.Hide
        MsgBox "Change password success (ถูกแล้วล่ะ)", vbInformation, "Success"
        Exit Sub
      Else
        MsgBox "New password no match", vbCritical, "No match"
      End If
    Else
    End If
    passdb2.MoveNext
Loop
txtlogin.Text = ""
txtold.Text = ""
MsgBox "Login Incorrect and Try Again (ผิดครับผม ลองใหม่นะ)", vbCritical, "Incorrect"
End Sub
Private Sub Form_Load()
Set orDerWs2 = DBEngine.Workspaces(0)
Set orDerdb2 = orDerWs2.OpenDatabase("d:\db1.mdb", False, False)
Set passdb2 = orDerdb2.OpenRecordset("SELECT * FROM table1", dbOpenDynaset)
Set namedb2 = orDerdb2.OpenRecordset("SELECT name,password FROM table1",
dbOpenDynaset)
Exit Sub
End Sub
```

ส่วน Login

```
Public orDerdb As Database
Public orDerWs As Workspace
Public namedb As Recordset
Public passdb As Recordset
Option Explicit
Public LoginSucceeded As Boolean
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Private Sub cmdCancel_Click()
   'set the global var to false
   'to denote a failed login
   LoginSucceeded = False
   txtUserName.Text = ""
   txtPassword.Text = ""
End
End Sub
Private Sub cmdOK_Click()
passdb.MoveFirst
Do While Not passdb.EOF
   If ((passdb.Fields("name")) = txtUserName.Text) And ((passdb.Fields("Password")) =
txtPassword.Text) Then
      'txtUserName.Text = ""
      'txtPassword.Text = ""
      LoginSucceeded = True
      Me.Hide
      frmTerminal.Show
      Exit Sub
   End If
   passdb.MoveNext
Loop
txtUserName.Text = ""
txtPassword.Text = ""
MsgBox "Login Incorrect and Try Again (มันผิดล่ะ ใส่ใหม่จะดีไหม)", vbCritical, "Incorrected"
End Sub   'check รหัสผ่าน
Private Sub Form_Load()
Set orDerWs = DBEngine.Workspaces(0)
Set orDerdb = orDerWs.OpenDatabase("d:\db1.mdb", False, False)
Set passdb = orDerdb.OpenRecordset("SELECT * FROM table1", dbOpenDynaset)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Set namedb = orDerdb.OpenRecordset("SELECT name,Password FROM table1",
dbOpenDynaset)
Exit Sub
End Sub
```

ส่วน Exit

```
Private Sub cmdCancle_Click()
Unload Me
End Sub
Private Sub cmdOK_Click()
Unload Me
Unload frmTerminal
End Sub
```

ส่วน Command

```
Private Sub cmdCancel_Click()
   strCommand = ""
   intCommandType = 0
   blnCancel = True
   Me.Hide
End Sub
Private Sub cmdOK_Click()
   Select Case True
      Case OptChannel(0).Value
         strChannel = "0"
         frmTerminal.cmdShowTemperature.Enabled = True
      Case OptChannel(1).Value
         strChannel = "1"
         frmTerminal.cmdShowTemperature.Enabled = False
      Case OptChannel(2).Value
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        strChannel = "2"
        frmTerminal.cmdShowTemperature.Enabled = False
    Case OptChannel(3).Value
        strChannel = "3"
        frmTerminal.cmdShowTemperature.Enabled = False
    Case OptChannel(4).Value
        strChannel = "4"
        frmTerminal.cmdShowTemperature.Enabled = False
    Case OptChannel(5).Value
        strChannel = "5"
        frmTerminal.cmdShowTemperature.Enabled = False
    Case OptChannel(6).Value
        strChannel = "6"
        frmTerminal.cmdShowTemperature.Enabled = False
    Case Else
        strChannel = "0"
        frmTerminal.cmdShowTemperature.Enabled = True
End Select
If OptCommand(0).Value Then
    strCommand = "3"
    intCommandType = 1
    frmTerminal.TmrCommand.Enabled = False
End If
If OptCommand(1).Value Then
    strCommand = "4"
    intCommandType = 2
    frmTerminal.TmrCommand.Enabled = True
    frmTerminal.TmrCommand.Interval = 60000
End If
If OptCommand(2).Value Then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        strCommand = "5"
        intCommandType = 3
        frmTerminal.TmrCommand.Enabled = True
        frmTerminal.TmrCommand.Interval = 60000
    End If
    blnCancel = False
    Me.Hide
End Sub
Private Sub Form_Load()
' -- Default
    OptCommand(0).Value = True
    strCommand = "r"
    intCommandType = 1
End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข.

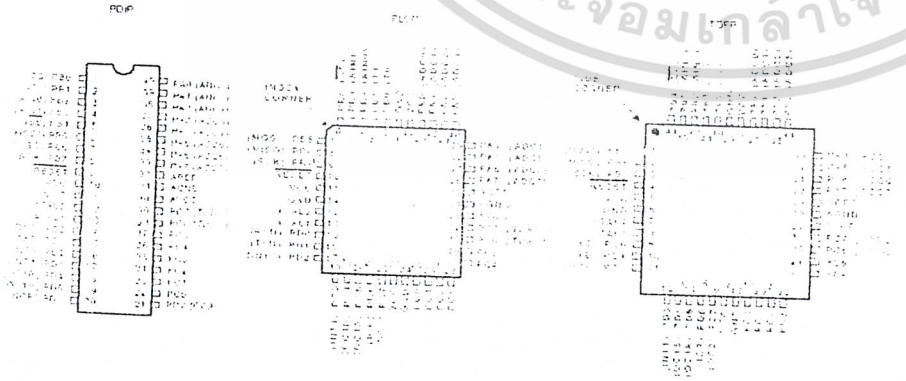
รายละเอียดอุปกรณ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Features

- AVR® - High-performance and Low-power RISC Architecture
  - 118 Powerful Instructions - Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Up to 8 MIPS Throughput at 8 MHz
- Data and Nonvolatile Program Memories
  - 4K/8K Bytes of In-System Programmable Flash
    SPI Serial Interface for In-System Programming
    Endurance: 1,000 Write/Erase Cycles
  - 256/512 Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 256/512 Bytes Internal SRAM
  - Programming Lock for Software Security
- Peripheral Features
  - 8-channel, 10-bit ADC
  - Programmable UART
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare and Capture Modes, and dual 8-, 9-, or 10-bit PWM
  - Programmable Watchdog Timer with On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset Circuit
  - Real Time Clock (RTC) with Separate Oscillator and Counter Mode
  - External and Internal Interrupt Sources
  - Three Sleep Modes: Idle, Power Save, and Power Down
- Power Consumption at 4 MHz, 3V, 20°C
  - Active: 6.4 mA
  - Idle Mode: 1.9 mA
  - Power Down Mode: <1 µA
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-pin PLCC and 44-pin TQFP
- Operating Voltages
  - V_{CC}: 4.0 - 6.0V AT90S4434/AT90S8535
  - V_{CC}: 2.7 - 6.0V AT90LS4434/AT90LS8535
- Speed Grades:
  - 0 - 8 MHz AT90S4434/AT90S8535
  - 0 - 4 MHz AT90LS4434/AT90LS8535

Pin Configurations



8-bit AVR® Microcontroller with 4K/8K Bytes In-System Programmable Flash

AT90S4434
AT90LS4434
AT90S8535
AT90LS8535

Preliminary

Rev. 1041E-04/98

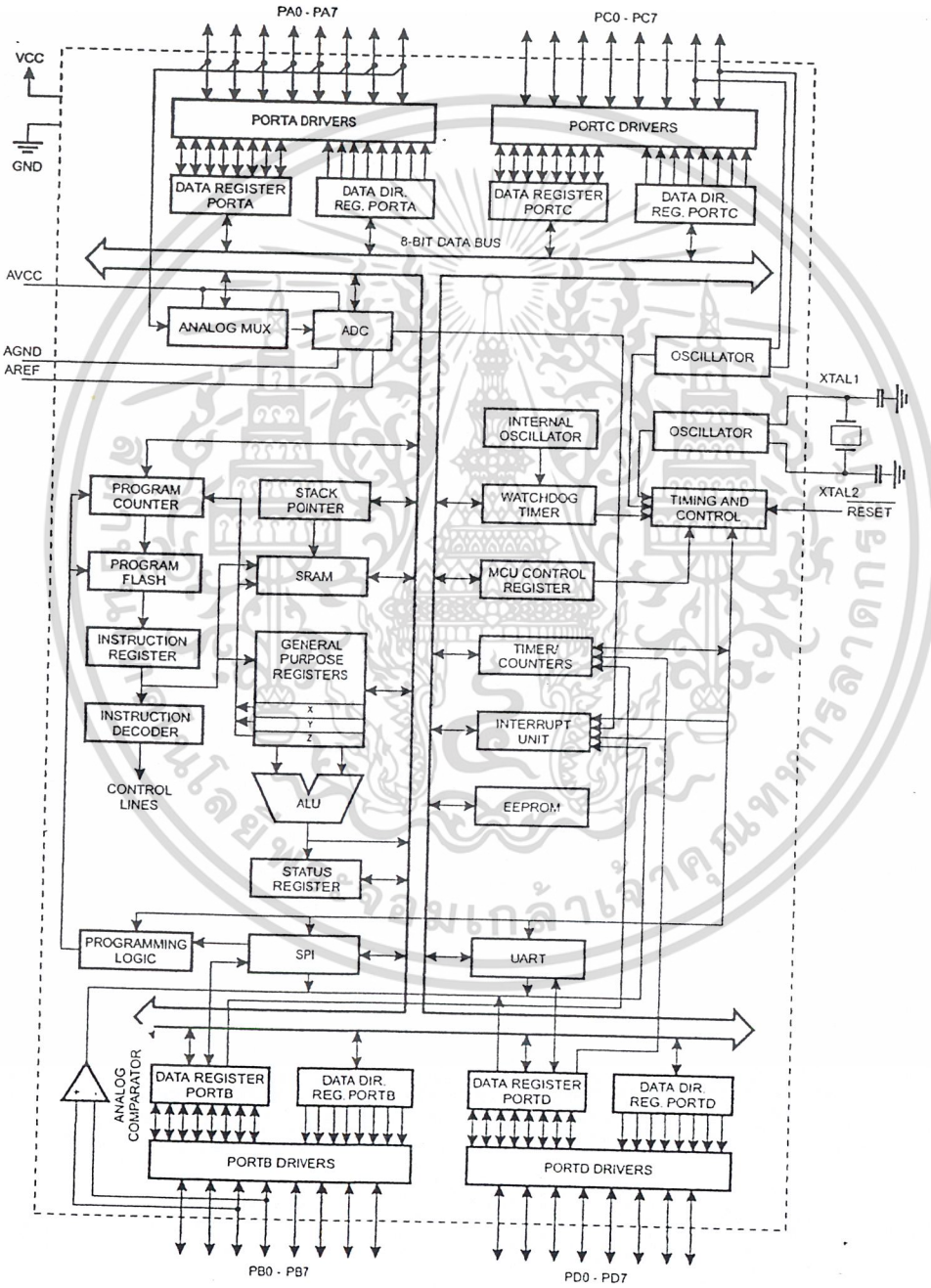
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Description

The AT90S4434/8535 is a low-power CMOS 8-bit microcontroller based on the AVR RISC architecture. By executing powerful instructions in a single clock cycle, the AT90S4434/8535 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 1. The AT90S4434/8535 Block Diagram



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The AT90S4434/8535 provides the following features: 4K/8K bytes of In-System Programmable Flash, 256/512 bytes EEPROM, 256/512 bytes SRAM, 32 general purpose I/O lines, 32 general purpose working registers, Real Time Clock (RTC), three flexible timer/counters with compare modes, internal and external interrupts, a programmable serial UART, 8-channel, 10-bit ADC, programmable Watchdog Timer with internal oscillator, an SPI serial port, and three software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, timer/counters, SPI port, and interrupt system to continue functioning. The Power Down mode saves the register contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power Save mode, the timer oscillator continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping.

The device is manufactured using Atmel's high density nonvolatile memory technology. The on-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface or by a conventional nonvolatile memory programmer. By combining an 8-bit RISC CPU with In-System Programmable Flash on a monolithic chip, the Atmel AT90S4434/8535 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The AT90S4434/8535 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

Comparison between AT90S4434 and AT90S8535

The AT90S4434 has 4K bytes of In-System Programmable Flash, 256 bytes of EEPROM, and 256 bytes of internal SRAM.

The AT90S8535 has 8K bytes of In-System Programmable Flash, 512 bytes of EEPROM, and 512 bytes of internal SRAM.

Table 1 summarizes the different memory sizes for the two devices.

Table 1. Memory Size Summary

Part	Flash	EEPROM	SRAM
AT90S4434	4K bytes	256 bytes	256 bytes
AT90S8535	8K bytes	512 bytes	512 bytes

Pin Descriptions

VCC

Digital supply voltage

GND

Digital ground

Port A (PA7..PA0)

Port A is an 8-bit bi-directional I/O port. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers can sink 20mA and can drive LED displays directly. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.

Port A also serves as the analog inputs to the A/D Converter.

The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7..PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors. The Port B output buffers can sink 20 mA. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. Port B also serves the functions of various special features of the AT90S4434/8535 as listed on page 68.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

The Port B pins are tristated when a reset condition becomes active, even if the clock is not running.

Port C (PC7..PC0)

Port C is an 8-bit bi-directional I/O port with internal pullup resistors. The Port C output buffers can sink 20 mA. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. Two Port C pins can alternatively be used as oscillator for Timer/Counter2.

The port C pins are tristated when a reset condition becomes active, even if the clock is not running.

Port D (PD7..PD0)

Port D is an 8-bit bidirectional I/O port with internal pull-up resistors. The Port D output buffers can sink 20 mA. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated.

Port D also serves the functions of various special features of the AT90S4434/8535 as listed on page 77.

The port D pins are tristated when a reset condition becomes active, even if the clock is not running.

$\overline{\text{RESET}}$

Reset input. An external reset is generated by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than 50 ns will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier

AVCC

This is the supply voltage pin for the A/D Converter. It should be externally connected to V_{CC} via a low-pass filter. See page 59 for details on operation of the ADC.

AREF

This is the analog reference input for the A/D Converter. For ADC operations, a voltage in the range AGND to AV_{CC} must be applied to this pin.

AGND

Analog ground. If the board has a separate analog ground plane, this pin should be connected to this ground plane. Otherwise, connect to GND.

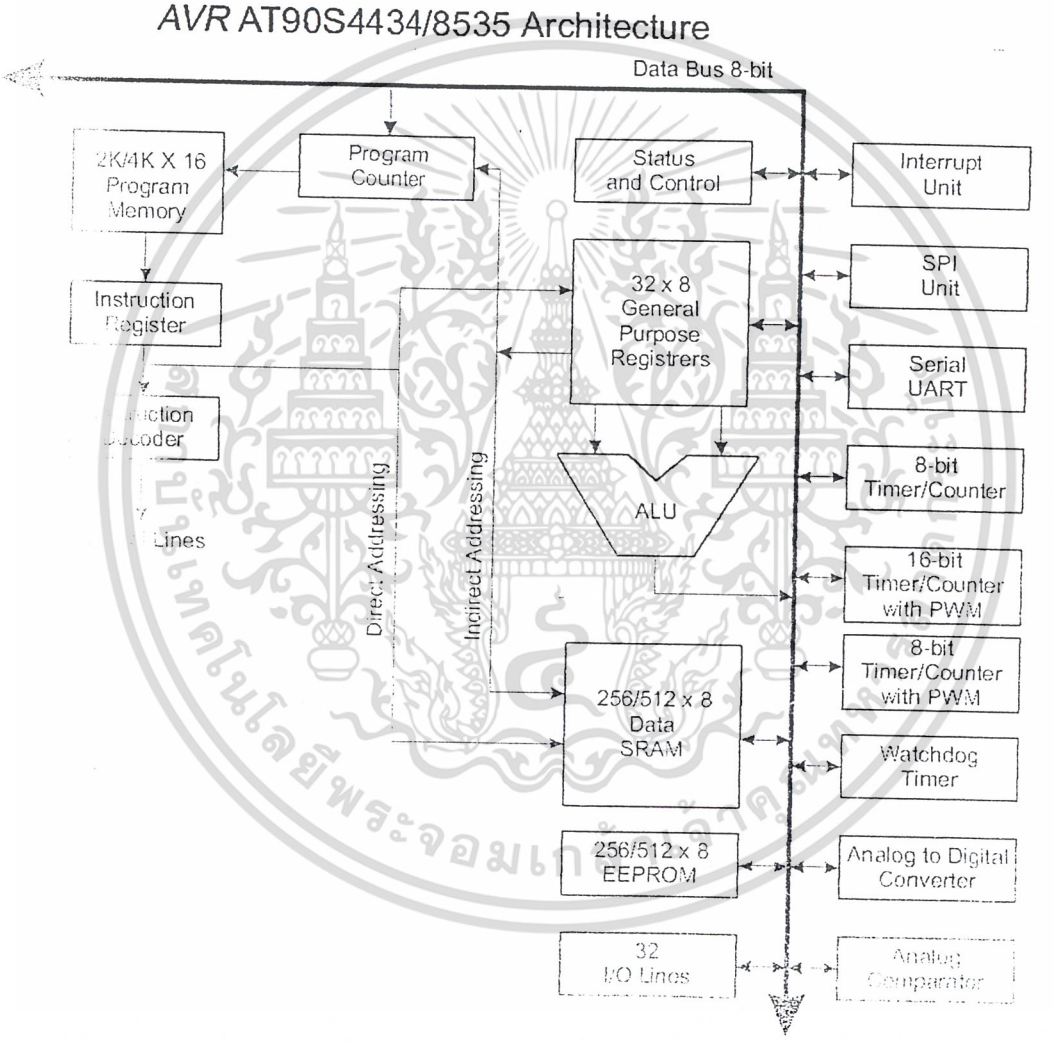
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Architectural Overview

The fa[illegible] register file concept contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This means that during one single clock cycle, one Arithmetic Logic Unit (ALU) operation is executed. Two operands are ou[illegible] the register file, the operation is executed, and the result is stored back in the register file - in one clock cycle.

Six of the [illegible] can be used as three 16-bits indirect address register pointers for Data Space addressing - enabling efficie[illegible]tions. One of the three address pointers is also used as the address pointer for the constant table look up fu[illegible] added function registers are the 16-bits X-register, Y-register and Z-register.

Figure 4. The AT90S4434/8535 AVR RISC Architecture



[illegible]rd logic functions between registers or between a constant and a register. Single register [illegible] the ALU. Figure 4 shows the AT90S4434/8535 AVR RISC microcontroller architecture.

[illegible]ion, the conventional memory addressing modes can be used on the register file as well. [illegible]he register file is assigned the 32 lowermost Data Space addresses ($00 - $1F), allowing [illegible]ey were ordinary memory locations.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, Timer/Counters, A/D-converters, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the register file, $20 - $5F.

The AVR uses a Harvard architecture concept - with separate memories and buses for program and data. The program memory is executed with a two stage pipeline. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is in-system downloadable Flash memory.

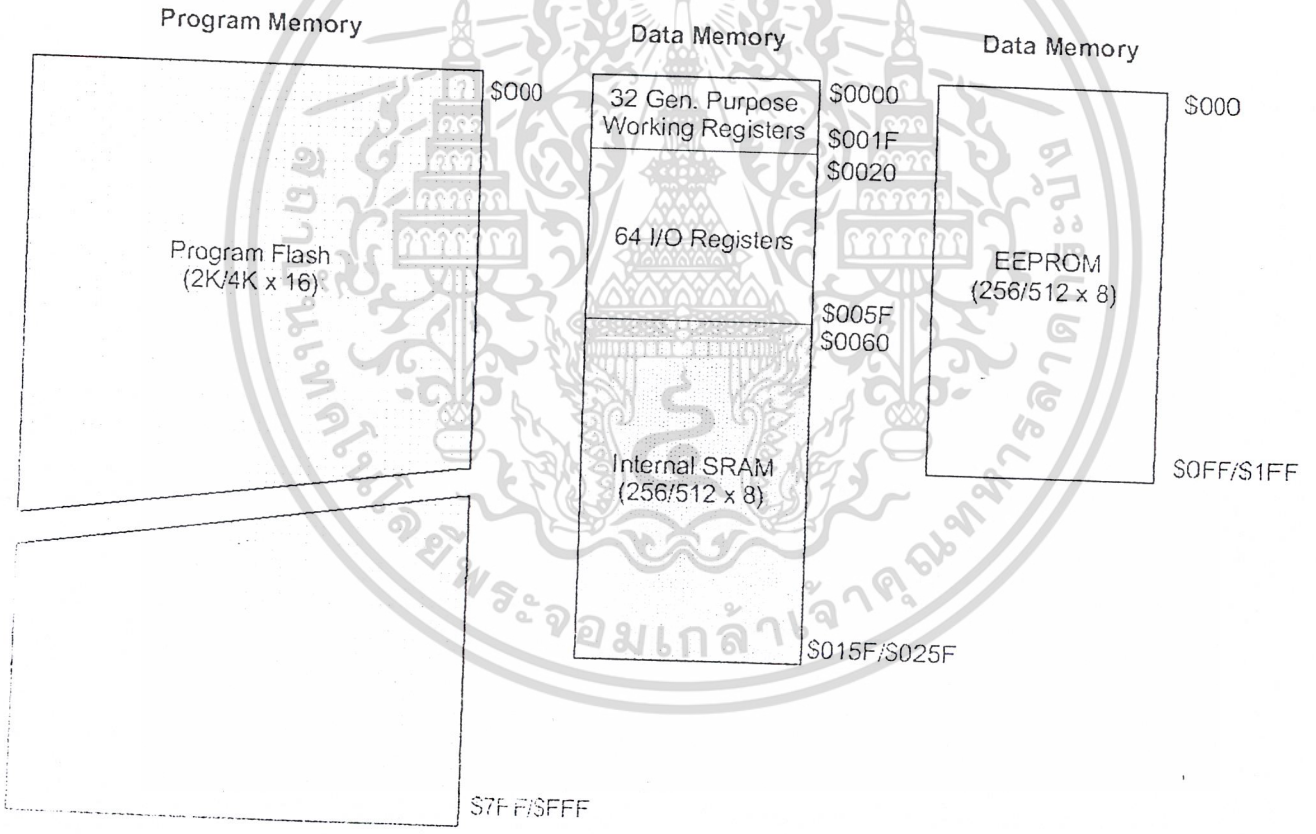
With the relative jump and call instructions, the whole 2K/4K address space is directly accessed. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

During interrupts and subroutine calls, the return address program counter (PC) is stored on the stack. The stack is effectively allocated in the general data SRAM, and consequently the stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The 9/10-bit stack pointer SP is read/write accessible in the I/O space.

The 256/512 bytes data SRAM can be easily accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

Figure 5. Memory Maps



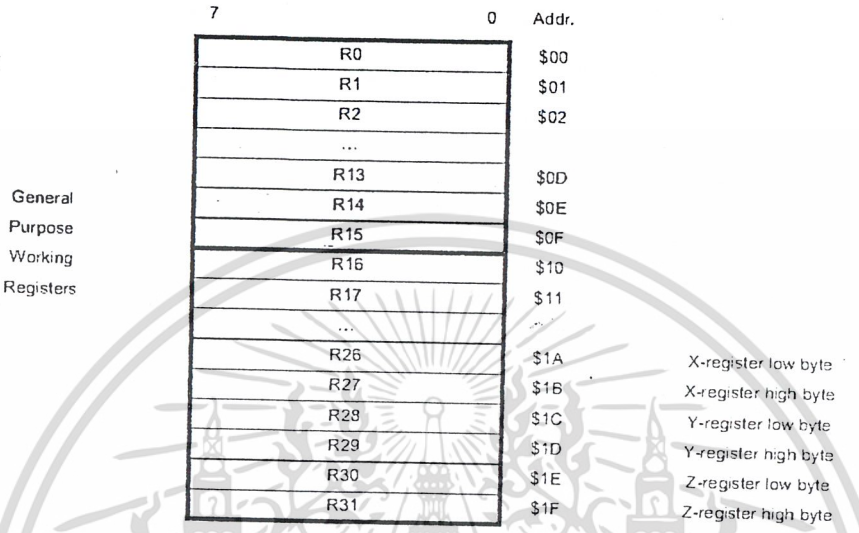
A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. All the different interrupts have a separate interrupt vector in the interrupt vector table at the beginning of the program memory. The different interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

General Purpose Register File

Figure 6 shows the structure of the 32 general purpose working registers in the CPU.

Figure 6. AVR CPU General Purpose Working Registers



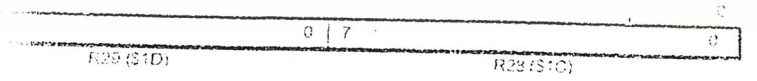
All the register operating instructions in the instruction set have direct and single cycle access to all registers. The only exception [illegible] constant arithmetic and logic instructions SBCI, SUBI, CPI, ANDI and ORI between a constant and a register and the [illegible] instruction for load immediate constant data. These instructions apply to the second half of the registers in the register file - R16..R31. The general SBC, SUB, CP, AND and OR and all other operations between two registers or [illegible] apply to the entire register file.

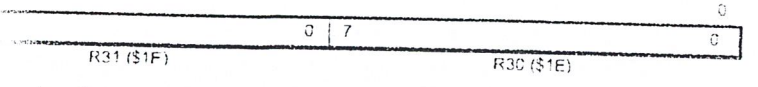
As [illegible] each register is also assigned a data memory address, mapping them directly into the first 32 locations [illegible] space. Although not being physically implemented as SRAM locations, this memory organization [illegible] access of the registers, as the X, Y and Z registers can be set to index any register in the file.

[illegible] And Z-register

[illegible] have some added functions to their general purpose usage. These registers are address pointers [illegible] the Data Space. The three indirect address registers X, Y and Z are defined as:

[illegible] isters





In [illegible] modes these address registers have functions as fixed displacement, automatic increment and [illegible] for the different instructions).

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ALU - Arithmetic Logic Unit

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, ALU operations between registers in the register file are executed. The ALU operations are divided into three main categories - arithmetic, logical, and bit-functions.

In-System Programmable Flash Program Memory

The AT90S4434/8535 contains 4K/8K bytes on-chip In-System Programmable Flash memory for program storage. Since all instructions are 16-or 32-bit words, the Flash is organized as 2K/4K x 16. The Flash memory has an endurance of at least 1000 write/erase cycles. The AT90S4434/8535 Program Counter (PC) is 11/12 bits wide, thus addressing the 2048/4096 program memory addresses.

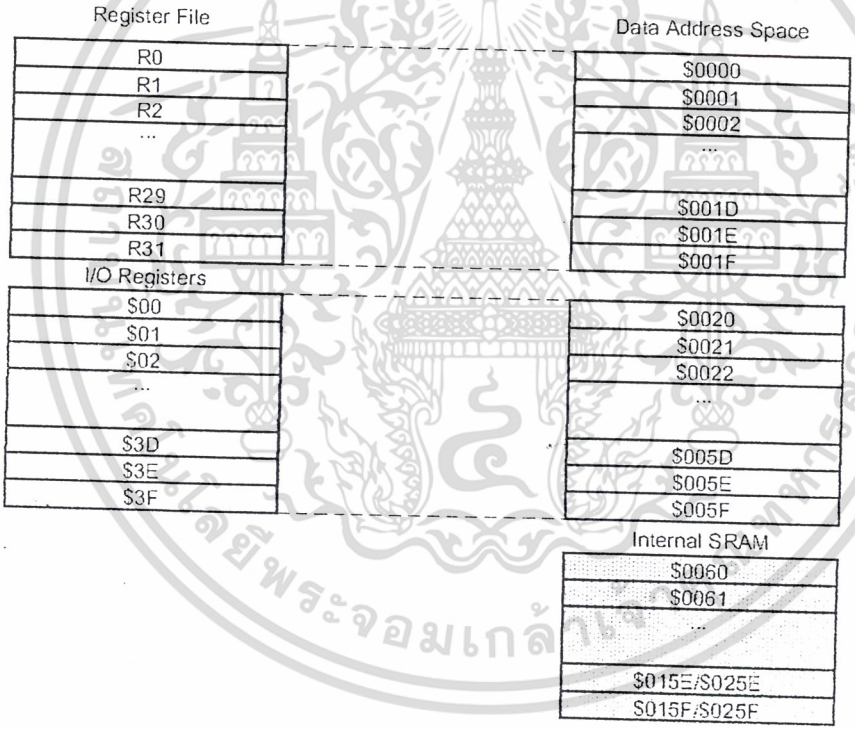
See page 89 for a detailed description on Flash data downloading.

See page 10 for the different program memory addressing modes.

SRAM Data Memory

The following figure shows how the AT90S4434/8535 SRAM Memory is organized:

Figure 8. SRAM Organization



e lower 352/608 Data Memory locations address the Register file, the I/O Memory, and the internal data SRAM. The first locations address the Register File + I/O Memory, and the next 256/512 locations address the internal data SRAM.

e five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Decrement and Indirect with Post-increment. In the register file, registers R26 to R31 feature the indirect addressing inter registers.

e direct addressing reaches the entire data space.

e Indirect with Displacement mode features a 63 address locations reach from the base address given by the Y or Z-jister.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

EEPROM Data Memory

The AT90S4434/8535 contains 256/512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described on page 46 specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register.

For the SPI data downloading, see page 89 for a detailed description.

Memory Access Times and Instruction Execution Timing

This section describes the general access timing concepts for instruction execution and internal memory access.

The AVR CPU is driven by the System Clock Ø, directly generated from the external clock crystal for the chip. No internal clock division is used.

Figure 20 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access register file concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 20. The Parallel Instruction Fetches and Instruction Executions

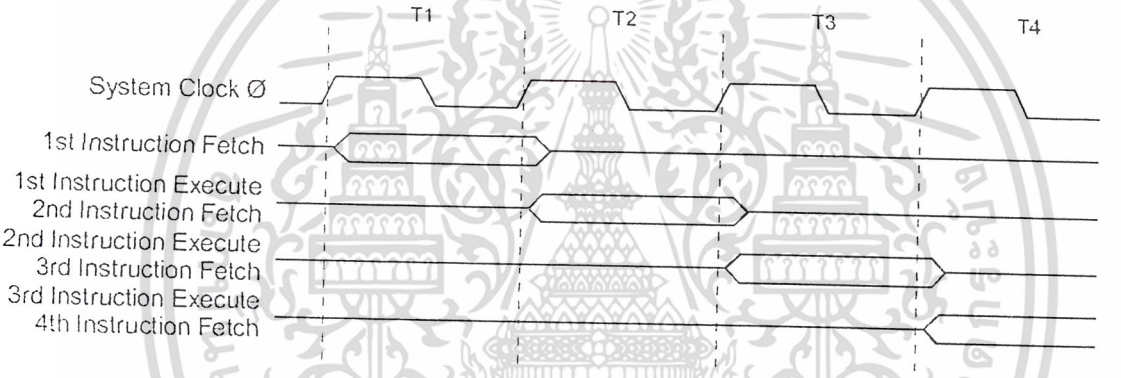
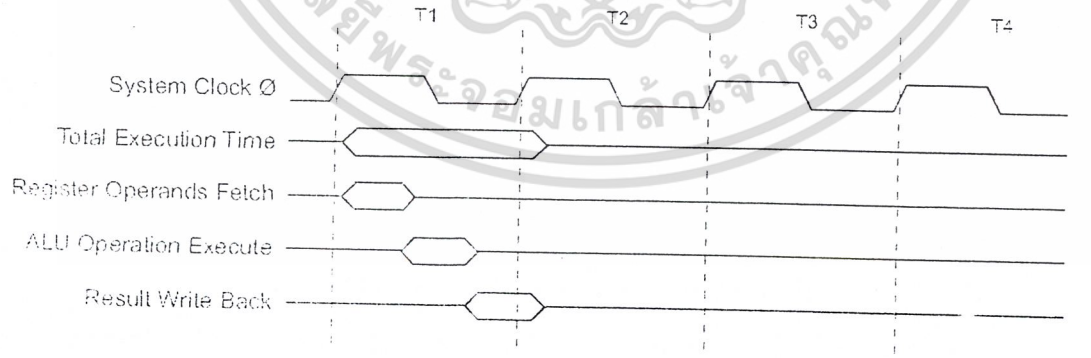


Figure 21 shows the internal timing concept for the register file. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

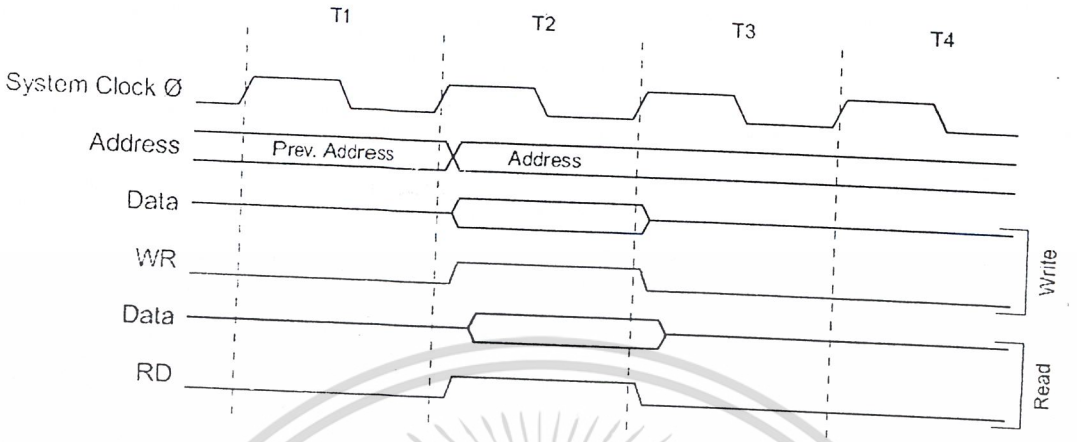
Figure 21. Single Cycle ALU Operation



The internal data SRAM access is performed in two System Clock cycles as described in Figure 22.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 22. On-Chip Data SRAM Access Cycles



I/O Memory

The I/O space definition of the AT90S4434/8535 is shown in the following table:

Table 2. AT90S4434/8535 I/O Space

I/O Address (SRAM Address)	Name	Function
$3F ($5F)	SREG	Status REGister
$3E ($5E)	SPH	Stack Pointer High
$3D ($5D)	SPL	Stack Pointer Low
$3B ($5B)	GIMSK	General Interrupt MaSK register
$3A ($5A)	GIFR	General Interrupt Flag Register
$39 ($59)	TIMSK	Timer/Counter Interrupt MaSK register
$38 ($58)	TIFR	Timer/Counter Interrupt Flag register
$35 ($55)	MCUCR	MCU general Control Register
$34 ($54)	MCUSR	MCU general Status Register
$33 ($53)	TCCR0	Timer/Counter0 Control Register
$32 ($52)	TCNT0	Timer/Counter0 (8-bit)
$2F ($4F)	TCCR1A	Timer/Counter1 Control Register A
$2E ($4E)	TCCR1B	Timer/Counter1 Control Register B
$2D ($4D)	TCNT1H	Timer/Counter1 High Byte
$2C ($4C)	TCNT1L	Timer/Counter1 Low Byte
$2B ($4B)	OCR1AH	Timer/Counter1 Output Compare Register A High Byte
$2A ($4A)	OCR1AL	Timer/Counter1 Output Compare Register A Low Byte
$29 ($49)	OCR1BH	Timer/Counter1 Output Compare Register B High Byte
$28 ($48)	OCR1BL	Timer/Counter1 Output Compare Register B Low Byte
$27 ($47)	ICR1H	T/C 1 Input Capture Register High Byte
$26 ($46)	ICR1L	T/C 1 Input Capture Register Low Byte
$25 ($45)	TCCR2	Timer/Counter2 Control Register

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Table 2. AT90S4434/8535 I/O Space (Continued)

I/O Address (SRAM Address)	Name	Function
$24 ($44)	TCNT2	Timer/Counter2 (8-bit)
$23 ($43)	OCR2	Timer/Counter2 Output Compare Register
$22 ($42)	ASSR	Asynchronous Mode Status Register
$21 ($41)	WDTCR	Watchdog Timer Control Register
$1F ($3E)	EEARH	EEPROM Address Register High Byte
$1E ($3E)	EEARL	EEPROM Address Register Low Byte
$1D ($3D)	EEDR	EEPROM Data Register
$1C ($3C)	EECR	EEPROM Control Register
$1B ($3B)	PORTA	Data Register, Port A
$1A ($3A)	DDRA	Data Direction Register, Port A
$19 ($39)	PINA	Input Pins, Port A
$18 ($38)	PORTB	Data Register, Port B
$17 ($37)	DDRB	Data Direction Register, Port B
$16 ($36)	PINB	Input Pins, Port B
$15 ($35)	PORTC	Data Register, Port C
$14 ($34)	DDRC	Data Direction Register, Port C
$13 ($33)	PINC	Input Pins, Port C
$12 ($32)	PORTD	Data Register, Port D
$11 ($31)	DDRD	Data Direction Register, Port D
$10 ($30)	PIND	Input Pins, Port D
$0F ($2F)	SPDR	SPI I/O Data Register
$0E ($2E)	SPSR	SPI Status Register
$0D ($2D)	SPCR	SPI Control Register
$0C ($2C)	UDR	UART I/O Data Register
$0B ($2B)	USR	UART Status Register
$0A ($2A)	UCR	UART Control Register
$09 ($29)	UBRR	UART Baud Rate Register
$08 ($28)	ACSR	Analog Comparator Control and Status Register
$07 ($27)	ADMUX	ADC Multiplexer Select Register
$06 ($26)	ADCSR	ADC Control and Status Register
$05 ($25)	ADCH	ADC Data Register High
$04 ($24)	ADCL	ADC Data Register Low

Note: Reserved and unused locations are not shown in the table.

All AT90S4434/8535 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions transferring data between the 32 general purpose working registers and the I/O space. I/O registers within the address range $00 - $1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set chapter for more details.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

When using the I/O specific commands IN, OUT the I/O addresses $00 - $3F must be used. When addressing I/O registers as SRAM, $20 must be added to these addresses. All I/O register addresses throughout this document are shown with the SRAM address in parentheses.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers $00 to $1F only.

The I/O and peripherals control registers are explained in the following sections.

Status Register - SREG

The AVR status register - SREG - at I/O space location $3F ($5F) is defined as:

Bit	7	6	5	4	3	2	1	0	
$3F ($5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 - I: Global Interrupt Enable**

The global interrupt enable bit must be set (one) for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the global interrupt enable register is cleared (zero), none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts.

- **Bit 6 - T: Bit Copy Storage**

The bit copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T bit as source and destination for the operated bit. A bit from a register in the register file can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the register file by the BLD instruction.

- **Bit 5 - H: Half Carry Flag**

The half carry flag H indicates a half carry in some arithmetic operations. See the Instruction Set Description for detailed information.

- **Bit 4 - S: Sign Bit, S = N ⊕ V**

The S-bit is always an exclusive or between the negative flag N and the two's complement overflow flag V. See the Instruction Set Description for detailed information.

- **Bit 3 - V: Two's Complement Overflow Flag**

The two's complement overflow flag V supports two's complement arithmetics. See the Instruction Set Description for detailed information.

- **Bit 2 - N: Negative Flag**

The negative flag N indicates a negative result from an arithmetical or logical operation. See the Instruction Set Description for detailed information.

- **Bit 1 - Z: Zero Flag**

The zero flag Z indicates a zero result from an arithmetical or logic operation. See the Instruction Set Description for detailed information.

- **Bit 0 - C: Carry Flag**

The carry flag C indicates a carry in an arithmetical or logical operation. See the Instruction Set Description for detailed information.

Note that the status register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt routine. This must be handled by software.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Stack Pointer - SP

The AT90S4434/8535 Stack Pointer is implemented as two 8-bit registers in the I/O space locations $3E ($5E) and $3D ($5D). As the AT90S4434/8535 data memory has $15F/$25F locations, 9/10 bits are used.

Bit	15	14	13	12	11	10	9	8	
$3E ($5E)	-	-	-	-	-	-	SP9	SP8	SPH
$3D ($5D)	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

The Stack Pointer points to the data SRAM stack area where the Subroutine and Interrupt Stacks are located. This Stack space in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. The stack pointer must be set to point above $60. The Stack Pointer is decremented by one when data is pushed onto the Stack with the PUSH instruction, and it is decremented by two when an address is pushed onto the Stack with subroutine calls and interrupts. The Stack Pointer is incremented by one when data is popped from the Stack with the POP instruction, and it is incremented by two when an address is popped from the Stack with return from subroutine RET or return from interrupt RETI.

Reset and Interrupt Handling

The AT90S4434/8535 provides 16 different interrupt sources. These interrupts and the separate reset vector, each have a separate program vector in the program memory space. All interrupts are assigned individual enable bits which must be set (one) together with the I-bit in the status register in order to enable the interrupt.

The lowest addresses in the program memory space are automatically defined as the Reset and Interrupt vectors. The complete list of vectors is shown in Table 3. The list also determines the priority levels of the different interrupts. The lower the address the higher is the priority level. RESET has the highest priority, and next is INT0 - the External Interrupt Request 0.

Reset and Interrupt Vectors

Program Address	Source	Interrupt Definition
$000	RESET	Hardware Pin, Power-On Reset and Watchdog Reset
$001	INT0	External Interrupt Request 0
$002	INT1	External Interrupt Request 1
$003	TIMER2 COMP	Timer/Counter2 Compare Match
$004	TIMER2 OVF	Timer/Counter2 Overflow
$005	TIMER1 CAPT	Timer/Counter1 Capture Event
$006	TIMER1 COMPA	Timer/Counter1 Compare Match A
	TIMER1 COMPB	Timer/Counter1 Compare Match B
	TIMER1 OVF	Timer/Counter1 Overflow
	TIMER0 OVF	Timer/Counter0 Overflow
	SPI, STC	SPI Serial Transfer Complete
$0013	UART, RX	UART, Rx Complete
	UART, UDRE	UART Data Register Empty
	UART, TX	UART, Tx Complete

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Table 3. Reset and Interrupt Vectors (Continued)

Vector No.	Program Address	Source	Interrupt Definition
15	$00E	ADC	ADC Conversion Complete
16	$00F	EE_RDY	EEPROM Ready
17	$010	ANA_COMP	Analog Comparator

The ... general program setup for the Reset and Interrupt Vector Addresses are:

```
Labels    Code                Comments
$000      rjmp    RESET       ; Reset Handler
$001      rjmp    EXT_INT0    ; IRQ0 Handler
$002      rjmp    EXT_INT1    ; IRQ1 Handler
$003      rjmp    TIM2_COMP   ; Timer2 Compare Handler
$004      rjmp    TIM2_OVF    ; Timer2 Overflow Handler
$005      rjmp    TIM1_CAPT   ; Timer1 Capture Handler
$006      rjmp    TIM1_COMPA  ; Timer1 CompareA Handler
$007      rjmp    TIM1_COMPB  ; Timer1 CompareB Handler
$008      rjmp    TIM1_OVF    ; Timer1 Overflow Handler
$009      rjmp    TIM0_OVF    ; Timer0 Overflow Handler
$00A      rjmp    SPI_STC;    ; SPI Transfer Complete Handler
$00B      rjmp    UART_RXC    ; UART RX Complete Handler
$00C      rjmp    UART_DRE    ; UDR Empty Handler
$00D      rjmp    UART_TXC    ; UART TX Complete Handler
$00E      rjmp    ADC         ; ADC Conversion Complete Interrupt Handler
$00F      rjmp    EE_RDY      ; EEPROM Ready Handler
$010      rjmp    ANA_COMP    ; Analog Comparator Handler
MAIN:     ldi     r16, high(RAMEND); Main program start
          out     SPH,r16
          ldi     r16, low(RAMEND);
          out     SPL,r16
          <instr> xxx
          ...     ...
```

... three sources of reset:

...U is reset when the supply voltage is below the power-on reset threshold (V_{POT}).

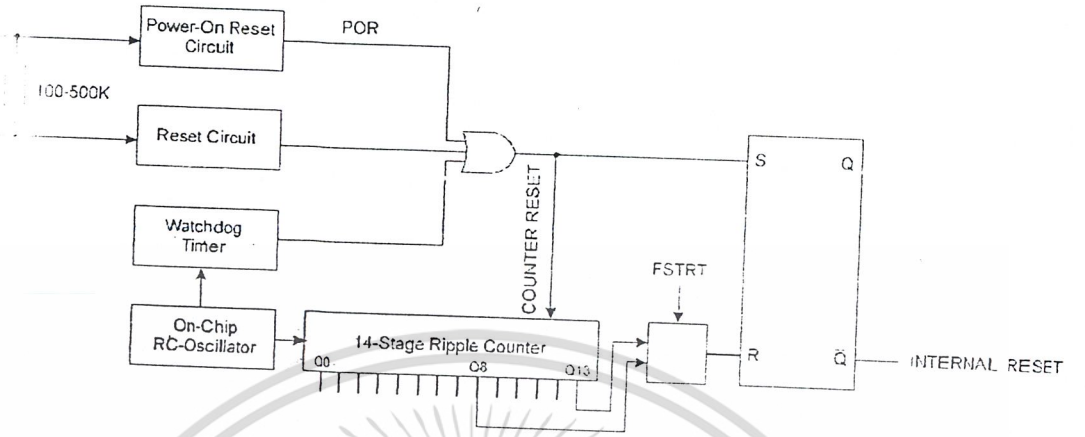
... reset when a low level is present on the $\overline{\text{RESET}}$ pin for more than 50 ns.

...U is reset when the Watchdog timer period expires, and the Watchdog is enabled

...rs are set to their initial values, and the program starts execution from address $000. The ...000 must be an RJMP - relative jump - instruction to the reset handling routine. If the pro... source, the interrupt vectors are not used, and regular program code can be placed at ...am in Figure 23 shows the reset logic. Table 4 defines the timing and electrical parameters

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Fig[illegible] Logic



Ta[illegible] acteristics (V_{CC} = 5.0V)

	Min	Typ	Max	Units
[illegible] Reset Threshold (rising)	1.0	1.4	1.8	V
[illegible]set Threshold (falling)	0.4	0.6	0.8	V
[illegible]shold Voltage		$0.6V_{CC}$		V
[illegible] Out Period [illegible]rammed	11	16	21	ms
[illegible]ut Period [illegible]od	1.0	1.1	1.2	ms

[illegible] will not work unless the supply voltage has been below V_{POT} (falling)

[illegible] Oscillator Cycles

Time-out at V_{CC} = 5V	Number of WDT cycles
1.1ms	1K
16.0ms	16K

[illegible]nsures that the device is reset from power-on. As shown in Figure 23, an internal timer [illegible] oscillator prevents the MCU from starting until after a certain period after V_{CC} has [illegible]oltage - V_{POT}, regardless of the V_{CC} rise time (see Figure 24).

[illegible]e according to typical oscillator start-up time. The number of WDT oscillator cycles is [illegible] the watchdog oscillator is voltage dependent as shown in "Typical Characteristics" on

[illegible]t, $\overline{\text{RESET}}$ can be connected to V_{CC} directly or via an external pull-up resistor. By hold- [illegible]as been applied, the Power-On Reset period can be extended. Refer to Figure 25 for

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 24. MCU Start-Up, $\overline{\text{RESET}}$ Tied to V_{CC}.

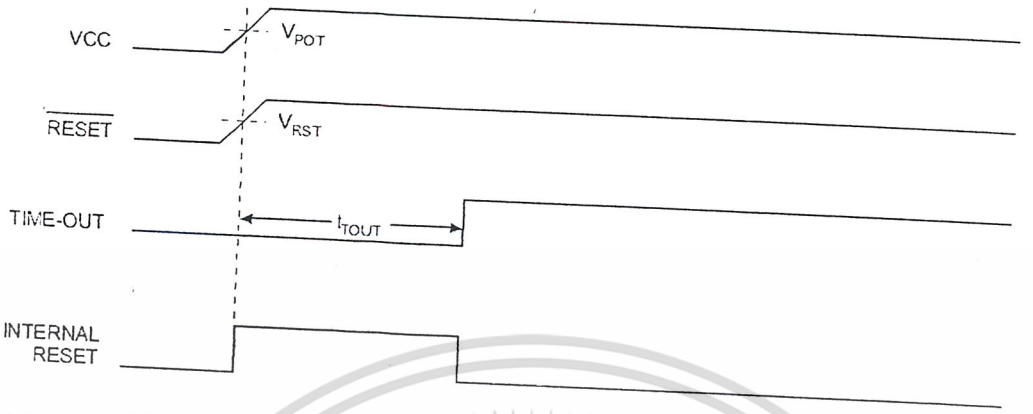
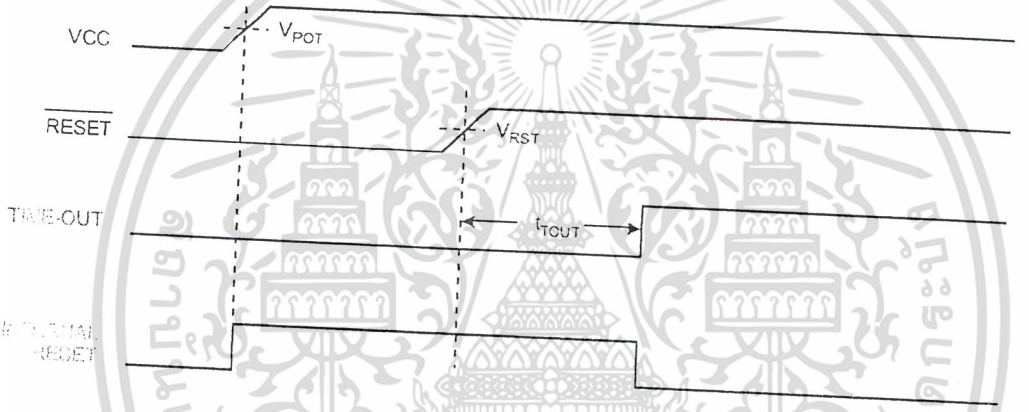


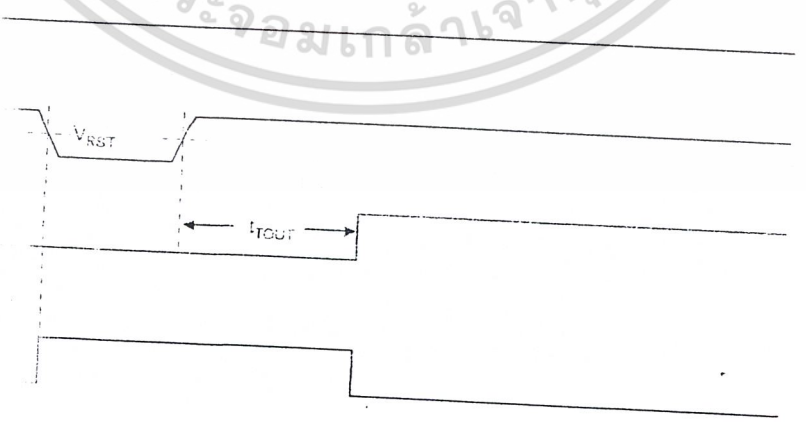
Figure 25. MCU Start-Up, $\overline{\text{RESET}}$ Controlled Externally



External Reset

...d by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than 50 ns will generate a reset, even ...horter pulses are not guaranteed to generate a reset. When the applied signal reaches the ...RST on its positive edge, the delay timer starts the MCU after the Time-out period t_{TOUT} has ...

...ring Operation

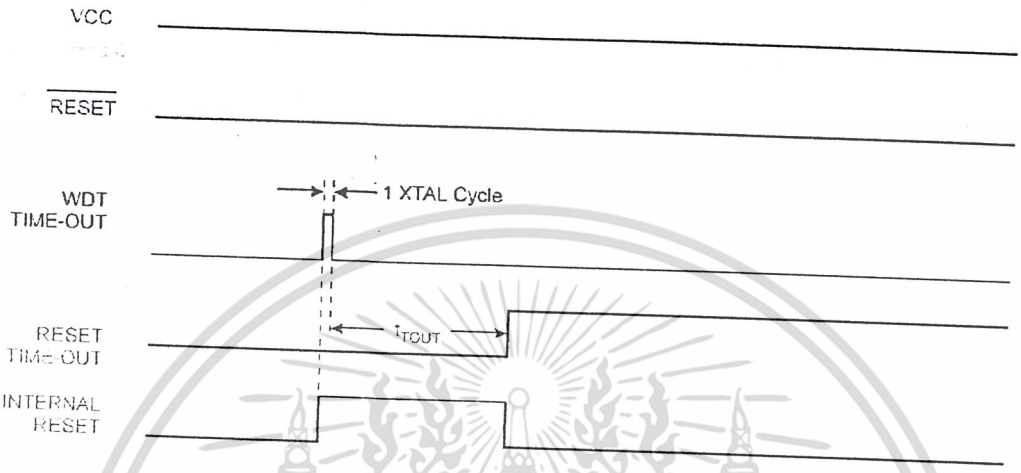


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Watchdog Reset

When the Watchdog times out, it will generate a short reset pulse of 1 XTAL cycle duration. On the falling edge of this pulse, the delay timer starts counting the Time-out period t_{TOUT}. Refer to page 44 for details on operation of the Watchdog.

Figure 27. Watchdog Reset During Operation



MCU Status Register - MCUSR

The MCU Status Register provides information on which reset source caused an MCU reset.

7	6	5	4	3	2	1	0	
-	-	-	-	-	-	EXTRF	PORF	MCUSR
R	R	R	R	R	R	R/W	R/W	
0	0	0	0	0	0	See bit description		

Bits 7..2 - Reserved Bits

These bits are reserved bits in the AT90S4434/8535 and always read as zero.

Bit 1 - EXTRF: External Reset Flag

This bit is undefined (X). It can only be set by an external reset. A watchdog reset will leave this bit unchanged. The bit is reset by writing a logical one to the bit.

Bit 0 - PORF: Power-on Reset Flag

This bit is set by a power-on reset. A watchdog reset or an external reset will leave this bit unchanged. The bit is reset by writing a logical one to the bit.

The following table shows the value of these two bits after the three modes of reset:

Table. Values after Reset

EXTRF	PORF
undefined	1
1	unchanged
unchanged	unchanged

To identify a reset condition, the user software should clear both the PORF and EXTRF bits as soon as possible. Checking the PORF and EXTRF values is done before the bits are cleared. If the bit is cleared when a watchdog reset occurs, the source of reset can be found by using the following truth table:

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Table 9. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Reserved
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

Note: When changing the ISC11/ISC10 bits, INT1 must be disabled by clearing its Interrupt Enable bit in the GIMSK Register. Otherwise an interrupt can occur when the bits are changed.

The value on the INT pin is sampled before detecting edges. If edge interrupt is selected, pulses that last longer than one CPU clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt. If enabled, a level triggered interrupt will generate an interrupt request as long as the pin is held low.

- **Bit 1, 0 - ISC01, ISC00: Interrupt Sense Control 0 bit 1 and bit 0**

The External Interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask is set. The level and edges on the external INT0 pin that activate the interrupt are defined in the following table:

Table 10. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Reserved
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

Note: When changing the ISC01/ISC00 bits, INT0 must be disabled by clearing its Interrupt Enable bit in the GIMSK Register. Otherwise an interrupt can occur when the bits are changed.

The value on the INT pin is sampled before detecting edges. If edge interrupt is selected, pulses that last longer than one CPU clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt. If enabled, a level triggered interrupt will generate an interrupt request as long as the pin is held low.

Sleep Modes

To enter any of the three sleep modes, the SE bit in MCUCR must be set (one) and a SLEEP instruction must be executed. The SM0 and SM1 bits in the MCUCR register select which sleep mode (Idle, Power Down, or Power Save) will be activated by the SLEEP instruction, see Table 8.

If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up, executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the register file, SRAM, and I/O memory are unaltered. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset vector.

Idle Mode

When the SM1/SM0 bits are set to 00, the SLEEP instruction makes the MCU enter the Idle Mode, stopping the CPU but allowing SPI, UARTs, Analog Comparator, ADC, Timer/Counters, Watchdog and the interrupt system to continue operating. This enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow and UART Receive Complete interrupts. If wake-up from the Analog Comparator interrupt is not required, the analog comparator can be powered down by setting the ACD-bit in the Analog Comparator Control and Status register - ACSR. This will reduce power consumption in Idle Mode. When the MCU wakes up from Idle mode, the CPU starts program execution immediately.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Power Down Mode

When the SM1/SM0 bits are set to 10, the SLEEP instruction makes the MCU enter the Power Down Mode. In this mode, the external oscillator is stopped, while the external interrupts and the Watchdog (if enabled) continue operating. Only an external reset, a watchdog reset (if enabled), or an external level interrupt can wake up the MCU.

Note that if a level trigged interrupt is used for wake-up from Power Down Mode, the changed level must be held for some time to wake up the MCU. This makes the MCU less sensitive to noise. The changed level is sampled twice by the watchdog oscillator clock, and if the input has the required level during this time, the MCU will wake up. The period of the watchdog oscillator is 1 µs (nominal) at 5.0V and 25°C. The frequency of the watchdog oscillator is voltage dependent as shown in section "Typical Characteristics" on page 96.

When waking up from Power Down Mode, there is a delay from the wake-up condition occurs until the wake-up becomes effective. This allows the clock to restart and become stable after having been stopped. The wake-up period is equal to the reset period, as shown in Table 4.

If the wake-up condition disappears before the MCU wakes up and starts to execute, e.g. a low level on is not held long enough, the interrupt causing the wake-up will not be executed.

Power Save Mode

When the SM1/SM0 bits are 11, the SLEEP instruction makes the MCU enter the Power Save Mode. This mode is identical to Power Down, with one exception:

If Timer/Counter2 is clocked asynchronously, i.e. the AS2 bit in ASSR is set, Timer/Counter2 will run during sleep. In addition to the Power Down wake-up sources, the device can also wake up from either Timer Overflow or Output Compare event from Timer/Counter2 if the corresponding Timer/Counter2 interrupt enable bits are set in TIMSK and the global interrupt enable bit in SREG is set.

When waking up from Power Save Mode by an external interrupt, 2 instruction cycles are executed before the interrupt flags are updated. When waking up by the asynchronous timer, 3 instruction cycles are executed before the flags are updated. During these cycles, the processor executes instructions, but the interrupt condition is not readable, and the interrupt routine has not started yet.

When waking up from Power Save Mode by an asynchronous timer interrupt, the part will wake up even if global interrupts are disabled.

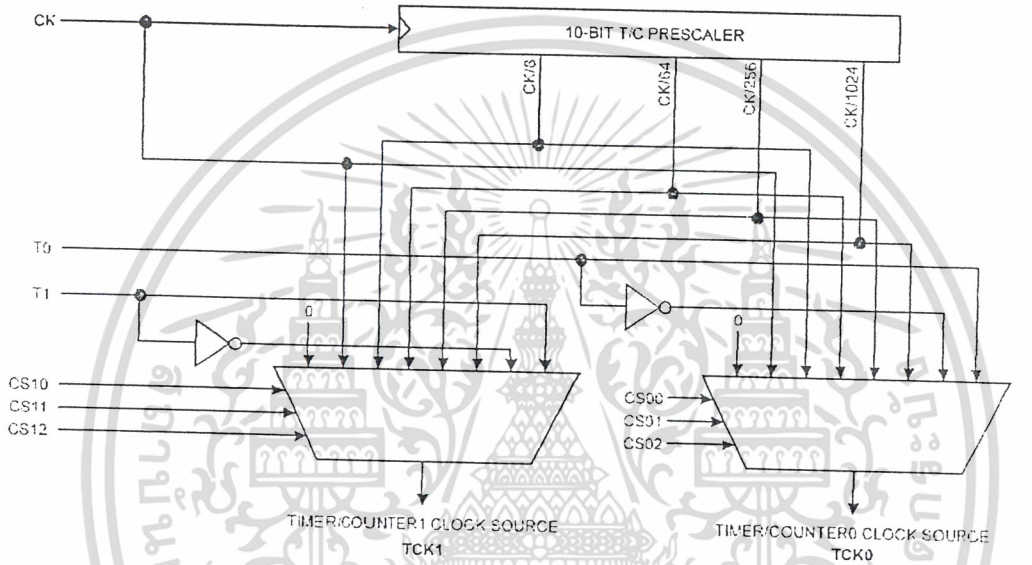
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Timer / Counters

The AT90S4434/8535 provides three general purpose Timer/Counters - two 8-bit T/Cs and one 16-bit T/C. Timer/Counter2 can optionally be asynchronously clocked from an external oscillator. This oscillator is optimized for use with a 32.768 kHz watch crystal, enabling use of Timer/Counter2 as a Real Time Clock (RTC). Timer/Counters 0 and 1 have individual prescaling selection from the same 10-bit prescaling timer. Timer/Counter2 has its own prescaler. These Timer/Counters can either be used as a timer with an internal clock timebase or as a counter with an external pin connection which triggers the counting.

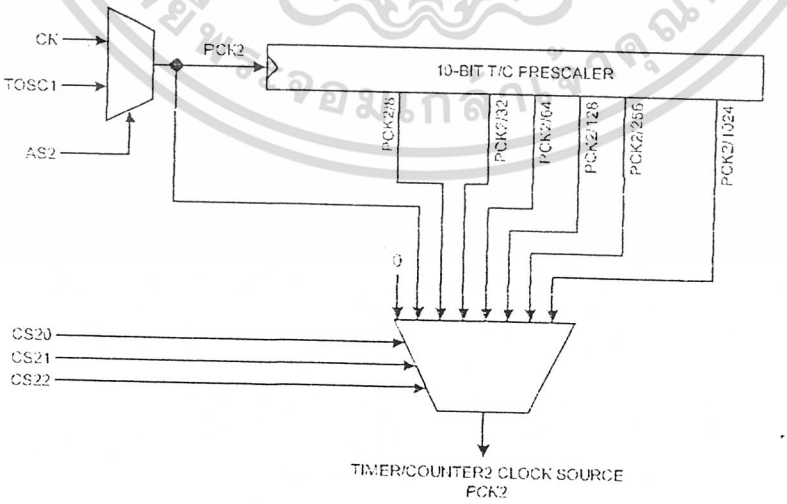
Timer/Counter Prescalers

Figure 28. Prescaler for Timer/Counter0 and 1



For Timer/Counters 0 and 1, the four different prescaled selections are: CK/8, CK/64, CK/256 and CK/1024, where CK is the oscillator clock. For the two Timer/Counters 0 and 1, CK, external source, and stop, can also be selected as clock sources.

Figure 29. Timer/Counter2 Prescaler



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

The clock source for Timer/Counter2 prescaler is named PCK2. PCK2 is by default connected to the main system clock CK. By setting the AS2 bit in ASSR, Timer/Counter2 prescaler is asynchronously clocked from the PC6(TOSC1) pin. This enables use of Timer/Counter2 as a Real Time Clock (RTC). When AS2 is set, pins PC6(TOSC1) and PC7(TOSC2) are disconnected from Port C. A crystal can then be connected between the PC6(TOSC1) and PC7(TOSC2) pins to serve as an independent clock source for Timer/Counter2. The oscillator is optimized for use with a 32.768 kHz crystal. Alternatively, an external clock signal can be applied to PC6(TOSC1). The frequency of this clock must be lower than one fourth of the CPU clock and not higher than 256 kHz.

8-bit Timer/Counter0

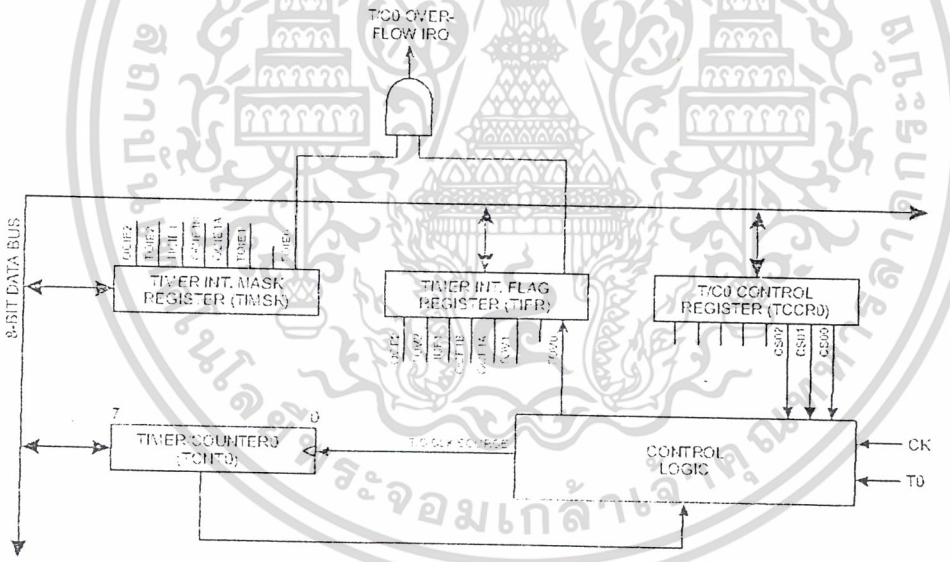
Figure 30 shows the block diagram for Timer/Counter0.

The 8-bit Timer/Counter0 can select clock source from CK, prescaled CK, or an external pin. In addition it can be stopped as described in the specification for the Timer/Counter0 Control Register - TCCR0. The overflow status flag is found in the Timer/Counter Interrupt Flag Register - TIFR. Control signals are found in the Timer/Counter0 Control Register - TCCR0. The interrupt enable/disable settings for Timer/Counter0 are found in the Timer/Counter Interrupt Mask Register - TIMSK.

When Timer/Counter0 is externally clocked, the external signal is synchronized with the oscillator frequency of the CPU. To assure proper sampling of the external clock, the minimum time between two external clock transitions must be at least one internal CPU clock period. The external clock signal is sampled on the rising edge of the internal CPU clock.

The 8-bit Timer/Counter0 features both a high resolution and a high accuracy usage with the lower prescaling opportunities. Similarly, the high prescaling opportunities make the Timer/Counter0 useful for lower speed functions or exact timing functions with infrequent actions.

Figure 30. Timer/Counter0 Block Diagram



Timer/Counter0 Control Register - TCCR0

Bit	7	6	5	4	3	2	1	0	
$33 ($53)	-	-	-	-	-	CS02	CS01	CS00	TCCR0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Bits 7..3 - Res: Reserved bits

These bits are reserved bits in the AT90S4434/8535 and always read zero.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **Bits 2,1,0 - CS02, CS01, CS00: Clock Select0, bit 2,1 and 0**

The Clock Select0 bits 2,1 and 0 define the prescaling source of Timer0.

Table 11. Clock 0 Prescale Select

CS02	CS01	CS00	Description
0	0	0	Stop, Timer/Counter0 is stopped.
0	0	1	CK
0	1	0	CK / 8
0	1	1	CK / 64
1	0	0	CK / 256
1	0	1	CK / 1024
1	1	0	External Pin T0, falling edge
1	1	1	External Pin T0, rising edge

The Stop condition provides a Timer Enable/Disable function. The prescaled CK modes are scaled directly from the CK oscillator clock. If the external pin modes are used, the corresponding setup must be performed in the actual data direction control register (cleared to zero gives an input pin).

Timer Counter 0 - TCNT0

Bit	7	6	5	4	3	2	1	0	
$32 ($52)	MSB							LSB	TCNT0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

The Timer/Counter0 is realized as an up-counter with read and write access. If the Timer/Counter0 is written and a clock source is present, the Timer/Counter0 continues counting in the clock cycle following the write operation.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

UART

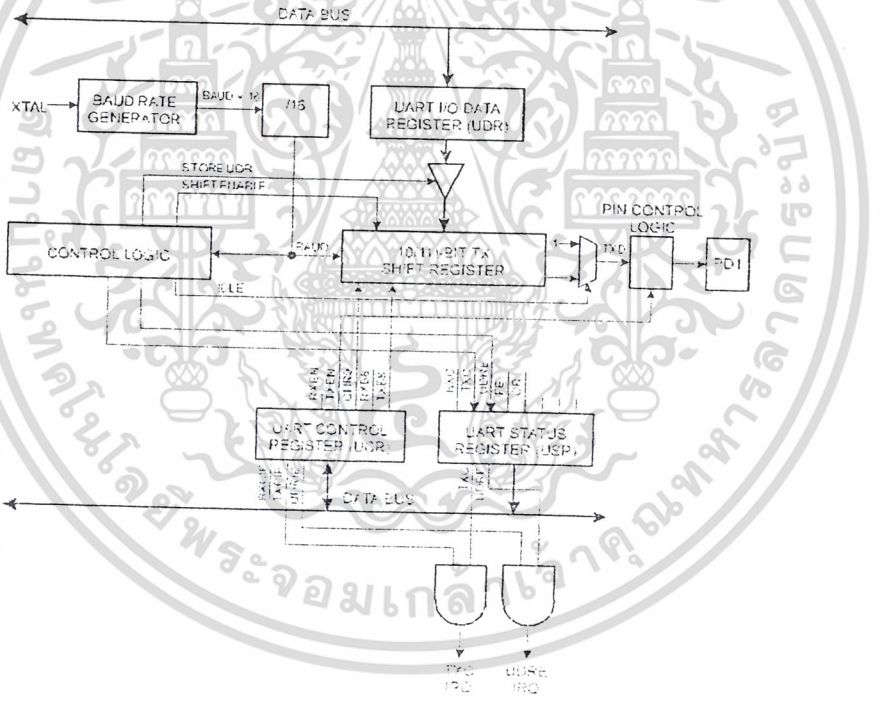
The AT90S4434/8535 features a full duplex (separate receive and transmit registers) Universal Asynchronous Receiver and Transmitter (UART). The main features are:

- Baud rate generator that can generate a large number of baud rates (bps)
- High baud rates at low XTAL frequencies
- 8 or 9 data bits
- No parity
- Overrun detection
- Framing error detection
- False Start Bit detection
- Three separate interrupts on TX Complete, TX Data Register Empty, and RX Complete
- Buffered Transmit and Receive.

Data Transmission

A block diagram of the UART transmitter is shown in Figure 41.

Figure 41. UART Transmitter



...ed by writing the data to be transmitted to the UART I/O Data Register, UDR. Data is transferred ...shift register when:

- ...to UDR after the stop bit from the previous character has been shifted out. The shift register is ...
- ...to UDR before the stop bit from the previous character has been shifted out. The shift register ...bit of the character currently being transmitted is shifted out.

...shift register is empty, data is transferred from UDR to the shift register. The UDRE (UART ...the UART Status Register, USR, is set. When this bit is set (one), the UART is ready to receive

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

nection between the analog comparator and the Input Capture function is given. To make the comparator trigger the Timer/Counter1 Input Capture interrupt, the TICIE1 bit in the Timer Interrupt Mask Register (TIMSK) must be set (one).

- **Bits 1,0 - ACIS1, ACIS0: Analog Comparator Interrupt Mode Select**

These bits determine which comparator events that trigger the Analog Comparator interrupt. The different settings are shown in Table 26.

Table 26. ACIS1/ACIS0 Settings

ACIS1	ACIS0	Interrupt Mode
0	0	Comparator Interrupt on Output Toggle
0	1	Reserved
1	0	Comparator Interrupt on Falling Output Edge
1	1	Comparator Interrupt on Rising Output Edge

Note: When changing the ACIS1/ACIS0 bits, The Analog Comparator Interrupt must be disabled by clearing its Interrupt Enable bit in the ACSR register. Otherwise an interrupt can occur when the bits are changed.

Caution: Using the SBI or CBI instruction on other bits than ACI in this register, will write a one back into ACI if it is read as set, thus clearing the flag.

Analog to Digital Converter

Feature list:

- 10-bit Resolution
- ±2 LSB Accuracy
- 0.5 LSB Integral Non-Linearity
- 65 - 260 µs Conversion Time
- Up to 15 kSPS
- 8 Multiplexed Input Channels
- Rail-to-Rail Input Range
- Free Run or Single Conversion Mode
- Interrupt on ADC Conversion Complete
- Sleep Mode Noise Canceler

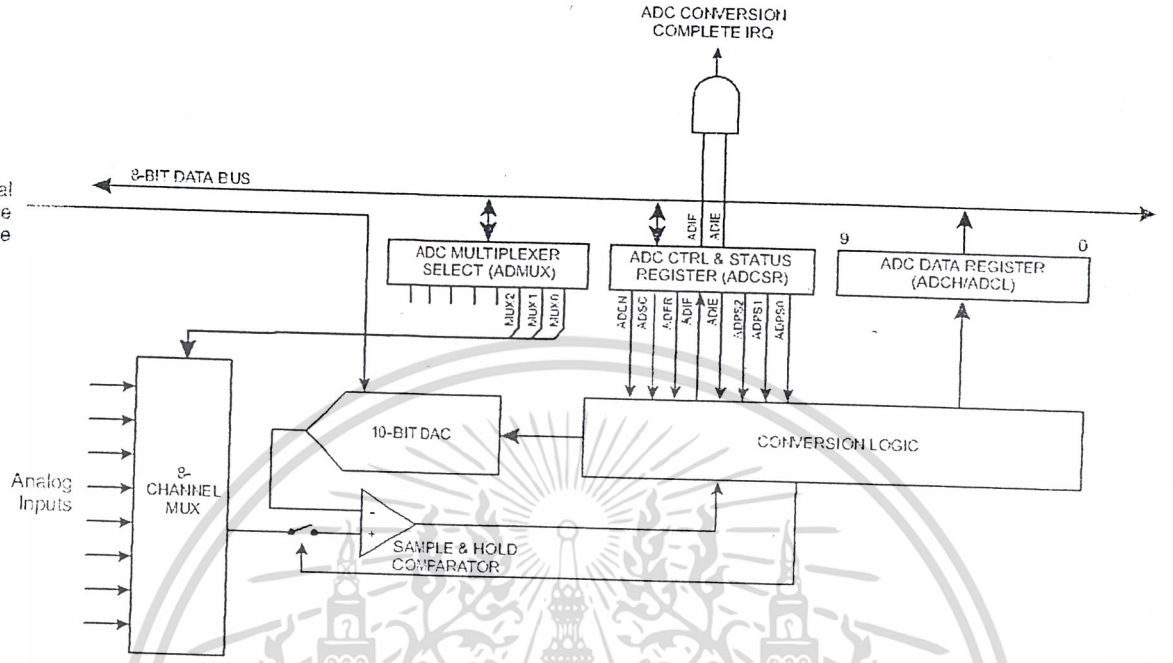
The AT90S4434/8535 features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows each pin of Port A to be used as an input for the ADC. The ADC contains a Sample and Hold Amplifier which ensures that the input voltage to the ADC is held at a constant level during conversion. A block dingram of the ADC is shown in Figure 45.

The ADC has two separate analog supply voltage pins, AV_{CC} and AGND. AGND must be connected to GND, and the voltage on AV_{CC} must not differ more than ±0.3V from V_{CC}. See the paragraph ADC Noise Canceling Techniques on how to connect these pins.

An external reference voltage must be applied to the AREF pin. This voltage must be in the range AGND ... AV_{CC}

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 45. Analog to Digital Converter Block Schematic



Operation

The ADC can operate in two modes - Single Conversion and Free Run Mode. In Single Conversion Mode, each conversion ...ed by the user. In Free Run Mode, the ADC is constantly sampling and updating the ADC Data Register. ...ADCSR selects between the two available modes.

...by writing a logical one to the ADC Enable bit, ADEN in ADCSR. The first conversion that is started ...will be preceded by a dummy conversion to initialize the ADC. To the user, the only difference will ...sion takes 12 more ADC clock pulses than a normal conversion.

...By writing a logical one to the ADC Start Conversion bit, ADSC. This bit stays high as long as the ...ress and will be set to zero by hardware when the conversion is completed. If a different data channel ...nversion is in progress, the ADC will finish the current conversion before performing the channel ...

...10-bit result, two data registers, ADCH and ADCL, must be read to get the result when the conver... ...ata protection logic is used to ensure that the contents of the data registers belong to the same ...ead. This mechanism works as follows:

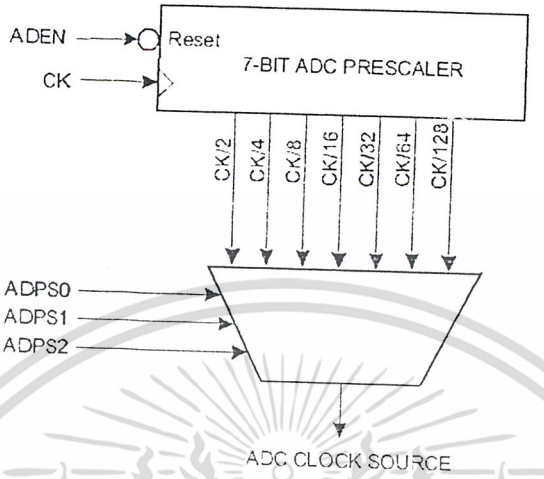
...must be read first. Once ADCL is read, ADC access to data registers is blocked. This means ...and a conversion completes before ADCH is read, none of the registers are updated and the ...lost. When ADCH is read, ADC access to the ADCH and ADCL registers is re-enabled.

...ADIF, which can be triggered when a conversion completes. When ADC access to the data ...reading of ADCH and ADCL, the interrupt will trigger even if the result gets lost

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Prescaling

Figure 46. ADC Prescaler

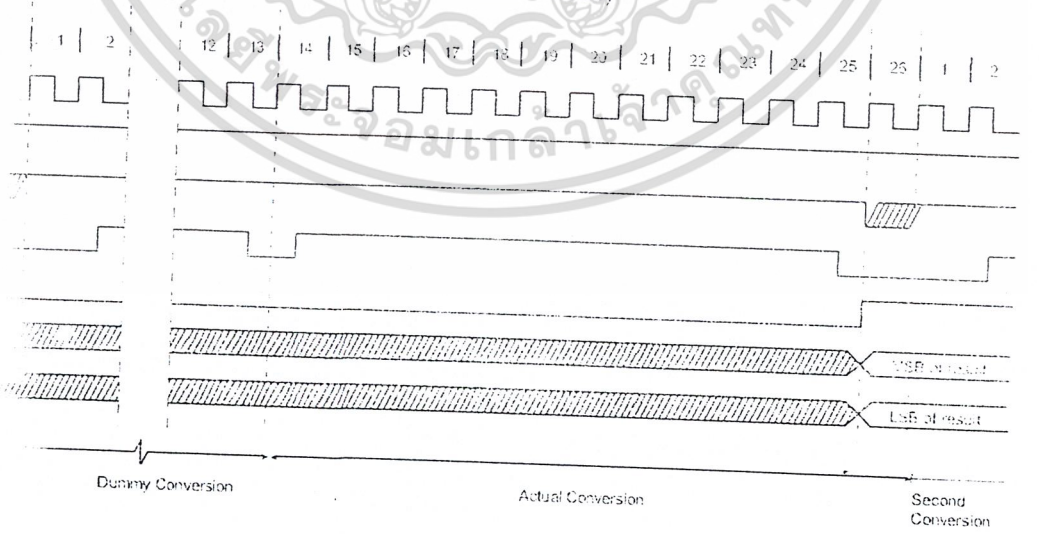


The ADC contains a prescaler, which divides the system clock to an acceptable ADC clock frequency. The ADC accepts input clock frequencies in the range 50 - 200 kHz. Applying a higher input frequency will result in a poorer accuracy. See "ADC Characteristics" on page 66.

The ADPS0 - ADPS2 bits in ADCSR are used to generate a proper ADC clock input frequency from any XTAL frequency above 100 kHz. The prescaler starts counting from the moment the ADC is switched on by setting the ADEN bit in ADCSR. The prescaler keeps running for as long as the ADEN bit is set, and is continuously reset when ADEN is low.

When initiating a conversion by setting the ADSC bit in ADCSR, the conversion starts at the following rising edge of the ADC clock cycle. The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of the conversion. The result is ready and written to the ADC Result Register after 13 cycles. In single conversion mode, the ADC needs one more clock cycle before a new conversion can be started, see Figure 47. If ADSC is set high in this period, the ADC will start the new conversion immediately. In Free Run Mode, a new conversion will be started immediately after the result is written to the ADC Result Register. Using Free Run Mode and an ADC clock frequency of 200 kHz gives the lowest conversion time, 65 µs, or 15.4 kSPS. For a summary of conversion times, see Table 27.

Figure 47. ADC Timing Diagram, first conversion (Single Conversion Mode)



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Table 27. ADC Conversion Time

Condition	Sample Cycle Number	Result Ready (cycle number)	Total Conversion Time (cycles)	Total Conversion Time (µs)
1st Conversion, Free Run	14	25	25	125 - 500
1st Conversion, Single	14	25	26	130 - 520
Free Run Conversion	2	13	13	65 - 260
Single Conversion	2	13	14	70 - 280

Figure 48. ADC Timing Diagram, single conversion

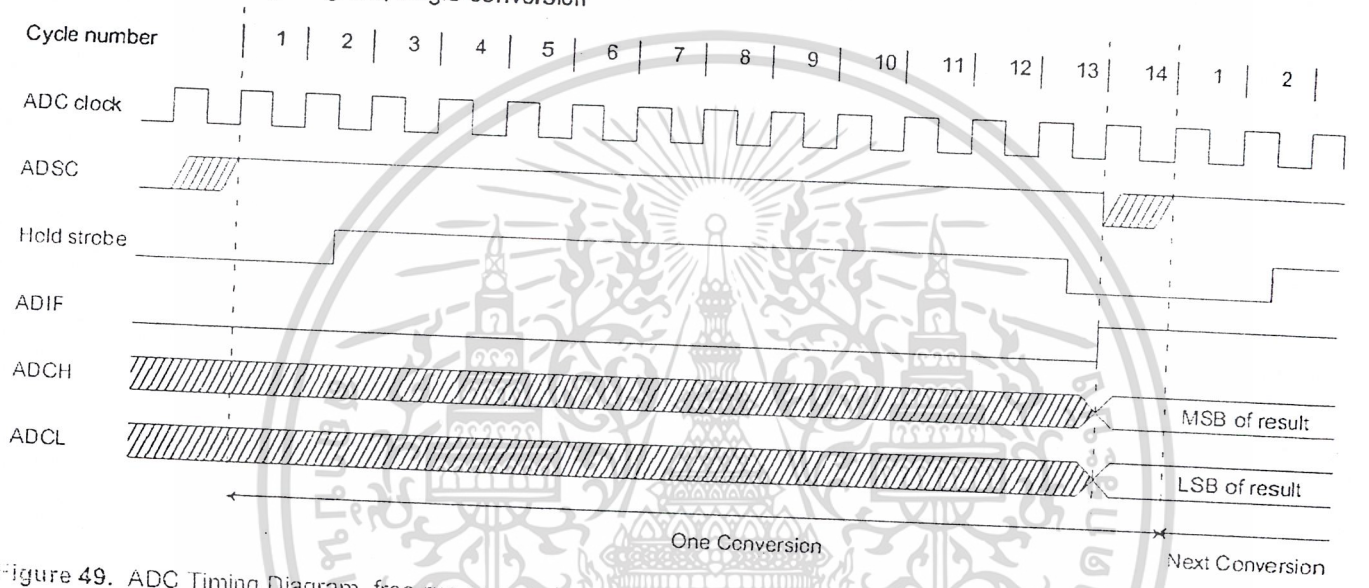
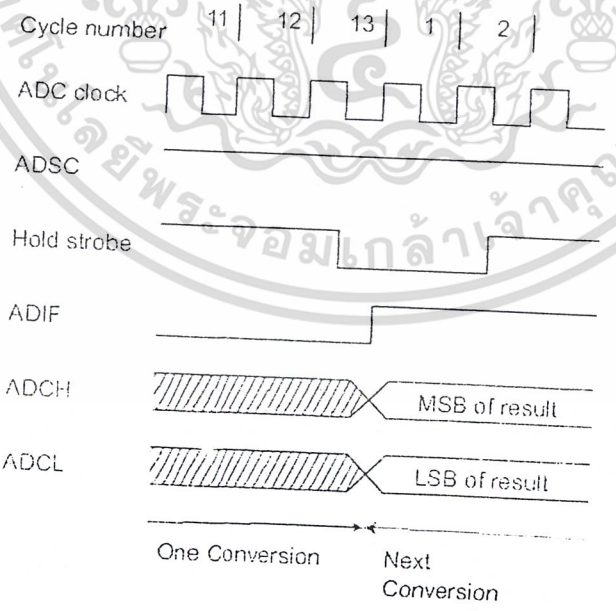


Figure 49. ADC Timing Diagram, free run conversion



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ADC Noise Canceler Function

The ADC features a noise canceler that enables conversion during idle mode to reduce noise induced from the CPU core. To make use of this feature, the following procedure should be used:

1. Make sure that the ADC is enabled and is not busy converting. Single Conversion Mode must be selected and the ADC conversion complete interrupt must be enabled.
   ADEN = 1
   ADSC = 0
   ADFR = 0
   ADIE = 1
2. Enter idle mode. The ADC will start a conversion once the CPU has been halted.
3. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the MCU and execute the ADC conversion complete interrupt routine.

ADC Multiplexer Select Register - ADMUX

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	-	-	-	-	-	MUX2	MUX1	MUX0	ADMUX
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bits 7..3 - Res: Reserved Bits**

These bits are reserved bits in the AT90S4434/8535 and always read as zero.

- **Bits 2..0 - MUX2..MUX0: Analog Channel Select Bits 2-0**

The value of these three bits selects which analog input 7-0 is connected to the ADC.

ADC Control and Status Register - ADCSR

Bit	7	6	5	4	3	2	1	0	
$06 ($26)	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

- **Bit 7 - ADEN: ADC Enable**

Writing a logical '1' to this bit enables the ADC. By clearing this bit to zero, the ADC is turned off. Turning the ADC off while a conversion is in progress, will terminate this conversion.

- **Bit 6 - ADSC: ADC Start Conversion**

In Single Conversion Mode, a logical '1' must be written to this bit to start each conversion. In Free Run Mode, a logical '1' must be written to this bit to start the first conversion. The first time ADSC has been written after the ADC has been enabled, or if ADSC is written at the same time as the ADC is enabled, a dummy conversion will precede the initiated conversion. This dummy conversion performs initialization of the ADC.

ADSC remains high during the conversion. ADSC goes low after the actual conversion is finished, but before the result is written to the ADC Data Registers. This allows a new conversion to be initiated before the current conversion is complete. The new conversion will then start immediately after the current conversion completes. When a dummy conversion precedes a real conversion, ADSC will stay high until the real conversion is finished.

Writing a 0 to this bit has no effect.

- **Bit 5 - ADFR: ADC Free Run Select**

When this bit is set (one) the ADC operates in Free Run Mode. In this mode, the ADC samples and updates the data registers continuously. Clearing this bit (zero) will terminate Free Run Mode.

- **Bit 4 -ADIF: ADC Interrupt Flag**

This bit is set (one) when an ADC conversion completes and the data registers are updated. The ADC Conversion Complete Interrupt is executed if the ADIE bit and the I-bit in SREG are set (one). ADIF is cleared by hardware when executing

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

the corresponding interrupt handling vector. Alternatively, ADIF is cleared by writing a logical one to the flag. Beware that if doing a read-modify-write on ADCSR, a pending interrupt can be disabled. This also applies if the SBI and CBI instructions are used.

- **Bit 3 - ADIE: ADC Interrupt Enable**

When this bit is set (one) and the I-bit in SREG is set (one), the ADC Conversion Complete Interrupt is activated.

- **Bits 2..0 - ADPS2..ADPS0: ADC Prescaler Select Bits**

These bits determine the division factor between the XTAL frequency and the input clock to the ADC.

Table 28. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

ADC Data Register - ADCL AND ADCH

Bit	15	14	13	12	11	10	9	8	
$05 ($25)	-	-	-	-	-	-	ADC9	ADC8	ADCH
$04 ($24)	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

When an ADC conversion is complete, the result is found in these two registers. In Free Run Mode, it is essential that both registers are read, and that ADCL is read before ADCH.

Scanning Multiple Channels

Since change of analog channel always is delayed until a conversion is finished, the Free Run Mode can be used to scan multiple channels without interrupting the converter. Typically, the ADC Conversion Complete interrupt will be used to perform the channel shift. However, the user should take the following fact into consideration:

The interrupt triggers once the result is ready to be read. In Free Run Mode, the next conversion will start immediately when the interrupt triggers. If ADMUX is changed after the interrupt triggers, the next conversion has already started, and the old setting is used.

ADC Noise Canceling Techniques

Digital circuitry inside and outside the AT90S4434/8535 generates EMI which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

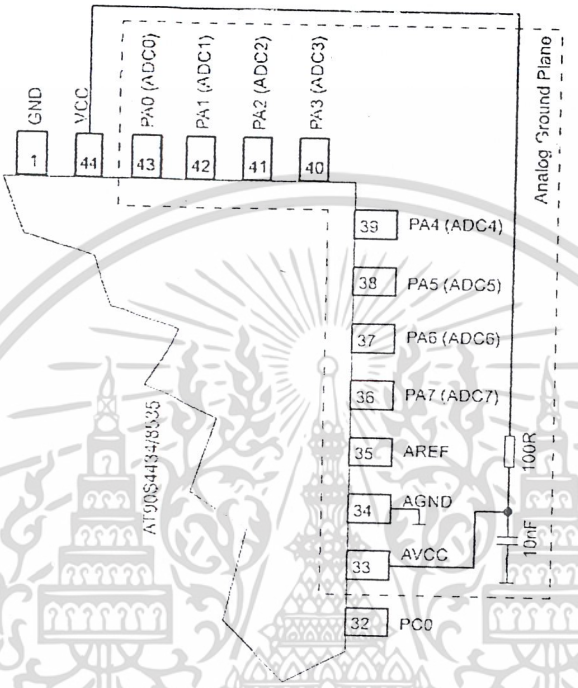
The analog part of the AT90S4434/8535 and all analog components in the application should have a separate analog ground plane on the PCB. This ground plane is connected to the digital ground plane via a single point on the PCB.

Keep analog signal paths as short as possible. Make sure analog tracks run over the analog ground plane, and keep them well away from high-speed switching digital tracks.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. The AV_{CC} pin on the AT90S4434/8535 should be connected to the digital V_{CC} supply voltage via an RC network as shown in Figure 50.
4. Use the ADC noise canceler function to reduce induced noise from the CPU.
5. If some Port A pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress.

Figure 50. ADC Power Connections



Note that since AV_{CC} feeds the Port A output drivers, the RC network shown should not be employed if any Port A serve as outputs.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ADC Characteristics

T_A = -40°C to 85°C

Symbol		Condition	Min	Typ	Max	Units
				10		Bits
	Absolute accuracy	VREF = 4V ADC clock = 200 kHz		1	2	LSB
	Absolute accuracy	VREF = 4V ADC clock = 1 MHz		4		LSB
	Absolute accuracy	VREF = 4V ADC clock = 2 MHz		16		LSB
	Integral Non-Linearity	V_{REF} > 2V		0.5		LSB
	Differential Non-Linearity	V_{REF} > 2V		0.5		LSB
	Zero Error (Offset)			1		LSB
	Conversion Time		65		260	µs
	Clock Frequency		50		200	kHz
AV_{CC}	Analog Supply Voltage		V_{CC} - 0.3[1]		V_{CC} + 0.3[2]	V
	Voltage		AGND		AV_{CC}	V
	Input Resistance		6	10	13	kΩ
	Resistance			100		MΩ

1. If V_{CC} is 2.7V.
2. If V_{CC} is 6.0V.

...ad-Modify-Write functionality when used as general digital I/O ports. This means that the direc-...hanged without unintentionally changing the direction of any other pin with the SBI and CBI ...es for changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if

...I/O port.

...cations are allocated for Port A, one each for the Data Register - PORTA, S1B(S3B), Data ...A(S3A) and the Port A Input Pins - PINA, S19(S39). The Port A Input Pins address is read ...nd the Data Direction Register are read/write.

...lectable pull-up resistors. The Port A output buffers can sink 20mA and thus drive LED dis-... to PA7 are used as inputs and are externally pulled low, they will source current if the ...ivated.

...as analog inputs for the ADC. If some Port A pins are configured as outputs, it is essential ... conversion is in progress. This might corrupt the result of the conversion.

... schmitt trigger of the digital input is disconnected. This allows analog signals that are close to ...erdown without causing excessive power consumption.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

National Semiconductor

August 1999

LM35
Precision Centigrade Temperature Sensors

General Description

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55° to +150°C temperature range, while the LM35C is rated for a −40° to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface mount small outline package and a plastic TO-220 package.

Features

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteeable (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±¼°C typical
- Low impedance output, 0.1 Ω for 1 mA load

Typical Applications

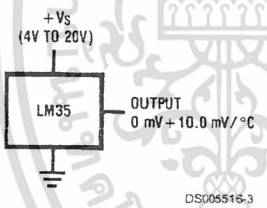
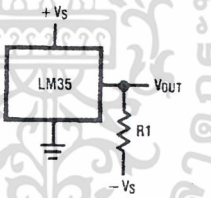


FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)



Choose $R_1 = -V_S/50\ \mu A$

V_{OUT} = +1,500 mV at +150°C
= +250 mV at +25°C
= −550 mV at −55°C

FIGURE 2. Full-Range Centigrade Temperature Sensor

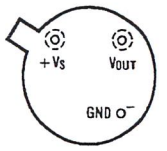
TRI-STATE® is a registered trademark of National Semiconductor Corporation.

© 2000 National Semiconductor Corporation

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Connection Diagrams

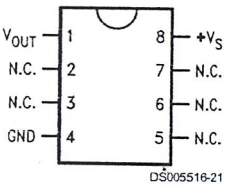
TO-46
Metal Can Package*



*Case is connected to negative pin (GND)

Order Number LM35H, LM35AH, LM35CH, LM35CAH or LM35DH
See NS Package Number H03H

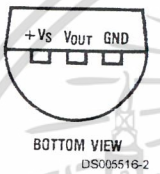
SO-8
Small Outline Molded Package



N.C. = No Connection

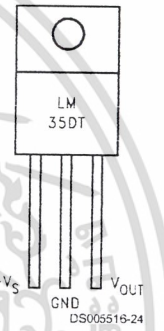
Top View
Order Number LM35DM
See NS Package Number M08A

TO-92
Plastic Package



Order Number LM35CZ, LM35CAZ or LM35DZ
See NS Package Number Z03A

TO-220
Plastic Package*



*Tab is connected to the negative pin (GND).
Note: The LM35DT pinout is different than the discontinued LM35DP.

Order Number LM35DT
See NS Package Number TA03F

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Absolute Maximum Ratings (Note 10)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Supply Voltage	+35V to −0.2V
Output Voltage	+6V to −1.0V
Output Current	10 mA
Storage Temp.;	
TO-46 Package,	−60°C to +180°C
TO-92 Package,	−60°C to +150°C
SO-8 Package,	−65°C to +150°C
TO-220 Package,	−65°C to +150°C
Lead Temp.:	
TO-46 Package, (Soldering, 10 seconds)	300°C
TO-92 and TO-220 Package, (Soldering, 10 seconds)	260°C
SO Package (Note 12)	
Vapor Phase (60 seconds)	215°C
Infrared (15 seconds)	220°C
ESD Susceptibility (Note 11)	2500V
Specified Operating Temperature Range: T_{MIN} to T_{MAX} (Note 2)	
LM35, LM35A	−55°C to +150°C
LM35C, LM35CA	−40°C to +110°C
LM35D	0°C to +100°C

Electrical Characteristics

(Notes 1, 6)

Parameter	Conditions	LM35A			LM35CA			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy (Note 7)	T_A=+25°C	±0.2	±0.5		±0.2	±0.5		°C
	T_A=−10°C	±0.3			±0.3		±1.0	°C
	T_A=T_{MAX}	±0.4	±1.0		±0.4	±1.0		°C
	T_A=T_{MIN}	±0.4	±1.0		±0.4		±1.5	°C
Nonlinearity (Note 8)	T_{MIN}≤T_A≤T_{MAX}	**±0.18**		**±0.35**	**±0.15**		**±0.3**	°C
Sensor Gain (Average Slope)	T_{MIN}≤T_A≤T_{MAX}	**+10.0**	**+9.9, +10.1**		**+10.0**		**+9.9, +10.1**	mV/°C
Load Regulation (Note 3) 0≤I_L≤1 mA	T_A=+25°C	±0.4	±1.0		±0.4	±1.0		mV/mA
	T_{MIN}≤T_A≤T_{MAX}	**±0.5**		**±3.0**	**±0.5**		**±3.0**	mV/mA
Line Regulation (Note 3)	T_A=+25°C	±0.01	±0.05		±0.01	±0.05		mV/V
	4V≤V_S≤30V	**±0.02**		**±0.1**	**±0.02**		**±0.1**	mV/V
Quiescent Current (Note 9)	V_S=+5V, +25°C	56	67		56	67		µA
	V_S=+5V	**105**		**131**	**91**		**114**	µA
	V_S=+30V, +25°C	56.2	68		56.2	68		µA
	V_S=+30V	**105.5**		**133**	**91.5**		**116**	µA
Change of Quiescent Current (Note 3)	4V≤V_S≤30V, +25°C	0.2	1.0		0.2	1.0		µA
	4V≤V_S≤30V	**0.5**		**2.0**	**0.5**		**2.0**	µA
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.5**	**+0.39**		**+0.5**	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, I_L=0	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	T_J=T_{MAX}, for 1000 hours	±0.08			±0.08			°C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Electrical Characteristics

(Notes 1, 6)

Parameter	Conditions	LM35			LM35C, LM35D			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy, LM35, LM35C (Note 7)	T_A=+25°C	±0.4	±1.0		±0.4	±1.0		°C
	T_A=−10°C	±0.5			±0.5		±1.5	°C
	T_A=T_{MAX}	±0.8	±1.5		±0.8		±1.5	°C
	T_A=T_{MIN}	±0.8		±1.5	±0.8		±2.0	°C
Accuracy, LM35D (Note 7)	T_A=+25°C				±0.6	±1.5		°C
	T_A=T_{MAX}				±0.9		±2.0	°C
	T_A=T_{MIN}				±0.9		±2.0	°C
Nonlinearity (Note 8)	T_{MIN}≤T_A≤T_{MAX}	**±0.3**		**±0.5**	**±0.2**		**±0.5**	°C
Sensor Gain (Average Slope)	T_{MIN}≤T_A≤T_{MAX}	**+10.0**	**+9.8, +10.2**		**+10.0**		**+9.8, +10.2**	mV/°C
Load Regulation (Note 3) 0≤I_L≤1 mA	T_A=+25°C	±0.4	±2.0		±0.4	±2.0		mV/mA
	T_{MIN}≤T_A≤T_{MAX}	**±0.5**		**±5.0**	**±0.5**		**±5.0**	mV/mA
Line Regulation (Note 3)	T_A=+25°C	±0.01	±0.1		±0.01	±0.1		mV/V
	4V≤V_S≤30V	**±0.02**		**±0.2**	**±0.02**		**±0.2**	mV/V
Quiescent Current (Note 9)	V_S=+5V, +25°C	56	80		56	80		µA
	V_S=+5V	**105**		**158**	**91**		**138**	µA
	V_S=+30V, +25°C	56.2	82		56.2	82		µA
	V_S=+30V	**105.5**		**161**	**91.5**		**141**	µA
Change of Quiescent Current (Note 3)	4V≤V_S≤30V, +25°C	0.2	2.0		0.2	2.0		µA
	4V≤V_S≤30V	**0.5**		**3.0**	**0.5**		**3.0**	µA
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.7**	**+0.39**		**+0.7**	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, I_L=0	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	T_J=T_{MAX}, for 1000 hours	±0.08			±0.08			°C

Note 1: Unless otherwise noted, these specifications apply: −55°C≤T_J≤+150°C for the LM35 and LM35A; −40°≤T_J≤+110°C for the LM35C and LM35CA; and 0°≤T_J≤+100°C for the LM35D. V_S=+5Vdc and I_{LOAD}=50 µA, in the circuit of *Figure 2*. These specifications also apply from +2°C to T_{MAX} in the circuit of *Figure 1*. Specifications in **boldface** apply over the full rated temperature range.

Note 2: Thermal resistance of the TO-46 package is 400°C/W, junction to ambient, and 24°C/W junction to case. Thermal resistance of the TO-92 package is 180°C/W junction to ambient. Thermal resistance of the small outline molded package is 220°C/W junction to ambient. Thermal resistance of the TO-220 package is 90°C/W junction to ambient. For additional thermal resistance information see table in the Applications section.

Note 3: Regulation is measured at constant junction temperature, using pulse testing with a low duty cycle. Changes in output due to heating effects can be computed by multiplying the internal dissipation by the thermal resistance.

Note 4: Tested Limits are guaranteed and 100% tested in production.

Note 5: Design Limits are guaranteed (but not 100% production tested) over the indicated temperature and supply voltage ranges. These limits are not used to calculate outgoing quality levels.

Note 6: Specifications in **boldface** apply over the full rated temperature range.

Note 7: Accuracy is defined as the error between the output voltage and 10mv/°C times the device's case temperature, at specified conditions of voltage, current, and temperature (expressed in °C).

Note 8: Nonlinearity is defined as the deviation of the output-voltage-versus-temperature curve from the best-fit straight line, over the device's rated temperature range.

Note 9: Quiescent current is defined in the circuit of *Figure 1*.

Note 10: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications do not apply when operating the device beyond its rated operating conditions. See Note 1.

Note 11: Human body model, 100 pF discharged through a 1.5 kΩ resistor.

Note 12: See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" or the section titled "Surface Mount" found in a current National Semiconductor Linear Data Book for other methods of soldering surface mount devices.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

1. สุวัฒน์ ปุณณชัยยะ , ต้น ตัณฑ์สุทธิวงศ์ และสุพจน์ ปุณณชัยยะ, เปิดโลกของ TCP/IP และโปรโตคอลของ อินเตอร์เน็ต, พิมพ์ครั้งที่ 1, ปีที่พิมพ์ 2543, ISBN 974-7821-81-8
2. ข้อมูลทางอินเตอร์เน็ต http://pirun.ku.ac.th/~b40num/micro/CPU/index.html และ http://www.atmel.com/atmel/products/prod200.htm
3. สัจจะ จรัสรุ่งรวีวร, Visual Basic6 Basic & Advanced, พิมพ์ครั้งที่ 1, สำนักพิมพ์อินโฟเพรส, ปีที่พิมพ์ 2542
4. Peter W. Gofton, จิรศักดิ์ เหลืองอุไร, คัมภีร์การใช้งานการสื่อสารอนุกรมบน PC Mastering Serial Communications, สำนักพิมพ์ซีเอ็ดยูเคชั่น, ปีที่พิมพ์ 2538
5. ผ.ศ.สมเกียรติ ศุภเดช, ทฤษฎีและการออกแบบวงจรพัลซ์, หน้า 67-73, อิเล็กทรอนิกเวิร์ด, 2526
6. สว่าง ประกายรุ้ง, รวมบทความ ทฤษฎี และการประยุกต์ใช้งานอุปกรณ์สารกึ่งตัวนำ, หน้า 113-124, สำนักพิมพ์ซีเอ็ดยูเคชั่น, ปีที่พิมพ์ 2538

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้