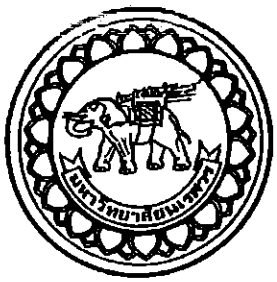


พัดลมกึ่งอัตโนมัติ

SPEED FAN BY TEMPERATURE CONTROL

นาย พรนิมิตร แก้วเพียร รหัส 48380240

นาย สุธีย์ หม่องมูล รหัส 48380249

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2551

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน การควบคุมพัดลมกึ่งอัตโนมัติ

ผู้ดำเนินโครงงาน นายพรนิมิตร แก้วเพียร รหัส 48380240

นายสุธีย์ หม่องมูล รหัส 48380249

ที่ปรึกษาโครงงาน ดร. อัครพันธ์ วงศ์กังแห

สาขาวิชา วิศวกรรมไฟฟ้า

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2551

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

.....ที่ปรึกษาโครงงาน

(ดร. อัครพันธ์ วงศ์กังแห)

.....กรรมการ

(ดร. ชัยรัตน์ พินทอง)

.....กรรมการ

(ดร. ศุภวรรณ พลพิทักษ์ชัย)

ชื่อหัวข้อโครงงาน	การควบคุมพัดลมกึ่งอัตโนมัติ		
ผู้ดำเนินโครงงาน	นาย พรนิมิตร	แก้วเพียร	รหัส 48380240
	นาย สุธีย์	หม่องมูล	รหัส 48380249
ที่ปรึกษาโครงงาน	ดร. อัครพันธ์ วงศ์กังแห		
สาขาวิชา	วิศวกรรมไฟฟ้า		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2551		

.....

บทคัดย่อ

งานโครงการนี้เป็นการศึกษาและออกแบบชุดควบคุมการทำงานของพัดลมไฟฟ้า โดยใช้อุณหภูมิเป็นตัวควบคุม ผู้ใช้สามารถเลือกการควบคุมได้สองโหมด คือ โหมดใช้มือ (manual mode) และอัตโนมัติ (auto mode) ในโหมดใช้มือ (manual mode) ผู้ใช้สามารถเลือกกด ปิด-เปิด และเลือกความเร็วของพัดลมได้สามระดับและในโหมดอัตโนมัติ (auto mode) พัดลมจะปรับระดับความเร็วโดยอัตโนมัติสามระดับ ขึ้นอยู่กับอุณหภูมิห้องจากการทดลองวงจรควบคุมพัดลมนั้น พัดลมสามารถทำงานตามอุณหภูมิห้องในขณะนั้น โดยที่อุณหภูมิตั้งแต่ติดลบพัดลมจะไม่ทำงาน ที่อุณหภูมิมากกว่าอุณหภูมิที่กำหนดครั้งที่ 1พัดลมจะทำงานที่ระดับความเร็ว1 ที่อุณหภูมิมากกว่าอุณหภูมิที่กำหนดครั้งที่ 2 พัดลมทำงานที่ระดับความเร็ว 2 ที่อุณหภูมิมากกว่า อุณหภูมิที่กำหนดครั้งที่ 3 พัดลมจะทำงานที่ระดับความ เร็ว 3 ซึ่งช่วงค่าเวลาการกำหนดในการควบคุมอุณหภูมิสามารถเปลี่ยนแปลงได้ โดยการตั้งค่าที่โปรแกรมผ่านไฮเปอร์เทอมินอล

Project title	Speed Fan by Temperature Control
Name	Mr. Phornnimit Keawpean ID. 48380240
	Mr. Sutee Mongmoon ID. 48380249
Project advisor	Dr. Akaraphunt Vongkunghae
Major	Electrical Engineering
Department	Electrical and Computer Engineering
Academic year	2008

.....

Abstract

This project is to study and design a speed controller for a small electrical fan. The controller allows the fan speed can be adjusted by it's ambient temperature. When the temperature is high, the speed is fast. The speed will be slow when the speed temperature goes lower. To setting up the temperature for each step of the fan speed, the hyper terminal program is used for interfacing to us. Also we can monitor the temperature on the hyper terminal program.

กิตติกรรมประกาศ

ในนามผู้จัดทำโครงงาน เรื่องการควบคุมพัดลมกึ่งอัตโนมัติขอแสดงความขอบคุณอย่างสูงต่ออาจารย์ที่ปรึกษาอาจารย์ ดร. อัครพันธ์ วงศ์กังแห โครงการที่ให้คำปรึกษาชี้แนะแนวทางการทำงาน ตลอดจนให้การอบรมสั่งสอนทั้งทางด้านวิชาการ คุณธรรม และจริยธรรม ทำให้การดำเนินโครงการสำเร็จลุล่วงไปด้วยดี และใคร่ขอขอบคุณอย่างสูงต่ออาจารย์ผู้ร่วมประเมินภาควิชาวิศวกรรมไฟฟ้า ที่ให้ความอนุเคราะห์ในการจัดสรรอุปกรณ์และเครื่องมือในการทำโครงการท้ายที่สุดนี้กราบขอบพระคุณทุกท่านที่มีส่วนเกี่ยวข้อง ให้ความช่วยเหลือและให้กำลังใจจนโครงการสำเร็จลุล่วง

นายพรนิมิตร แก้วเพียร

นายสุธีย์ หม่องมูล

สารบัญ

หน้า

สารบัญรูป

รูปที่ หน้า

บทที่ 1
บทนำ

พัดลมเป็นสิ่งที่ช่วย ในการระบายความร้อนให้กับผู้ใช้ โดยส่วนใหญ่พัดลมนั้นนิยมใช้กันอย่างแพร่หลายกันทุกฐานะ เนื่องจากหาซื้อได้ง่ายและราคาไม่ถือว่าแพงมากนัก ซึ่งในโครงงานนี้จะช่วยให้การใช้พัดลมนั้นสะดวก มากขึ้น พัดลมนี้จะต่างจากพัดลมที่ใช้กันทั่วไป อาจทำให้ผู้ใช้พึงพอใจมากขึ้นในการใช้งานพัดลม ไม่ว่าจะเป็นการประหยัดพลังงาน การบรรเทาความร้อน การปรับเปลี่ยนความเร็วของพัดลมเองจึงสะดวกแก่การใช้งานมากขึ้น

1.1 ความเป็นมาและความสำคัญของปัญหา

พัดลมซึ่งเป็นอุปกรณ์เครื่องใช้ไฟฟ้าที่ใช้กันอย่างแพร่หลายหากเราจะทำการปิด-เปิดหรือปรับเปลี่ยนความเร็วของพัดลมบ่อยนั้นคงจะไม่สะดวกมากนักในขณะที่กำลังนอนหลับอยู่

โครงงานนี้จึงทำการศึกษาขึ้นมาเป็นโครงงานพัดลมกึ่งอัตโนมัติโดยอาศัยหลักการเปลี่ยนแปลงของอุณหภูมิโดยใช้ตัวรับรู้โดยจะเปลี่ยนแปลงตามค่าอุณหภูมิที่เราตั้งไว้ที่เปลี่ยนไปเพื่อนำไปควบคุมการเปลี่ยนความเร็วของมอเตอร์พัดลมได้

1.2 วัตถุประสงค์

1.2.1 ศึกษาและทำความเข้าใจการทำงานของมอเตอร์พัดลม

1.2.2 ศึกษาและทำความเข้าใจเกี่ยวกับบอร์ดไมโครคอนโทรลเลอร์และการเขียนโปรแกรมภาษาซี

1.2.3 ศึกษาและทำความเข้าใจเกี่ยวกับตัวรับรู้อุณหภูมิและการเปลี่ยนความเร็วของมอเตอร์พัดลมแต่ละเกียร์

1.3 ขอบเขตการดำเนินโครงงาน

1.3.1 ศึกษาและทำความเข้าใจเกี่ยวกับการทำงานและการเปลี่ยนความเร็วของมอเตอร์พัดลม

1.3.2 ศึกษาและทำความเข้าใจเกี่ยวกับการทำงานและการเปลี่ยนแปลงของความเร็วของมอเตอร์พัดลมแต่เกียร์ โดยอาศัยชุดควบคุมจากตัวรับรู้ค่าของอุณหภูมิ

1.3.3 ศึกษาและทำความเข้าใจเกี่ยวกับการทำการประกอบชุดทดลองและทดสอบการใช้งาน

1.4 ขั้นตอนการดำเนินโครงงาน

1.4.1 ทำความเข้าใจเกี่ยวกับการทำงานและการเปลี่ยนแปลงความเร็วของมอเตอร์

1.4.2 ศึกษาความเข้าใจเกี่ยวกับการเขียนโปรแกรมของบอร์ดไมโครคอนโทรลเลอร์เพื่อควบคุมการเปลี่ยนความเร็วของมอเตอร์พัดลมตามอุณหภูมิที่กำหนด

1.4.2 ศึกษาออกแบบวงจรตัวรับรู้อุณหภูมิและทำการออกแบบวงจร

1.4.3 ดำเนินการจัดหาอุปกรณ์ที่จำเป็นต้องใช้

1.4.4 เริ่มดำเนินประกอบส่วนต่างๆให้เป็นชิ้นงานทดลองแล้วทำการทดลอง

1.4.5 ทำรายงานสรุปผลการทำโครงการทั้งหมด

1.5 ระยะเวลาดำเนินงาน

การดำเนินงาน	เดือน ปี					
	ธ.ค. 2552	ม.ค. 2553	ก.พ. 2553	มี.ค. 2553	เม.ย. 2553	พ.ค. 2553
1.5.1 ศึกษาการทำงานของมอเตอร์พัดลม	←	→				
1.5.2 ศึกษาการออกแบบวงจรควบคุมโดยใช้ตัวรับรู้	←	—	→			
1.5.3 จัดหาอุปกรณ์และส่วนประกอบของส่วนต่าง ๆ		←	—	→		
1.5.4 ประกอบวงจรชุดควบคุมของมอเตอร์พัดลม			←	—	→	
1.5.5 ทดสอบการทำงานของมอเตอร์พัดลม					←	→
1.5.6 จัดทำรายงานและสรุปผล					←	→

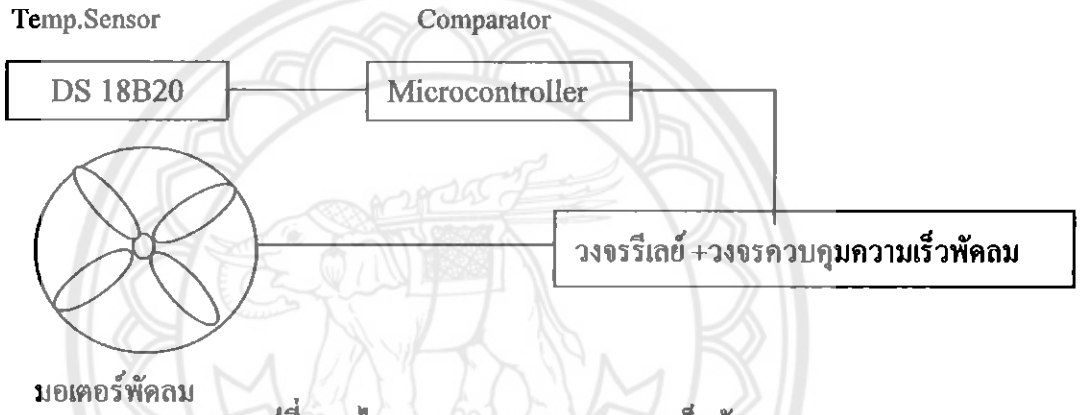
1.6 งบประมาณของโครงการ

ชุดควบคุมมอเตอร์พัดลม	2500 บาท
ค่าจัดทำรายงานและทำรูปเล่ม	500 บาท
ถัวเฉลี่ยทุกรายการ	500 บาท
รวมเงินทั้งสิ้น	3500 บาท

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 ขั้นตอนการออกแบบวงจรควบคุมความเร็วของพัดลม

ในการออกแบบวงจรการควบคุมความเร็วของมอเตอร์ผู้จัดทำโครงงานได้ทำการออกแบบวงจรมาโดยใช้ไอซี DS 18B20 และวงจรเปรียบเทียบแรงดันเพื่อควบคุมความเร็วของพัดลม ดังแสดงตามไดอะแกรม



รูปที่ 2.1 ไดอะแกรมการควบคุมความเร็วพัดลม

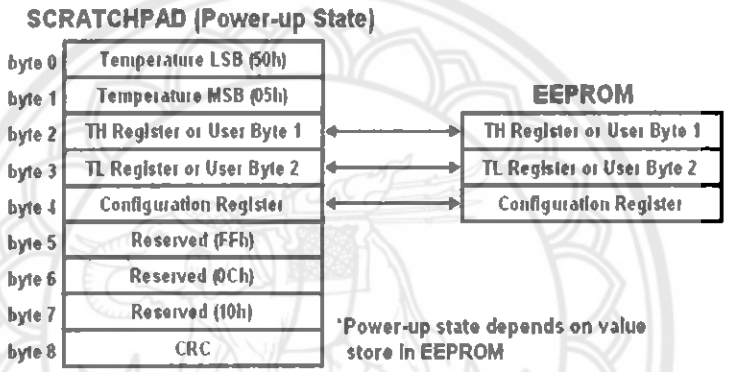
2.2 ระบบตรวจวัดอุณหภูมิด้วย ดิจิตอลเทอร์โมมิเตอร์ DS18B20

DS18B20 เป็น IC วัดอุณหภูมิแบบดิจิตอลของ Dallas Semiconductor สามารถวัดอุณหภูมิเป็นหน่วยองศา °C ในช่วง -55°C ถึง 125°C ที่ความละเอียด 9 - 12 บิต และมีความแม่นยำอยู่ที่ 0.5°C ในช่วง -10°C ถึง 85°C ในกรณีที่เป็นตัวถังแบบ TO - 92 นั้นจะมีโครงสร้าง ดังแสดงในรูปที่ 2.2

PIN	SYMBOL	Description
1	GND	Ground
2	DQ	Data Input/ Output pin
3	Vdd	Optional Vdd pin

รูปที่ 2.2 โครงสร้างและขาของ DS 18B20

การสื่อสารและควบคุม DS18B20 นั้นสามารถทำได้โดยใช้บัสข้อมูลแบบ 1-wire ของ Dallas Semiconductor ซึ่งใช้สายสัญญาณเพียงแค่เส้นเดียวเท่านั้น ภายใน DS18B20 แต่ละตัวมีโค๊ดประจำตัวขนาด64 บิตทำให้สามารถใช้งาน DS18B20 หลายตัวทำงานบนบัสแบบ1- wire พร้อมกันได้ นอกจากนี้ DS18B20 ยังสามารถทำงานในโหมดพาราสิต (Parasite Power Mode) ซึ่งเป็นการทำงานโดยไม่ใช้ไฟเลี้ยง แต่ใช้พลังงานจากสายสัญญาณ 1-wire ซึ่งมีประโยชน์มากสำหรับการวัดอุณหภูมิระยะไกลหรือในการใช้งานในที่ๆมีเนื้อที่จำกัดในพื้นที่ โครงสร้างรีจิสเตอร์ภายในของ DS18B20 มีลักษณะดังแสดงในรูปที่ 2.3 จะเห็นได้ว่าประกอบไปด้วย SRAM Scratchpad ขนาด 9 ไบต์ และ EEPROM ขนาด 3 ไบต์ ซึ่งใช้เก็บค่าอุณหภูมิสูงสุด (TH) ต่ำสุด (TL) สำหรับเปรียบเทียบการเกิดสัญญาณเตือน และรีจิสเตอร์ควบคุม (Configuration Register)



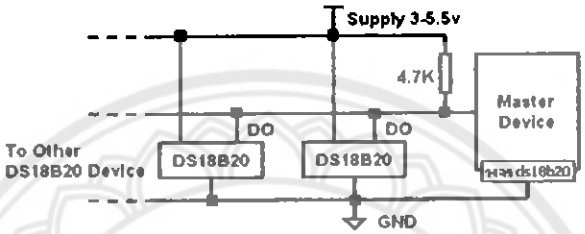
รูปที่ 2.3 โครงสร้างรีจิสเตอร์ภายในของ DS18B20

ข้อมูลอุณหภูมิที่วัดได้จะถูกเก็บอยู่ในรีจิสเตอร์ Temperature ซึ่งมีขนาด 16 บิต ดังแสดงในรูปที่ 2.3 ถ้าข้อมูลอุณหภูมิเป็นบวก S จะเป็น "1" แต่ถ้าข้อมูลอุณหภูมิเป็นลบ S จะเป็น "0" ในกรณีที่ DS18B20 ทำงานในโหมดความละเอียด 12 บิต บิตทุกบิตในรีจิสเตอร์ Temperature จะถูกใช้หมด แต่ในกรณีที่ทำงานในโหมด 9-11 บิต บิตล่าง (บิต 0 – บิต 2) จะไม่ถูกใช้งาน ซึ่งในการกำหนดโหมดความละเอียดการทำงานของ DS18B20 นั้นสามารถกำหนดได้ที่รีจิสเตอร์ Configuration ซึ่งโดยปกติเริ่มต้น DS18B20 จะทำงานในโหมด 12 บิต

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
LS Byte	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
MS Byte	S	S	S	S	S	2^6	2^5	2^4

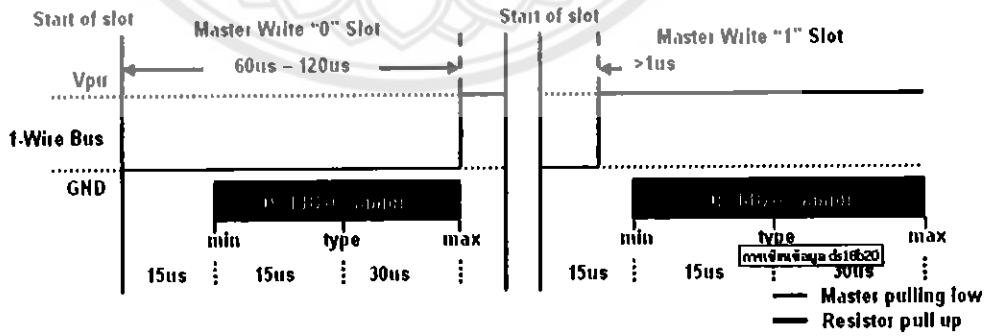
รูปที่ 2.4 โครงสร้างภายในรีจิสเตอร์ Temperature LSB และ MSB

การสื่อสารแบบ 1-wire เป็นระบบบัสข้อมูลแบบ Half-duplex นั้นคือสามารถสื่อสารได้ 2 ทิศทาง แต่ไม่สามารถรับ และส่งข้อมูลพร้อมกันในช่วงเวลาเดียวกันได้ ระบบบัสมีการทำงานเป็นแบบ Master/Slave โดยอุปกรณ์ Master จะเป็นตัวควบคุมสถานะ และจังหวะการรับส่งของบัสข้อมูล ในขณะที่อุปกรณ์ Slave จะทำงานตามการควบคุมของอุปกรณ์ Master เท่านั้น ในการใช้งานบัสแบบ 1 wire นี้ สายสัญญาณข้อมูล DQ จะต้องมีสภาวะปกติที่ลอจิกสูง สามารถทำได้โดยการต่อตัวต้านทานประมาณ 5 กิโลโอห์มพูลอัฟไว้กับไฟเลี้ยง หรือในกรณีที่ใช้บัสแบบ 1 wire ต่อร่วมกับอุปกรณ์ DS18B20 หลายตัว ก็สามารถทำได้ดังแสดงในรูปที่ 2.4



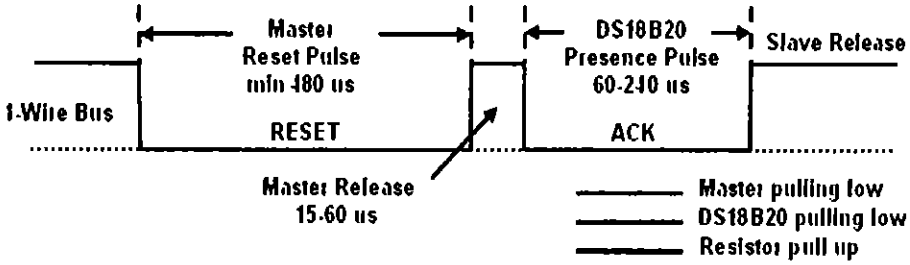
รูปที่ 2.5 การต่อใช้งาน DS 18B20

รูปแบบของสัญญาณบนบัส 1-wire สามารถแบ่งออกได้เป็น 6 รูปแบบ คือ Reset pulse, Presence pulse, write 0, write 1, read 0, read 1ในกระบวนการเริ่มต้นการสื่อสารแบบ 1-wire ทั้งหมดนั้น อุปกรณ์ Master ต้องขอเริ่มการสื่อสารด้วยการสร้าง Reset pulse ก่อน เมื่ออุปกรณ์ Slave ได้รับ Reset pulse ก็จะสร้าง Presence pulse เพื่อตอบรับการขอเริ่มการสื่อสารนั้น ซึ่งมีรายละเอียดของช่วงเวลาต่าง ๆ ดังแสดงในรูปที่ 2.5



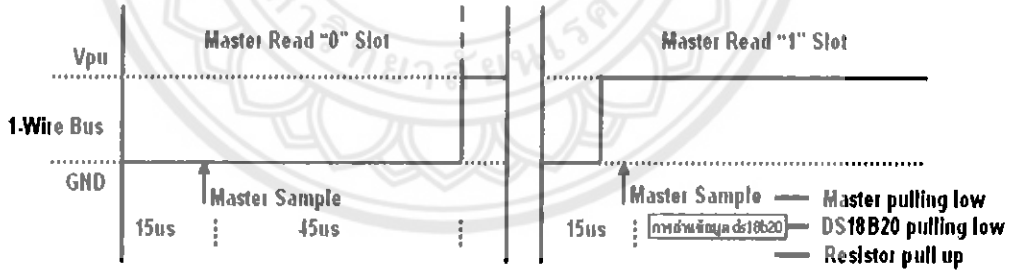
รูปที่ 2.6 การเริ่มการติดต่อสื่อสารแบบ 1-wire ด้วย Reset Pulse และ Presence Pulse

ในการเขียนข้อมูลแบ่งออกเป็น 2 ชนิดคือการเขียนข้อมูล "1" และการเขียนข้อมูล "0" ดังแสดงในรูปที่ 2.6 การเขียนข้อมูลลง DS18B20 ต้องใช้ช่วงเวลาของไทม์สล๊อตอย่างต่ำ 60 µsec และต้องมีช่วงเวลาระหว่างไทม์สล๊อตอย่างต่ำ 1 µsec



รูปที่ 2.7 การเขียนข้อมูลลง DS18B20

การเขียนข้อมูลทั้ง 2 ชนิด เริ่มแรกอุปกรณ์ Master ต้องดึงสัญญาณบนบัส 1-wire ลงมาให้อยู่ในสถานะลอจิกต่ำก่อน ในกรณีที่ต้องการเขียนข้อมูล "0" ลงใน DS18B20 อุปกรณ์ Master ต้องดึงสัญญาณบนบัสให้เป็นลอจิกต่ำต่อ จนกว่าจะครบช่วงเวลาไทม์สล๊อต (อย่างต่ำ 60 µsec) ส่วนในกรณีที่ต้องการเขียนข้อมูล "1" ลง DS18B20 อุปกรณ์ Master ต้องปล่อยบัส เพื่อให้บัสกลับไปอยู่ในสถานะลอจิกสูงก่อนการ Sampling ของ DS18B20 ซึ่งจะอยู่ในช่วง 15 usec-60 µsecหลังจากที่อุปกรณ์Masterดึงสัญญาณบัส1-wireลงมาในการอ่านค่าภายใน SRAM ของ DS18B20 สามารถทำได้ก็ต่อเมื่ออุปกรณ์ Master ได้เขียนข้อมูลเพื่อขอทำการอ่านค่าใน SRAM (Read Scratchpad) ซึ่งมีค่าเป็น 0xBE ลงไปที่ DS18B20 เสียก่อน จากนั้นจึงเริ่มอ่านข้อมูลจากบัส 1-wire โดยไทม์สล๊อตของการอ่านต้องมีช่วงเวลาอย่างต่ำ 60 sec และต้องมีช่วงเวลาระหว่างไทม์สล๊อตอย่างต่ำ 1 µsec ดังแสดงในรูปที่ 2.7



รูปที่ 2.8 การอ่านข้อมูลจาก DS18B20

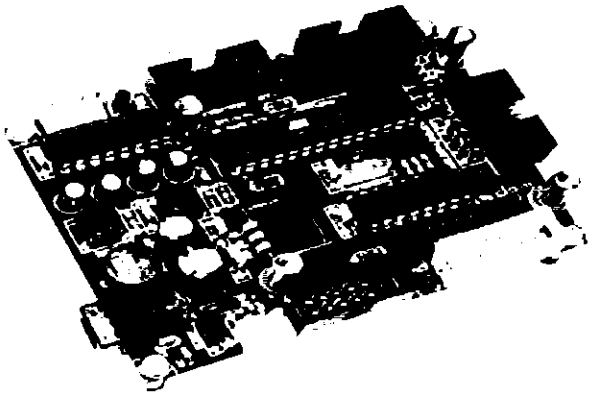
การอ่านข้อมูลจากบัส 1-wire เริ่มแรกอุปกรณ์ Master จะต้องดึงบัส 1-wire ลงให้อยู่ในสถานะลอจิกต่ำเป็นช่วงเวลาอย่างน้อย 1 µsec จากนั้นจึงค่อยปล่อยบัส ในกรณีที่ DS18B20 ส่งข้อมูล "0" DS18B20 จะดึงบัสให้เป็นลอจิกต่ำจนจนกว่าจะสิ้นสุดไทม์สล๊อตถึงจึงจะปล่อยบัสให้กลับไปอยู่ในสถานะลอจิกสูง ส่วนในกรณีที่ DS18B20 ส่งข้อมูล "1" DS18B20 จะปล่อยบัสให้อยู่ในสภานะลอจิกสูงตลอด ในการ Sample เพื่อรับข้อมูลจาก DS18B20 ควรทำภายใน 15 µsec หลังจากจุดเริ่มของไทม์สล๊อตดังแสดงในรูปที่ 2.7 ขั้นตอนการเข้าใช้งาน DS18B20 มี 3 ขั้นตอนด้วยกันคือ 1. Initialization 2. ROM Command 3. DS18B20 Function Commandการ Initialization ประกอบไป

ด้วยการส่ง Reset Pulse จากอุปกรณ์ Master ตามด้วย Presence Pulse ซึ่งตอบรับโดย DS18B20 เพื่อบ่งบอกว่าอุปกรณ์พร้อมทำงาน หลังจากการทำ Initialization เสร็จเรียบร้อยแล้วอุปกรณ์ Master ต้องส่งROM Commandไปยัง DS18B20 ROM Command นั้นแบ่งออกได้เป็น 5 คำสั่งด้วยกันคือ SEARCH ROM [F0h], READ ROM [33h], MATCH ROM [55h], SKIP ROM [CCh], ALARM SEARCH [ECh] ซึ่งในกรณีที่ต่อใช้งาน DS18B20 เพียงตัวเดียวนั้นจะใช้ ROM Command ได้แค่ 2 คำสั่งนั้นคือ READ ROM ซึ่งเป็นการอ่านค่า ROM Code ขนาด 64 บิต บนตัว DS18B20 อีกคำสั่งคือ SKIP ROM ซึ่งเป็นคำสั่งที่ใช้ในกรณีที่อุปกรณ์ Master ต้องการส่งคำสั่งควบคุม DS18B20 ทุกตัว ซึ่งไม่จำเป็นต้องระบุ ROM Code หลังจากที่อุปกรณ์ Master ส่ง Rom Command ไปยัง DS18B20 แล้วอุปกรณ์ Master จะสามารถใช้ Function Command เพื่อเข้าไปควบคุมการทำงานของDS18B20 ได้ประกอบไปด้วย CONVERT T[44h], WRITE SCRATCHPAD[4Eh], READ SCRATCHPAD[BEh], COPY SCRATCHPAD [48h], RECALL E2[B8h], READ POWER SUPPLY [B4h]

2.3 ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) เป็นชื่อของอุปกรณ์อิเล็กทรอนิกส์แบบหนึ่งที่บรรจุเอา ความสามารถมากมายไม่ว่าจะเป็นหน่วยประมวลผลหน่วยคำนวณทางคณิตศาสตร์และลอจิกวงจรรับสัญญาณอินพุตวงจรขับสัญญาณออกทางเอาท์พุตหน่วยความจำ วงจรกำเนิดสัญญาณนาฬิกาทำให้ไมโครคอนโทรลเลอร์สามารถนำไปประยุกต์ใช้งานแทนวงจร์ที่ซับซ้อนได้เป็นอย่างดี โดยช่วยลดจำนวนอุปกรณ์และขนาดของระบบลงในขณะที่ขีดความสามารถสูงขึ้น ภายใต้งบประมาณที่เหมาะสม

ในโครงการจะใช้ไมโครคอนโทรลเลอร์ของ ET-BASE AVR EASY 168 เป็นบอร์ดไมโครคอนโทรลเลอร์ ของ ATMEL ในตระกูล AVR เบอร์ ATMEGA168 โดยมีจุดเด่นเป็นไมโครคอนโทรลเลอร์ขนาดเล็กแต่เพียบพร้อมไปด้วยทรัพยากรพื้นฐานต่างๆ ครบ เช่น RUN ความถี่สูงสุด 20 MHz ที่ 1 CLOCK/MACHINE CYCLE, EEPROM 512 BYTE, SRAM อีก 1 KBYTE, SPI,UART, I2C,WATCHDOG, ATO D ฯลฯ นอกจากนี้บอร์ดยังสามารถโหลดโปรแกรมเข้าตัวไมโครคอนโทรลเลอร์ได้ทางพอร์ท RS232 ได้โดยตรง ไม่ต้องจัดซื้อบอร์ด DOWNLOAD อื่นๆ เพิ่มเติม ด้วยการออกแบบโครงสร้างของบอร์ดให้สามารถต่อเข้ากับบอร์ด I/O ต่างๆ ของทาง อีทีที ได้โดยตรง เช่น ชุด ET-MINI I/O ต่างๆ เช่นET-MINI - ENC28J60 ต่อเป็นระบบ LAN ได้เป็นต้น



รูปที่ 2.9 รูปแสดงของบอร์ด ET – BASE AVR EASY 168

2.3.1. รูปแบบ ในการพัฒนา ET-BASE AVR EASY 168

1. รูปแบบโปรแกรมการพัฒนาด้วย ภาษา ซี (C++) ของ Arduino Projectในแบบ OPEN SOURCE โดยตัว MCU ของทาง อีทีที นี้ ได้ติดตั้งโปรแกรม BOOTLOADER ไว้ในตัว MCU เรียบร้อยแล้ว สามารถ DOWNLOAD ได้โดยตรงผ่านทางRS232 PORT (ในกรณีต้องการผ่านทางพอร์ท USB ก็สามารถเพิ่มเติมการใช้งานได้ด้วยชุด ET-USB/RS232 MINI)

2. รูปแบบโปรแกรมการพัฒนาด้วย AVR ปกติ ซึ่งสามารถเลือกใช้งานในรูปแบบโปรแกรมภาษาใดๆ ที่ทำงานรองรับ AVR เช่น ภาษาเบสิก, ภาษา ซี หรือด้วยWIN AVR ฯลฯ โดยใช้การ DOWNLOAD ผ่านทาง BOOTLOADER ทาง PORT RS232 หรือผ่านทางขั้วต่อ AVR ISP แบบ IDE 10 PIN ที่มีอยู่แล้วบนบอร์ด ใช้ร่วมกับบอร์ดET-AVR PROG MINI, ET-AVR ISP USB V1 ฯลฯ

คุณสมบัติของบอร์ด ET-BASE AVR EASY88/168

1. เลือกใช้ MCU ตระกูล AVR เบอร์ ATMEGA88 ในบอร์ด รุ่น ET-BASE AVR EASY88 และ เบอร์ ATMEGA168 ในรุ่น ET-BASE AVR EASY168
2. ATMEGA88 มีหน่วยความจำ FLASH 8 KBYTE และATMEGA168 มีหน่วยความจำ FLASH 16 KBYTE
3. SRAM 1 KBYTE, EEPROM 512 BYTE, RUN ความถี่ 19.6608 MHz
4. มี PORT I/O ขนาด 20 BIT จำนวน 3 PORT (PB 6 BIT), (PC 6 BIT), (PD 8 BIT) โดยเป็น RS232, SPI, I2C, TIMER/COUNTER, A TO D 10 BIT 6 ช่อง
5. ขั้วต่อใช้งาน 10 PIN ET 3 ชุด, และขั้วต่อ OUTPUT ด้วย 74HC595 แบบ 10PIN IDEET อีก 1 ชุด
6. SW RESET และ SW BL (PD2) สำหรับใช้รีเซ็ตบอร์ดเข้าในการทำงานแบบ BOOTLOADER ผ่านทาง PORT RS232

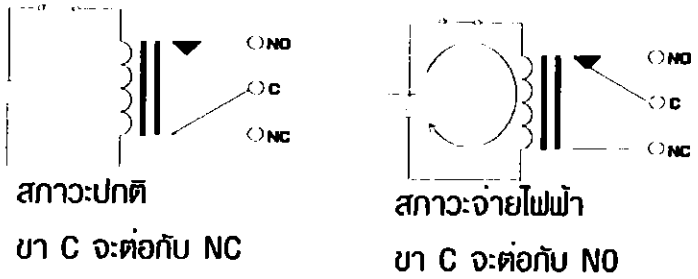
7. RS232 PORT แบบ 4 PIN ET ใช้งานและใช้ DOWNLOAD โปรแกรม
8. 10 PIN IDE มาตรฐาน AVR ISP สำหรับโปรแกรมแบบไม่ผ่าน PORT RS232
9. มีฐานยึดบนบอร์ดใช้ติดตั้งบอร์ดในการทดลองในตระกูล ET-MINI I/O ต่างๆ ได้โดยตรง สะดวกในการทดลอง
10. POWER SUPPLY 7 - 10 VDC, ใช้ LM2940 (LOW DROP) ON BOARD
11. ขนาด PCB 8 X 6 CM

2.4 รีเลย์ (Relay)

รีเลย์อิเล็กทรอนิกส์ Relay รีเลย์ เป็นอุปกรณ์อิเล็กทรอนิกส์เชิงกล ชนิดหนึ่งซึ่งทำหน้าเป็นสวิตช์ แต่รีเลย์นั้นจะถูกควบคุมด้วย กระแสไฟฟ้ารูป 2.9 รีเลย์ และ สัญลักษณ์ของรีเลย์

รูปที่ 2.10 รีเลย์ และ สัญลักษณ์ของรีเลย์

การทำงานของรีเลย์ คือ เมื่อมีกระแสไฟฟ้าไหลผ่านขดลวด จะทำให้ขดลวดเกิดสนามแม่เหล็กไปดึง แผ่นหน้าสัมผัสให้ดึงลงมา แตะหน้าสัมผัสอีกอันทำให้มีกระแสไหลผ่านหน้าสัมผัสไปได้ดังรูปที่ 2.10



รูปที่ 2.11 รูปแสดงสภาวะการทำงานของรีเลย์

ขาของรีเลย์จะประกอบไปด้วยตำแหน่งต่างๆดังนี้คือ

ขาจ่ายแรงดันใช้งาน ซึ่งจะมีอยู่ 2 ขา จากรูปจะเห็นสัญลักษณ์ขดลวดแสดงตำแหน่งขา coil หรือ ขา

ต่อแรงดันใช้งาน

ขา C หรือ COM หรือ ขาคอมมอน จะเป็นขาต่อระหว่าง NO และ NC

ขา NO (Normally opened หรือ ปกติ เปิด) โดยปกติขานี้จะเปิดเอาไว้ จะทำงานเมื่อเราป้อนแรงดันให้รีเลย์

ขา NC (Normally closed หรือ ปกติ ปิด) โดยปกติขานี้จะต่อกับขา C ในกรณีที่เราไม่ได้จ่ายแรงดัน หน้าสัมผัสของ C และ NC จะต่อถึงกัน

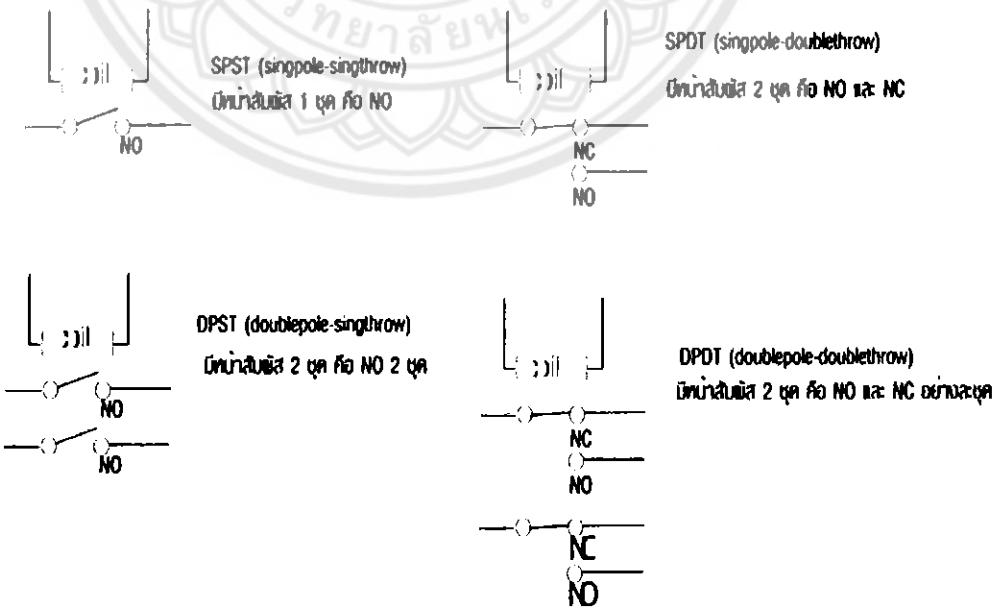
ข้อคำนึงในการใช้งานรีเลย์ทั่วไป

1. แรงดันใช้งาน หรือแรงดันที่ทำให้รีเลย์ทำงานได้ หากเราดูที่ตัวรีเลย์จะระบุค่า แรงดันใช้งานไว้ (หากใช้ในงานอิเล็กทรอนิกส์ ส่วนมากจะใช้แรงดันกระแสตรงในการใช้งาน) เช่น 12 VDC คือต้องใช้แรงดันที่ 12 VDC เท่านั้นหากใช้มากกว่านี้ ขดลวดภายใน ตัวรีเลย์อาจจะขาดได้ หรือหากใช้แรงดันต่ำกว่ามาก รีเลย์จะไม่ทำงาน ส่วนในการต่อวงจรนั้นสามารถต่อขั้วใดก็ได้ เพราะตัวรีเลย์ จะไม่ระบุขั้วต่อไว้ (นอกจากชนิดพิเศษ)

2. การใช้งานกระแสผ่านหน้าสัมผัส ซึ่งที่ตัวรีเลย์จะระบุไว้ เช่น 10A 220 AC คือ หน้าสัมผัสของรีเลย์นั้นสามารถทนกระแสได้ 10 แอมแปร์ที่ 220 VAC ครับ แต่การใช้ก็ควรจะใช้งานที่ระดับกระแสต่ำกว่านี้จะเป็นการดีกว่าครับ เพราะถ้ากระแสมากหน้าสัมผัส ของรีเลย์จะละลายเสียหายได้

3. จำนานหน้าสัมผัสการใช้งาน ควรดูว่ารีเลย์นั้นมีหน้าสัมผัสให้ใช้งานกี่อัน และมีขั้วคอมมอนด้วยหรือเปล่า

ปกติแล้วรีเลย์จะมีหน้าสัมผัสและการเรียกจำนวนหน้าสัมผัสดังรูป 2.11

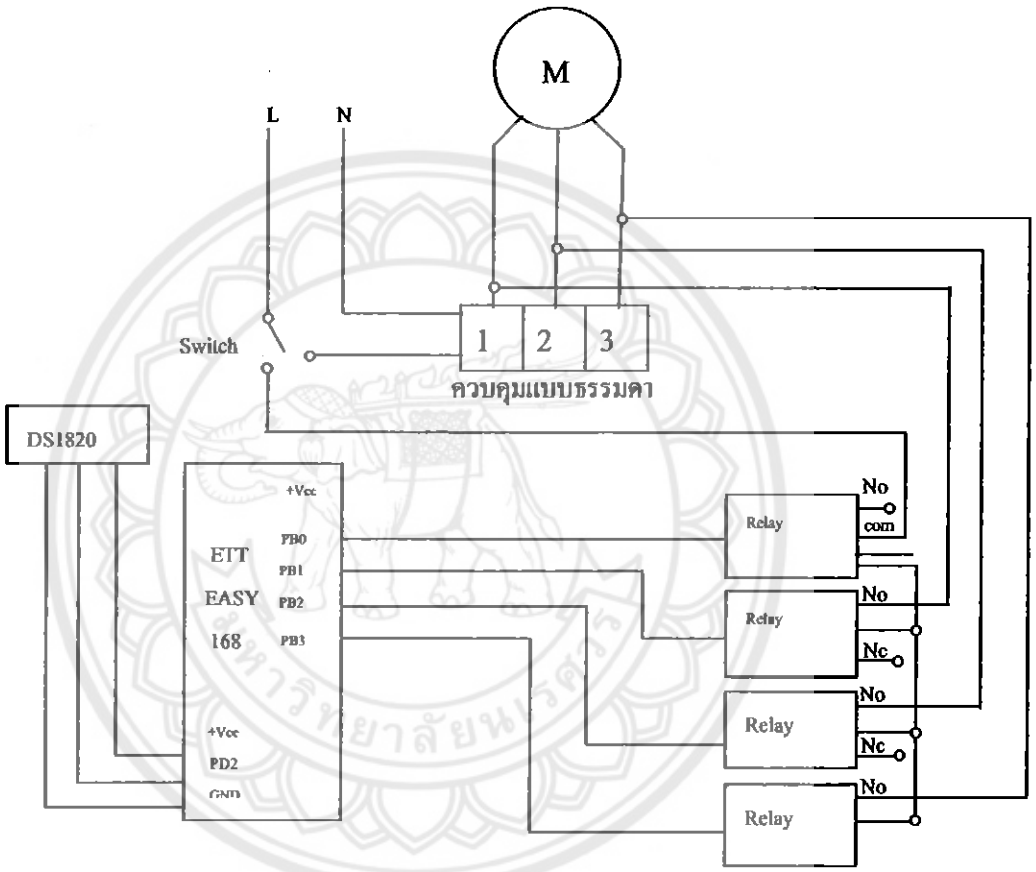


รูป 2.12 หน้าสัมผัสและการเรียกจำนวนหน้าสัมผัส

บทที่ 3
วิธีการดำเนินงาน

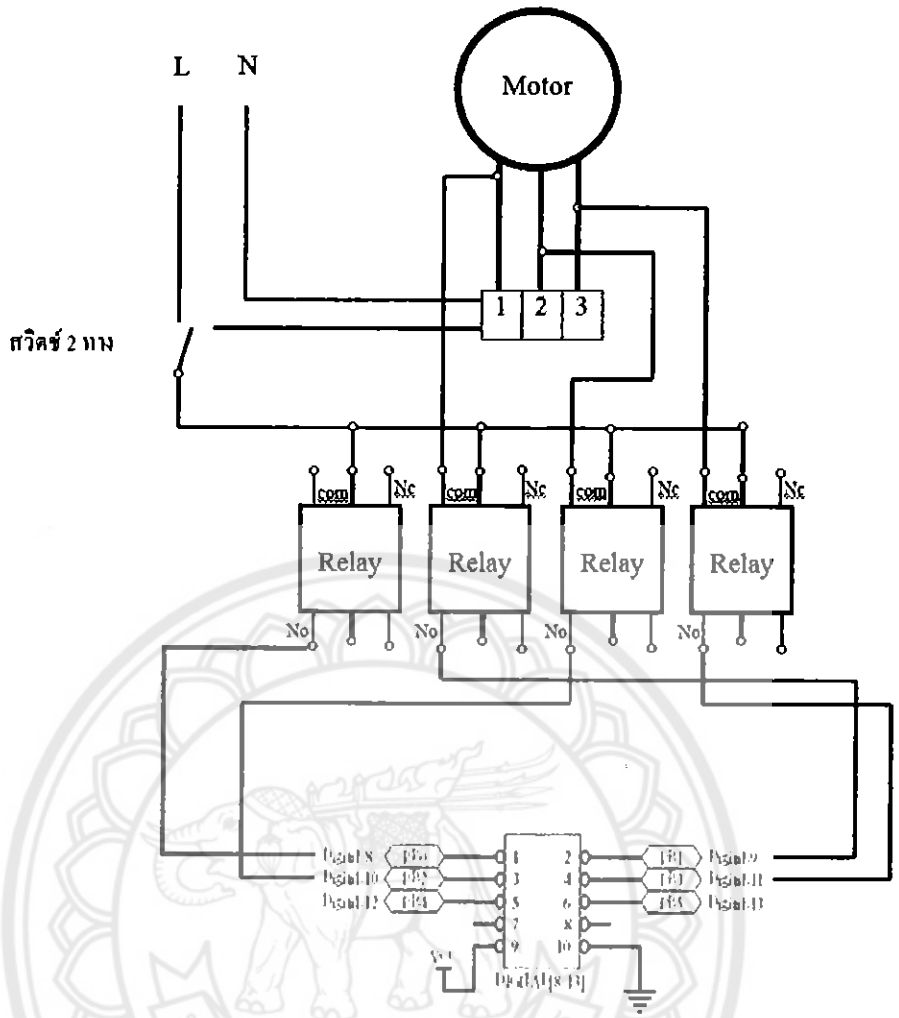
3.1 การออกแบบวงจรควบคุมพัดลม

ทำการออกแบบวงจรควบคุม



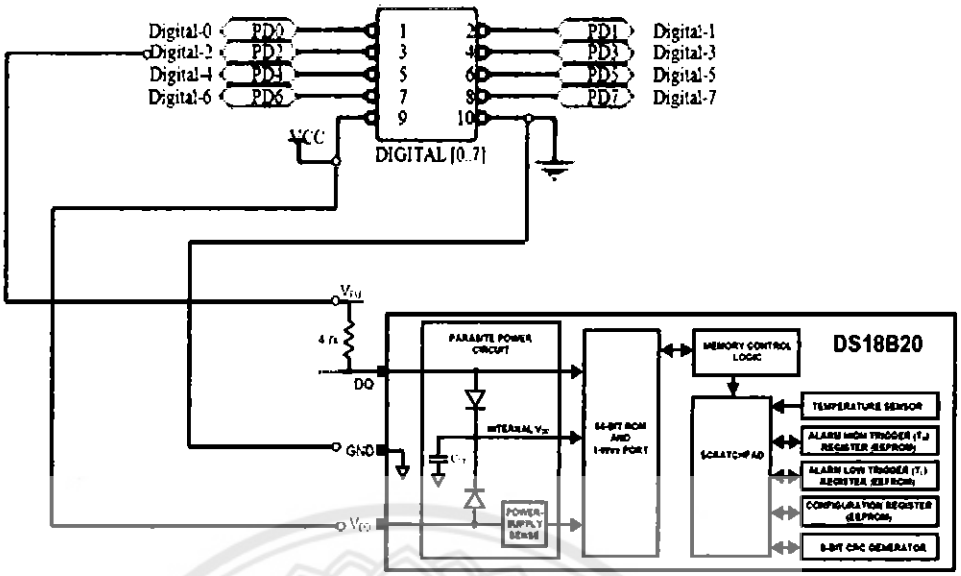
รูปที่ 3.1 วงจรควบคุมพัดลม

จากรูปอธิบายการทำงานของวงจรได้ดังนี้ เมื่อเสียบปลั๊กพัดลมจะมีสวิตช์สองทางว่าเราเลือกจะเลือกใช้แบบธรรมดาหรือว่าให้แบบอัตโนมัติเมื่อเลือกแบบอัตโนมัติแล้วไฟจะเข้าบอร์ดไปส่งให้ DS 18B20 ทำงานและโปรแกรมที่เขียนควบคุมจะทำงานเพื่อขับไปรีเลย์แต่ตัวซึ่งเมื่อโปรแกรมให้รีเลย์ตัวไหนทำงานรีเลย์ตัวนั้นก็จะไปขับให้มอเตอร์ของพัดลมทำงานตามเบอร์ที่กำหนดไวนั้นเอง



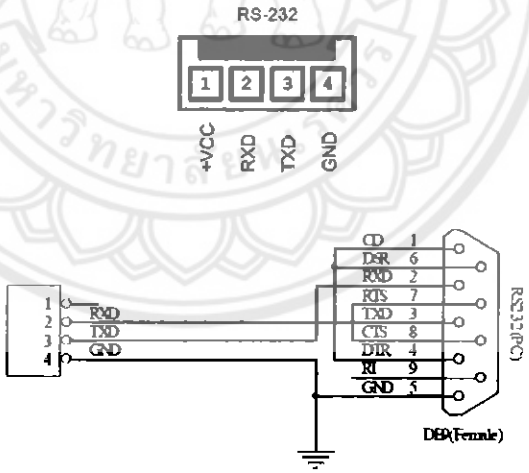
รูปที่ 3.2 การต่อวงจรรีเลย์ต่อเข้ากับขาพอร์ทของบอร์ด

จากรูปอธิบายได้ว่าขาพอร์ท Digital 8–11 ซึ่งได้มาจากรูปของวงจรในภาคผนวกของบอร์ดต่อเข้ากับรีเลย์ที่ขา No ของเมื่อขาพอร์ทไหนทำงานรีเลย์จะทำหน้าที่เป็นสวิตช์ทำให้ขา com (ขาไฟ) ของรีเลย์เข้าไปปิดวงจรทำให้ไฟไหลเข้าไปในมอเตอร์ของพัดลมได้ตามบทที่ 2 ทฤษฎีของรีเลย์



รูปที่ 3.3 การต่อ DS 18B20 เข้ากับขาพอร์ทของบอร์ด

จากรูปที่ 3.4 จะเห็นได้ว่ากำหนดให้ขาพอร์ท PD 2 ซึ่งได้มาจากรูปวงจรของบอร์ดไมโครในภาคผนวกเป็นตัวรับสัญญาณจาก DS 18B20 ซึ่งเราจะกำหนดในโปรแกรมเองว่าจะใช้ขาพอร์ทไหนของบอร์ด และเขียนโปรแกรมให้ DS 18B20 ติดต่อกับบอร์ดรับค่าสัญญาณจาก DS 18B20 ได้



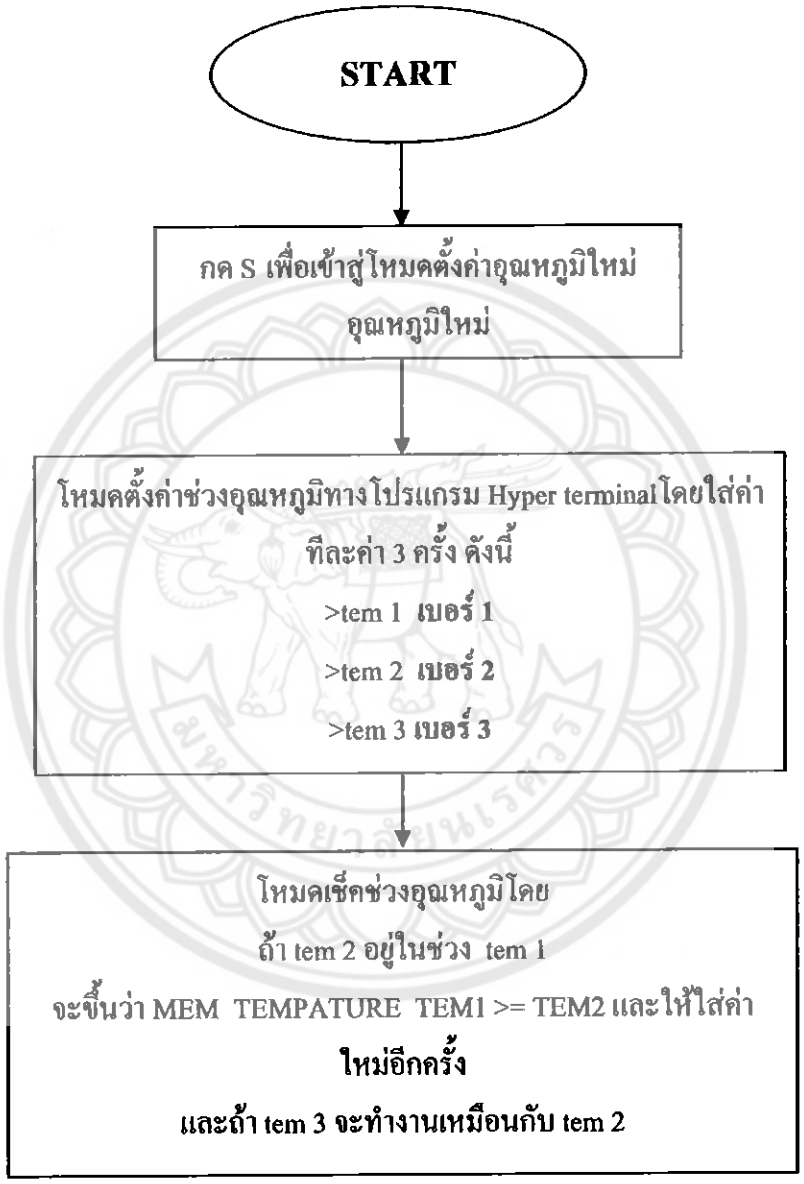
รูปที่ 3.4 การต่อขั้ว RS 232 เข้ากับขาต่อช่อง RS 232 ของ คอมพิวเตอร์

จากรูปแสดงให้เห็นว่าการต่อขา RS 232 เข้ากับช่องต่อ RS 232 ของคอมพิวเตอร์ว่าต่ออย่างไร แต่โดยส่วนมากเมื่อเราซื้อบอร์ดมาจะได้ขั้วต่อของ RS 232 อยู่แล้ว

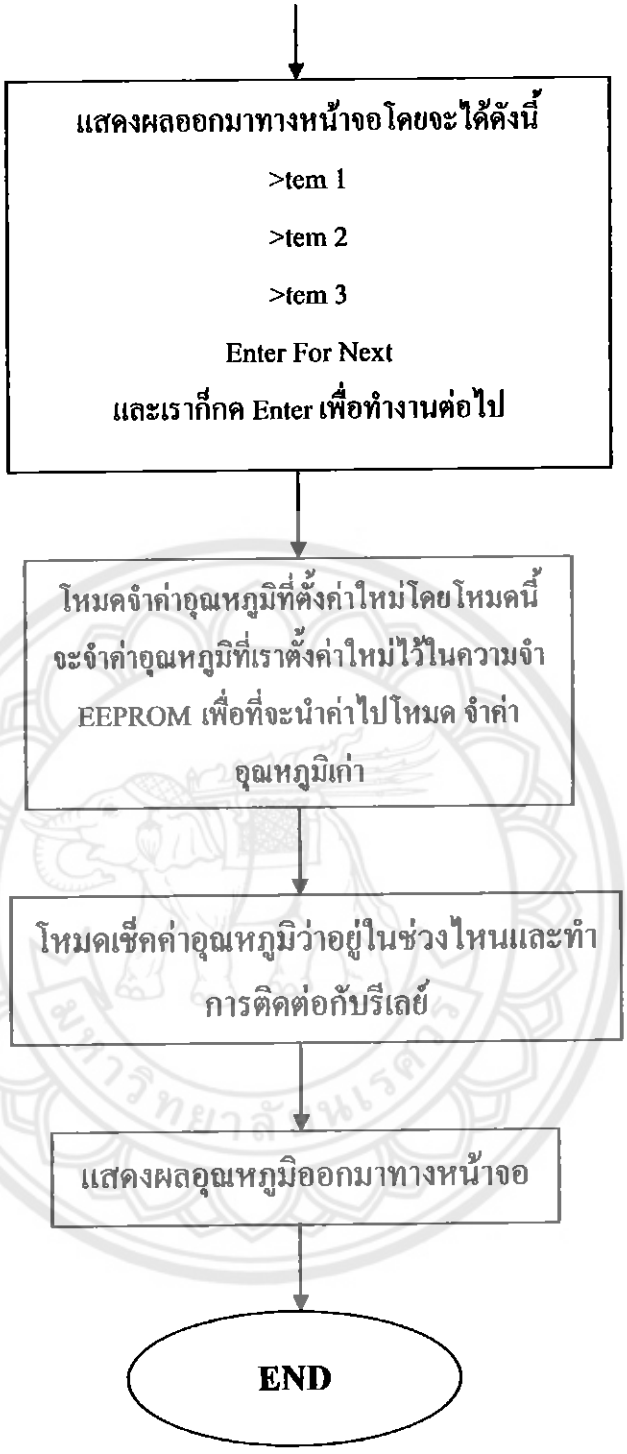
3.2 แผนผังการทำงานของโปรแกรมควบคุมพัดลม

โปรแกรมควบคุมพัดลมอัตโนมัติจะมีอยู่ 2 โหมดคือ โหมด กด S เพื่อตั้งค่าช่วงอุณหภูมิใหม่ และ โหมดใช้ความจำเดิม

โหมด กด S

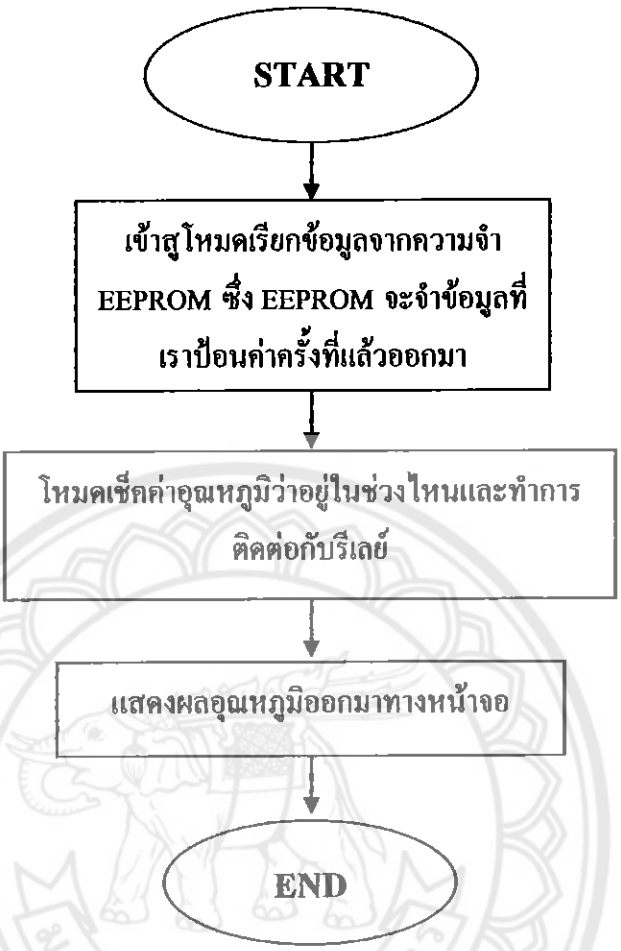


รูปที่ 3.5 ไดอะแกรมโหมดกด S



รูปที่ 3.6 ไดอะแกรมโหมดกด S (ต่อ)

โหมด ใช้ความจำเดิม



รูปที่ 3.7 ไดอะแกรมโหมดใช้ความจำเดิม

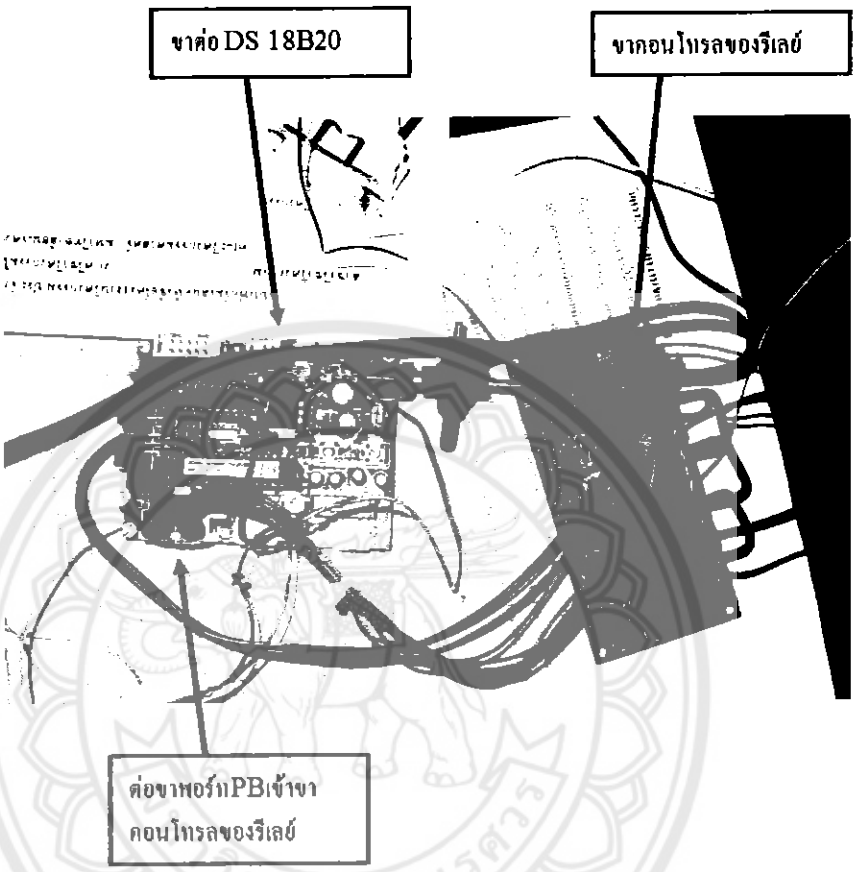
3.3 ทำการเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์

สำหรับไมโครคอนโทรลเลอร์ที่จะใช้ในโครงการคือไมโครคอนโทรลเลอร์ของบริษัท ETT เบอร์ ATMEGA 168 ซึ่งสามารถทำการเขียนโปรแกรม คอมไพล์โปรแกรมและอัดโปรแกรมโดยใช้โปรแกรมที่ชื่อว่า Arduino แล้วเราก็จะได้ไมโครคอนโทรลเลอร์นำไปใช้งานได้ตามต้องการและยังสามารถนำกลับมาลบแล้วเขียนโปรแกรมอัดเข้าไปใหม่ได้อีกหลายครั้ง

ทำการเขียนโปรแกรมลงบอร์ดไมโครคอนโทรลเลอร์ ETT EASY 168 โดยใช้โปรแกรม Arduino ในการคอมไฟล์ และรันโปรแกรมซึ่งได้โปรแกรมควบคุมโดยแสดงไว้ในภาคผนวก

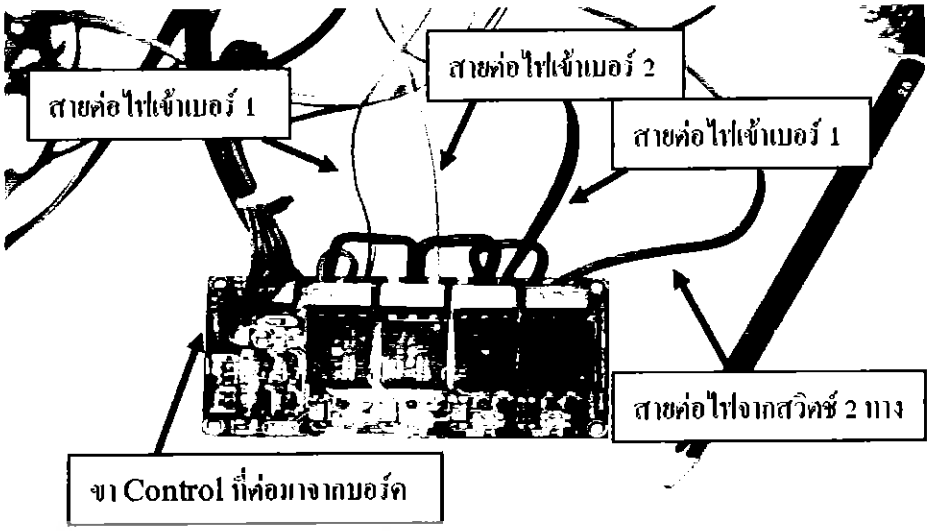
3.4 ทำการประกอบชุดควบคุมเข้ากับพัดลม

เมื่อทำการเขียนโปรแกรมเสร็จแล้วก็ทำการประกอบบอร์ดของ AVR กับดีเลย์เพื่อที่จะประกอบเข้าเป็นชุดควบคุมและต่อเข้ากับพัดลมโดยประกอบตามวงจรควบคุมดังรูปที่ 3.1

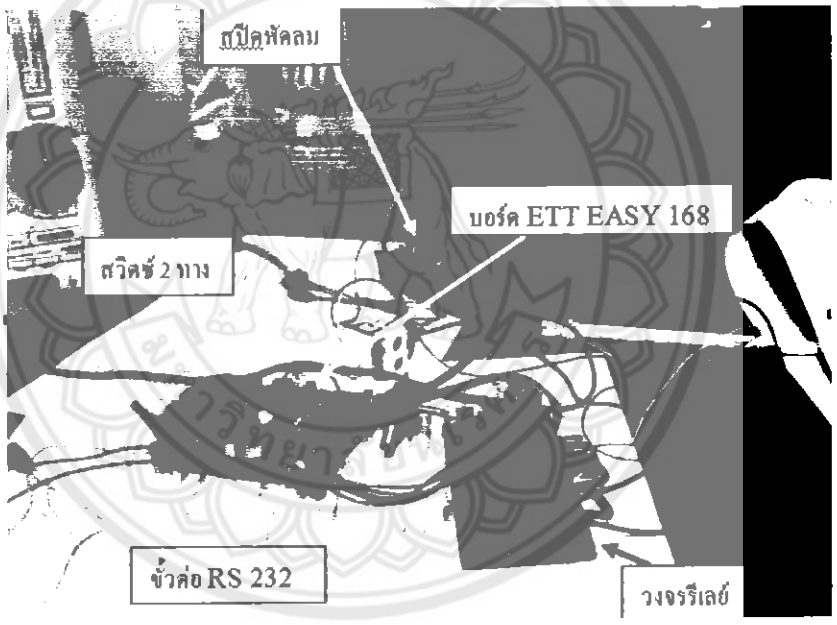


รูปที่ 3.8 ประกอบบอร์ดเข้ากับวงจรรีเลย์

จากรูปที่ 3.5 เราจะเห็นได้ว่าเราจะประกอบวงจรทั้ง 2 วงจรตามรูปที่ 3.3 เมื่อประกอบบอร์ดเข้ากับวงจรของรีเลย์แล้วเราจะเรียกวงจรรวมว่าเป็นวงจรชุดควบคุมเพื่อให้เรียกได้ง่ายขึ้นในการอ้างถึงบอร์ดรีเลย์และบอร์ดคอนโทรลที่ต่อเข้าด้วยกัน



รูปที่ 3.9 บอร์ดรีเลย์ต่อเข้า Speed พัดลม



รูปที่ 3.10 ประกอบวงจรชุดควบคุมเข้ากับ Speed พัดลมแต่ละเกียร์

บทที่ 4

ผลการทดลอง

4.1 วัดค่าทางไฟฟ้าของพัดลม

ทำการตรวจวัดค่าทางไฟฟ้าที่ค่าแรงดันที่ 220 V ได้ค่ากระแส และค่ากำลังทางไฟฟ้าจากตัวพัดลมตั้งโต๊ะ ได้ค่าดังนี้

ที่แรงดัน 220 V

ได้ค่ากระแส ที่ - Speed 1 กระแส 0.19 A

ที่ - Speed 2 กระแส 0.20 A

ที่ - Speed 3 กระแส 0.22 A

ได้ค่ากำลังไฟฟ้า ที่ - Speed 1 กำลังไฟฟ้า 44 W

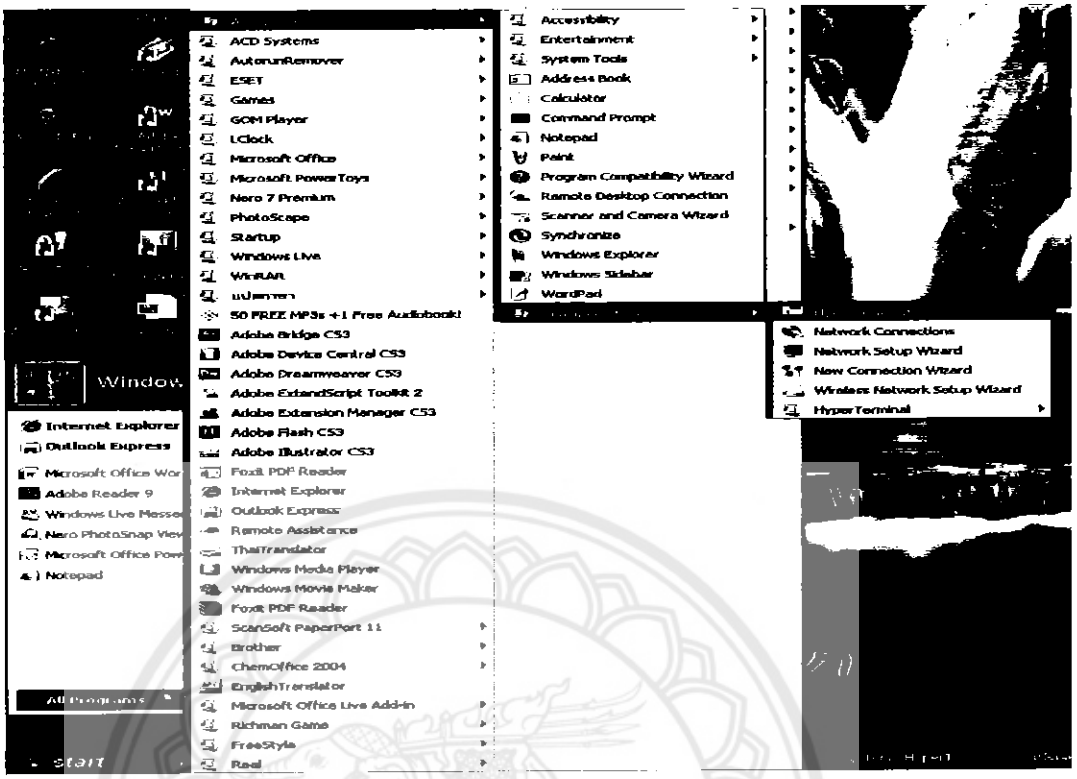
ที่ - Speed 2 กำลังไฟฟ้า 48 W

ที่ - Speed 3 กำลังไฟฟ้า 51 W

4.2 ทำการต่อชุดควบคุมเข้ากับคอมพิวเตอร์

ทำการต่อชุดควบคุมเข้ากับคอมพิวเตอร์เพื่อทำการตั้งค่าของอุณหภูมิ โดยโปรแกรม Hyper Terminal ซึ่งโปรแกรม Hyper Terminal นี้จะมีอยู่ประจำเครื่องอยู่แล้วผ่านทางพอร์ท RS 232 โดยได้ดังนี้

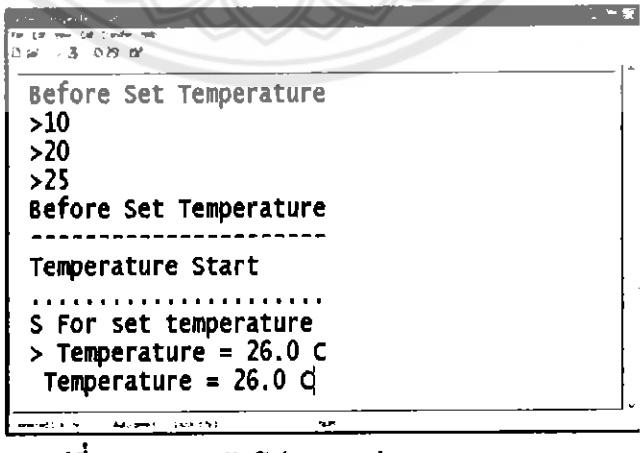
4.2.1 เปิดโปรแกรม Hyper Terminal



รูปที่ 4.1 เปิดโปรแกรม Hyper Terminal

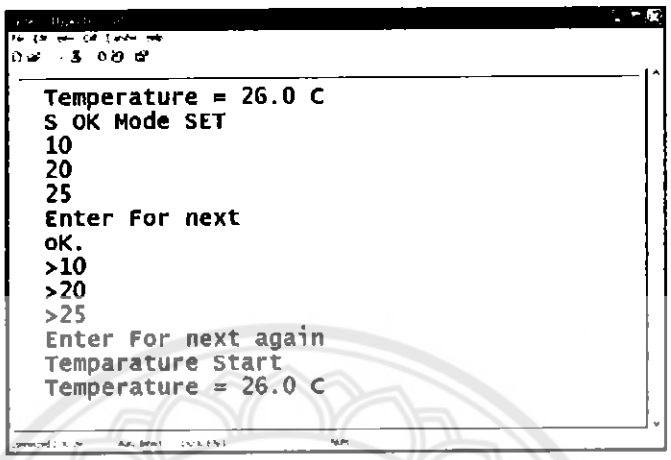
4.2.2 ทำการเซ็ต โปรแกรม Hyper Terminal

เมื่อเข้าสู่โปรแกรม Hyper Terminal ทำการเซ็ตค่าโดยตั้งชื่อคอมพิวเตอร์ และไปตามขั้นตอนและตั้งค่า bit per second เป็น 19200 แล้วทำการรีเซ็ตบอร์ดโดยกดปุ่มรีเซ็ตที่บอร์ด 1ครั้ง แล้วจะได้หน้าต่างดังนี้



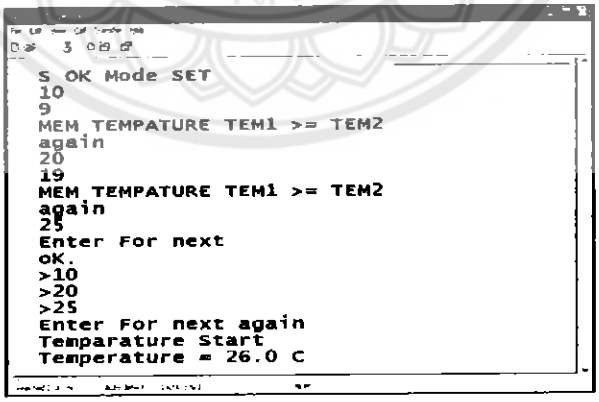
รูปที่ 4.2 ทำการรันโปรแกรมผ่านทาง Hyper Terminal

เมื่อทำการกดรีเซ็ตที่บอร์ดแล้วจะได้ดังรูปที่ 4.2 แสดงว่าโปรแกรมจากบอร์ดทำงานเป็นไปตามที่เขียนโปรแกรมมาและทำการตั้งค่าอุณหภูมิโดยป้อนตัวเลขผ่าน โปรแกรม Hyper Terminal ซึ่งได้ผลดังนี้



รูปที่ 4.3 ทำการตั้งค่าอุณหภูมิใหม่

เมื่อทำการป้อนค่า โดยเบอร์ 1ของพัดลมใส่เลขครั้งที่ 1และครั้งที่ 2 โดยครั้งที่ 1 เป็นค่า มากว่าอุณหภูมินี้แล้วทำงาน และครั้งที่ 2 เป็นค่าที่น้อยกว่าอุณหภูมินี้ไม่ทำงานโดยเบอร์ 2 และเบอร์ 3 ของพัดลมจะตั้งค่าเหมือนเบอร์ 1เป็นอุณหภูมิทั้งหมด 3 ค่าแล้วกด Enter ครั้งแรกจะแสดงอุณหภูมิที่เราตั้งค่า และเมื่อกด Enter ครั้งที่ 2จะทำการรันโปรแกรมตามเดิมแสดงว่าโปรแกรมที่เขียนมาใช้งานได้ตามที่ออกแบบไว้ตั้งแต่แรกตาม Flow Chart รูปที่ 3.2 และรูปที่ 3.3

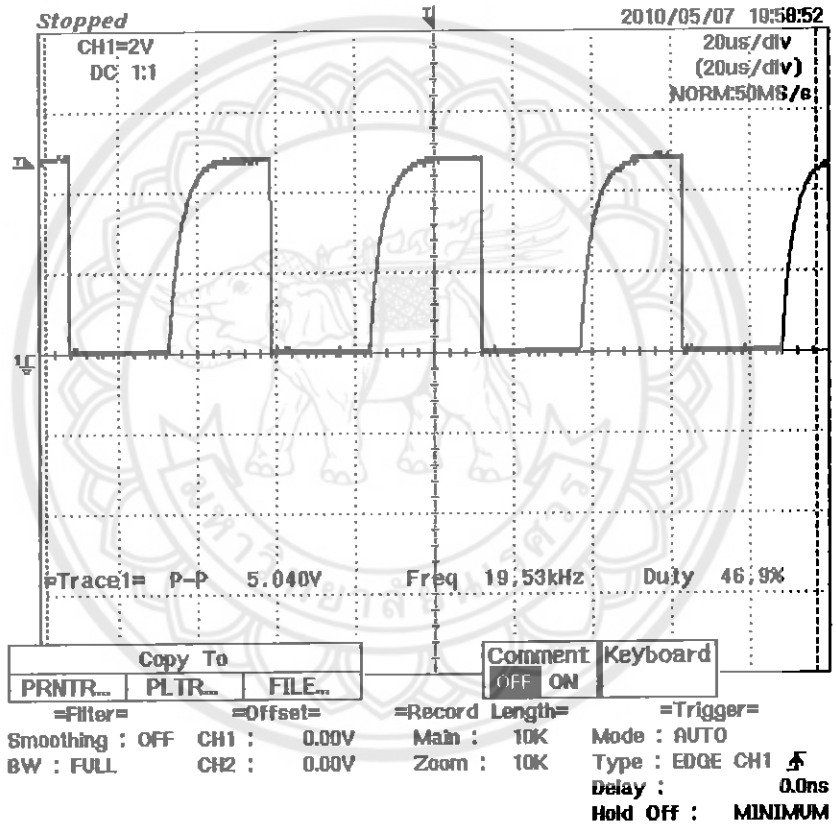


รูปที่ 4.4 แสดงการตั้งค่าอุณหภูมิซ้ำ

จากรูปที่ 4.4 จะเห็นว่าเมื่อเราทำการป้อนค่าตัวเลขซ้ำภายในช่วงของเบอร์ 1 โปรแกรมจะทำการฟ้องว่าคุณป้อนช่วงค่าอุณหภูมิซ้ำและให้ทำการป้อนค่าใหม่จนกว่าช่วงอุณหภูมิไม่ซ้ำกับเบอร์ 1 และถ้าเข้าสู่ช่วงเบอร์ 3 ก็เช่นกันเมื่อเราป้อนค่าที่ซ้ำกับเบอร์ 2 โปรแกรมก็จะฟ้องและให้ป้อนค่าใหม่

4.3 การวัดสัญญาณของ DS 18B20

ทำการวัดค่าสัญญาณ ของ DS 18B20 ว่า DS 18B20 ทำงานหรือไม่ โดยวัดสัญญาณของ ขา DS1820 เทียบกับ กราวด์ซึ่งได้กราฟสัญญาณการทำงานของ DS 18B20 ดังนี้



รูปที่ 4.5 กราฟแสดงสัญญาณของ DS 18B20

จากรูปจะเห็นได้ว่าเมื่อ DS 18B20 ส่งสัญญาณแรงดันของ DS 18B20 จะพุ่งขึ้นสูงที่ นั่นคือ DS 18B20 ส่งสัญญาณให้กับบอร์ด

บทที่ 5
สรุป

5.1 สรุป

จากการทดลองวงจรควบคุมพัดลมนั้น พัดลมสามารถทำงานตามอุณหภูมิห้องในขณะนั้น เมื่อตั้งค่าทางโปรแกรม Hyper Terminal โดยไม่ต้องแก้ไขตัวโปรแกรมที่เขียนแล้วโหลดลงบอร์ดใหม่ ซึ่งเราป้อนค่า อุณหภูมิตรงตามที่เราต้องการพัดลมจะทำงานตามอุณหภูมิที่เราตั้งค่าไว้ โดยทำการป้อนค่า โดยเบอร์ 1 ของพัดลมใส่เลขครั้งที่ 1และครั้งที่ 2 โดยครั้งที่ 1 เป็นค่า มากกว่าอุณหภูมินี้แล้วทำงาน และครั้งที่ 2 เป็นค่าที่น้อยกว่าอุณหภูมินี้ไม่ทำงานโดยเบอร์ 2 และเบอร์ 3 ของพัดลมจะตั้งค่าเหมือนเบอร์ 1 เป็นอุณหภูมิทั้งหมด 3 ค่าแล้วกด Enter ครั้งแรกจะแสดงอุณหภูมิที่เราตั้งค่าและเมื่อกด Enter ครั้งที่ 2 จะทำการรันโปรแกรมตามเดิมโดยมีเงื่อนไขการตั้งค่าคือไม่ตั้งค่าอุณหภูมิอยู่ในช่วงเดียวกันมิฉะนั้นโปรแกรมจะแจ้งให้ใส่ค่าใหม่อีกที

5.2 ปัญหา

ปัญหาในเรื่องของโปรแกรมที่ใช้ในการควบคุมไมโครคอนโทรลเลอร์ เนื่องจากผู้ทำโครงการต้องศึกษาเองทั้งหมด จึงทำให้ในส่วนของโปรแกรมเกิดการล่าช้า

References หน้า 24

MISSING

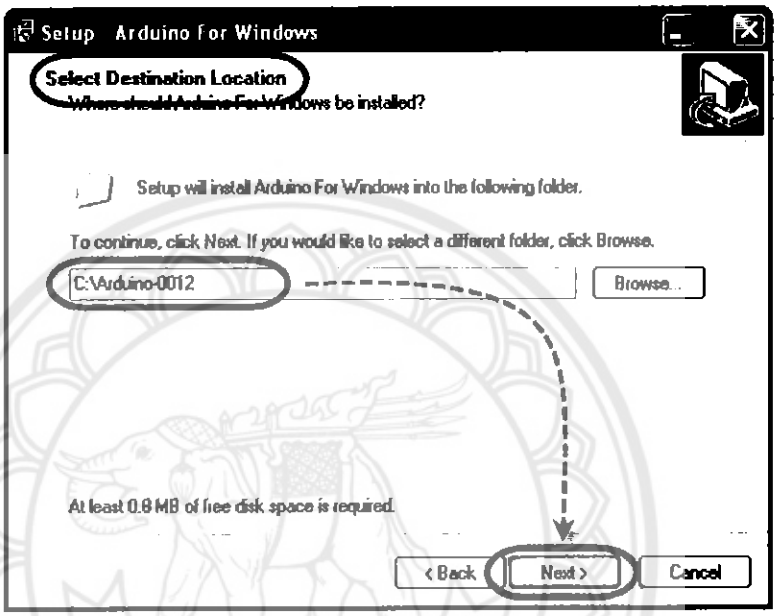
ภาคผนวก

การติดตั้งโปรแกรม Arduino

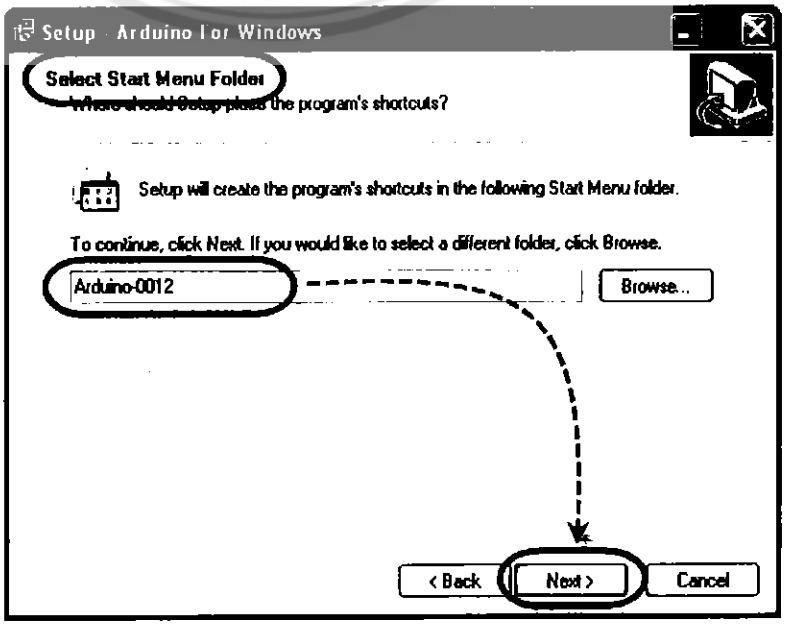
สำหรับโปรแกรม Arduino นั้น ได้รับการพัฒนาขึ้นมาให้สามารถใช้งานกับระบบปฏิบัติการแบบต่างๆ ได้หลาย Platform ซึ่งปัจจุบัน (เดือน ธันวาคม พ.ศ.2551) โปรแกรมของ Arduino ได้รับการปรับปรุงเป็นรุ่น เวอร์ชั่น "Arduino-0012" แล้ว โดยมีโปรแกรมให้เลือกใช้งาน 4 Platform ทั้ง Windows,Mac OSxและ Linux โดยผู้อ่านสามารถเข้าไป ตรวจสอบ หรือ Download โปรแกรมรุ่นใหม่ๆของ Arduino มาใช้งานได้ฟรีโดยไม่เสียค่าใช้จ่ายใดๆจาก"http://arduino.cc/" หรือ "http://arduino.cc/en/Main/Software" ซึ่งเป็นเว็ปไซต์ที่ได้รวบรวมรายละเอียดและข่าวความเคลื่อนไหวต่างๆ เกี่ยวกับ Arduino มากมาย ซึ่งข้อมูลต่างๆจะได้รับการปรับปรุงอย่างต่อเนื่องเป็นประจำสำหรับกรณี ที่ผู้อ่านใช้งานกับบอร์ด Arduino ของ บริษัท อีทีที นั้น โปรแกรมต่างๆจะถูกรวบรวมไว้ในแผ่น CD ROM เป็นที่เรียบร้อยแล้ว โดยโปรแกรมดังกล่าวจะเป็น รุ่นที่ได้รับการปรับแต่งรายละเอียดให้สามารถใช้งานร่วมกันกับบอร์ดรุ่นต่างๆที่ทางบริษัท อีทีที ผลิตขึ้นมาใหม่ได้ด้วย นอกจากนี้แล้วทางบริษัทอีทีที ยังได้ทำการเพิ่มเติม Library ส่วนที่ทาง อีทีที พัฒนาปรับปรุงขึ้นมาใหม่รวมไว้ในชุดโปรแกรมดังกล่าวด้วยแล้ว พร้อมกับทำการเพิ่ม Install Shield เข้าไปด้วยเพื่อให้ผู้ใช้สามารถทำการติดตั้งใช้งานโปรแกรมได้โดยง่ายเช่นเดียวกันกับการ Install โปรแกรมทั่วๆไปสำหรับขั้นตอนการติดตั้งโปรแกรมนั้น สามารถทำได้ตามขั้นตอนของ Wizard ในการติดตั้งของโปรแกรมได้ทันที โดยมีลำดับขั้นตอนดังนี้

1. สั่ง Run ไฟล์ "ET-ARDUINO-0012-WIN.EXE" ซึ่งจะได้ผลดังรูป

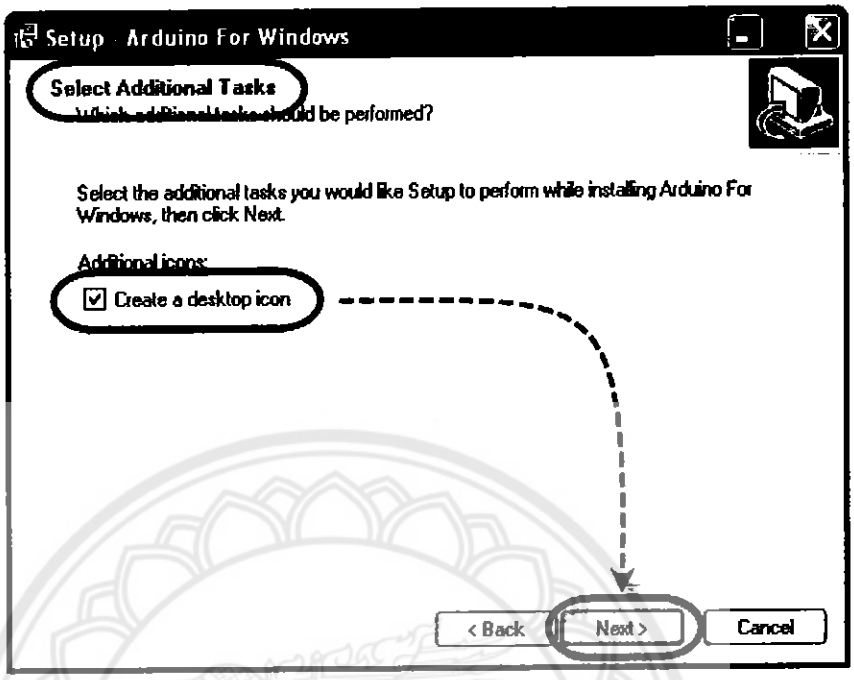
2. ในขั้นตอนนี้โปรแกรมจะให้กำหนด ตำแหน่งโฟลเดอร์ที่จะใช้สำหรับติดตั้งโปรแกรม ซึ่งให้เลือกกำหนดตามค่า Default ของโปรแกรมการติดตั้ง คือ C:\Arduino-0012 แล้วเลือก Next ดังรูป



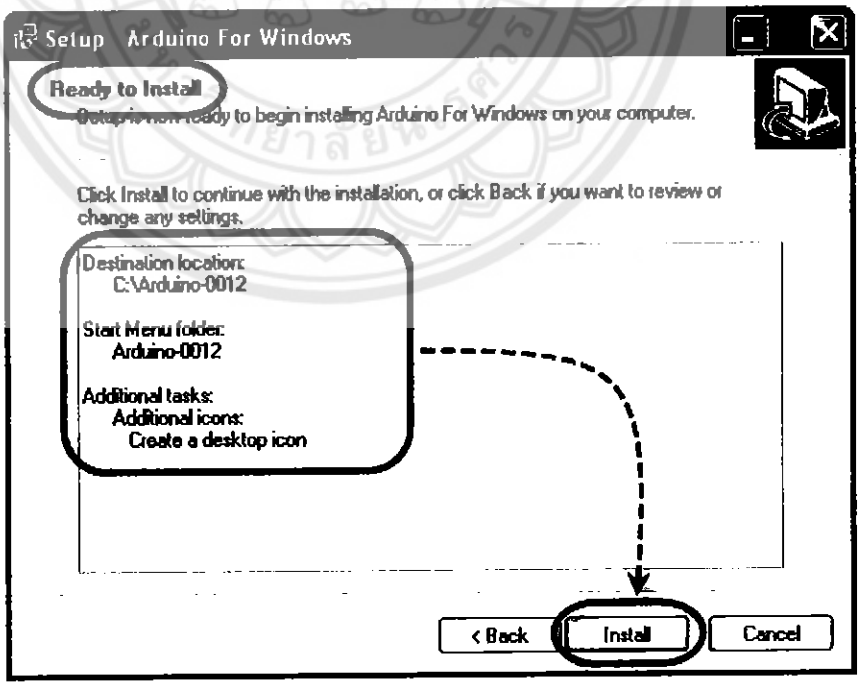
3. ในขั้นตอนนี้โปรแกรมจะให้กำหนด ชื่อโฟลเดอร์ที่จะใช้สำหรับเรียกใช้โปรแกรมผ่านทางเมนูคำสั่งของ Windows ซึ่งในขั้นตอนนี้แนะนำให้เลือกกำหนดตามค่า Default ของโปรแกรมการติดตั้ง คือC:\Arduino-0012 แล้วเลือก Next ดังรูป



4. ในขั้นตอนนี้ให้เลือก Create a desktop icon ด้วย เพื่อให้โปรแกรมสร้าง Icon สำหรับเรียกใช้งานโปรแกรมที่หน้า Desktop ให้ด้วย แล้วเลือก Next ดังรูป



5. เมื่อถึงขั้นตอนนี้ โปรแกรมก็พร้อมทำการติดตั้งแล้ว โดยโปรแกรมจะแสดงค่าตัวเลือกต่างๆ ที่ถูกกำหนดไว้ในขั้นตอนก่อนหน้านี้ให้ทราบ เมื่อทุกอย่างถูกต้องให้เลือก Install ซึ่งโปรแกรมก็จะเริ่มทำการติดตั้งทันที

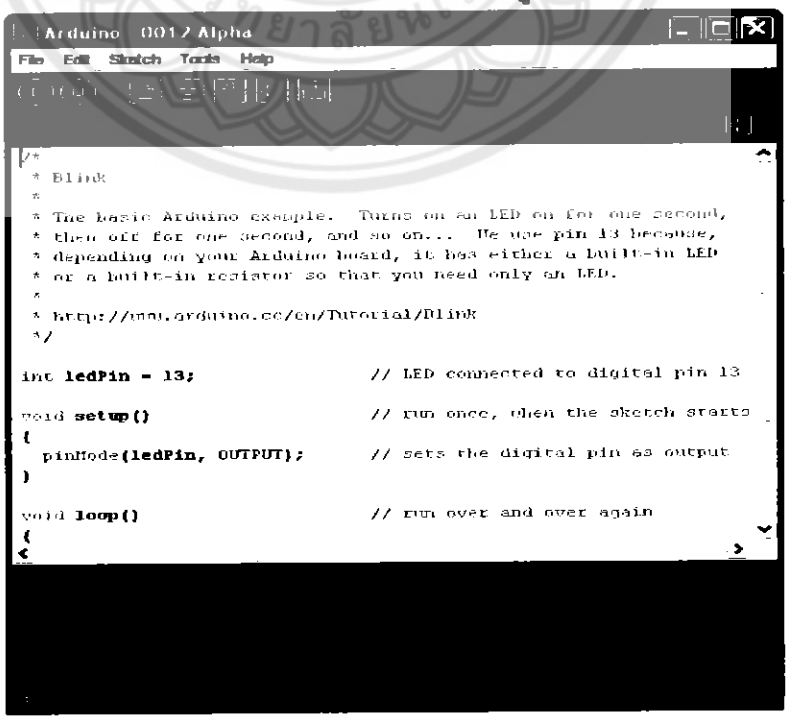


6. ให้รอจนกระทั่งขั้นตอนการติดตั้งเรียบร้อยแล้วเลือก Finish ดังรูป

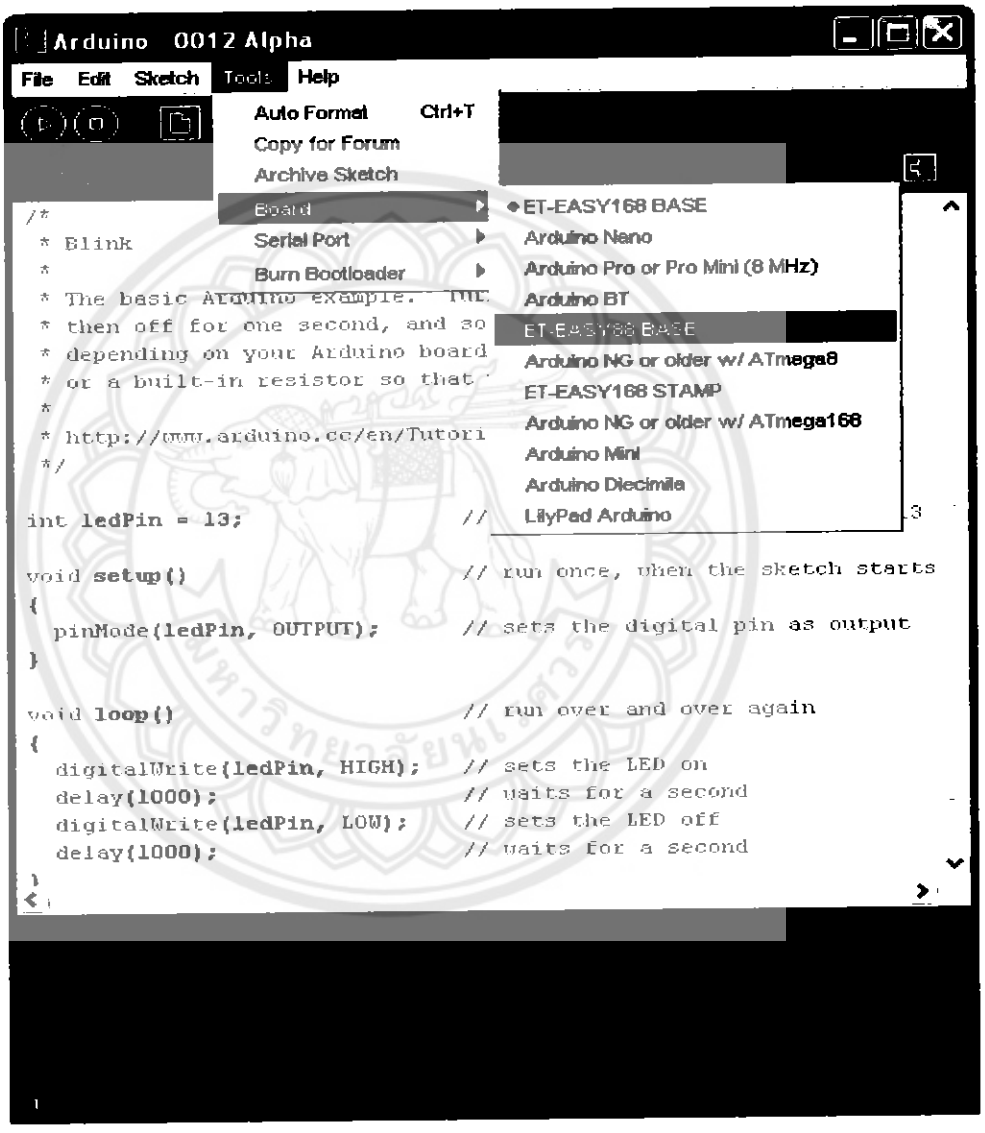
ทดสอบเขียนโปรแกรมใช้งานด้วย Arduino

หลังจากที่เราได้ทำการติดตั้งโปรแกรม Arduino เป็นที่เรียบร้อยแล้ว ก็เป็นอันเสร็จสิ้นขั้นตอนของการเตรียมการแล้ว ลำดับขั้นตอนต่อจากนี้เป็นต้นไป ก็เป็นเรื่องของการใช้งาน การเขียนโปรแกรม และการศึกษาเรียนรู้ต่างๆตามความต้องการแล้ว แต่ก่อนอื่นเราจะต้องทำการติดตั้งโปรแกรมของ Arduino เพื่อใช้เป็นโปรแกรมสำหรับศึกษาเรียนรู้ ซึ่งมีลำดับขั้นตอนดังต่อไปนี้

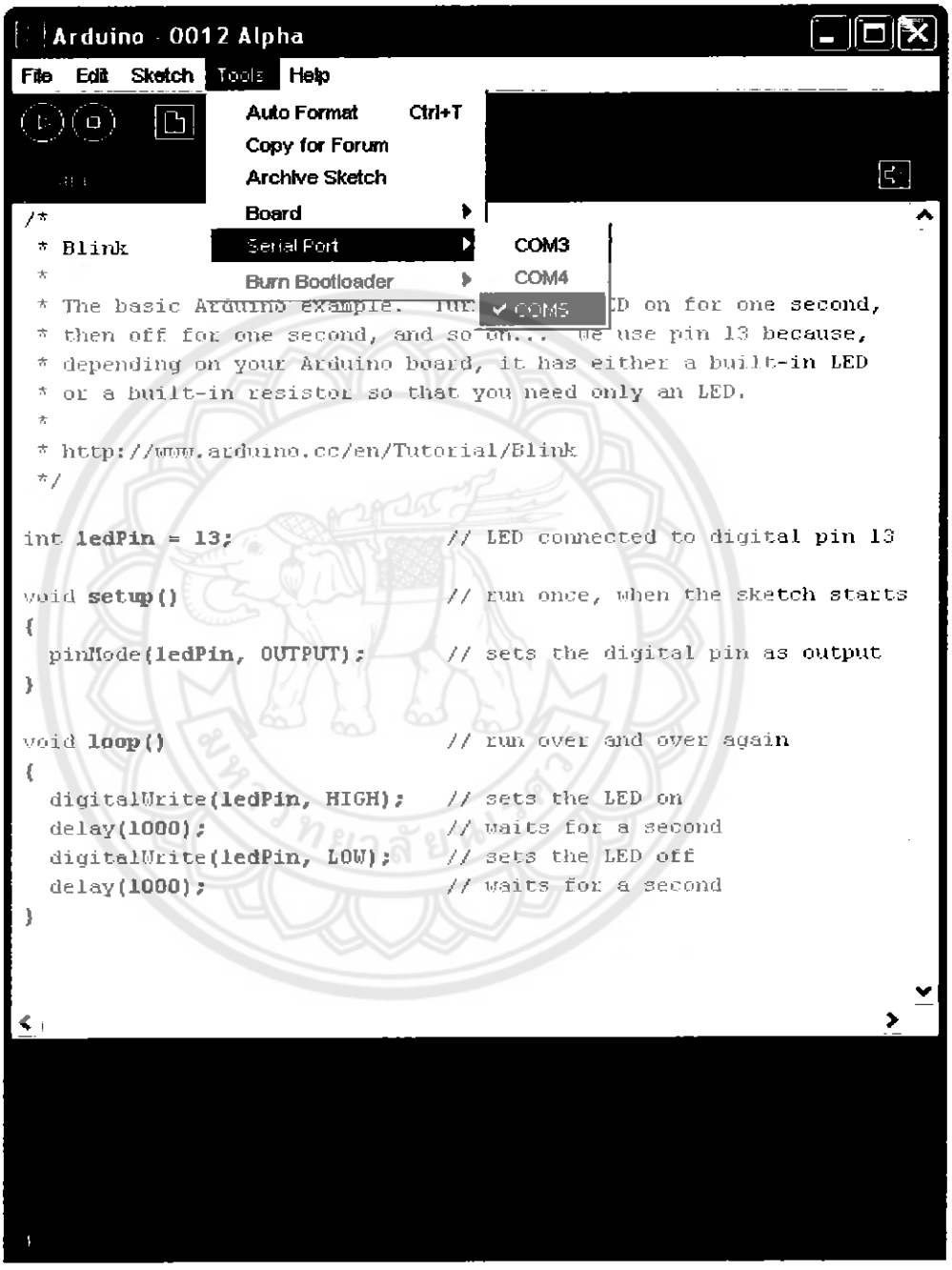
1. ทำการสั่ง Run โปแกรม "arduino.exe" จะได้ผลดังรูป



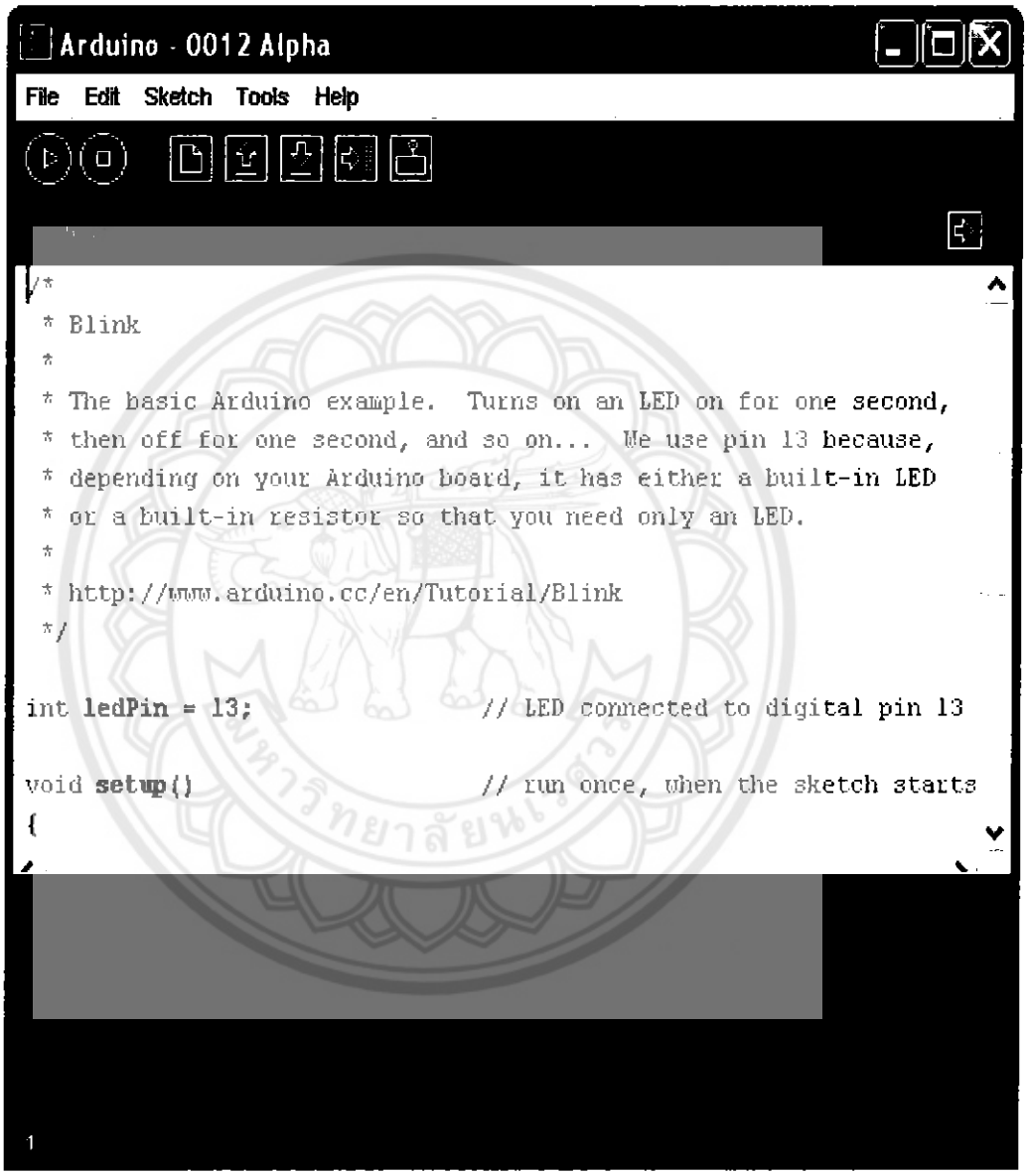
2. ในครั้งแรกของการเรียกใช้งานโปรแกรม ให้ทำการกำหนดระบบฮาร์ดแวร์ที่จะใช้งานกับโปรแกรมของ Arduino ให้เรียบร้อยเสียก่อน เนื่องจากในปัจจุบันนี้ มีการออกแบบวงจรและสร้างฮาร์ดแวร์บอร์ดแบบต่างๆสำหรับนำมาใช้งานร่วมกับโปรแกรมพัฒนาของ Arduino ไว้มากมายหลายรุ่น โดยในกรณีของบอร์ด ET-BASE AVR EASY88 ให้ทำการเลือกกำหนดชื่อบอร์ดเป็น "EASY88 BASE"โดยคลิกเมาส์ที่ "Tools ⟶ Board ⟶ "ET-EASY88 BASE" ดังรูป



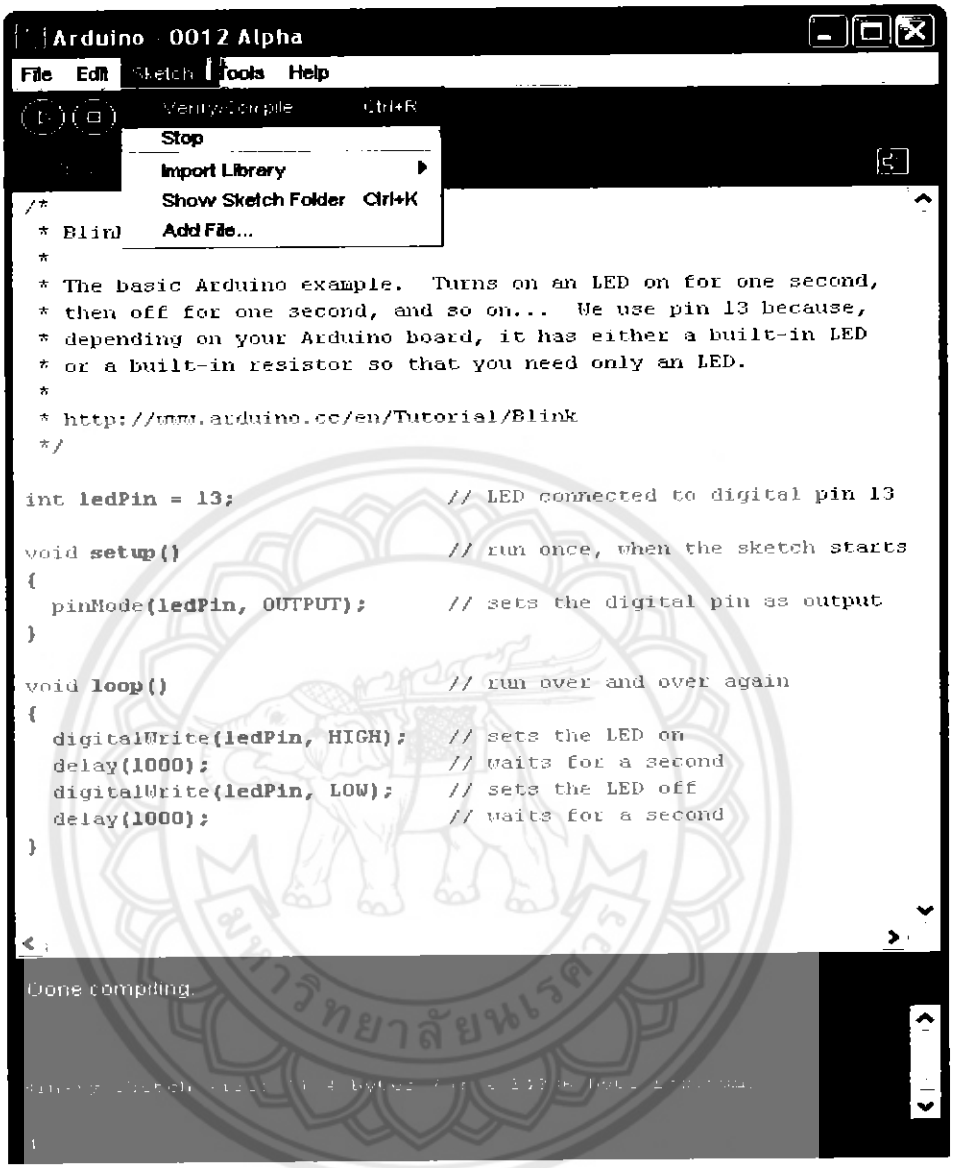
3. เลือกกำหนดหมายเลขพอร์ต สำหรับติดต่อสื่อสารกับบอร์ด ให้ตรงกับหมายเลข Comport ที่ต่อใช้งานไว้จริงในเครื่องคอมพิวเตอร์ PC เช่น ถ้าหมายเลข Comport ของเครื่องคอมพิวเตอร์ PC เป็น COM5 ให้คลิกเมาส์ที่ Tools → Serial Port → COM5 ดังรูป



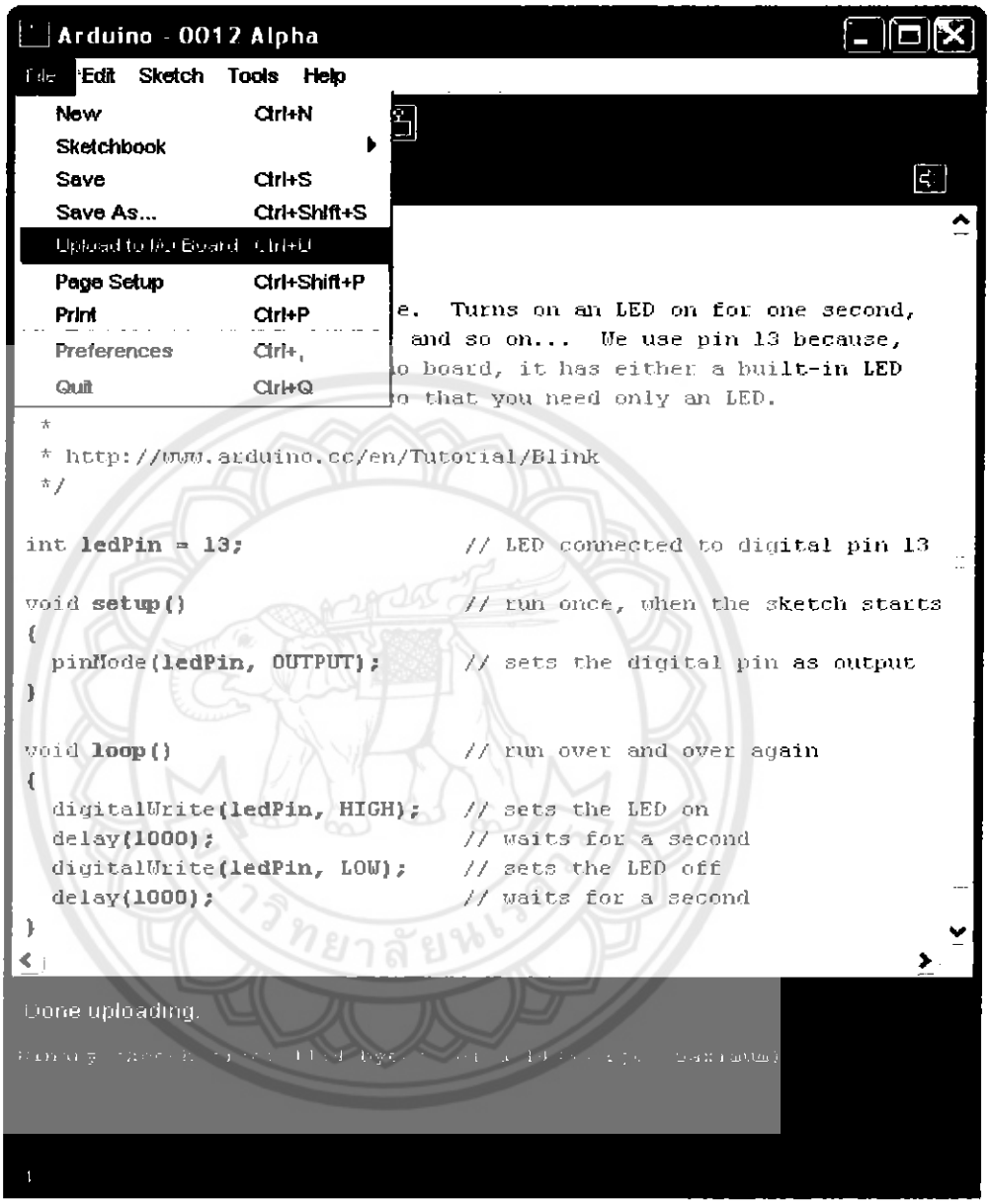
4. ทดสอบเขียนโปรแกรม โดยคลิกเมาส์ที่ File ⟶New แล้วพิมพ์โปรแกรมทดสอบ หรืออาจใช้การสั่งเปิดไฟล์ตัวอย่างที่สร้างไว้แล้วขึ้นมาแทนก็ได้ โดยในที่นี้ขอแนะนำให้ทดสอบด้วยโปรแกรมไฟกระพริบ โดยให้เลือก "File ⟶ sketchbook ⟶ Examples ⟶ Digital ⟶ Blink" ซึ่งจะได้ดังรูป



5. สั่งแปลโปรแกรมโดยคลิกเมาส์ที่ “Sketch ⟶ Verify/Compile” ดังตัวอย่าง

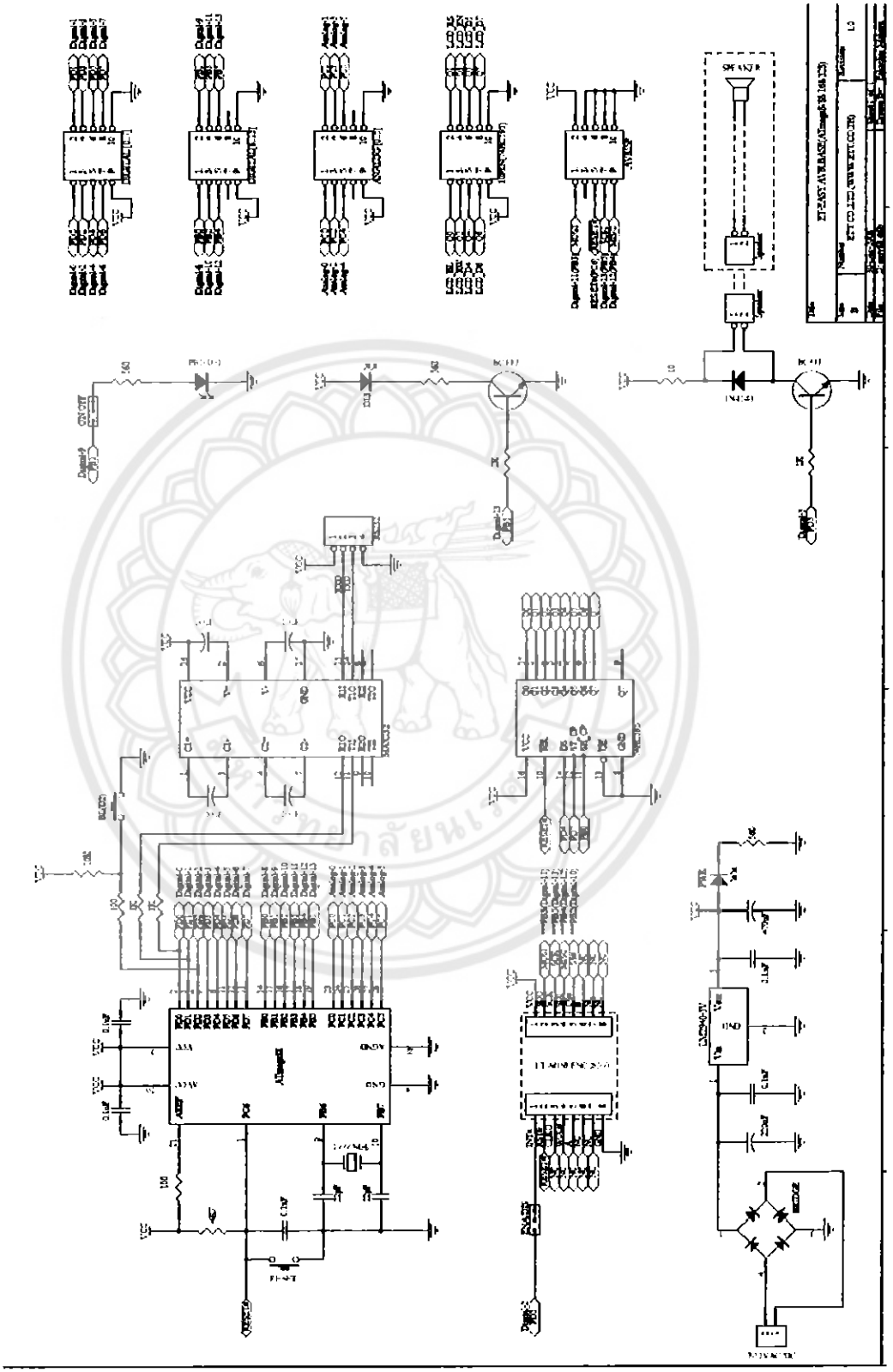


6. สั่ง Download Code ให้กับบอร์ด โดยคลิกเมาส์เลือกที่ "File → Upload to I/O Board" แล้วรอสักครู่จนโปรแกรมทำงานเสร็จ ซึ่งควรได้ผลดังรูป



7. หลังจากที่ทำการ Upload Code ให้กับบอร์ดเป็นที่เรียบร้อยแล้ว บอร์ดก็จะเริ่มต้นทำงานตามคำสั่งที่เขียนไว้ในโปรแกรมทันที โดยจะสังเกตเห็น LED กระพริบ ติด และ ดับ สลับกันไปมา ด้วยความเร็วประมาณ 1 วินาที ตลอดเวลา

รูปวงจร ETT EASY 168



ทำการศึกษา Serial module ใน SoftSerial libraly

SoftwareSerial Library เป็น Module หนึ่งของ Arduino ที่ใช้งานการเชื่อมต่อจากคอมพิวเตอร์ผ่านทาง Serial Port หรือก็คือ การติดต่อสื่อสารกันระหว่าง Arduino และ อุปกรณ์คอมพิวเตอร์ต่าง ๆ โดยใช้สาย Serial บน Digital Pins 0 (RX) and 1 (TX) โดยคำสั่งใน Serial module นี้จะมีด้วยกันอยู่ดังนี้

Serial.begin(speed) คือ การเซตค่า Baud rate ในการส่งค่าไปตามสาย Serial ซึ่งเราจะใช้กันแค่ 9600

Serial.available() คือ จำนวน Character ที่อ่านได้จากบัฟเฟอร์ซึ่งเกิดจาก คำสั่ง Serial.read()

Serial.read() คือ การอ่านค่าที่ได้รับจาก RX

Serial.print(data) คือ การส่งค่าไปยัง TX

Serial.println(data) คือ เหมือนกับคำสั่ง Serial.print() แต่ทำการขึ้นบรรทัดใหม่ด้วย

เอกสารนี้นำมาจากโครงการ Robot positioning

ส่วนของโปรแกรม

```
#include <OneWire.h>
#include <EEPROM.h>
```
ใช้นามสกุล OneWire.h และ EEPROM.h

```
OneWire  ds(2);
```
ต่อ DS 1820 ที่ขา 2 กับบอร์ด

```
char  ReadByte;
```
ประกาศตัวแปรReadByte เป็นประเภท Character เพื่อ เก็บคำสั่งให้อ่านค่าเมื่อตรงเงื่อนไขที่ตั้งไว้

```
char mem0[20];
char mem1[20];
char mem2[20];
char mem3[20];
char mem4[20];
char mem5[20];
char mem6[20];
```
ประกาศตัวแปร mem0, mem1, mem2, mem3, mem4, mem5, mem6 เป็นประเภท Character (ตัวอักขระ) ซึ่งแต่ละตัวแปรจะเก็บค่าได้ 20 ตำแหน่ง เพื่อเก็บข้อมูลไว้ใน EEPROM

```
int  SD1 ;
int  SD2 ;
int  SD3 ;
int  SD4 ;
int  SD5 ;
int  SD6 ;
```
ประกาศตัวแปร SD1, SD2, SD3, SD4, SD5, SD6 เป็นประเภท Integer (จำนวนเต็ม) เพื่อจะได้ใส่ค่าข้อมูลของอุณหภูมิ (ตัวเลข)

int count_1 ;	ประกาศตัวแปร count_1 เป็นประเภท Integer (จำนวนเต็ม) เพื่อคำนวณตัวเลขของตัวแปรSD 1-6
int check_1 ;	ประกาศตัวแปร check_1 เป็นประเภท Integer (จำนวนเต็ม) เพื่อเช็คตัวเลขของตัวแปร SD 1-6 ว่าตรงตามเงื่อนไขหรือไม่

int ledPin = 8;
int ledPin2 = 9;
int ledPin3 = 10;
int ledPin4 = 11;
	ประกาศตัวแปร ledPin = 8,ledPin2 = 9, ledPin3 =10, ledPin4 = 11 เพื่อกำหนดขาพอร์ทของบอร์ดเพื่อต่อเข้า หลอด LED ของ DELAY ทั้งสี่ตัว

char ascii[32];	ประกาศตัวแปร ascii[32] เป็น Character เพื่อเก็บ frac
int frac;	ประกาศตัวแปร frac เป็น Integer เพื่อเก็บค่าที่คำนวน
frac=(int)(num*10)%10;	num คูณ 10 จากนั้นเป็น int แล้ว mod (หารเอาเฉพาะเศษ) ได้เศษเท่าไรเก็บไว้ที่ frac เอาตัวเลขหลังจุด 1 ตำแหน่ง
itoa((int)num,ascii,10);	num เป็น int หรืออาจจะเรียกว่าเอาค่าหลังจุดออกไป (กลายเป็นจำนวนเต็ม) จากนั้น ใช้ itoa เปลี่ยนค่าจำนวณเต็ม เป็น String ASCII และเก็บในตัวแปร ascii เลข 10 เป็นตัวบอก ฟังชัน itoa ว่า
strcat(ascii,".");	
itoa(frac,&ascii[strlen(ascii)],10);	เปลี่ยนค่า frac เป็น String แล้วจับต่อท้ายค่าที่มีอยู่แล้วใน ascii

```
return ascii;                                   ส่ง ascii ซึ่งตอนนี้เป็นข้อความตัวเลข
}
void setup()                                    เข้าสู่โหมด เซ็ต
{
Serial.begin(19200);                            เซ็ตความถี่ที่ 19200 ต่อ วินาที
Serial.println("Befor Set  Temparature");       พิมม์ Befor Set  Temparature เพื่อแจ้งให้รู้ว่า
                                                ก่อนเซ็ตอุณหภูมิมีค่าเท่าไหร่
SD1=   EEPROM.read(0) ;                         ประกาศ SD1 เท่ากับค่าความจำใน EEPROM(0)
SD2=   EEPROM.read(1) ;                         ประกาศ SD2 เท่ากับค่าความจำใน EEPROM(1)
SD3=   EEPROM.read(2) ;                         ประกาศ SD3 เท่ากับค่าความจำใน EEPROM(2)
SD4=   EEPROM.read(3) ;                         ประกาศ SD4 เท่ากับค่าความจำใน EEPROM(3)
SD5=   EEPROM.read(4) ;                         ประกาศ SD5 เท่ากับค่าความจำใน EEPROM(4)
SD6=   EEPROM.read(5) ;                         ประกาศ SD6 เท่ากับค่าความจำใน EEPROM(5)
Serial.print(">");                              พิมพ์แจ้งค่าตัวแปร (SD = ตัวเลขอุณหภูมิที่
Serial.print( SD1);                             ความจำ EEPROM)
Serial.print(".....");                          >SD1    < SD2
Serial.print( SD2);                             >SD3    <SD4
Serial.println("<");                           >SD5    <SD6
Serial.print(">");                              เพื่อให้รู้ว่าค่าที่เราเซ็ตอุณหภูมิไว้ที่แรกมีค่า
Serial.print( SD3);                             เท่าไหร่
```

```
Serial.print(".....");
Serial.print( SD4);
Serial.println("<");
Serial.print(">");
Serial.print( SD5);
Serial.print(".....");
Serial.print( SD6);
Serial.println("<");
pinMode(ledPin3, OUTPUT);
pinMode(ledPin4, OUTPUT);
}
void loop(void)
{
tem_tem1 :
while(1)
{
byte data[9];
int temp;

if (Serial.available() > 0)
```

พิมพ์แจ้งค่าตัวแปร (SD = ตัวเลขอุณหภูมิที่ความจำ EEPROM)

>SD1 < SD2

>SD3 <SD4

>SD5 <SD6

เพื่อให้รู้ว่าค่าที่เราเซ็ตอุณหภูมิไว้ที่แรกมีค่าเท่าไหร่

เข้าสู่โหมดลูปเพื่อ กด S เพื่อเซ็ตอุณหภูมิ

เข้าโหมดอุณหภูมิ

ใช้หน่วยความจำ 9 ไบต์ ในการจำค่าอุณหภูมิ

ตั้งค่าตัวแปร temp เป็นประเภท Integer เพื่อเก็บค่าอุณหภูมิ

ถ้า ascii > 0

```
Serial.println("Before Set  Temperature");     พิมพ์ Before Set  Temperature ขึ้นบรรทัดใหม่

Serial.println("----------------------");       พิมพ์ ---------- และขึ้นบรรทัดใหม่

Serial.println("  Temperature  Start ");        พิมพ์ Temperature Start  และขึ้นบรรทัดใหม่

StartSerial.println("....................");    พิมพ์ ..... และขึ้นบรรทัดใหม่

Serial.println("S For set Temparature");       พิมพ์ S For set Temparature เพื่อบอกว่าให้กด เพื่อ
                                               เข้าสู่ โหมด เซ็ต อุณหภูมิ และขึ้นบรรทัดใหม่

Serial.print(">");                             พิมพ์ > เพื่อบอกให้รู้ว่าคุณกำลังเข้าสู่ การทำงาน

pinMode(ledPin, OUTPUT);                       }
pinMode(ledPin2, OUTPUT);                      }  สั่งให้ขาของบอร์ดที่ต่อกับ LED ของดีเลย์
                                               }  ทำงาน

{

ReadByte = Serial.read();                      }  ตั้งค่า ReadByte = ค่าที่รับจาก ascii และ ส่งค่า
Serial.print(ReadByte,BYTE);                   }  ReadByte ไว้ใน Byte


if ((ReadByte == 0x53)||(ReadByte =='S'))      ถ้าค่า ascii = S

{
Serial.println("  OK Mode SET ");              พิมพ์ OK Mode SET

Serial.println("  Please Enter Number  ");     พิมพ์  Please Enter Number

goto SET_1_1 ;                                 เข้าสู่โหมด เซ็ต DS 1820

}

}

if(ds.reset())                                 ถ้า DS 1820 reset

{
```

```
ds.write(0xCC);                    เขียนค่าจากหน่วยความจำ
ds.write(0x44);                    เริ่มการเปลี่ยนอุณหภูมิ
ds.reset();                        เปลี่ยนคำสั่งใหม่
ds.write(0xCC);                    เขียนค่าใส่หน่วยความจำ
ds.write(0xBE);                    เขียนค่าใส่หน่วยความจำชั่วคราว

for ( int i = 0; i<9; i++)         ใช้หน่วยความจำ 9 ไบต์สำหรับอ่านหน่วยความจำ
                                   ชั่วคราว

data[i] = ds.read();

}
if (ds.crc8(data,8) != data[8])    พิสูจน์ว่าทั้ง 9 ไบต์ ของ data 0-8 ถ้าไม่เท่ากันให้
{                                  พิมพ์คำว่า Read Device CRC.....Error
Serial.println("Read Device CRC.....Error");
}
else                               ถ้าไม่ใช่

for ( int i=0; i<9; i++)           ใช้พื้นที่ 9ไบต์ สำหรับพิมพ์หรือแสดง
{
if (data[i] < 0x10)                ถ้า data[i] น้อยกว่า 10
{
}
}
Serial.print(" Temperature = ");   พิมพ์ Temperature =
temp=(data[1]<<8)+data[0];         ถ้า data[1]ไม่เท่า 0 (เพื่อตรวจสอบค่าของอุณหภูมิว่า
                                   เป็น + หรือ -)
if(data[1]!=0x00)
```

```
{
temp = (~temp)+1;                          ให้กลับค่า temp แล้ว+1
}
else                                       ถ้าไม่
{
Serial.print("+");                         พิมพ์ +
}
temp =  ((temp*0.05)*-1)   ;               นำ temp ไปคูณ0.05แล้วคูณด้วย-1

Serial.print(DoubleToAscii((double)temp));  พิมพ์ temp เป็นตัวเลขประเภท double
Serial.println(" C");                       พิมพ์ C เพื่อแสดงเป็น สัญลักษณ์ องศา

if(temp<0)                                 ถ้า temp < 0 ให้
{
digitalWrite(ledPin, HIGH);                LED0 ของดีเลย์มีสถานะเป็น HIGH (ดับ)
digitalWrite(ledPin2, HIGH);               LED2 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
digitalWrite(ledPin3, HIGH);               LED3 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
digitalWrite(ledPin4, HIGH);               LED4 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
}
else if((temp>=0))                         ถ้า temp >=0  ให้
{
digitalWrite(ledPin, LOW);                 LED0 ของดีเลย์มีสถานะเป็น LOW(ติด)
digitalWrite(ledPin2, HIGH);               LED2 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
digitalWrite(ledPin3, HIGH);               LED3 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
digitalWrite(ledPin4, HIGH);               LED4 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
}
```

```
else if((temp>=SD1) )
{
digitalWrite(ledPin, HIGH);
digitalWrite(ledPin2, LOW);
digitalWrite(ledPin3, HIGH);
digitalWrite(ledPin4, HIGH);
}
```

ถ้า temp >=SD1 ให้
LED0 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
LED2 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
LED3 ของดีเลย์มีสถานะเป็น LOW(ติด)
LED4 ของดีเลย์มีสถานะเป็น HIGH(ดับ)

```
else if((temp>=SD2) )
{
digitalWrite(ledPin, HIGH);
digitalWrite(ledPin2, HIGH);
digitalWrite(ledPin3, LOW);
digitalWrite(ledPin4, HIGH);
}
```

ถ้า temp >=SD2 ให้
LED0 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
LED2 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
LED3 ของดีเลย์มีสถานะเป็น LOW(ติด)
LED4 ของดีเลย์มีสถานะเป็น HIGH(ดับ)

```
else if((temp>=SD3)
{
digitalWrite(ledPin, HIGH);
digitalWrite(ledPin2, HIGH);
digitalWrite(ledPin3, HIGH);
digitalWrite(ledPin4, LOW);
 }
else
{
}
}
}
```

ถ้า temp >=SD3 ให้
LED0 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
LED2 ของดีเลย์มีสถานะเป็น HIGH(ดับ)
LED3 ของดีเลย์มีสถานะเป็น LOW(ติด)
LED4 ของดีเลย์มีสถานะเป็น HIGH(ดับ)

```
 else                                          ถ้าไม่ใช่  ให้พิมพ์ Search Device...Not
{                                              Found (ในกรณีDS1820ติดต่อกับบอร์ด
Serial.println("Search Device...Not Found");   ไม่ได้)
}
delay(800);                                    ตั้งค่าดีเลย์ ประมาณ 1วินาที เพื่อให้บอร์ดทำงาน
}
SET_1_1 :                                      เข้าสู่โหมด check

while(1)                                       ในขณะที่
}

if (Serial.available() > 0)                    ถ้า ascii > 0 ให้ส่งค่า(พิมพ์) ascii
{                                              ในช่องของไบต์
ReadByte = Serial.read();
Serial.print(ReadByte,BYTE);

if (ReadByte == 0x0D)                          ถ้า ReadByte = 0 ถึง 7
{                                              ถ้า ReadByte =7 ให้ส่งค่าออก
if(check_1 ==7)                                หน้าจอว่า Temparature  Start
{                                              และถ้า ReadByte = 0 ให้กลับไปสู่
Serial.println("  Temparature  Start ");       วนฟังก์ชั่น temp_temp1

check_1 = 0 ;
goto  tem_tem1 ;
}
```

```
if(check_1 ==6)
{
EEPROM.write(0,SD1);
EEPROM.write(1,SD2);
EEPROM.write(2,SD3);
EEPROM.write(3,SD4);
 EEPROM.write(4,SD5);
EEPROM.write(5,SD6);
delay(300) ;
Serial.println("OK.");
Serial.print(">");
Serial.print( SD1);
Serial.print(".....");
Serial.print( SD2);
Serial.println("<");
 Serial.print(">");
Serial.print( SD3);
Serial.print(".....");
Serial.print( SD4);
Serial.println("<");
Serial.print(">");
Serial.print( SD5);
Serial.print(".....");
Serial.print( SD6);
Serial.println("<");
```

ถ้า ReadByte = 6

ให้พิมพ์SD1จากหน่วยความจำ EEPROM 0

ให้พิมพ์SD2จากหน่วยความจำ EEPROM 1

ให้พิมพ์SD3จากหน่วยความจำ EEPROM 2

ให้พิมพ์SD4จากหน่วยความจำ EEPROM 3

ให้พิมพ์SD5จากหน่วยความจำ EEPROM 4

ให้พิมพ์SD6จากหน่วยความจำ EEPROM 5

ดีเลย์ 300

พิมพ์ OK แล้วขึ้นบรรทัดใหม่

พิมพ์ >SD1

>SD2

>SD3

```
SD1=   EEPROM.read(0) ;
SD2=   EEPROM.read(1) ;
SD3=   EEPROM.read(2) ;
SD4=   EEPROM.read(3) ;
SD5=   EEPROM.read(4) ;
SD6=   EEPROM.read(5) ;
count_1 = 0 ;
check_1 = check_1+ 1 ;
Serial.println("Enter  For  Next 2 ");
}
```

ให้ EEPROM0 เก็บค่า SD1 ไว้
ให้ EEPROM1 เก็บค่า SD2 ไว้
ให้ EEPROM2 เก็บค่า SD3 ไว้
ให้ EEPROM3 เก็บค่า SD4 ไว้
ให้ EEPROM4 เก็บค่า SD5 ไว้
ให้ EEPROM5 เก็บค่า SD6 ไว้
แล้วตรวจดูอีกรอบแล้ว เพิ่ม ReadByte อีก 1
แล้วพิมพ์ Enter For Next 2 เพื่อเข้าไปสู่ขั้นตอน
แจ้งที่เราเซ็ตตัวเลขอุณหภูมิ

```
if(check_1 ==5)
{
delay(300) ;
SD6 = atof(mem5);
Serial.println( SD6);
count_1 = 0 ;
check_1 = check_1+ 1 ;
Serial.println("Enter  For  Next 1 ");

}
```

ถ้า ReadByte = 5 แล้วให้ดีเลย์ 300
แล้วเก็บ SD6 เก็บไว้ที่ mem5
พิมพ์ SD 6 แล้วตรวจเช็ค แล้วเพิ่ม
ReadByte อีก 1

```
if(check_1 ==4)
{
delay(300) ;
SD5 = atof(mem4);
Serial.println( SD5);
count_1 = 0 ;
check_1 = check_1+ 1 ;
 }
```

ถ้า ReadByte = 4 แล้วให้ดีเลย์ 300
แล้วเก็บ SD5 เก็บไว้ที่ mem4
พิมพ์ SD 5 แล้วตรวจเช็ค แล้วเพิ่ม
ReadByte อีก 1

```
if(check_1 ==3)
{
delay(300) ;
SD4 = atof(mem3);
Serial.println( SD4);
count_1 = 0 ;
check_1 = check_1+ 1 ;
 }
```

ถ้า ReadByte = 3 แล้วให้ดีเลย์ 300 แล้วเก็บ SD4 เก็บไว้ที่ mem3 พิมพ์ SD 4 แล้วตรวจเช็ค แล้วเพิ่ม ReadByte อีก 1

```
if(check_1 ==2)
{
delay(300) ;
SD3 = atof(mem2);
 Serial.println( SD3);
count_1 = 0 ;
check_1 = check_1+ 1 ;
}
```

ถ้า ReadByte = 2 แล้วให้ดีเลย์ 300 แล้วเก็บ SD3 เก็บไว้ที่ mem2 พิมพ์ SD 3 แล้วตรวจเช็ค แล้วเพิ่ม ReadByte อีก 1

```
if(check_1 ==1)
{
delay(300) ;
SD2 = atof(mem1);
Serial.println( SD2);
count_1 = 0 ;
check_1 = check_1+ 1 ;
}
```

ถ้า ReadByte = 1 แล้วให้ดีเลย์ 300 แล้วเก็บ SD2 เก็บไว้ที่ mem1 พิมพ์ SD 2 แล้วตรวจเช็ค แล้วเพิ่ม ReadByte อีก 1

```
if(check_1 ==0)
{
delay(300) ;
SD1 = atof(mem0);
 Serial.println( SD1);
count_1 = 0 ;
check_1 = check_1+ 1 ;

}
}
```

ถ้า ReadByte = 0 แล้วให้ดีเลย์ 300 แล้วเก็บ SD1 เก็บไว้ที่ mem0 พิมพ์ SD 1 แล้วตรวจเช็ค แล้วเพิ่ม ReadByte อีก 1

```
if(check_1 ==0)
{
mem0[count_1] = ReadByte ;
count_1 = count_1+1 ;
}
if(check_1 ==1)
{
mem1[count_1] = ReadByte ;
count_1 = count_1+1 ;
}
if(check_1 ==2)
{
mem2[count_1] = ReadByte ;
count_1 = count_1+1 ;
}
```

ถ้า ReadByte = 0 ให้ เก็บไว้ที่ mem0[count_1]แล้วบวกค่า count_1 อีก 1 เพื่อให้ไปวนลูป SET_1_1 ขึ้นไป

ถ้า ReadByte = 1 ให้ เก็บไว้ที่ mem1 [count_1]แล้วบวกค่า count_1 อีก 1 เพื่อให้ไปวนลูป SET_1_1 ขึ้นไป

ถ้า ReadByte = 2 ให้ เก็บไว้ที่ mem2 [count_1] แล้วบวกค่า count_1 อีก 1 เพื่อให้ไปวนลูป SET_1_1 ขึ้นไป

```
if(check_1 ==3)
{
mem3[count_1] = ReadByte ;
count_1 = count_1+1 ;
}
```

ถ้า ReadByte = 3 ให้ เก็บไว้ที่ mem3[count_1]แล้วบวกค่า count_1 อีก 1 เพื่อให้ไปวนลูป SET_1_1 ขึ้นไป

```
if(check_1 ==4)
{
mem4[count_1] = ReadByte ;
count_1 = count_1+1 ;
}
```

ถ้า ReadByte = 4 ให้ เก็บไว้ที่ mem4[count_1]แล้วบวกค่า count_1 อีก 1 เพื่อให้ไปวนลูป SET_1_1 ขึ้นไป

```
if(check_1 ==5)
{
mem5[count_1] = ReadByte ;
count_1 = count_1+1 ;
}
```

ถ้า ReadByte = 5 ให้ เก็บไว้ที่ mem5[count_1]แล้วบวกค่า count_1 อีก 1 เพื่อให้ไปวนลูป SET_1_1 ขึ้นไป

```
if(check_1 ==6)
{
mem6[count_1] = ReadByte ;
count_1 = count_1+1 ;
}
}
}
}
```

ถ้า ReadByte = 6 ให้ เก็บไว้ที่ mem6[count_1]แล้วบวกค่า count_1 อีก 1 เพื่อให้ไปวนลูป SET_1_1 ขึ้นไป

เนื่องจากเป็นบอร์ดของ AVR จึงใช้โค้ดคำสั่งของ AVR ที่ติดต่อกับบอร์ดโดยตรงเพื่อให้ง่ายต่อการติดต่อกับบอร์ดและเมื่อได้โปรแกรมแล้วก็ทำการโหลดโปรแกรมลงบอร์ด

ประวัติผู้ดำเนินโครงงาน

ชื่อ นายสุธีย์ หม่องมูล
ภูมิลำเนา 113/1 หมู่ 1 ต.กลางดง อ. ทุ่งเสลี่ยม จ. สุโขทัย
ประวัติการศึกษา

- จบการศึกษาจากโรงเรียนทุ่งเสลี่ยมชนูปถัมภ์
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 5 สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

Email: sutee83@hotmail.com

ชื่อ นายพรนิมิตร แก้วเพียร
ภูมิลำเนา 241 หมู่ 8 ต. ศิลา อ. หล่มเก่า จ. เพชรบูรณ์
ประวัติการศึกษา

- จบการศึกษาจากโรงเรียนหล่มเก่าพิทยาคม
- ปัจจุบันกำลังศึกษาในระดับปริญญาตรีชั้นปีที่ 5 สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

Email: m_koby02@hotmail.com