เครื่องควบคุมอุณหภูมิแบบพีไอดี

TEMPERATURE CONTROL WITH PID

นายนพพล งามนิล รหัส 49364127

นายพิเชฐ สุขศรีศักดิ์สิทธิ์ รหัส 49364172

นายสุรชัย แสวงสาย รหัส 49364349

2554

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2554

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน เครื่องควบคุมอุณหภูมิแบบพีไอดี

ผู้ดำเนินโครงงาน นายนพพล งามนิล รหัส 49364127

นายพิเชฐ สุขศรีศักดิ์สิทธิ์ รหัส 49364172

นายสุรชัย แสวงสาย รหัส 49364349

ที่ปรึกษาโครงงาน นายเศรษฐา ตั้งค้าวานิช

สาขาวิชา วิศวกรรมไฟฟ้า

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2554

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

เศรษฐา ตั้งค้าวานิช .....ที่ปรึกษาโครงงาน

(นายเศรษฐา ตั้งค้าวานิช)

.....กรรมการ

(ดร.มุฑิตา สงฆ์จันทร์)

.....กรรมการ

(ดร.ศุภวรรณ พลพิทักษ์ชัย)

ชื่อหัวข้อโครงงาน	เครื่องปรับอากาศรักษาความชื้น		
ผู้ดำเนินโครงงาน	นายนพพล	งามนิล	รหัส 49364127
	นายพิเชฐ	สุขศรีศักดิ์สิทธิ์	รหัส 49364172
	นายสุรชัย	แสวงสาย	รหัส 49364349
ที่ปรึกษาโครงงาน	นายเศรษฐา	ตั้งค้าวานิช	
สาขาวิชา	วิศวกรรมไฟฟ้า		
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์		
ปีการศึกษา	2554		

บทคัดย่อ

โครงงานนี้เป็นการประดิษฐ์เครื่องต้มน้ำควบคุมอุณหภูมิแบบปรับค่าได้ โดยการควบคุมการทำงานของขดลวดทำความร้อนแบบพีไอดี (PID CONTROLLER) และศึกษาทฤษฎีของการควบคุมแบบพีไอดี ในการปรับตั้งค่าพารามิเตอร์ของการควบคุมแบบพีไอดี โดยวิธีการปรับจูนค่าเกนของตัวควบคุมพีไอดีสำหรับระบบที่ไม่ทราบแบบจำลองทางคณิตศาสตร์ หรือ ไม่ทราบสมการระบบ ซึ่งสามารถปรับตั้งค่าพารามิเตอร์ Kp, Ki และKd ได้โดยการทดลองควบคุมอุณหภูมิของน้ำ เพื่อทำให้เครื่องต้มน้ำควบคุมอุณหภูมิมีความสามารถในการแก้ไขค่าความผิดพลาดของระบบ คือ ลดค่าโอเวอร์ชูต (overshoots) และ ลดค่าการแกว่งของระบบ (oscillation) ให้เหลือน้อยที่สุด โดยมีเนื้อหาการศึกษาทดลองเกี่ยวกับการนำขดลวดความร้อนมาพัฒนาให้มีความสามารถตั้งค่าอุณหภูมิได้ตั้งแต่อุณหภูมิเริ่มต้นของน้ำที่นำมาต้มจนถึง 100 องศาเซลเซียส จากการทดลองเครื่องต้มน้ำควบคุมอุณหภูมิมีความสามารถในการต้มน้ำให้ได้อุณหภูมิตามที่ตั้งค่าไว้โดยมีค่าความผิดพลาดในสถานะอยู่ตัวประมาณ 5 เปอร์เซ็นต์

Project title	Temperature Control with PID		
Name	Mr. Nopphon	Ngamnil	ID. 49364127
	Mr. Pichet	Suksrisaksit	ID. 49364172
	Mr. Surachai	Sawaengsai	ID. 49364349
Project advisor	Mr. Settha Thangkawanit		
Major	Electrical Engineering		
Department	Electrical and Computer Engineering		
Academic year	2011		

.....

Abstract

This project is an invention of water boiler with temperature controllable, Including a study of how to control PID to operated boiling machine by tuning parameters in PID controller for a system without mathematical model or equation. To minimize the oscillation and overshoot value, we are focusing on how to adjust Kp, Ki and Kd parameter through the experiment subject, which is water. Based on developing heat coil to adjust temp value between currently temp to one-hundred Celsius degree. The trial and study was come to conclusion with temperature controllable boiling machine and error value at 5 percent.

กิตติกรรมประกาศ

โครงงานนี้สำเร็จลุล่วงไปได้ด้วยความกรุณาเป็นอย่างยิ่งจาก อ. เศรษฐา ตั้งค้าวานิช ซึ่งเป็นอาจารย์ที่ปรึกษาโครงงาน และให้ความกรุณาในการตรวจทานปริญญานิพนธ์ คณะผู้ดำเนินโครงงานขอขอบพระคุณเป็นอย่างสูงและขอระลึกถึงความกรุณาของท่านไว้ตลอดไป

ขอขอบคุณคณาจารย์ทุกท่านที่ประสิทธิ์ประสาทวิชาความรู้ให้กับคณะผู้ดำเนินงาน

นอกจากนี้คณะผู้ดำเนินงานขอขอบคุณภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ ที่ให้ยืมอุปกรณ์มาใช้ทดลองจนทำให้โครงงานนี้สำเร็จลุล่วง

เหนือสิ่งอื่นใด คณะผู้ดำเนินโครงงานขอกราบขอบพระคุณ คุณพ่อ คุณแม่ ผู้มอบความรักความเมตตา สติปัญญา รวมทั้งเป็นผู้ให้ทุกสิ่งทุกอย่างตั้งแต่เยาว์วัยจวบจนปัจจุบัน ซึ่งคอยเป็นทั้งผู้ส่งเสริมและให้กำลังใจจนได้รับความสำเร็จในทุกวันนี้ และขอขอบคุณทุกๆคนในครอบครัวของคณะผู้ดำเนินโครงงาน รวมถึงผู้ที่คอยช่วยเหลือคณะผู้ดำเนินงานที่ไม่ได้กล่าวไว้ ณ ที่นี้ด้วย

นายนพพล งามนิล

นายพิเชฐ สุขศรีศักดิ์สิทธิ์

นายสุรชัย แสวงสาย

สารบัญ

หน้า

สารบัญรูป

รูปที่ หน้า

สารบัญรูป (ต่อ)

รูปที่ หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

เครื่องควบคุมอุณหภูมิแบบพีไอดี (PID CONTROLLER) ที่สามารถตั้งค่าความร้อนได้ สามารถนำมาใช้ในงานหลายประเภท เช่นใช้ในห้องทดลอง ใช้ในการควบคุมอุณหภูมิของสารต่างๆให้คงที่ในระหว่างการแยกสาร การผสมสารบางชนิดหรือใช้ในการแยกน้ำออกจากน้ำมันที่มีการรักษาอุณหภูมิที่ต้องการใช้ได้ดีและมีประสิทธิภาพ ซึ่งมีราคาค่อนข้างสูงและหาซื้อยาก

การควบคุมแบบพีไอดี (PID CONTROLLER) หมายถึงระบบควบคุมแบบสัดส่วนปริพันธ์-อนุพันธ์ การควบคุมแบบพีไอดีเป็นระบบควบคุมแบบป้อนกลับที่ใช้กันอย่างกว้างขวาง ซึ่งค่าที่นำไปใช้ในการคำนวณเป็นค่าความผิดพลาดที่หามาจากความแตกต่างของตัวแปรในกระบวนการและค่าที่ต้องการตัวควบคุมจะพยายามลดค่าผิดพลาดให้เหลือน้อยที่สุดด้วยการปรับค่าสัญญาณขาเข้าของกระบวนการ ค่าตัวแปรของพีไอดีคอนโทรลเลอร์ที่ใช้จะปรับเปลี่ยนตามธรรมชาติของระบบวิธีคำนวณของพีไอดีคอนโทรลเลอร์ขึ้นอยู่กับสามตัวแปรคือ ค่าสัดส่วนซึ่งกำหนดจากผลของความผิดพลาดในปัจจุบัน ค่าปริพันธ์ซึ่งกำหนดจากผลบนพื้นฐานของผลรวมความผิดพลาดที่ซึ่งพึ่งผ่านพ้นไป และค่าอนุพันธ์กำหนดจากผลบนพื้นฐานของอัตราการเปลี่ยนแปลงของค่าความผิดพลาด น้ำหนักที่เกิดจากการรวมกันของทั้งสามนี้จะใช้ในการปรับกระบวนการ โดยการปรับค่าคงที่ใน การควบคุมแบบพีไอดีตัวควบคุมสามารถปรับรูปแบบการควบคุมให้เหมาะกับกระบวนการที่ต้องการได้ การตอบสนองของตัวควบคุมจะอยู่ในรูปของการไหวตัวของตัวควบคุมจนถึงค่าความผิดพลาดของโอเวอร์ชูต (overshoots) และค่าแกว่งของระบบ (oscillation) ซึ่งวิธีพีไอดีคอนโทรลเลอร์นี้ไม่รับประกันได้ว่าจะเป็นระบบควบคุมที่เหมาะสมที่สุดหรือสามารถทำให้กระบวนการมีความเสถียรแน่นอน

ระบบควบคุมที่ได้จัดทำขึ้นมาแบ่งออกเป็น2ส่วนหลัก โดยส่วนแรกจะเป็นส่วนควบคุมโดยจะมีไมโครคอนโทรลเลอร์ (ATMEGA 32) เป็นส่วนประกอบหลัก ใช้เซ็นเซอร์วัดอุณหภูมิ DS1820 และมีส่วนของวงจรเกี่ยวเนื่องกับระบบควบคุม ในส่วนที่สองจะเป็นอุปกรณ์ทำความร้อนโดยใช้ขดลวดความร้อนขนาด 900 วัตต์

ผู้จัดทำโครงงานสนใจในการนำขดลวดความร้อนมาพัฒนาให้มีความสามารถตั้งค่าอุณหภูมิ และเพิ่มการควบคุมแบบพีไอดีคอนโทรลเลอร์ ฉะนั้นเครื่องควบคุมอุณหภูมิที่ได้จัดทำ

ขึ้น จึงมีความสามารถในการตั้งค่าของอุณหภูมิที่แม่นยำและมีความคงที่ ทำให้ผู้จัดทำสามารถนำเครื่องควบคุมอุณหภูมิไปใช้ในการกำหนดอุณหภูมิของน้ำในรูปแบบต่างๆที่กล่าวมาข้างต้นได้อย่างมีประสิทธิภาพ อีกทั้งราคายังไม่แพง และสามารถนำไปใช้ประโยชน์ในห้องทดลองต่างๆได้อีกด้วย

1.2 วัตถุประสงค์ของโครงงาน

เพื่อประดิษฐ์เครื่องต้มน้ำควบคุมอุณหภูมิแบบปรับค่าได้ โดยการควบคุมแบบพีไอดี (PID Controller)

1.3 ขอบเขตของโครงงาน

1) ศึกษาระบบควบคุมแบบพีไอดี (PID Controller)
2) สร้างวงจรควบคุมโดยใช้ไมโครคอนโทรลเลอร์ (ATmega32)ตระกูล AVR
3) เขียนคำสั่งควบคุมระบบไมโครคอนโทรลเลอร์ด้วยโปรแกรมภาษาซี
4) สร้างเครื่องต้มน้ำควบคุมอุณหภูมิแบบปรับค่าอุณหภูมิ,ปรับตั้งค่าพารามิเตอร์ Kp, Ki และKd ได้ ซึ่งสามารถปรับอุณหภูมิได้ตั้งแต่ 10 – 100 องศาเซลเซียสในปริมาณน้ำ 100 – 1000 มิลลิลิตร

1.4 ขั้นตอนและแผนการดำเนินงาน

รายละเอียด	ปี 2553				ปี 2554				
	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.
1) รวบรวมข้อมูล									
2) ศึกษาการทำงานและออกแบบชิ้นงาน									
3) ประดิษฐ์ชิ้นงานและทดสอบการทำงาน									
4) ดำเนินการวิเคราะห์ผลที่ได้จากการปรับค่าพารามิเตอร์									
5) สรุปผลการดำเนินงาน									
6) จัดทำปริญญานิพนธ์ฉบับสมบูรณ์									

1.5 ประโยชน์ที่คาดว่าจะได้รับจากโครงงาน

1) เข้าใจหลักการทำงานของไมโครคอนโทรลเลอร์

2) สามารถสร้างเครื่องต้มน้ำควบคุมอุณหภูมิแบบพีไอดีคอนโทรล

3) สามารถปรับค่าอุณหภูมิในการต้มน้ำให้อยู่ในระดับอุณหภูมิตั้งแต่ 10 – 100 องศาเซลเซียส

4) สามารถทำการปรับค่าพารามิเตอร์ Kp, Ki และKdที่ทำให้ค่าความผิดพลาดของโอเวอร์ชูต (overshoots) และค่าแกว่งของระบบ (oscillation) น้อยที่สุด

1.6 งบประมาณ

1) ค่าอุปกรณ์ไฟฟ้าและอิเล็กทรอนิกส์	2,500 บาท
2) ค่าเอกสาร	300 บาท
3) ค่าวัสดุอื่นๆ	200 บาท
รวมเป็นเงินทั้งสิ้น (สามพันบาทถ้วน)	3,000 บาท

หมายเหตุ: ถัวเฉลี่ยทุกรายการ

บทที่ 2
หลักการและทฤษฎี

ในบทที่ 2 นี้จะอธิบายถึงหลักการและทฤษฎี ที่รวบรวมเพื่อนำมาใช้ประกอบการทำโครงงานโดยมีใจความสำคัญที่อธิบายถึงองค์ประกอบของอุปกรณ์ต่างๆและการทำงานของโปรแกรมรวมถึงทฤษฎีที่นำมาใช้งานควบคู่กับการเขียนโปรแกรม เพื่อให้ได้ผลงานออกมาตามที่ตั้งเป้าหมายไว้ตามจุดประสงค์ดังที่กล่าวไว้ในบทที่ 1

2.1 หลักการทำงานของกาต้มน้ำ

กาต้มน้ำไฟฟ้า เป็นเครื่องใช้ในการทำน้ำร้อนเพื่อใช้ประโยชน์ต่างๆ ภายในบ้านตามความต้องการ เช่น ชงเครื่องดื่ม ใช้ผสมน้ำอาบ ทำความสะอาดภาชนะ เป็นต้น กาต้มน้ำไฟฟ้าที่นิยมใช้ในปัจจุบัน มี 3 แบบ คือ แบบธรรมดา แบบอัตโนมัติ และแบบปล่อยน้ำด้วยแรงกดอากาศ แต่ที่เราเลือกนำมาใช้งานคือ กาต้มน้ำแบบธรรมดา กาต้มน้ำไฟฟ้าแบบธรรมดา มีส่วนประกอบเพียงอย่างเดียว คือ ลวดความร้อน หรือ ฮีทเตอร์ เป็นลวดนิโครมห่อด้วยผงแมกนีเซียมออกไซด์แล้วหุ้มด้วยท่อโลหะ ลวดความร้อนนี้จะขดโค้งอยู่ที่ก้นกา มีขั้วไฟฟ้า 2 ขั้ว ต่อออกมาข้างนอก เพื่อต่อกับปลั๊กทำให้น้ำร้อน เมื่อมีกระแสไฟฟ้าไหลผ่านลวดความร้อนจะทำให้ลวดความร้อนมีอุณหภูมิสูงขึ้นเรื่อยๆ และความร้อนจะถูกถ่ายเทไปให้กับน้ำจนอุณหภูมิของน้ำสูงสุดถึงจุดเดือด ถ้าน้ำเดือดแล้วยังไม่ดึงปลั๊กออกน้ำจะเดือดต่อไปเรื่อยๆจนระเหยเป็นไอหมด

ในทางปฏิบัติเส้นลวดที่นำมาใช้ทำเป็นขดลวดความร้อนนี้ ทำมาจากโลหะผสมระหว่างนิกเกิล ประมาณร้อยละ 60 โครเมียม ประมาณร้อยละ 15 และเหล็กประมาณร้อยละ 25 ได้โลหะใหม่ที่ เรียกว่า นิโครม เป็นโลหะที่มีความต้านทานประมาณ 50 เท่าของลวดทองแดง และจุดหลอมเหลวอยู่ 1,500 องศาเซลเซียส ลวดนิโครมที่ประกอบอยู่ในเครื่องใช้ไฟฟ้าอาจทำเป็นเส้น-ลวดหรือแถบทั้งนี้ขึ้นอยู่กับลักษณะการใช้งาน โดยทั่วไปจะนิยมทำเป็นลวดขดสปริง เพราะความยาวของขดลวดทำให้ขดลวดมีความต้านทานและความร้อนสูงขึ้นด้วย

2.1.1 อุณหภูมิ (Temperature)

อุณหภูมิ (Temperature) คือ การวัดค่าเฉลี่ยของพลังงานจลน์ของอนุภาคในสสารใดๆ ซึ่งสอดคล้องกับความร้อนหรือเย็นของสสารนั้น

ในอดีตมีแนวคิดเกี่ยวกับอุณหภูมิเกิดขึ้นเป็น 2 แนวทาง คือตามแนวทางของหลักอุณหพลศาสตร์ และตามการอธิบายเชิงจุลภาคทางฟิสิกส์เชิงสถิติ อุณหพลศาสตร์นั้นเกี่ยวข้องกับการวัดในเชิงมหภาค ดังนั้นคำจำกัดความอุณหภูมิในเชิงอุณหพลศาสตร์ในเบื้องแรก ซึ่งกำหนดขึ้นโดยลอร์ดเคลวิน จึงระบุเกี่ยวกับค่าตัวแปรต่างๆ ที่สามารถตรวจวัดได้จากการสังเกต ส่วนฟิสิกส์สถิติจะให้ความเข้าใจในเชิงลึกยิ่งกว่าอุณหพลศาสตร์ โดยอธิบายถึงการสะสมจำนวนอนุภาคขนาดใหญ่ และตีความพารามิเตอร์ต่างๆ ในอุณหพลศาสตร์ (เชิงมหภาค) ในฐานะค่าเฉลี่ยทางสถิติของพารามิเตอร์ของอนุภาคในเชิงจุลภาค

2.2 ไมโครคอนโทรลเลอร์ [6]

ไมโครคอนโทรลเลอร์ (อังกฤษ: Microcontroller) คือ อุปกรณ์ควบคุมขนาดเล็ก ซึ่งบรรจุความสามารถที่คล้ายคลึงกับระบบคอมพิวเตอร์ โดยในไมโครคอนโทรลเลอร์ได้รวมเอาซีพียู หน่วยความจำและพอร์ตซึ่งเป็นส่วนประกอบหลักสำคัญของระบบคอมพิวเตอร์เข้าไว้ด้วยกันโดยทำการบรรจุเข้าไว้ในตัวถังเดียวกัน

โครงสร้างโดยทั่วไปของไมโครคอนโทรลเลอร์นั้นสามารถแบ่งออกมาได้เป็น 5 ส่วนใหญ่ๆ ดังต่อไปนี้

1) หน่วยประมวลผลกลางหรือซีพียู (CPU: Central Processing Unit)

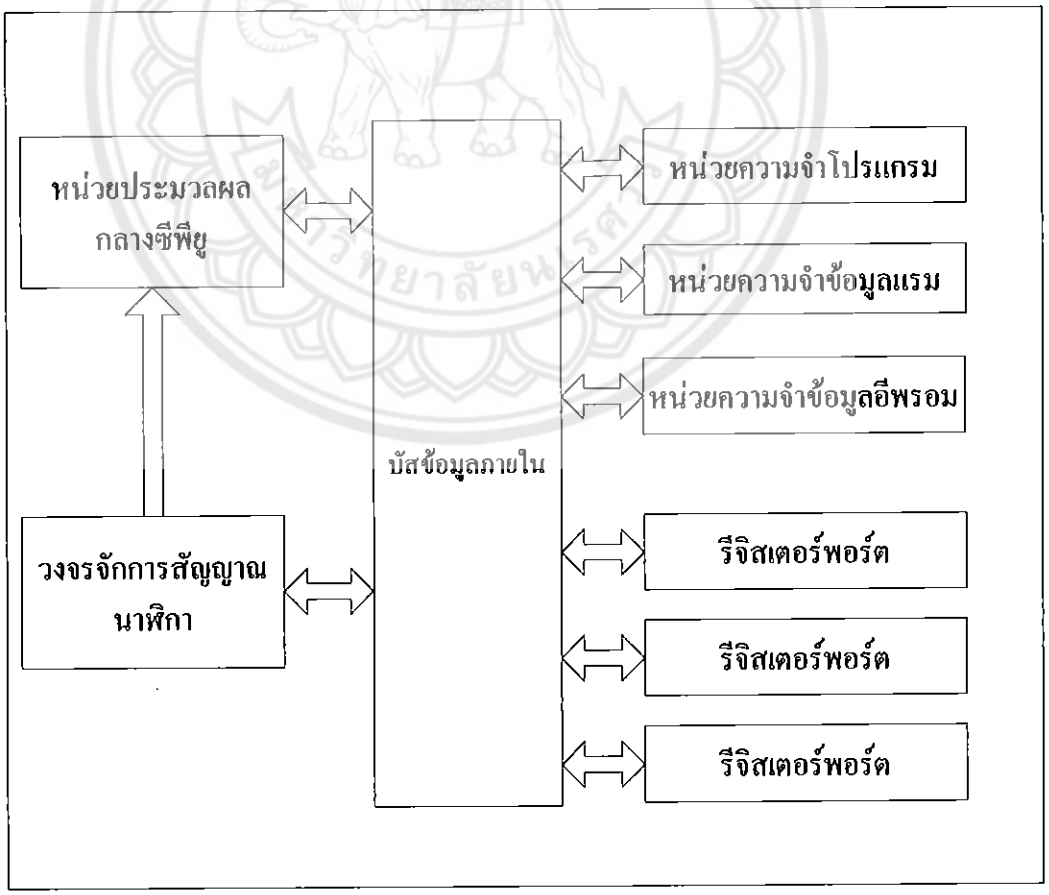
2) หน่วยความจำ (Memory) สามารถแบ่งออกเป็น 2 ส่วน คือ หน่วยความจำที่มีไว้สำหรับเก็บโปรแกรมหลัก (Program Memory) เปรียบเสมือนฮาร์ดดิสก์ของเครื่องคอมพิวเตอร์ตั้งโต๊ะ คือข้อมูลใดๆ ที่ถูกเก็บไว้ในนี้จะไม่สูญหายไปแม้ไม่มีไฟเลี้ยง อีกส่วนหนึ่งคือหน่วยความจำข้อมูล (Data Memory) ใช้เป็นเหมือนกระดาษทดในการคำนวณของซีพียู และเป็นที่พักข้อมูลชั่วคราวขณะทำงาน แต่หากไม่มีไฟเลี้ยง ข้อมูลก็จะหายไปคล้ายกับหน่วยความจำแรม (RAM) ในเครื่องคอมพิวเตอร์ทั่วๆ ไป แต่สำหรับไมโครคอนโทรลเลอร์สมัยใหม่ หน่วยความจำข้อมูลจะมีทั้งที่เป็นหน่วยความจำแรม ซึ่งข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยง และมีหน่วยความจำรอมเป็นอีอีพรอม (EEPROM : Erasable Electrically Read-Only Memory) ซึ่งสามารถเก็บข้อมูลได้แม้ไม่มีไฟเลี้ยง

3) ส่วนติดต่อกับอุปกรณ์ภายนอก หรือพอร์ต (Port) มี 2 ลักษณะคือ พอร์ตรับสัญญาณหรือพอร์ตอินพุต (Input Port) และพอร์ตส่งสัญญาณหรือพอร์ตเอาท์พุต (Output Port) ส่วนนี้จะใช้

ในการเชื่อมต่อกับอุปกรณ์ภายนอก ถือว่าเป็นส่วนที่สำคัญมาก ใช้ร่วมกันระหว่างพอร์ตอินพุต เพื่อรับสัญญาณ อาจจะด้วยการกดสวิตช์ เพื่อนำไปประมวลผลและส่งไปพอร์ตเอาท์พุต เพื่อแสดงผล เช่น การติดดับของหลอดไฟ เป็นต้น

4) ช่องทางเดินของสัญญาณ หรือบัส (Bus) คือเส้นทางการแลกเปลี่ยนสัญญาณข้อมูลระหว่าง ซีพียู หน่วยความจำและพอร์ต เป็นลักษณะของสายสัญญาณ จำนวนมากอยู่ภายในตัวไมโครคอนโทรลเลอร์ โดยแบ่งเป็นบัสข้อมูล (Data Bus), บัสแอดเดรส (Address Bus) และบัสควบคุม (Control Bus)

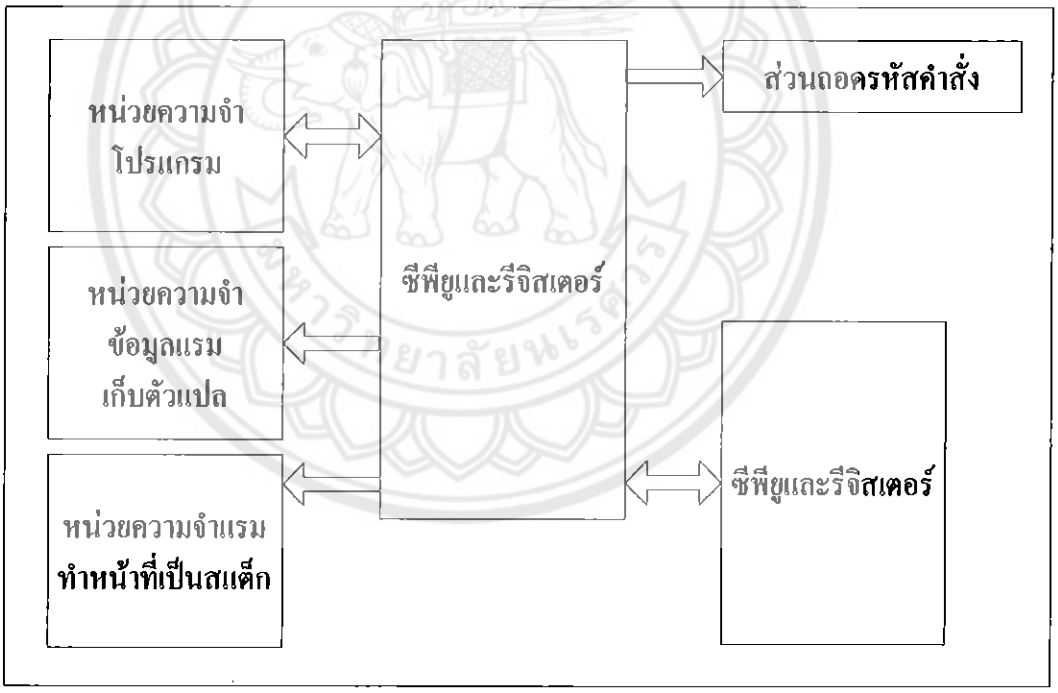
5) วงจรกำเนิดสัญญาณนาฬิกา นับเป็นส่วนประกอบที่สำคัญมากอีกส่วนหนึ่ง เนื่องจากการทำงานที่เกิดขึ้นในตัวไมโครคอนโทรลเลอร์ จะขึ้นอยู่กับการกำหนดจังหวะ หากสัญญาณนาฬิกามีความถี่สูงจังหวะการทำงานก็จะสามารถทำได้ถี่ขึ้นส่งผลให้ไมโครคอนโทรลเลอร์ตัวนั้นมีความเร็วในการประมวลผลสูงตามไปด้วยเนื่องด้วยการควบคุมในโครงงานนี้จะใช้การควบคุมระบบจากไมโครคอนโทรลเลอร์จึงขอกล่าวเกี่ยวกับระบบและทฤษฎีของไมโครคอนโทรลเลอร์ดังนี้



รูปที่ 2.1 โครงสร้างและส่วนประกอบหลักเบื้องต้นของไมโครคอนโทรลเลอร์ [6]

2.2.1 สถาปัตยกรรมของไมโครคอนโทรลเลอร์ [6]

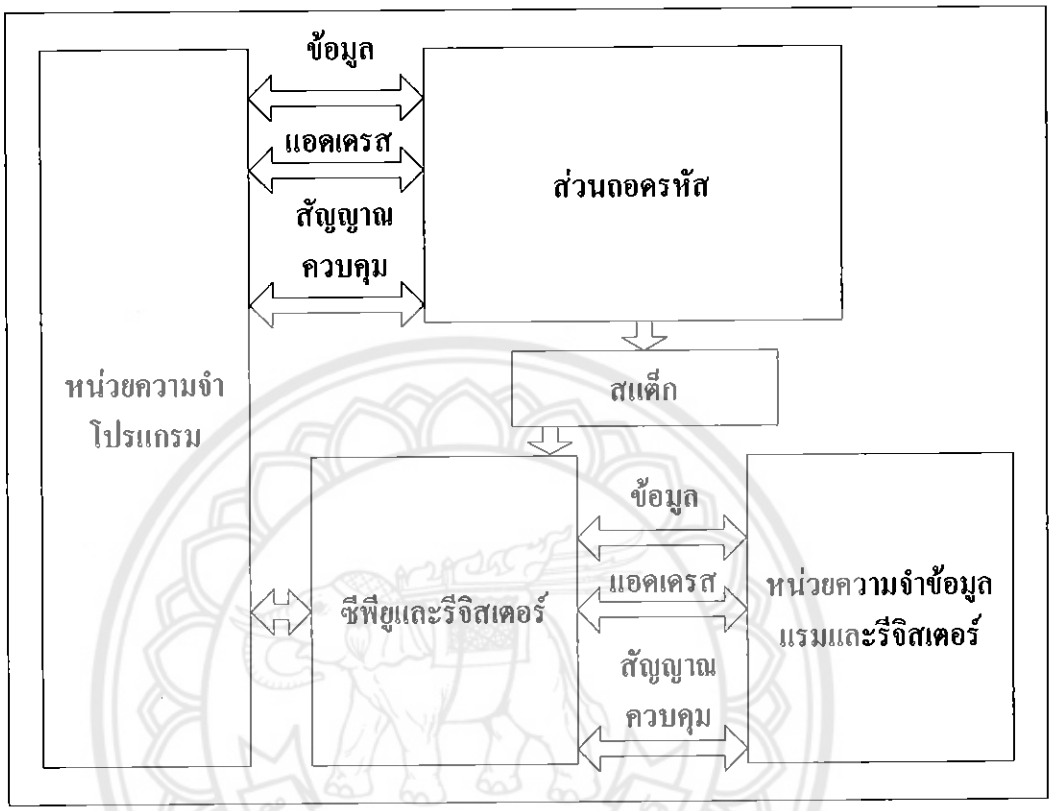
เป็นที่ยอมรับกันแพร่หลายว่าสถาปัตยกรรมของไมโครคอนโทรลเลอร์มีด้วยกัน 2 แบบ คือ พรินซ์ตัน (Princeton) หรือฟอนนิวแมน (Von Neumann) และฮาร์วาร์ด (Harvard) ในภาพที่ 2.2 และ 2.3 แสดงการจัดสรรหน่วยความจำและรีจิสเตอร์ในสถาปัตยกรรมของไมโครคอนโทรลเลอร์ทั้งสองแบบพิจารณาในภาพที่ 2-2 เป็นการจัดสรรในสถาปัตยกรรมแบบพรินซ์ตันจะเห็นได้ว่ามีโครงสร้างที่เรียบง่ายไม่ซับซ้อนส่วนของหน่วยความจำโปรแกรมกับหน่วยความจำข้อมูลจะได้รับการจัดสรรให้อยู่รวมกันติดต่อกับซีพียูผ่านส่วนจัดการเชื่อมต่อหน่วยความจำและภายในซีพียูจะมีรีจิสเตอร์บรรจุอยู่ข้อดีของสถาปัตยกรรมแบบนี้คือออกแบบง่ายเพราะหน่วยความจำทั้งหมดอยู่รวมกันสามารถเข้าถึงได้ง่ายหน่วยความจำมีขนาดใหญ่เพียงพอที่จะเก็บได้ทั้งโปรแกรมควบคุมการทำงานและข้อมูลของตัวแปรในการประมวลผลข้อด้อยของสถาปัตยกรรมนี้คือความเร็วในการประมวลผลเนื่องจากหน่วยความจำอยู่รวมกันจึงต้องติดต่อหน่วยความจำโปรแกรมสลับกับหน่วยความจำข้อมูลส่งผลให้ซีพียูต้องใช้จำนวนไซเคิลในการทำงานมาก



รูปที่ 2.2 โครงสร้างสถาปัตยกรรมของไมโครคอนโทรลเลอร์แบบพรินซ์ตันหรือฟอนนิวแมน [6]

ในขณะที่สถาปัตยกรรมแบบฮาร์วาร์ดซึ่งแสดงในภาพที่ 2.3 จะแยกส่วนของหน่วยความจำข้อมูลและรีจิสเตอร์ออกจากหน่วยความจำโปรแกรมทำให้ไซเคิลการทำงานลดลงเนื่องจากสามารถติดต่อหน่วยความจำโปรแกรมและหน่วยความจำข้อมูลได้เร็วกว่านอกจากนั้นใน

สถาปัตยกรรมนี้ในขณะที่ซีพียูกำลังเอ็กซิคิวต์คำสั่งปัจจุบันอยู่สามารถที่จะเฟตช์คำสั่งถัดไปได้ยิ่งทำให้ไมโครคอนโทรลเลอร์ทำงานได้เร็วขึ้น

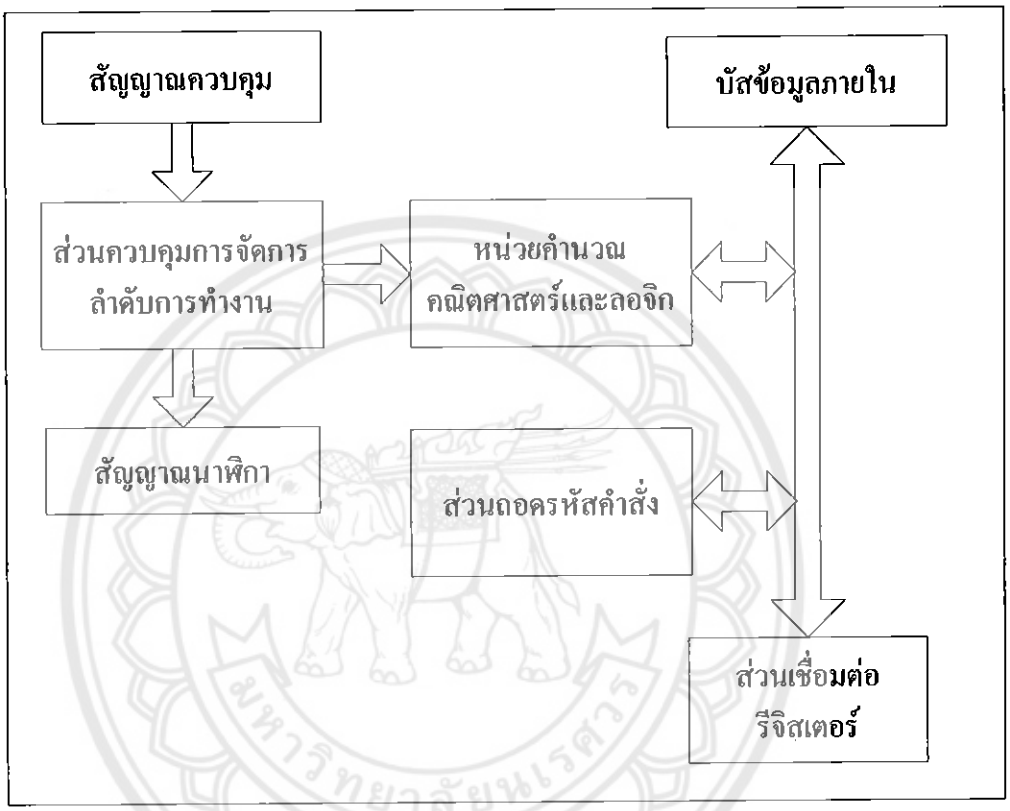


รูปที่ 2.3 โครงสร้างสถาปัตยกรรมของไมโครคอนโทรลเลอร์แบบฮาร์วาร์ด [6]

2.2.2 หน่วยประมวลผลกลางหรือซีพียู [6]

ซีพียูเป็นเสมือนมันสมองของไมโครคอนโทรลเลอร์ซึ่งทำหน้าที่ประมวลผลข้อมูลที่เข้ามาในระบบแล้วทำการส่งต่อไปยังส่วนต่างๆเพื่อควบคุมการทำงานต่อไปในภาพที่ 2.4 แสดงส่วนประกอบพื้นฐานของซีพียูในไมโครคอนโทรลเลอร์ทั่วๆไปจะเห็นได้ว่าหัวใจหลักของซีพียูคือหน่วยคำนวณทางคณิตศาสตร์และลอจิก (ALU: Arithmetic and Logic Unit) ซึ่งได้รับการกำหนดจังหวะการทำงานจากส่วนควบคุมลำดับการทำงานโดยจังหวะการทำงานนั้นจะสัมพันธ์กับสัญญาณนาฬิกาเมื่อซีพียูทำการติดต่อหน่วยความจำสิ่งที่ปรากฏขึ้นบนบัสข้อมูลภายในซีพียูคือรหัสคำสั่ง (Instruction Code) ซึ่งต้องผ่านการทำงานของส่วนถอดรหัสคำสั่ง (Instruction Decoder) เสียก่อนจะได้เป็นข้อมูลคำสั่งที่ซีพียูเข้าใจและสามารถดำเนินการต่อได้หลังจากที่หน่วยคำนวณทางคณิตศาสตร์และลอจิกประมวลผลแล้วก็ส่งข้อมูลมายังส่วนเชื่อมต่อรีจิสเตอร์ภายในซีพียูเพื่อติดต่อกับส่วนอื่นๆต่อไป

การทำงานของซีพียูมีด้วยกัน 2 จังหวะเฟตช์ (Fetch) และเอ็กซิคิวต์ (Executed) โดยการทำงานจะเริ่มจากการเฟตช์คือการเรียกหรือการเข้าถึงคำสั่งแล้วทำการถอดรหัสเป็นภาษาเครื่องเพื่อเตรียมการประมวลผลจากนั้นจะเป็นจังหวะของการเอ็กซิคิวต์คือการกระทำตามคำสั่งที่กำหนดให้จนเสร็จสิ้น



รูปที่ 2.4 ส่วนประกอบหลักของซีพียูในไมโครคอนโทรลเลอร์ [6]

2.2.3 หน่วยความจำ [6]

ในไมโครคอนโทรลเลอร์จะประกอบด้วยหน่วยความจำ 3 แบบคือหน่วยความจำโปรแกรม (Program Memory) หน่วยความจำข้อมูลแรมและหน่วยความจำข้อมูลอีอีพรอม

1) หน่วยความจำโปรแกรม

หน่วยความจำโปรแกรมเป็นที่สำหรับเก็บข้อมูลคำสั่งของโปรแกรมควบคุมที่ผู้พัฒนาเขียนขึ้นหรือเรียกว่าโปรแกรมมอนิเตอร์ (Monitor Program) ซีพียูจะเข้ามาติดต่อเพื่ออ่านข้อมูลรหัสคำสั่งจากหน่วยความจำในส่วนนี้แล้วนำไปประมวลผลเพื่อควบคุมการทำงานของระบบทั้งหมดต่อไปหน่วยความจำโปรแกรมนี้มักมีขนาดใหญ่ยิ่งมีขนาดมากเท่าใดก็จะสามารถบรรจุ

โปรแกรมที่มีความซับซ้อนหรือสามารถเก็บตารางข้อมูลที่ใช้ในการประมวลผลได้มากตามไปด้วย โดยทั่วไปมีความจุไม่น้อยกว่า 512 ไบต์ขนาดของหน่วยความจำโปรแกรมจะแปรตามความก้าวหน้าทางเทคโนโลยีชนิดของหน่วยความจำโปรแกรมที่ใช้ในไมโครคอนโทรลเลอร์ที่นิยมมีด้วยกันอยู่3 แบบ คือ แบบอีพรอม (EPROM Erasable Programmable Read-Only Memory) แบบอีอีพรอม (EEPROM: Electrically Erasable Programmable Read-Only Memory) และแบบแฟลช (Flash Memory) ความแตกต่างของทั้ง 3 แบบอยู่ที่จำนวนครั้งในการลบและเขียนข้อมูลทับลงไปใหม่โดยสามารถสรุปได้ดังนี้แบบอีพรอมแบ่งออกเป็น 2 แบบคือแบบโปรแกรมได้หลายครั้งและแบบโปรแกรมได้เพียงครั้งเดียว โดยถ้าหากเป็นแบบโปรแกรมได้หลายครั้งนั้นบนตัวถังของไมโครคอนโทรลเลอร์จะมีหน้าต่างกระจกติดอยู่สามารถมองเห็นชิปภายในได้เวลาลบข้อมูลต้องลบด้วยแสงอัลตร้าไวโอเลตจำนวนรอบในการโปรแกรมใหม่อยู่ระหว่าง 10-100 ครั้งแต่ถ้าเป็นแบบโปรแกรมได้ครั้งเดียว (One–Time Programmable, OTP) จะไม่สามารถลบได้ตัวถังของมันจะปิดมิดชิดเหมือนกับไอซีธรรมดาแบบอีอีพรอมหน่วยความจำแบบนี้จะลบและเขียนใหม่ได้ด้วยสัญญาณไฟฟ้าในอดีตเป็นที่นิยมมากเนื่องจากสามารถเขียนใหม่ได้เป็นหลักร้อยรอบขึ้นไปในบางตระกูลถึง 1 ล้านครั้งแต่ในปัจจุบันไม่เป็นที่นิยมใช้ในไมโครคอนโทรลเลอร์เนื่องจากต้นทุนสูง แบบแฟลชหน่วยความจำโปรแกรมชนิดนี้สามารถลบและเขียนได้ด้วยสัญญาณไฟฟ้าแตกต่างกับแบบอีอีพรอมในเชิงการใช้งานตรงที่กระบวนการลบข้อมูลหน่วยความจำโปรแกรมแบบแฟลชจะไม่สามารถเลือกลบเฉพาะเจาะจงบางตำแหน่งได้เมื่อทำการลบข้อมูลจะต้องลบทั้งหมดหน่วยความจำแบบนี้ได้รับความนิยมมากเนื่องจากราคาไม่สูงและสามารถโปรแกรมได้เป็นร้อยครั้งขึ้นไปในบางรุ่นสูงเป็นหมื่นครั้งและเป็นแสนครั้งขึ้นอยู่กับแรงดันที่ใช้ในโปรแกรม

2) หน่วยความจำข้อมูลแรม

เป็นหน่วยความจำที่ต้องมีในไมโครคอนโทรลเลอร์ทุกตัวเพราะใช้เป็นพื้นที่สำหรับเก็บข้อมูลทั้งในระหว่างและหลังการประมวลผลยิ่งมีมากยิ่งช่วยในการทำงานสะดวกขึ้นเพราะหน่วยความจำแรมมีอัตราเร็วในการอ่านเขียนสูงมากและไม่จำกัดจำนวนรอบในการอ่านเขียนในพื้นที่ของหน่วยความจำข้อมูลแรมจะแบ่งออกเป็น 2 ส่วนคือส่วนของข้อมูลทั่วไปสำหรับเก็บค่าตัวแปรและส่วนของรีจิสเตอร์โดยปกติแล้วหน่วยความจำข้อมูลแรมจะมีความจุไม่มากเมื่อเทียบกับหน่วยความจุโปรแกรมในบางตัวอยู่ในหลักสิบไบต์แต่ถ้าไมโครคอนโทรลเลอร์มีความสามารถสูงขึ้นความจุของหน่วยความจำข้อมูลแรมก็เพิ่มมากขึ้นตามทั้งนี้เพราะต้องเพิ่มในส่วนของรีจิสเตอร์ตามความสามารถที่สูงขึ้นของไมโครคอนโทรลเลอร์

3) หน่วยความจำข้อมูลอีอีพรอม

เป็นหน่วยความจำข้อมูลพิเศษที่ในไมโครคอนโทรลเลอร์บางเบอร์บางรุ่นบางตระกูลใช้สำหรับเก็บข้อมูลที่ต้องการรักษาไว้เมื่อไม่มีการจ่ายไฟเลี้ยงให้แก่ไมโครคอนโทรลเลอร์การติดต่อเพื่ออ่านจะมีลักษณะพิเศษขึ้นอยู่กับไมโครคอนโทรลเลอร์แต่ละเบอร์ขนาดของหน่วยความจำแบบนี้มักเท่ากับ 8 บิตส่วนความจุก็จะแตกต่างกันไปตั้งแต่ไม่กี่สิบไบต์จนถึงเป็นกิโลไบต์การอ่านเขียนหน่วยความจำแบบนี้จะใช้สัญญาณไฟฟ้าทั้งหมดและสามารถรักษาข้อมูลล่าสุดไว้แม้ว่าจะไม่มีการจ่ายไฟเลี้ยงให้แก่ไมโครคอนโทรลเลอร์แล้วก็ตามสำหรับจำนวนรอบในการเขียนโดยปกติอยู่ในหลักล้านครั้งขึ้นไป

2.2.4 หน่วยความจำพิเศษ [6]

เป็นหน่วยความจำที่ใช้เก็บข้อมูลพิเศษไม่ได้ใช้สำหรับเก็บชุดคำสั่งของโปรแกรมหรือข้อมูลอื่นๆมีหลายชนิดแตกต่างกันขึ้นอยู่กับตระกูลของไมโครคอนโทรลเลอร์ได้แก่

1) รีจิสเตอร์

เป็นหน่วยความจำพิเศษที่มีบทบาทสูงมากในการทำงานของไมโครคอนโทรลเลอร์เพื่อให้ไมโครคอนโทรลเลอร์สามารถที่จะอ่านและเขียนข้อมูลได้ตลอดเวลาจนกว่าจะหยุดจ่ายไฟเลี้ยงให้แก่ไมโครคอนโทรลเลอร์ ส่วนหน้าที่หลักของรีจิสเตอร์ คือ ใช้เก็บข้อมูลในการทำงานของไมโครคอนโทรลเลอร์โดยข้อมูลที่เก็บนี้มีทั้งข้อมูลแสดงสถานะการทำงาน, ข้อมูลสำหรับควบคุมการทำงานโมดูลย่อยต่างๆภายในไมโครคอนโทรลเลอร์, ข้อมูลที่รับเข้ามาจากพอร์ตอินพุตหรือข้อมูลที่ต้องส่งออกไปยังอุปกรณ์ภายนอกผ่านพอร์ตเอาท์พุตโดยข้อมูลแต่ละประเภทก็จะจัดเก็บลงในรีจิสเตอร์ที่แตกต่างกันตามหน้าที่การทำงานหน่วยความจำที่นำมาใช้ทำรีจิสเตอร์มีด้วยกัน 2 ลักษณะขึ้นอยู่กับสถาปัตยกรรมของไมโครคอนโทรลเลอร์หากเป็นแบบพรินซ์ตันรีจิสเตอร์จะมีอยู่ด้วยกัน 2 ส่วนส่วนแรกจะอยู่ร่วมกันกับซีพียูหรือเรียกว่ารีจิสเตอร์ซีพียูส่วนที่สองจะอยู่แยกต่างหากซึ่งมักเป็นรีจิสเตอร์ควบคุมพอร์ตอินพุตเอาท์พุตและรีจิสเตอร์แสดงสถานะแต่ในสถาปัตยกรรมแบบฮาร์วาร์ดนั้นจะใช้เพียงบางส่วนในหน่วยความจำข้อมูลแรมภายในไมโครคอนโทรลเลอร์

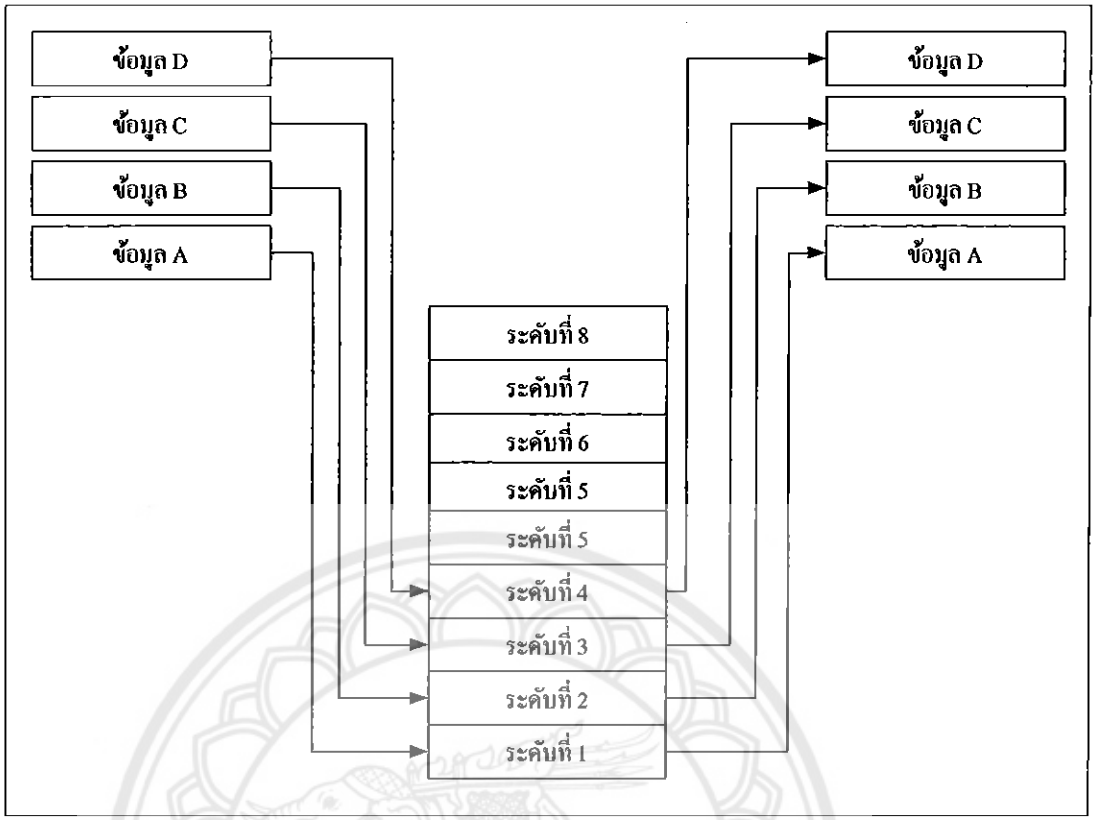
2) โปรแกรมเคาน์เตอร์

การที่ซีพียูสามารถติดต่อกับหน่วยความจำโปรแกรมเพื่ออ่านข้อมูลคำสั่งได้อย่างถูกต้องเป็นผลมาจากรีจิสเตอร์หน้าที่พิเศษตัวหนึ่งคือรีจิสเตอร์ตัวนับโปรแกรมหรือโปรแกรมเคาน์เตอร์ (Program Counter, PC) โดยโปรแกรมเคาน์เตอร์จะเป็นตัวชี้ตำแหน่งแอดเดรสของหน่วยความจำ

โปรแกรมที่ซีพียูจะต้องไปกระทำในลำดับถัดไปโดยปกติแล้วค่าของโปรแกรมเคาน์เตอร์จะเปลี่ยนแปลงโดยอัตโนมัติขึ้นอยู่กับผลการทำงานที่เกิดขึ้นในไมโครคอนโทรลเลอร์บางตระกูลสามารถเข้าถึงโปรแกรมเคาน์เตอร์เพื่อทำการอ่านเขียนได้แต่ในบางตระกูลก็ไม่สามารถทำได้

3) สแต็ก

สแต็ก (Stack) เป็นหน่วยความจำพิเศษที่ไมโครคอนโทรลเลอร์ทุกตัวต้องมีโดยหน้าที่ของมันคือเก็บข้อมูลที่ยังต้องการอยู่ของรีจิสเตอร์และเมื่อข้อมูลนั้นถูกนำมาเก็บไว้ในสแต็กแล้วก็สามารถที่จะเปลี่ยนข้อมูลในรีจิสเตอร์ตัวนั้นๆได้ทันทีหลังจากที่ทำงานเรียบร้อยแล้วจึงกลับมาอ่านข้อมูลเดิมกลับคืนจากสแต็กซึ่งมีกระบวนการทำงานดังในภาพที่ 2-5 การเก็บข้อมูลของสแต็กจะมีลักษณะเป็นระดับหรือเป็นชั้นข้อมูลที่จะเก็บเข้ามาก่อนและจะต้องอ่านออกทีหลังหรือเป็นแบบ FILO (First In Last Out) และจำนวนระดับหรือจำนวนชั้นของสแต็กก็มีจำกัดในไมโครคอนโทรลเลอร์ส่วนใหญ่จะมีความจุของสแต็กไม่น้อยกว่า 8 ระดับการที่ยิ่งมีขนาดของสแต็กมากหรือมีจำนวนระดับมากก็จะยิ่งช่วยให้การทำงานสะดวกขึ้นเพราะในการประมวลผลมีโอกาสมากที่จะต้องพักข้อมูลในรีจิสเตอร์หลักเพื่อไปทำงานอื่นก่อนหลังจากนั้นจึงกลับมาทำงานต่อโดยเฉพาะอย่างยิ่งกับงานที่มีการอินเทอร์รัพต์หรือขัดจังหวะซีพียูอยู่บ่อยๆรวมถึงการกระโดดไปทำงานที่โปรแกรมย่อยที่มีความต้องการเขียนข้อมูลลงในรีจิสเตอร์ตัวเดียวกันนี้หลังจากทำงานแล้วจึงกลับมาที่โปรแกรมหลักแล้วอ่านค่าเดิมก่อนหน้านี้กลับมาทำงานต่อหว่างงานบางลักษณะการกระโดดไปทำงานยังโปรแกรมย่อยซ้อนกัน 2-3 ชั้นทำให้ต้องมีการเก็บข้อมูลไว้ในสแต็กมากขึ้นหากความจุของสแต็กมีน้อยก็จะไม่สามารถรองรับการทำงานในลักษณะนี้ได้ขนาดของสแต็กโดยปกติจะต้องเท่ากับขนาดของรีจิสเตอร์ตัวนับโปรแกรมหรือ โปรแกรมเคาน์ เพราะมีโอกาสที่จะต้องเก็บค่าของ โปรแกรมเคาน์เตอร์ไว้ในสแต็กด้วย



รูปที่ 2.5 กลไกการทำงานของสแต็กอย่างง่าย [6]

2.2.5 การทำงานของไมโครคอนโทรลเลอร์ [6]

ไมโครคอนโทรลเลอร์จะสามารถทำงานได้เมื่อจ่ายไฟเลี้ยงและต่อวงจรกำเนิดสัญญาณนาฬิกาให้แก่มันจากนั้นซีพียูภายในไมโครคอนโทรลเลอร์จะติดต่อกับหน่วยความจำโปรแกรมเพื่ออ่านข้อมูลคำสั่งทำงานตามคำสั่งที่บรรจุอยู่ในหน่วยความจำโปรแกรม นั่นหมายความว่า ต้องมีการเขียนข้อมูลลงในหน่วยความจำโปรแกรมก่อนโดยไมโครคอนโทรลเลอร์แต่ละเบอร์จะมีรูปแบบของข้อมูลคำสั่งที่แตกต่างกันโดยภาษาที่ใช้เขียนโปรแกรมสามารถแบ่งได้ 2 ระดับ คือ ภาษาระดับสูง (High Level Language) และภาษาแอสเซมบลี (Assembly Language) ซึ่งโดยปกติแล้วไมโครคอนโทรลเลอร์ต้องโปรแกรมด้วยภาษาแอสเซมบลี เนื่องจากสามารถทำงานได้รวดเร็วผ่านกระบวนการแปลงข้อมูลคำสั่งเป็นเลขฐานสิบหกเพื่อทำงานตามคำสั่งเพียง 1 ขั้นตอนคือแปลงจากภาษาแอสเซมบลีเป็นข้อมูลฐานสิบหกที่เรียกว่าออปโค้ด (Opcode) แต่ข้อเสียของการเขียนภาษาแอสเซมบลีคือผู้เขียนต้องทำความเข้าใจในชุดคำสั่งของไมโครคอนโทรลเลอร์เบอร์นั้นๆอย่างถ่องแท้และเมื่อเปลี่ยนเบอร์ไมโครคอนโทรลเลอร์ก็จะต้องทำการเรียนรู้และทำความเข้าใจชุดคำสั่งใหม่ซึ่งอาจทำให้เสียเวลามากรวมทั้งการเขียนโปรแกรมด้วยภาษาแอสเซมบลีผู้เขียนต้องมีทักษะในการเขียนโปรแกรมสูงพอสมควรและเข้าใจถึงสถาปัตยกรรมของไมโครคอนโทรลเลอร์เป็น

อย่างดีในขณะที่การเขียนโปรแกรมด้วยภาษาระดับสูงอาทิเช่นภาษาซีภาษาเบสิกต้องผ่านกระบวนการที่เรียกว่าคอมไพล์ (Compile) เพื่อแปลงภาษาระดับสูงเหล่านั้นเป็นภาษาเครื่องหรือ ออปโค้ดของไมโครคอนโทรลเลอร์เบอร์นั้นๆเสียก่อนเมื่อใช้เครื่องมือทางซอฟต์แวร์ตัวนี้ทำให้ผู้เขียนโปรแกรมอาจไม่จำเป็นต้องศึกษาสถาปัตยกรรมและชุดคำสั่งของไมโครคอนโทรลเลอร์เบอร์นั้นๆอย่างลึกซึ้งเท่ากับการเขียนภาษาแอสเซมบลีทั้งนี้เพราะคอมไพเลอร์จะทำในส่วนนี้แทน ดังนั้นเมื่อผู้ใช้งานเปลี่ยนเบอร์ไมโครคอนโทรลเลอร์ก็เพียงจัดหาโปรแกรมคอมไพเลอร์ที่เหมาะสมมาใช้งานและศึกษาสถาปัตยกรรมของไมโครคอนโทรลเลอร์เบอร์ใหม่อีกเพียงเล็กน้อยก็สามารถใช้งานได้แต่ข้อเสียของการใช้คอมไพเลอร์คือราคาแพงมาก

2.2.6 ไมโครคอนโทรลเลอร์ตระกูล AVR [2]

ไมโครคอนโทรลเลอร์ AVR ผลิตจากบริษัท ATMEL ซึ่งในบ้านเราเองนิยมนำมาใช้งานกันอย่างกว้างขวางซึ่งส่วนประกอบ หรือสถาปัตยกรรมภาพรวมของไมโครคอนโทรลเลอร์ตระกูลนี้โดยเบอร์ต่างๆของไอซีจะมีสถาปัตยกรรมแต่ละตัวแตกต่างกันออกไป ซึ่งบางตัวมีให้ใช้งานบางตัวไม่มีให้ใช้งาน หรือ ตัวใดมีมากกว่าตัวใด สำหรับไมโครคอนโทรลเลอร์ตัวนั้นๆด้วยคุณสมบัติต่างๆของไมโครคอนโทรลเลอร์ที่เราควรทราบ เช่น ความถี่สูงสุดในการทำงาน หน่วยความจำประเภทต่างๆมีค่าเท่าไร แรงดันในการทำงาน จำนวนพอร์ตอินพุตและเอาท์พุต ตอบสนองการอินเทอร์รัพท์ได้กี่ครั้ง เป็นต้น

2.2.7 โครงสร้างและสถาปัตยกรรมของ ATmega 32 [2]

ไมโครคอนโทรลเลอร์ได้มีการคิดค้นและพัฒนามาอย่างต่อเนื่องเพื่อให้มีศักยภาพในการทำงานสูงขึ้นไมโครคอนโทรลเลอร์ตระกูล AVR ของบริษัท ATMEL เป็นไมโครคอนโทรลเลอร์ที่มีฟังก์ชันการใช้งานต่างๆมากมาย เช่น โมดูล Analog to Digital, Timer/Counter, USART, SPI และจัดเป็นไมโครคอนโทรลเลอร์ตระกูลใหม่ที่มีสถาปัตยกรรมแบบ RISC (Advanced RISC architecture) ซึ่งส่วนต่างๆ เหล่านี้จะถูกสร้างรวมอยู่ภายในซีพียูเพียงตัวเดียวทำให้สามารถทำงานได้หลายๆอย่างและสามารถลดในส่วนของฮาร์ดแวร์บางอย่างลงส่วนในเรื่องของความเร็วซีพียูตระกูลนี้จะใช้เวลาในการกระทำคำสั่งต่างๆ เพียง 1 ไซเคิลต่อคำสั่งเท่านั้น ทำให้มีความเร็วในการทำงานมากกว่าซีพียูทั่วไป (ที่ความถี่เดียวกัน) และมีสถาปัตยกรรมภายในดังรูปที่ 2.6 คุณสมบัติต่างๆของไมโครคอนโทรลเลอร์ AVR สามารถสรุปอย่างคร่าวๆ ได้ดังนี้

- สถาปัตยกรรมภายในเป็นแบบ Advanced RISC (Reduced Instruction Set Computer)

- มีคำสั่งควบคุมการทำงานมากกว่า 100 คำสั่ง โดยมีความเร็วในการประมวลผล 1 คำสั่งต่อ 1 สัญญาณนาฬิกา (1 MIP/1MHz)

- มีรีจิสเตอร์ใช้งานทั่วไปขนาด 8 บิต จำนวน 32 ตัว (ทำให้สะดวกต่อการพัฒนาโปรแกรม ด้วยภาษา C เป็นอย่างมาก)

- ความเร็วในการทำงาน 1 MIPS ต่อ 1 MHz และมากถึง 16 MIPS เมื่อใช้ความถี่ที่ 16 MHz (ความสามารถในการใช้งานความถี่สัญญาณนาฬิกาขึ้นอยู่กับเบอร์ที่เลือกใช้งาน)

- หน่วยความจำ ROM แบบ Flash (มีโหมดป้องกันหน่วยความจำ) ขนาด 32 กิโลไบต์ (เขียน/ลบ ได้ 10,000 ครั้ง)

- หน่วยความจำข้อมูลแบบ EEPROM (มีโหมดป้องกันหน่วยความจำ) ขนาด 1,024 ไบต์ (เขียน/ลบ) ได้ 100,000 ครั้ง)

- หน่วยความจำข้อมูลแบบ SRAM 1 กิโลไบต์

- ไทเมอร์/เคาน์เตอร์ทั้งแบบ 8 บิตและ 16 บิต พร้อมปรีสเกลเลอร์

- มีระบบตรวจสอบความผิดพลาดในการทำงานของซอฟต์แวร์ (Watchdog Timer with On-Chip Oscillator)

- โมดูลสร้างสัญญาณ PWM (Pulse Width Modulator) มีจำนวน 4 ช่อง

- มีโมดูลแปลงสัญญาณอะนาลอกเป็นดิจิตอล (ADC) ขนาด 10 บิต มากถึง 8 ช่อง

- โมดูลเปรียบเทียบแรงดันอะนาลอก (Analog Comparator)

- การสื่อสารข้อมูลอนุกรมมีทั้งแบบ UART (Universal Asynchronous Receiver Transmitters) หรือแบบ RS232, SPI (Serial Peripheral Interface) และแบบ I^2C เป็นต้น

- พอร์ตอินพุตเอาท์พุตขึ้นอยู่กับเบอร์ AVR ที่เลือกใช้งาน มีตั้งแต่ 8 ขา จนมากกว่า 100 ขาพอร์ต (ATmega 32 มีขาพอร์ตอินพุตเอาท์พุต 32 ขา)

- ทำงานที่ไฟเลี้ยง 4.5 ถึง 5.5 โวลต์

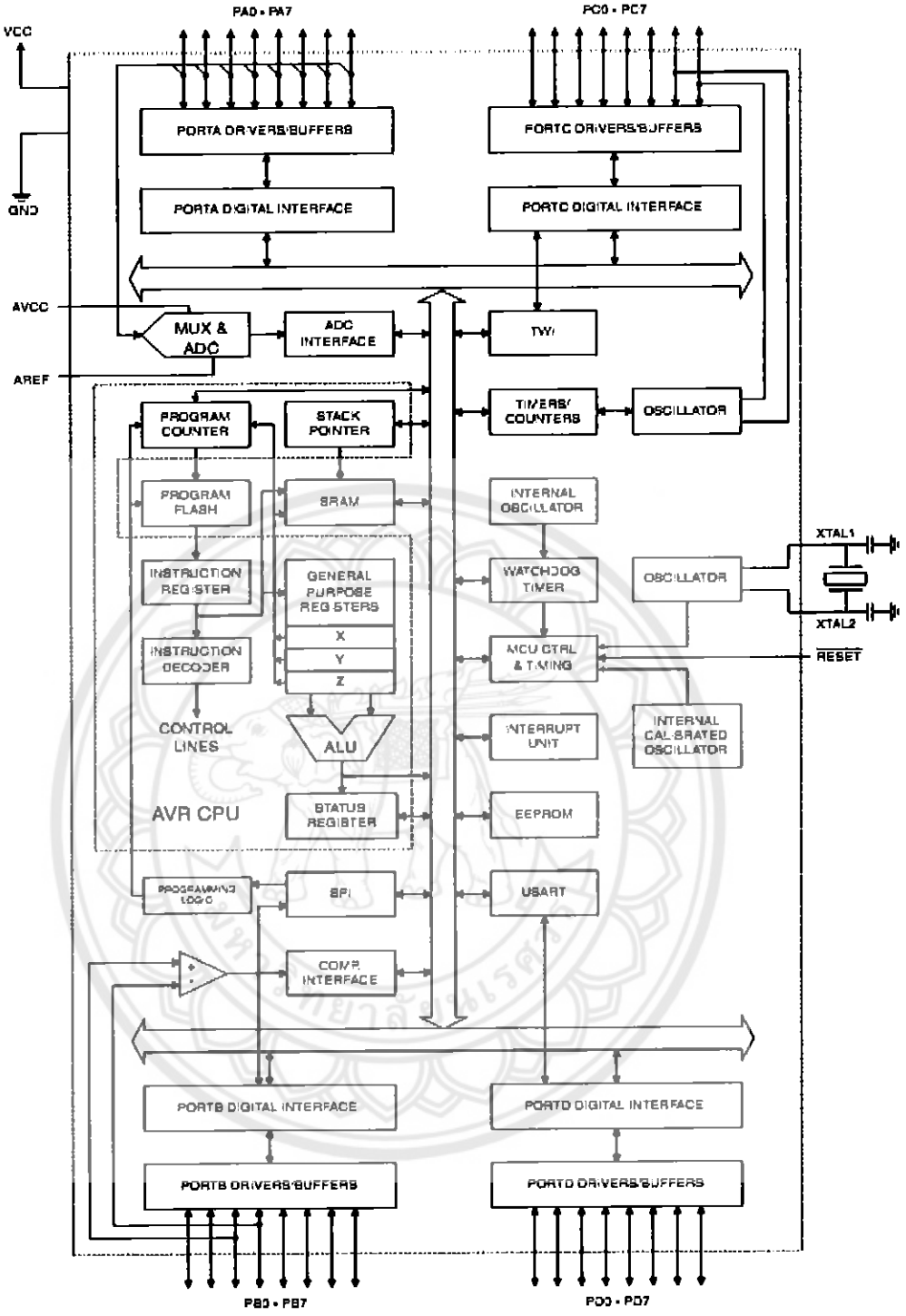
- มีพอร์ต I/O 4 พอร์ตประกอบด้วย A, B, C, D แต่ละพอร์ตจะมีจำนวนบิตเท่ากัน

- PORT A = PA0–PA7 จำนวน 8 บิต

- PORT B = PB0–PB7 จำนวน 8 บิต

- PORT C = PC0–PC7 จำนวน 8 บิต

- PORT D = PD0–PD7 จำนวน 8 บิต

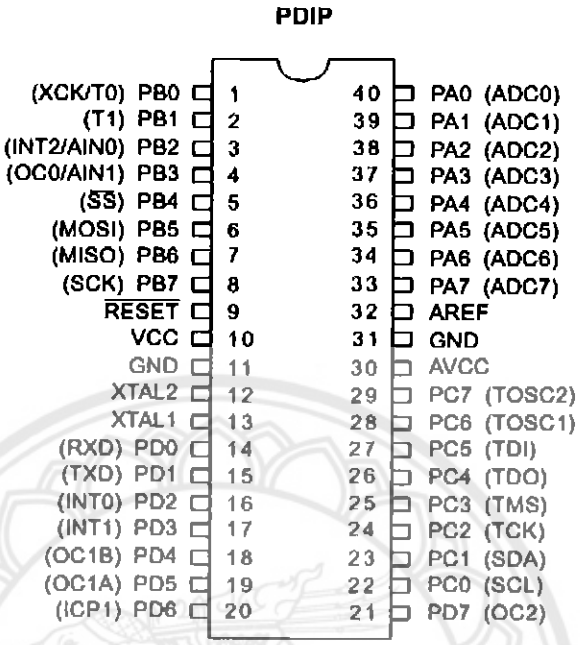


รูปที่ 2.6 สถาปัตยกรรมภายในของไมโครคอนโทรลเลอร์ ATmega 32 [1]

2.2.8 ตัวถังและตำแหน่งขาสัญญาณของ ATmega 32 [2]

ขาพอร์ตอินพุตเอาท์พุตของไมโครคอนโทรลเลอร์ ATmega 32 มีจำนวน 40 ขา โดยแบ่งเป็นขาพอร์ตอินพุตเอาท์พุตอิสระ จำนวน 32 ขา ประกอบไปด้วย PA, PB, PC, PD ขนาด 8 บิต

และขาพอร์ตที่เกี่ยวข้องกับสัญญาณอะนาลอกจำนวน 2 ขาพอร์ต คือ AREF และ AVCC รายละเอียดขาพอร์ตทั้งหมดแสดงดังรูปที่ 2.7 ตัวถังของ ATmega 32 และตำแหน่งขาสัญญาณต่างๆ



รูปที่ 2.7 ตัวถังของ ATmega 32 และตำแหน่งขาสัญญาณต่างๆ **[1]**

ตารางที่ 2.1 ชื่อขา ตำแหน่งขาชนิดขาและหน้าที่การทำงานของไมโครคอนโทรลเลอร์ ATmega 32

ชื่อขา	ตำแหน่ง	ชนิด	หน้าที่และการทำงาน
VCC	10	อินพุต	ขาแรงดันไฟตรง
GND	11	อินพุต	ขากราวด์
Port A (PA0.0-PA0.7)	33-40	อินพุต/เอาท์พุต	ขาพอร์ตเป็นอินพุตเอาท์พุตดิจิตอล กำหนดการพูลอัพภายในขาพอร์ตได้ (internal pull-up register) และสามารถกำหนดใช้งานเป็นพอร์ตอินพุตสัญญาณอะนาลอก (A/D Converter) ได้

ชื่อขา	ขาที่	ชนิด	หน้าที่และการทำงาน
Port B (PB0.0-PB0.7)	1-7	อินพุต/เอาท์พุต	ขาพอร์ตเป็นอินพุตเอาท์พุตดิจิตอล กำหนดการพูลอัพภายในขาพอร์ตได้ (internal pull-up register) และเป็นขาพอร์ตหน้าที่พิเศษ อีกด้วย เช่น ขาสำหรับการโปรแกรมชิพ ขาป้อนสัญญาณนาฬิกาภายนอก
Port C (PC0.0-PC0.7)	22-29	อินพุต/เอาท์พุต	ขาพอร์ตเป็นอินพุตเอาท์พุตดิจิตอล กำหนดการพูลอัพภายในขาพอร์ตได้ (internal pull-up register) และเป็นขาพอร์ตหน้าที่พิเศษ อีกด้วย เช่น ขาเชื่อมต่อกับดีบักและโปรแกรมด้วยการเชื่อมต่อแบบ JTAG
Port D (PD0.0-PD0.7)	14-21	อินพุต/เอาท์พุต	ขาพอร์ตเป็นอินพุตเอาท์พุตดิจิตอล กำหนดการพูลอัพภายในขาพอร์ตได้ (internal pull-up register) และเป็นขาพอร์ตหน้าที่พิเศษ อีกด้วย เช่น ขาเชื่อมต่อพอร์ตอนุกรม ขาอินเตอร์รัปต์เนื่องจากสัญญาณภายนอก
$\overline{\text{RESET}}$	9	อินพุต	ขารีเซตวงจร
XTAL1	13	อินพุต	ขาต่อคริสตอลออสซิเลเตอร์ ช่องที่ 1
XTAL2	12	เอาท์พุต	ขาต่อคริสตอลออสซิเลเตอร์ ช่องที่ 2
AVCC	30	เอาท์พุต	ขาแรงดันสำหรับพอร์ต A และโมดูลแปลงสัญญาณอะนาลอกเป็นดิจิตอล
AREF	32	เอาท์พุต	ขาแรงดันอะนาลอกอ้างอิงสำหรับโมดูลแปลงสัญญาณอะนาลอกเป็นดิจิตอล

2.3 เซนเซอร์วัดอุณหภูมิ [7]

เซนเซอร์อุณหภูมิ (Temperature Sensor) เป็นอุปกรณ์ที่ทำหน้าที่เปลี่ยนระดับอุณหภูมิเป็นระดับแรงดันไฟฟ้า ซึ่งมีมากมายหลายชนิดยกตัวอย่างเช่น

เทอมิสเตอร์ (Thermister) เป็นอุปกรณ์ที่เปลี่ยนอุณหภูมิให้กลายเป็นระดับความต้านทาน ซึ่งมีอยู่ 2 แบบ คือ เอ็นทีซี และ พีทีซี เทอมิสเตอร์ เป็นอุปกรณ์จับความร้อนที่ใช้สำหรับป้องกันอุณหภูมิที่สูงเกินกว่าค่าที่ กำหนดไว้ที่ตัวเซนเซอร์ จะทำงานร่วมกับรีเลย์

ไบเมทอลลิค หรือ เทอร์โมสตัท (Bimetallic / Thermostat) เป็นอุปกรณ์ตัวจับความร้อนที่ใช้สำหรับป้องกันอุณหภูมิที่สูงเกินกว่าค่าที่กำหนดไว้ที่ตัวเซนเซอร์ ไบเมทอลลิค ทำงานเหมือนเทอร์โมสตัทของเตารีดจะถูกติดตั้งไว้ที่ขดลวดบริเวณปลายคอยล์เนื่องจากมีขนาดค่อนข้างใหญ่เพราะตัวมันเองจะมีหน้าสัมผัสอยู่แล้ว การใช้งานจะนำไปต่อเข้ากับชุดคอนโทรลโดยตรง

อาร์ทีดี (RTD) มีหลายประเภท ประเภทที่นิยม คือ PT100 โดยที่ PT100 มีความหมายว่าที่อุณหภูมิ 0 องศาเซลเซียส ตัว PT100 จะมีค่าความต้านทาน 100 โอห์ม RTD ต้องใช้ร่วมกับรีเลย์เช่นกันสามารถเซ็ตได้เป็นทั้งชุดป้องกันอุณหภูมิสูง หรือใช้วัดค่าอุณหภูมิได้เลย ข้อเสียคือมีราคาค่อนข้างแพง

เทมเพอเรเจอร์เซนเซอร์โมดูล (Temperature Sensor Module) เป็นอุปกรณ์ที่ทำหน้าที่เปลี่ยนระดับอุณหภูมิเป็นสัญญาณไฟฟ้าที่มีความ เที่ยงตรงสูง เช่น ไอซีอุณหภูมิ เซนเซอร์แบบนี้จะให้ความเที่ยงตรงของค่าที่อ่านจากจากเทอมิสเตอร์ ไอซีตระกูล 335 เช่น LM135/LM235/LM335 มีความไวทางด้านเอาท์พุตเป็น 10 mV/°K, ไอซีตระกูล 34 เช่น เบอร์ LM34 จากบริษัท National Semiconductor

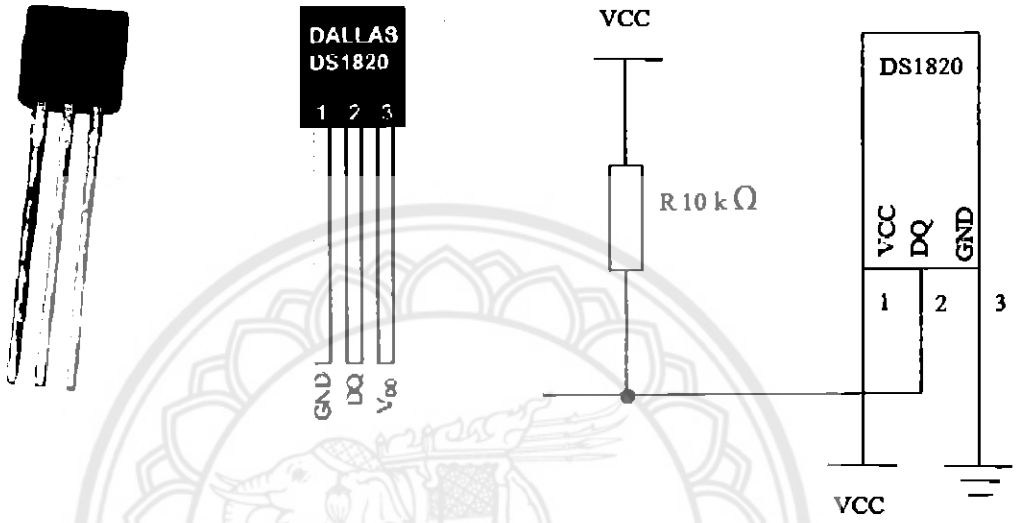
ไอซี DS1820 ซึ่งเป็นตัวแปลงอุณหภูมิ ให้เป็นค่าดิจิตอล สำหรับการอ่านค่าอุณหภูมิ ใช้สายสัญญาณเพียง 1 เส้นเท่านั้น ถ้ารวม VCC, GND เข้าไปด้วยก็จะมีขาใช้งานเพียง 3 ขาเท่านั้น รูปร่างหน้าตาก็จะคล้ายกับทรานซิสเตอร์ตัวถัง TO-92 จริงๆแล้วมันไม่ใช่แค่เซนเซอร์อุณหภูมิเท่านั้น แต่ยังมี เลเซอร์รอม (LASERED ROM) ขนาด 64 บิตโดย DS1820 ในแต่ละตัวจะมีค่าเลเซอร์รอมไม่ซ้ำกัน ทำให้สามารถต่อ DS1820 หลายๆตัวในสายสัญญาณเส้นเดียวกันได้ ย่านการวัดอุณหภูมิจะอยู่ในช่วง -55 องศาเซลเซียส ถึง +125 องศาเซลเซียส และ ยังมี Alarm Triger TH, TL ไว้คอยเตือนเราด้วยว่า อุณหภูมิเกิน ระดับที่เราต้องการควบคุม

2.3.1 การทำงานของเซนเซอร์วัดอุณหภูมิ DS1820

ไอซี DS1820 เป็นไอซีที่มีระบบการสื่อสารข้อมูลอนุกรมแบบ 1 สาย ซึ่งถือได้ว่าเป็นระบบที่มีความชาญฉลาดและใช้จำนวนสายสัญญาณเพียง 1 เส้นเท่านั้น โดยไม่ต้องมีสายสัญญาณคล๊อกมาควบคุมการถ่ายทอดข้อมูล เหมือนกับระบบสื่อสารข้อมูลอนุกรมในแบบอื่นๆ สายข้อมูลจะทำหน้าที่เสมือนเป็นสัญญาณคล๊อกในตัว ส่วนค่าของข้อมูลจะพิจารณาจากลักษณะของรูปสัญลักษณ์ ที่ปรากฏบนสายสัญญาณ ในแต่ละช่องของเวลา ซึ่งเรียกว่า “Time-Slot” โดยคาบเวลาต่ำสุดและสูงสุดของสถานะต่างๆ ในการสื่อสารข้อมูลในแต่ละ “Time-Slot” มีการกำหนดขอบเขตไว้อย่างชัดเจน การถ่ายทอดข้อมูลจะเกิดขึ้นในแต่ละ “Time-Slot” นั้นรูปแบบการถ่ายทอดข้อมูลจะ

เป็นแบบอะซิงโครนัสในระดับบิตไม่มีการกำหนดความยาวของข้อมูลเป็นระดับไบต์ ระบบสื่อสารแบบนี้เหมาะที่จะใช้ในการสื่อสารข้อมูลระหว่างไอซีบนแผงวงจรเดียวกัน

2.3.2 ขาสัญญาณสำหรับสื่อสารข้อมูลของ DS1280 [7]



รูปที่ 2.8 รูปร่างของ DS1280 และการต่อใช้งาน

1) ขากราวด์ (GND)

2) ขารับส่งข้อมูล (DQ)

3) ขาแหลงจ่ายไฟเข้า (VDD)

2.3.3 คุณสมบัติเด่นของไอซี DS1820

DS1820 จะให้สัญญาณเอาท์พุทออกมาเป็นแบบดิจิตอล และยังสามารถที่จะทำการโปรแกรมเข้าไปยังหน่วยความจำและควบคุมฟังก์ชันภายในไอซีได้ ซึ่งมีหน่วยความจำรอมภายในขนาด 64 บิตแบบเลเซอร์รอม ดังนั้นจึงสามารถที่จะทำการอ่านและเขียนข้อมูลต่างๆ เกี่ยวกับหน้าที่การทำงานในการตรวจวัดอุณหภูมิได้อย่างมากมายตามการประมวลผลของไมโครโปรเซสเซอร์ นอกจากนั้นแล้วยังสามารถติดตั้ง DS1820 เพื่อการตรวจวัดอุณหภูมิได้ในหลายลักษณะและหลายสถานที่ตำแหน่งการติดตั้งที่มีความแตกต่างอย่างมากกับอุปกรณ์ทั่วๆไปไม่ว่าจะเป็นการติดตั้ง

ภายในอาคาร, อุปกรณ์เครื่องใช้ต่างๆ หรือ ภายในเครื่องจักร และเอาท์พุตที่เป็นอนุกรมตัวเลขของ DS1820 สามารถต่อเอาท์พุตบนสายสัญญาณเพียงเส้นเดียวได้หลายๆ ชุดโดยไม่สับสนข้อมูลซึ่งกันและกัน ซึ่งมีคุณสมบัติเด่นๆ ดังนี้

1) อินเตอร์เฟสสัญญาณผ่านขาเอาท์พุทเพียงพอร์ตเดียวแบบ 1 สายข้อมูล (1 - WireTM)

2) ขยายจุดตรวจจับอุณหภูมิได้หลายๆ จุดบนสายข้อมูลเพียง 1 สายข้อมูล

3) ไม่ต้องใช้อุปกรณ์ภายนอกมาต่อร่วม

4) สามารถควบคุมการทำงานเพาเวอร์ออนได้ผ่านทางสายข้อมูล

5) เพาเวอร์ขณะสแตนด์บายเป็นศูนย์

6) ย่านการวัดอุณหภูมิตั้งแต่ -55 องศาเซลเซียสถึง +125 องศาเซลเซียสที่ 0.5 องศาต่อสเต็ป หรือตั้งแต่ย่าน -67 องศาฟาเรนไฮ ถึง +257 องศาฟาเรนไฮ ที่ 0.9 องศาฟาเรนไฮ ต่อสเต็ป

7) อุณหภูมิจะถูกอ่านออกมาเป็นค่าทางดิจิตอล 9 บิต

8) อัตราความเร็วในการแปลงจากอุณหภูมิมาเป็นค่าตัวเลขทางดิจิตอลเท่ากับ 200 มิลลิวินาที

9) ผู้ใช้งานสามารถกำหนดการเซตค่าเตือนย่านอุณหภูมิได้ในแบบ non – volatile

10) การเตือนย่านอุณหภูมินั้นสามารถกำหนดรหัสผ่านการสั่งการและแอดเดรสของอุปกรณ์ได้จากภายนอกพื้นที่ตรวจวัดอุณหภูมิผ่านทางโปรแกรมภายนอก

11) เหมาะกับการประยุกต์ใช้งานตรวจวัดอุณหภูมิและติดตั้งไว้ในอุปกรณ์ควบคุมเทอร์โมสแตติก, ระบบโรงงานอุตสาหกรรม, ผลิตภัณฑ์, เทอร์โมมิเตอร์ หรือระบบอื่นๆ ที่มีส่วนตรวจจับอุณหภูมิทำงานร่วมอยู่

2.3.4 รูปแบบการอ่านข้อมูลอุณหภูมิของ DS1280

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
LS Byte	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}
	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
MS Byte	S	S	S	S	S	S	S	S

TEMPERATURE	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+85.0°C	0000 0000 1010 1010	00AAh
+25.0°C	0000 0000 0011 0010	0032h
+5.0°C	0000 0000 0000 0001	0001h
0°C	0000 0000 0000 0000	0000h
-0.5°C	1111 1111 1111 1111	FFFFh
-25.0°C	1111 1111 1100 1110	FFCEh
-55.0°C	1111 1111 1001 0010	FF92h

รูปที่ 2.9 ตารางแสดงรูปแบบในการอ่านอุณหภูมิของ DS1280

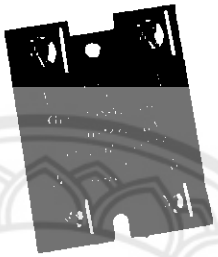
ข้อมูลอุณหภูมิที่วัดได้จะถูกเก็บอยู่ในรีจิสเตอร์อุณหภูมิ (Register Temperature) ซึ่งมีขนาด 16 บิต ดังแสดงในรูปที่ 2.9 ถ้าข้อมูลอุณหภูมิเป็นบวก S จะเป็น "1" แต่ถ้าข้อมูลอุณหภูมิเป็นลบ S จะเป็น "0" ในกรณีที่ DS1820 ทำงานในโหมดความละเอียด 12 บิต บิตทุกบิตในรีจิสเตอร์อุณหภูมิ (Register Temperature) จะถูกใช้หมด แต่ในกรณีที่ทำงานในโหมด 9-11 บิต บิตล่าง (บิต 0 – บิต 2) จะไม่ถูกใช้งาน ซึ่งในการกำหนดโหมดความละเอียดการทำงานของ DS18B20 นั้นสามารถกำหนดได้ที่รีจิสเตอร์กำหนดค่า (Register Configuration) ซึ่งโดยปกติเริ่มต้น DS1820 จะทำงานในโหมด 12 บิต

2.4 รีเลย์

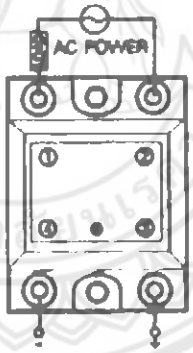
รีเลย์ (Relay) เป็นอุปกรณ์ใช้สำหรับปิดและเปิดวงจรเช่นเดียวกับสวิตช์แต่การทำงานของรีเลย์ทำงานด้วยการให้กระแสไฟฟ้าไหลผ่านเข้าไปในขดลวดของรีเลย์ทำให้เกิดอำนาจแม่เหล็กดูด-

ขั้วโลหะของรีเลย์ติดหรือขาดออกจากกันทำให้วงจรต่อกันหรือขาดออกจากกันเหมือนการปิดเปิดวงจรด้วยสวิตช์

รีเลย์ที่เลือกใช้งานคือ โซลิดสเตตรีเลย์ เป็นอุปกรณ์ที่ใช้เชื่อมต่อ (Interface) ระหว่างภาคควบคุม (Control) ซึ่งเป็นส่วนวงจรอิเล็กทรอนิกส์ กับวงจรภาคไฟฟ้ากำลัง (Power) โดยภาคทั้งสองจะมีระบบกราวด์ ที่แยกออกจากกันทำให้สามารถป้องกันการลัดวงจร และการรบกวนซึ่งกันและกันได้ โซลิดสเตตรีเลย์ อาจถือได้ว่าเป็นสิ่งประดิษฐ์ที่ออกแบบมาเพื่อใช้งานแทน อาร์เมเจอร์ริเลย์ แต่มีข้อดีกว่าคือมีขนาดเล็กกว่า มีความไวในการทำงานสูงกว่า มีอายุการใช้งานนานกว่า ฯลฯ



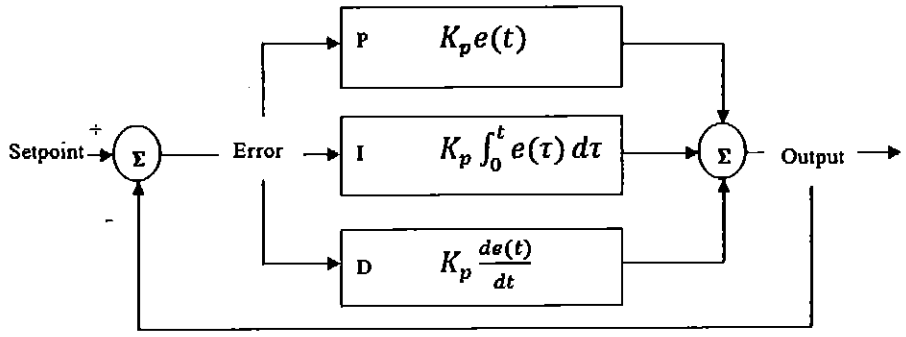
รูปที่ 2.10 โซลิดสเตตรีเลย์



รูปที่ 2.11 การต่อสายใช้งานโซลิดสเตตรีเลย์

2.5 การควบคุมแบบพีไอดี (PID Control)

การควบคุมแบบพีไอดี (PID Control) เป็นระบบควบคุมแบบป้อนกลับที่ใช้กันอย่างกว้างขวาง ซึ่งค่าที่นำไปใช้ในการคำนวณเป็นค่าความผิดพลาดที่หามาจากความแตกต่างของตัวแปรในกระบวนการและค่าที่ต้องการ ตัวควบคุมจะพยายามลดค่าผิดพลาดให้เหลือน้อยที่สุดด้วยการปรับค่าสัญญาณขาเข้าของกระบวนการ ค่าตัวแปรของการควบคุมแบบพีไอดี ที่ใช้จะปรับเปลี่ยนตามธรรมชาติของระบบ



รูปที่ 2.12 รูปแสดงแผนภาพบล็อกของการควบคุมแบบพีไอดี [4]

2.5.1 ทฤษฎีการควบคุมแบบพีไอดี [4]

ทฤษฎีการควบคุมแบบพีไอดี ได้ชื่อตามการรวมกันของเทอมของตัวแปรทั้งสามตามสมการ

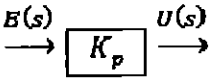
$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{d}{dt} e(t) \quad (2.1)$$

ถ้าเขียนอยู่ในรูปแบบผลการแปลงลาปลาซ จะได้ว่า

$$G(s) = K_p + \frac{K_i}{s} + K_d s \quad (2.2)$$

2.5.2 การควบคุมแบบสัดส่วน (Proportion Control) [4]

การควบคุมระบบแบบป้อนกลับโดยใช้ตัวควบคุมแบบสัดส่วนนั้น สัญญาณควบคุม (u) จะเป็นสัดส่วนโดยตรงกับค่าสัญญาณความผิดพลาด (e) ที่เกิดจากผลต่างระหว่างค่าสัญญาณอ้างอิงกับสัญญาณเอาท์พุตของระบบที่ทำการควบคุมนั้นดังแสดงในแผนภาพบล็อกดังนี้



รูปที่ 2.13 รูปแสดงแผนภาพบล็อกของการควบคุมแบบสัดส่วน [4]

ความสัมพันธ์ระหว่างสัญญาณเอาท์พุตจากตัวควบคุมและสัญญาณผิดพลาดที่ส่งเข้าไปในระบบสามารถเขียนได้ดังนี้

$$u(t) = K_p e(t) \quad (2.3)$$

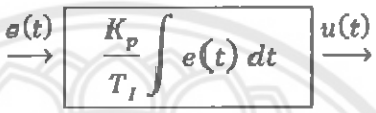
ถ้าเขียนอยู่ในรูปแบบผลการแปลงลาปลาซ จะได้ว่า

$$\frac{U(s)}{E(s)} = K_p \tag{2.4}$$

เมื่อค่าเกน K_p จะเป็นค่าอัตราขยายของตัวควบคุมนี้หรือจะเรียกว่าเกนสัดส่วน

2.5.3 การควบคุมแบบอินทิกรัล (Integral Control)

การควบคุมแบบอินทิกรัล (Integral Control) นั้นเพื่อต้องการลดค่าความผิดพลาดในสถานะอยู่ตัว ในขณะเดียวกันค่าความมีเสถียรภาพของระบบก็จะลดน้อยลง การควบคุมนั้นดังแสดงในแผนภาพบล็อกดังนี้



รูปที่ 2.14 รูปแสดงแผนภาพบล็อกของการควบคุมแบบอินทิกรัล [4]

$$u(t) = \frac{K_p}{T_I}\int e(t)\,dt \tag{2.5}$$

หรือในรูปแบบของสมการฟังก์ชันถ่ายโอน จะได้ว่า

(2.6)

ถ้าเขียนอยู่ในรูปแบบผลการแปลงลาปลาซ จะได้ว่า

$$\frac{U(s)}{E(s)} = \frac{K_p}{T_I s} \tag{2.7}$$

โดยที่ T_I = ช่วงเวลาอินทิกรัล

$\frac{1}{T_I}$ = อัตราการตั้งต้นใหม่ (reset rate)

การควบคุมแบบอินทิกรัล (Integral control) นี้จะเห็นว่าสัญญาณควบคุม u(t) อาจจะมีค่าค่อนข้างมากก็ได้โดยไม่ลดลงทั้งๆ ที่สัญญาณผิดพลาด e(t) มีค่าเป็นศูนย์ในภายหลังหรือเมื่อเวลาผ่านไปแล้วก็ตาม ทั้งนี้ก็เพราะว่าสัญญาณควบคุมในกรณีของการควบคุมแบบอินทิกรัล (Integral control) ขึ้นอยู่กับค่าในอดีต (past value) ไม่เหมือนกับตัวควบคุมเชิงสัดส่วน ซึ่งอยู่กับค่าปัจจุบัน

2.5.4 การควบคุมแบบอนุพันธ์ (Derivative Control) [4]

การควบคุมแบบอนุพันธ์ (Derivative Control) นี้จะช่วยในการเพิ่มค่าความหน่วง (damping) ให้กับระบบที่ต้องการจะควบคุม นั่นก็คือทำให้ระบบมีเสถียรภาพเพิ่มมากขึ้น แต่โดยทั่วไปแล้วตัวควบคุมเชิงอนุพันธ์นี้จะไม่ทำให้ค่าความผิดพลาดในสถานะอยู่ตัวมีค่าเป็นศูนย์ได้ ในขณะเดียวกันจะเห็นว่าสัญญาณเอาท์พุตที่ออกจากตัวควบคุมเชิงอนุพันธ์นี้เป็นสัญญาณที่เกิดจากการหาอนุพันธ์ของสัญญาณผิดพลาด $\frac{d(e(t))}{dt}$ การควบคุมนั้นดังแสดงในแผนภาพบล็อกดังนี้

$$e(t) \longrightarrow \boxed{K_p T_D \frac{d(e(t))}{dt}} \longrightarrow U(t)$$

รูปที่ 2.15 รูปแสดงแผนภาพบล็อกของการควบคุมแบบอนุพันธ์ [4]

$$u(t) = K_p T_D \frac{d(e(t))}{dt} \tag{2.8}$$

หรือในรูปแบบของสมการฟังก์ชันถ่ายโอน จะได้ว่า

$$E(s) \longrightarrow \boxed{K_p T_D S} \longrightarrow U(s) \tag{2.9}$$

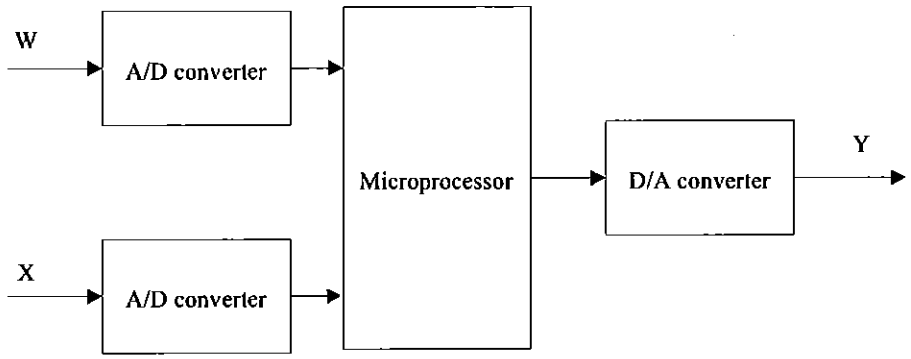
ถ้าเขียนอยู่ในรูปแบบผลการแปลงลาปลาซ จะได้ว่า

$$\frac{U(s)}{E(s)} = K_p T_D S \tag{2.10}$$

โดยที่ค่า T_D = ช่วงเวลาอนุพันธ์

การควบคุมแบบอนุพันธ์ (Derivative control) นี้ส่วนมากแล้วจะใช้ร่วมกับตัวควบคุมตัวอื่น หากสัญญาณผิดพลาดนี้มีสัญญาณรบกวนมาก สัญญาณเอาท์พุตที่ออกมาจากตัวควบคุมเชิงอนุพันธ์นี้จะกระเพื่อม (fluctuate) ค่อนข้างมาก (เนื่องจากค่าความชัน (slope) ของสัญญาณเปลี่ยนแปลงค่อนข้างมาก) ซึ่งจะทำให้ระบบควบคุมของเรามีเสถียรภาพได้

2.5.5 ตัวควบคุม พีไอดีแบบดิจิตอล [3]



รูปที่ 2.16 รูปแสดงแผนภาพการไหลของสัญญาณสำหรับตัวควบคุมแบบดิจิตอล [3]

ในรูปที่ 2.16 แสดงให้เห็นการไหลของสัญญาณสำหรับตัวควบคุมแบบดิจิตอล โดยที่ค่าของตัวแปลอ้างอิงและตัวแปลที่ถูกควบคุมต่างก็จะถูกอ่านค่าโดยผ่านตัวแปลงสัญญาณอานาลอกเป็นสัญญาณดิจิตอล ซึ่งตัวแปลงสัญญาณจะทำให้เกิดตัวแปรเอาท์พุตในลักษณะของค่าตัวเลข จากนั้นตัวไมโครโปรเซสเซอร์ก็จะปฏิบัติงานตามขั้นตอนดังต่อไปนี้

1) ทำการอ่านค่าของตัวแปรอ้างอิง และตัวแปรที่ถูกควบคุม

2) ทำการเปรียบเทียบค่าของตัวแปรอ้างอิง และตัวแปรที่ถูกควบคุม

3) ทำการประมวลผลสัญญาณ

4) ค่าเอาท์พุตจากการประมวลผลจะถูกส่งไปที่อินพุตของตัวแปลงสัญญาณดิจิตอลเป็นสัญญาณอานาลอก ต่อไป

ตัวแปลงสัญญาณดิจิตอลเป็นสัญญาณ อานาลอกจะทำให้ข้อมูลที่ได้จากการประมวลผลของไมโครโปรเซสเซอร์ ซึ่งถูกกำหนดเป็นค่าตัวแปรแก้ไขปรับกระบวนการ ถูกแปลงให้ไปอยู่ในรูปของสัญญาณแรงดันไฟฟ้า

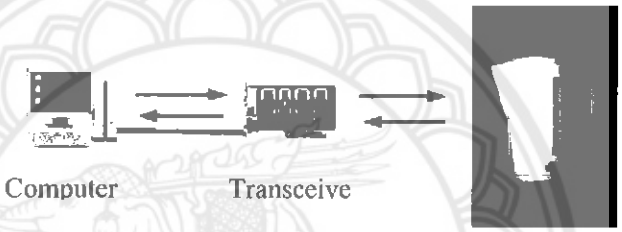
การปฏิบัติการทั้งในส่วนของการอ่านค่าสัญญาณ การประมวลผลและการส่งสัญญาณเอาท์พุตออกไปจะถูกกระทำให้เกิดขึ้นซ้ำๆกันตั้งแต่ 20 ครั้ง ถึงจะเป็นจำนวนหลายพันครั้งต่อวินาที ทั้งนี้ขึ้นอยู่กับชนิดของตัวควบคุมที่ถูกนำมาใช้

จากทฤษฎีที่ได้ศึกษามาแล้วทั้งหมดใน บทที่ 2 ทำให้มีความรู้ในเรื่อง คุณสมบัติ,การทำงาน และการต่อใช้งานของไอซีที่นำมาใช้ในโครงงานคือ ไมโครคอนโทรลเลอร์ตระกูล AVR(ATmega32),เซนเซอร์วัดอุณหภูมิ DS1820, รีเลย์ และมีความรู้ในเรื่องของทฤษฎีการควบคุมแบบพีไอดี เพื่อนำไปใช้ในการเขียนโปรแกรมภาษาซี เพื่อใช้ควบคุมการทำงาน และสร้างชิ้นงานใน บทที่ 3 ต่อไป

บทที่ 3
การประดิษฐ์เครื่องต้มน้ำควบคุมอุณหภูมิแบบพีไอดี

บทนี้อธิบายถึงการประดิษฐ์เครื่องต้มน้ำควบคุมอุณหภูมิแบบพีไอดีโดยนำหลักการและทฤษฎีจากบทที่ 2 มาใช้ในการดำเนินงานซึ่งมีหัวข้อหลัก ดังนี้

1. การออกแบบวงจรควบคุมระบบกาต้มน้ำด้วยไมโครคอนโทรลเลอร์
2. การเขียนโปรแกรมควบคุมการทำงานของไมโครคอนโทรลเลอร์
3. การประกอบบอร์ดควบคุมไมโครคอนโทรลเลอร์เข้ากับกาต้มน้ำ

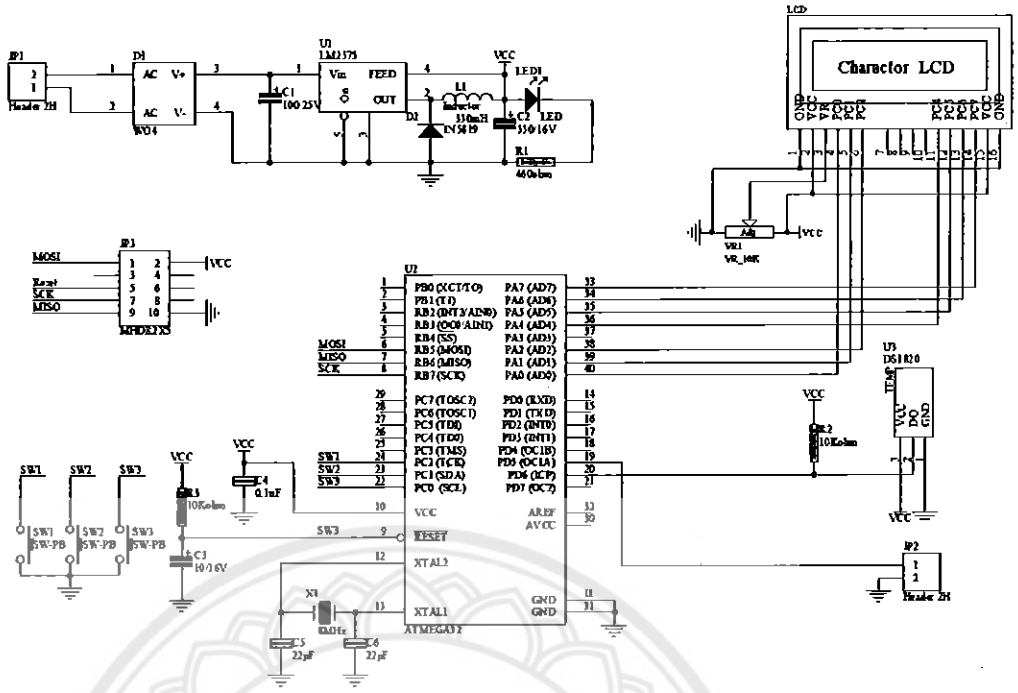


รูปที่ 3.1 ภาพรวมของระบบทั้งหมดในโครงงาน

หลักการทำงานของระบบในรูปที่ 3.1 นั้น เริ่มจากการเขียนโปรแกรมข้อมูลสั่งการจากคอมพิวเตอร์ ผ่านโปรแกรม Code Vision AVR C Compiler โดยการเขียนข้อมูลมาอยู่ในรูปแบบของภาษาซี แล้วนำส่งข้อมูลผ่านพอร์ตอนุกรม (RS-232) หลังจากที่คอมพิวเตอร์ได้ส่งข้อมูลออกมาทางพอร์ตอนุกรมแล้ว ทางด้านฝั่งรับข้อมูลจะเชื่อมต่อเข้ากับไมโครคอนโทรลเลอร์ตระกูล AVR เพื่อรับข้อมูลจากคอมพิวเตอร์เพื่อควบคุมการทำงานของอุปกรณ์ภายนอก คือ การรับค่าอุณหภูมิจากสวิตช์ การวัดค่าอุณหภูมิปัจจุบันด้วยไอซี DS1280 หลังจากนั้นจึงควบคุมการทำงานของขดลวดนิโครมภายในกาต้มน้ำเพื่อต้มน้ำให้ได้ตามอุณหภูมิที่ตั้งค่าไว้

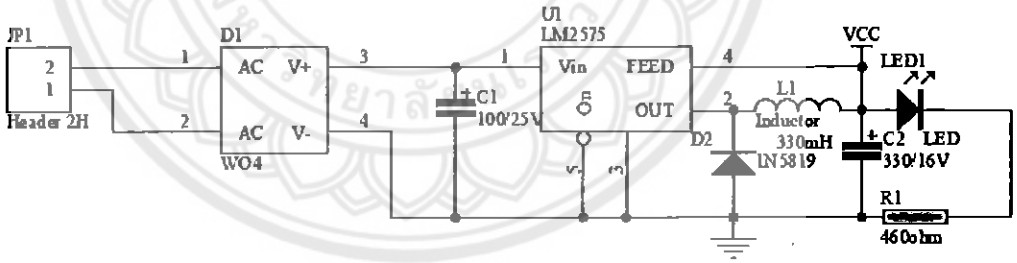
3.1 ออกแบบวงจรควบคุมระบบกาต้มน้ำด้วยไมโครคอนโทรลเลอร์

อุปกรณ์รับข้อมูลจากคอมพิวเตอร์ในโครงงานนี้คือ วงจรควบคุมอุณหภูมิด้วยการทำงานของไมโครคอนโทรลเลอร์ตระกูล AVR (ATmega 32) มีการทำงานรับข้อมูลจากคอมพิวเตอร์เป็นข้อมูลอักขระตัวอักษร เพื่อมาแปลงเป็นเลขฐาน 16 ใช้ในการสั่งการและแสดงผลและส่งข้อมูลต่างๆไปยังอุปกรณ์ควบคุมการทำงานในการต้มน้ำตามรูปภาพที่ 3.2



รูปที่ 3.2 รูปวงจรควบคุมอุณหภูมิด้วยไมโครคอนโทรลเลอร์

3.1.1 วงจรไฟเลี้ยงไมโครคอนโทรลเลอร์



รูปที่ 3.3 วงจรไฟเลี้ยงไมโครคอนโทรลเลอร์

- JP1 เป็นคอนเน็คเตอร์สำหรับต่อไฟเลี้ยงเข้าตั้งแต่ 7-40 โวลต์ DC หรือ AC ก็ได้ เพราะมีบริดจ์ เป็นตัวแปลงกระแสไฟ

- บริดจ์ มีหน้าที่แปลงไฟจากไฟกระแสสลับให้เป็นกระแสตรงเมื่อต่อไฟ AC เข้า แต่เมื่อต่อไฟ DC เข้า บริดจ์จะมีหน้าที่กลับไฟลบให้เป็นไฟบวกหากต่อผิดขั้ว

- C1 คือตัวเก็บประจุมีหน้าที่กรองแรงดันไฟให้เรียบเมื่อผ่าน บริดจ์ในกรณีที่ต่อไฟ AC เข้า แต่ถ้าต่อไฟ DC เข้าไม่จำเป็นต้องต่อ C1 ก็ได้

- LM2575-5 โวลต์ มีหน้าที่พยุงแรงดันไฟฟ้าเมื่อเข้ามา 7-40 โวลต์ จะทำให้เอาท์พุตออกมา 5 โวลต์

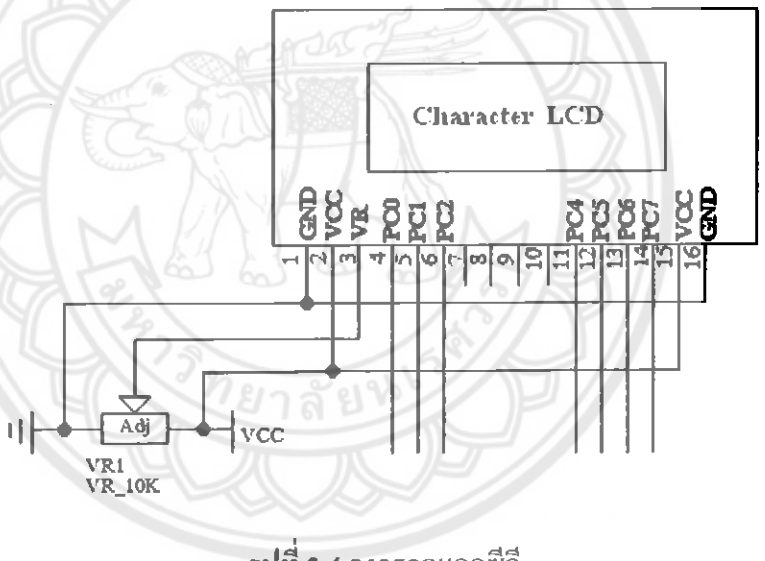
- D2 คือไดโอดมีหน้าที่ป้องกันกระแสย้อนกลับจาก L1 ที่เป็นตัวเหนี่ยวนำ

- L1 คือตัวเหนี่ยวนำซึ่งทำหน้าที่เป็นตัวหน่วงกระแสเพื่อไม่ให้แรงดันตกขณะที่ไฟ 5 โวลต์ จ่ายโหลดมากเกินความต้องการ

- C2 คือตัวเก็บประจุมีหน้าที่กรองแรงดันเอาท์พุตของ 5โวลต์ ที่ออกมาจาก LM2575

- LED1 เป็นแอลอีดีแสดงสถานะเพื่อให้รู้ว่ามีไฟ 5โวลต์ ออกมาจาก LM2575 หรือไม่ ส่วน R1 (ตัวต้านทาน) มีหน้าที่เป็นตัวต้านทานกระแสไม่ให้กระแสไหลผ่านหลอดแอลอีดีมากเกินไป ซึ่งจะทำให้หลอดแอลอีดีขาดได้

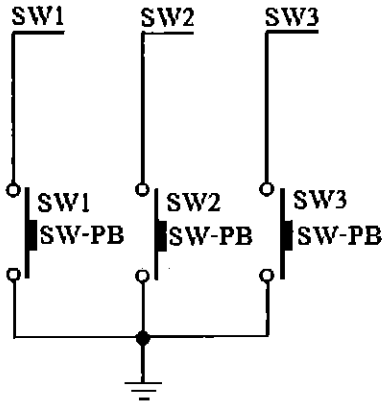
3.1.2 การต่อใช้งาน จอแอลซีดี 16x2



รูปที่ 3.4 วงจรจอแอลซีดี

จอแอลซีดีที่นำมาใช้งานเป็นจอแอลซีดีขนาด 16x2 คือ16 ตัวอักษร 2 บรรทัดในการต่อกับไมโครคอนโทรลเลอร์ ที่เป็นการต่อแบบ 4 บิต

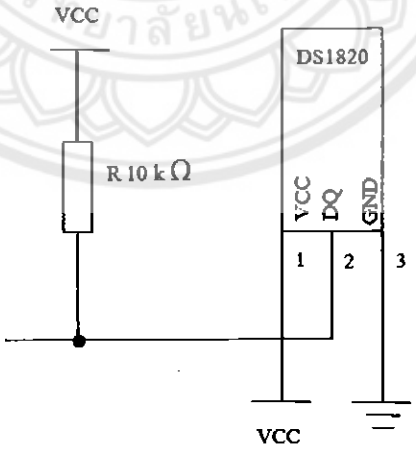
3.1.3 การต่อใช้งานสวิตช์



รูปที่ 3.5 วงจรสวิตช์

สวิตช์ เป็นสวิตช์แบบกดติดปล่อยดับ มี สวิตช์หนึ่ง สวิตช์สอง และสวิตช์สาม หลักการทำงาน คือ เมื่อมีการกดสวิตช์หนึ่ง อุณหภูมิจะเพิ่มขึ้นทีละ 0.5 องศาเซลเซียส ต่อการกดสวิตช์หนึ่งครั้ง เมื่อกดสวิตช์สอง อุณหภูมิจะลดลงทีละ 0.5 องศาเซลเซียสต่อการกดสวิตช์หนึ่งครั้ง และสวิตช์สามใช้ในการรีเซ็ต เพื่อเริ่มต้นการทำงานใหม่อีกครั้ง

3.1.4 การต่อใช้งานไอซีวัดอุณหภูมิ DS1820

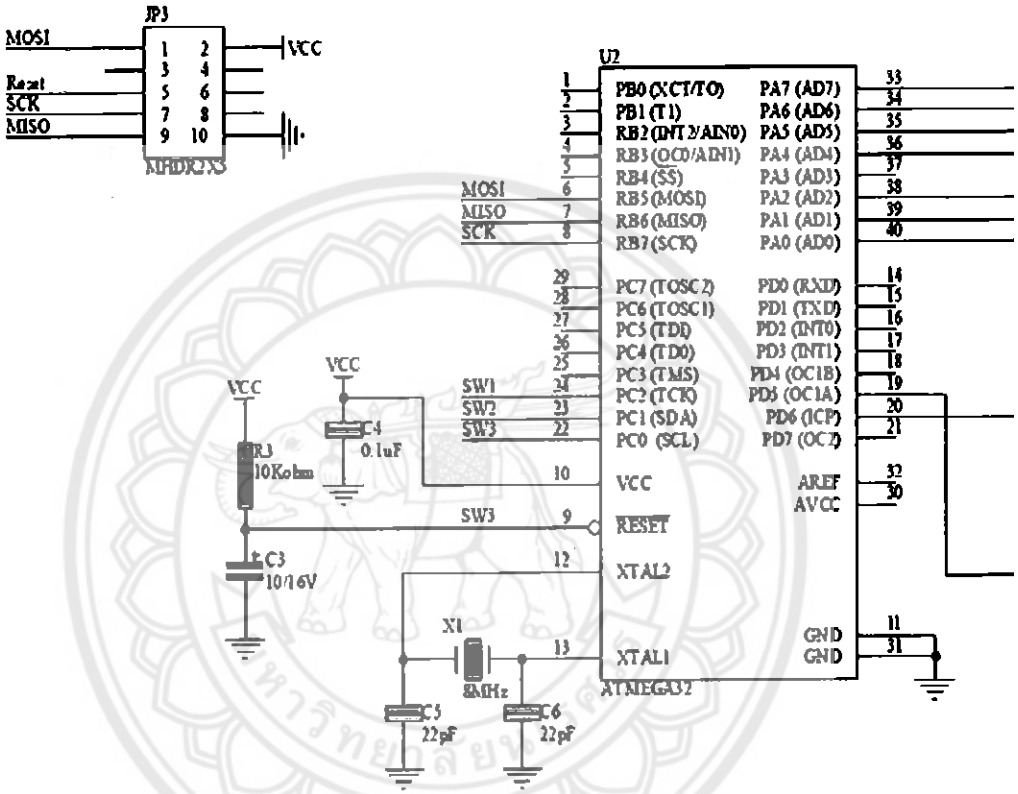


รูปที่ 3.6 วงจร DS1802 เป็นตัวตรวจวัดอุณหภูมิ

ไอซี DS1802 เป็นตัวตรวจวัดอุณหภูมิแบบสายเดียว มีสามขาคือ ขาแหล่งจ่ายไฟ ขากราวด์ และขารับข้อมูลค่าของอุณหภูมิ ซึ่งเป็นขารับข้อมูลสายเดียว ซึ่งทำหน้าที่ตรวจวัดอุณหภูมิ

ของน้ำว่ามีค่ากี่องศาเซลเซียส และนำค่าอุณหภูมิส่งไปยังไมโครคอนโทรลเลอร์เพื่อเก็บไว้ คำนวณหาค่าความแตกต่างระหว่างอุณหภูมิของน้ำขณะนั้นกับค่าอุณหภูมิของน้ำที่เราได้กำหนดไว้ เพื่อนำค่าที่ได้ไปใช้ในการควบคุมแบบพีไอดีต่อไป

3.1.5 การต่อใช้งานไมโครคอนโทรลเลอร์ตระกูล AVR (ATmega32)



รูปที่ 3.7 วงจรใช้งานไมโครคอนโทรลเลอร์

U2 เป็นไมโครคอนโทรลเลอร์ตระกูล AVR (ATmega32) ไมโครคอนโทรลเลอร์จะทำงานได้ต้องอาศัย 3 อย่างหลักๆ คือ

1. ไฟเลี้ยงไมโครคอนโทรลเลอร์

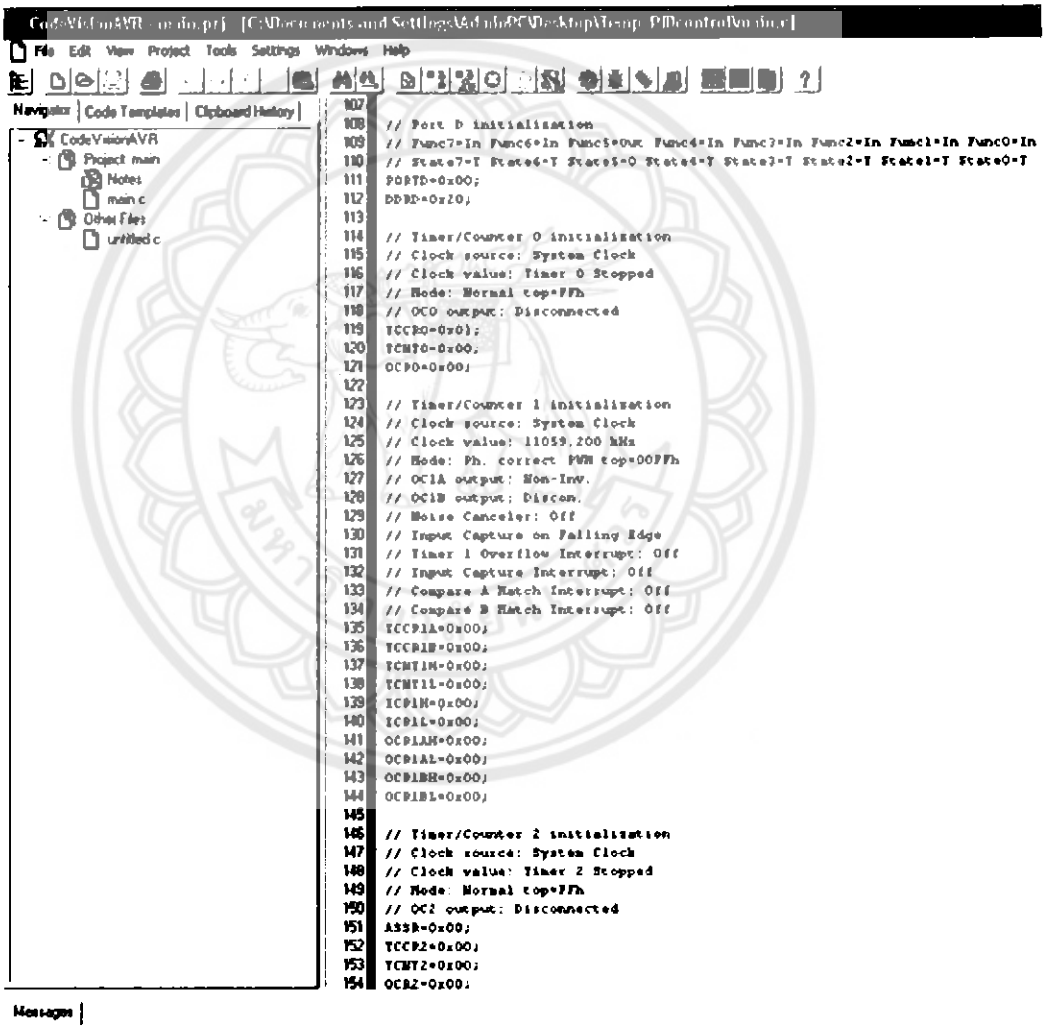
2. X-TAL หรือ คริสตัล มีหน้าที่สร้างสัญญาณนาฬิกาให้กับไมโครคอนโทรเลอร์ C5 (ตัวเก็บประจุ) และ C6 (ตัวเก็บประจุ) มีหน้าที่กรองสัญญาณนาฬิกา (CRYSTAL) ก่อนที่จะเข้าไปยังไมโครคอนโทรลเลอร์ C4 (ตัวเก็บประจุ) มีหน้าที่กรองสัญญาณรับกวนที่จะเกิดขึ้นกับไมโครคอนโทรลเลอร์

3. R3 (ตัวต้านทาน) และ C3 (ตัวเก็บประจุ) เป็นวงจรรีเซตไมโครคอนโทรลเลอร์ ได้มาจากแผ่นข้อมูลของไมโครคอนโทรลเลอร์ส่วน SW1, SW2, SW3 เป็นการต่อสวิตช์แบบแอคทีฟ

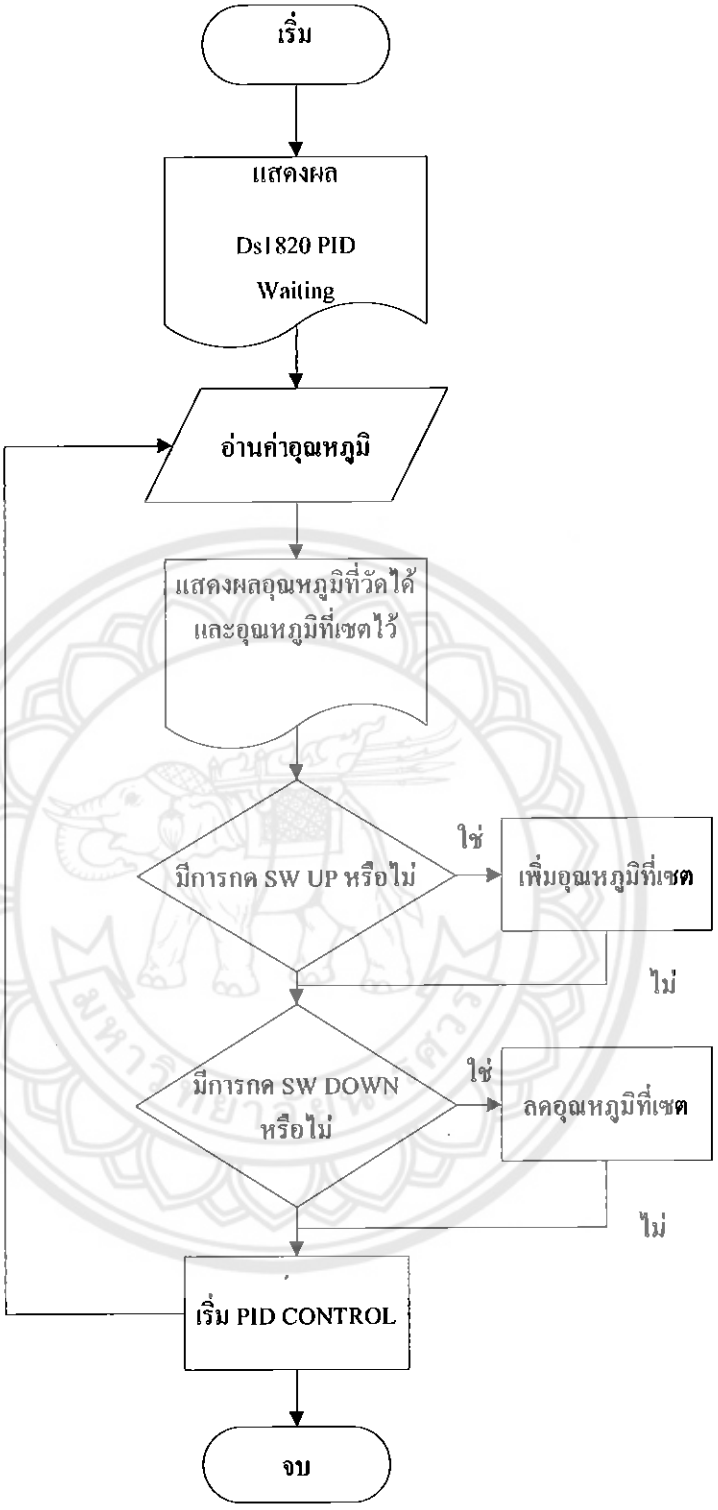
GND (กราวด์) คือการรับสัญญาณ GND (กราวด์) และ JP3 เป็นการเชื่อมต่อไว้สำหรับดาวน์โหลดโปรแกรมเข้าไมโครคอนโทรลเลอร์โดยผ่านสาย ISP Down ของบริษัท ETT

3.2 การเขียนโปรแกรมควบคุมการทำงานของไมโครคอนโทรลเลอร์

การทำงานของระบบควบคุมกาต้มน้ำแบบพีไอดีคอนโทรลซึ่งใช้โปรแกรมภาษาซีในการเขียนคำสั่งควบคุมไมโครคอนโทรลเลอร์ตระกูล AVR มีรูปแบบของโปรแกรมที่ใช้ และแผนผังการทำงานของโปรแกรม ตามที่แสดงไว้ในรูปที่ 3.8 และ 3.9



รูปที่ 3.8 รูปโปรแกรม Code Vision AVR C Compiler



รูปที่ 3.9 การทำงานของโปรแกรม

เริ่มต้นโปรแกรมจะแสดงผล Ds1820 PID waiting ที่หน้าจอแอลซีดีเพื่อแสดงว่าโปรแกรมเริ่มทำงานแล้วและจะเริ่มอ่านค่าอุณหภูมิที่ตั้งไว้กับอ่านค่าอุณหภูมิในน้ำที่วัดได้

ขณะนั้นเพื่อนำค่าอุณหภูมิไปแสดงผลบนจอแอลซีดี หลังจากนั้นโปรแกรมจะตรวจสอบว่ามีการกดสวิตช์เพิ่มอุณหภูมิหรือไม่ ถ้ามีการกดสวิตช์เพิ่มอุณหภูมิ อุณหภูมิที่ตั้งค่าไว้ก็จะเพิ่มทีละ 0.5 องศาเซลเซียสต่อการกดสวิตช์หนึ่งครั้ง แต่ถ้าไม่มีการกดสวิตช์เพิ่มอุณหภูมิโปรแกรมก็จะทำงานในขั้นตอนต่อไป คือ ตรวจสอบว่ามีการกดสวิตช์ลดอุณหภูมิหรือไม่ ถ้ามีการกดสวิตช์ลดอุณหภูมิ อุณหภูมิที่ตั้งค่าไว้ก็จะถูกลดค่า 0.5 องศาเซลเซียส ต่อการกดสวิตช์หนึ่งครั้ง หลังจากนั้นก็นำค่าอุณหภูมิที่วัดได้จากน้ำ และค่าอุณหภูมิที่ได้ตั้งค่าไว้ เข้าสู่โปรแกรมการคำนวณของการควบคุมแบบพีไอดี และวนกลับไปเริ่มอ่านค่าอุณหภูมิใหม่อีกครั้ง จนอุณหภูมิที่วัดได้จากน้ำมีค่าเท่ากับอุณหภูมิที่ตั้งค่าไว้ จึงจบการทำงาน

3.2.1 การทำงานของโปรแกรมการควบคุมแบบพีไอดี [4]

หลังจากดำเนินการทำอุปกรณ์รับส่งสัญญาณในระบบเรียบร้อยแล้ว ต่อไปเป็นขั้นตอนการดำเนินการคำนวณโดยใช้สมการ การควบคุมแบบพีไอดี ควบคุมไมโครคอนโทรลเลอร์ เพื่อเข้าไปสั่งการอุปกรณ์กาต้มน้ำเพื่อต้มน้ำให้ได้อุณหภูมิที่ต้องการ

การสร้างตัวควบคุมในที่นี้พิจารณาในรูปแบบของการใช้ฮาร์ดแวร์ที่เป็นไมโครคอนโทรลเลอร์ จากเนื้อหาเรื่องทฤษฎีระบบควบคุมที่นำเสนอก่อนหน้านี้ ทำให้ทราบว่าตัวควบคุมพีไอดี สามารถได้รับการเขียนแสดงในรูปความสัมพันธ์ทางคณิตศาสตร์ว่า

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d}{dt} e(t) \quad (3.1)$$

เมื่อ u(t) เป็นสัญญาณเอาท์พุตของตัวควบคุมและ e(t) เป็นสัญญาณค่าผิดพลาดที่เกิดขึ้นของระบบควบคุมตัวควบคุม สมการพีไอดีแบบดิจิตอลที่ใช้ควบคุม จึงแทนได้ด้วยความสัมพันธ์ดังสมการนี้

$$u(t) = (kp \times error) + (ki \times integral) + (kd \times derivative) \quad (3.2)$$

error คือ ค่าผิดพลาด ณ เวลาปัจจุบันมีค่าเป็น

$$error = setpoint - actual\_position \quad (3.3)$$

setpoint คือ ค่าที่กำหนด

actual_position คือ ค่าความอุณหภูมิจริงขณะนั้น

integral คือ ค่าความผิดพลาดของระบบที่ถูกอินทิเกรท สามารถเขียนแทนด้วยสมการ

$$integral = integral + (error \times dt) \quad (3.4)$$

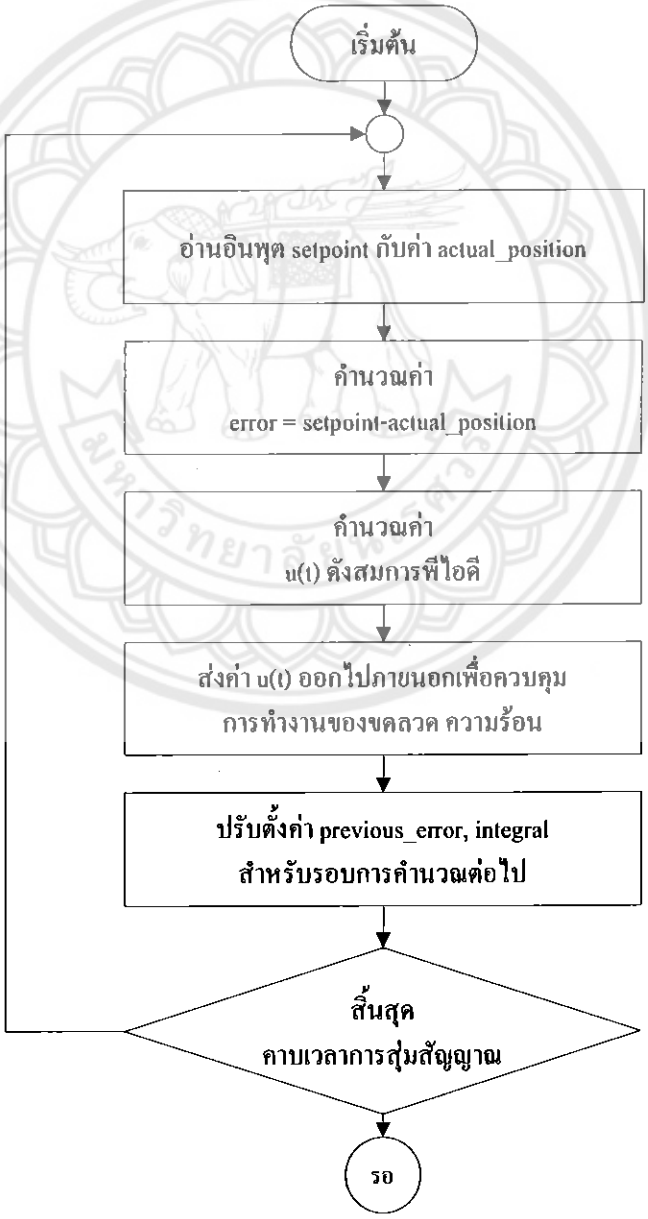
โดยที่ค่า integral เริ่มต้นจากศูนย์ตามการคำนวณของโปรแกรม และค่า dt คือคาบเวลาการสุ่มสัญญาณ

derivative คือ การหาอนุพันธ์ของค่าความผิดพลาดของระบบ เขียนแทนด้วยสมการ

$$derivative = \frac{(error - previous\_error)}{dt} \quad (3.5)$$

previous_error คือ ค่าความผิดพลาดก่อนหน้า โดยมีค่าเริ่มต้นเท่ากับค่า error ณ ปัจจุบัน

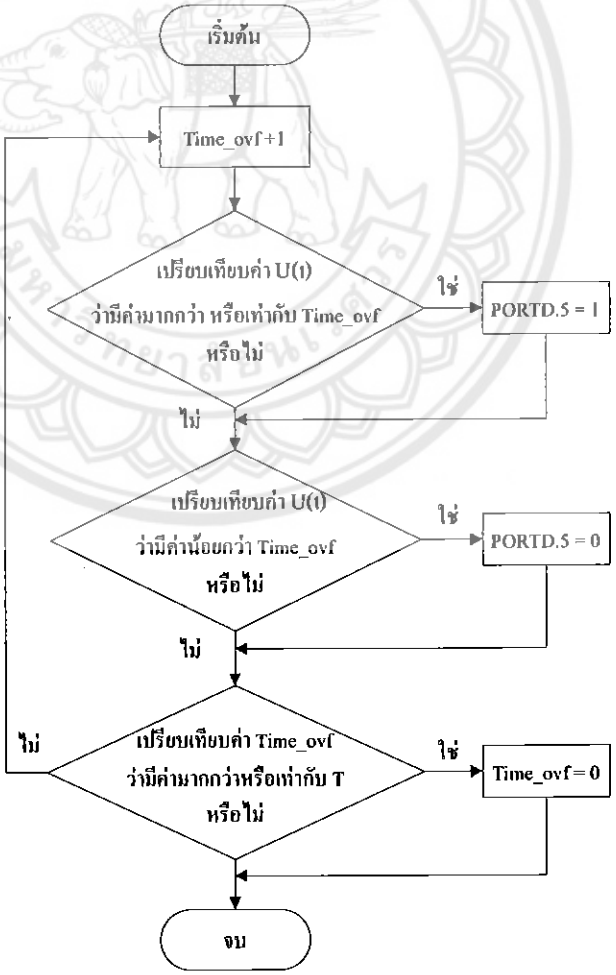
สามารถเขียนแผนภูมิการทำงานของโปรแกรมได้ดังนี้โดยการคำนวณแต่ละรอบจะต้องจำกัดอยู่ในคาบเวลาของการสุ่มสัญญาณ



รูปที่ 3.10 แผนภูมิการทำงานของโปรแกรมควบคุมพีไอดีแบบดิจิตอล **[4]**

การทำงานของโปรแกรมควบแบบคุมพีไอดี เริ่มจากการอ่านค่าอินพุต สองค่าคือค่าอุณหภูมิที่ตั้งไว้ และ อุณหภูมิจริงของน้ำขณะนั้น หลังจากนั้นนำสองค่าที่อ่านได้ไปคำนวณหาค่าความผิดพลาด โดยนำค่าอุณหภูมิที่ตั้งไว้ลบออกด้วยค่าอุณหภูมิที่วัดได้ขณะนั้น และนำค่าความผิดพลาดที่คำนวณได้ ไปเข้าสู่การคำนวณของทฤษฎีการควบคุมแบบพีไอดีเพื่อหาค่า u(t) (ตัวแปรแก้ไขระบบ) หลังจากนั้นนำค่า u(t) ที่ได้จากการคำนวณของโปรแกรมไปควบคุมการทำงานของขดลวดความร้อนภายในกาต้มน้ำเพื่อต้มน้ำให้ได้อุณหภูมิที่ตั้งค่าไว้เมื่อจบการทำงาน 1 คาบคือหนึ่งรอบการทำงานของโปรแกรม โปรแกรมพีไอดีก็จะกลับไปเริ่มทำงานอ่านค่าอินพุตใหม่ คืออุณหภูมิที่ตั้งไว้ และ อุณหภูมิจริงของน้ำขณะนั้น เพื่อใช้ในการทำงานของโปรแกรมอีกครั้งจะทำงานแบบนี้ต่อไปเรื่อยๆจนกว่าจะได้ค่าตัวแปรแก้ไขระบบที่ทำให้ระบบการควบคุมอุณหภูมิของน้ำอยู่ในสภาวะสมดุลคืออุณหภูมิที่ตั้งไว้เท่ากับอุณหภูมิของน้ำที่วัดได้ในขณะนั้น

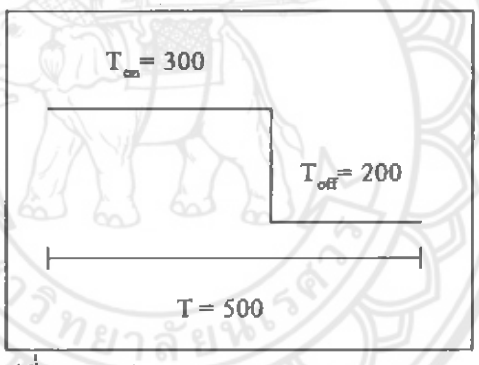
3.2.2 การเขียนโปรแกรมนำค่า U(t) ออกไปควบคุมอุปกรณ์ภายนอก



รูปที่ 3.11 แผนภูมิการทำงานของโปรแกรมนำค่า U(t) ออกไปควบคุมอุปกรณ์ภายนอก

การเขียนโปรแกรมกำหนดการทำงานของค่า U(t) กำหนดค่าตัวแปร T=500, U(t)=250 และ ตัวแปร Time_ovf ซึ่งชุดโปรแกรมนี้จะทำงานในรูปแบบของการอินเทอร์รัพท์ มีการทำงานดังนี้

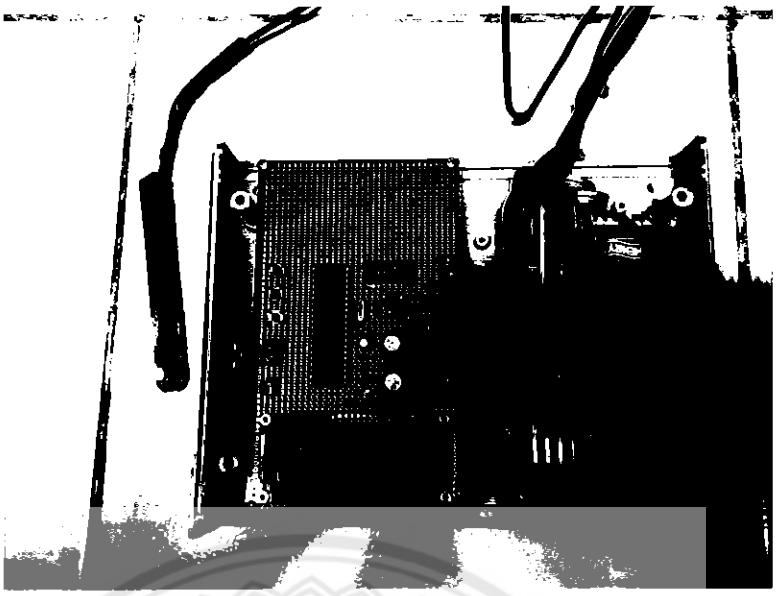
ให้ Time_ovf บวกหนึ่ง ต่อไปก็ทำการเปรียบเทียบค่า Time_ovf กับค่า U(t) ว่าค่า U(t) มีค่ามากกว่าหรือเท่ากับค่า Time_ovf หรือไม่ ถ้าใช่ก็ให้ส่งเอาท์พุตเท่ากับ 1 ออกทาง PORTD.5 แต่ถ้าค่า U(t) มีค่าน้อยกว่าค่า Time_ovf ก็ให้ส่งค่าเอาท์พุตท่ากับ 0 ออกทาง PORTD.5 ต่อไปก็ทำการเปรียบเทียบค่า Time_ovf กับค่า T ว่าค่า Time_ovf มีค่ามากกว่าหรือเท่ากับค่า T หรือไม่ ถ้าไม่ใช่ก็ให้ กลับไปเริ่มทำงานใหม่คือ บวกเพิ่มค่า Time_ovf ขึ้นจากเดิม +1 จนกว่าค่า Time_ovf จะมีค่ามากกว่าหรือเท่ากับค่า T ก็ให้กำหนดค่า Time_ovf เท่ากับ 0 จบการทำงาน จะได้การทำงานของขดลวดความร้อนอยู่ในรูปแบบของพัลส์วิดท์มอดูเลชั่น (Pulse Width Modulation) หรือ PWM ยกตัวอย่าง เช่น เมื่อเราคำนวณ ค่า U(t) จากสมการพีไอดีได้ค่าเท่ากับ 300 จะได้เอาท์พุตเป็น T_{on}= 300 ครั้ง และT_{off}= 200 ครั้ง ซึ่งการทำงานจะขึ้นอยู่กับค่า U(t) ที่คำนวณได้จากสมการพีไอดี คือค่า U(t)=T_{on} และได้รูป PWM ดังนี้



รูปที่ 3.12 รูปการทำงานแบบ PWM ของ U(t)

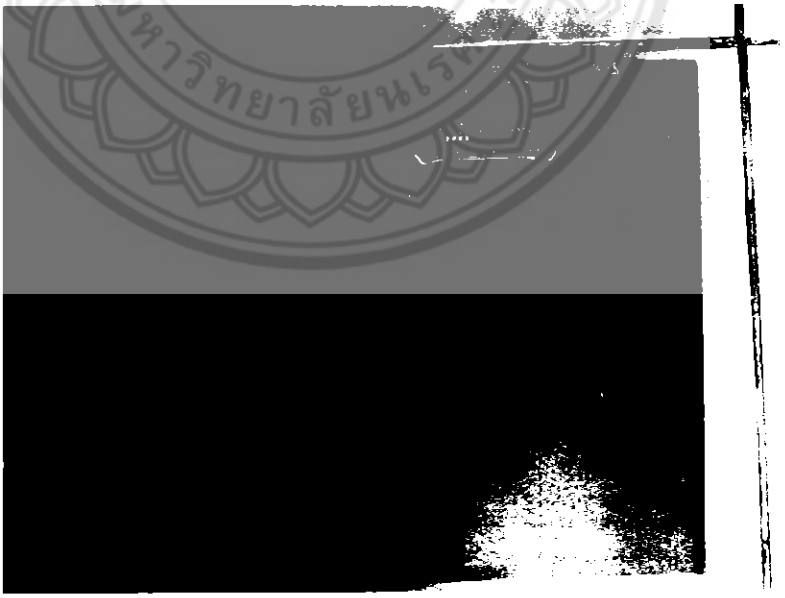
3.3 การประกอบบอร์ดควบคุมไมโครคอนโทรลเลอร์เข้ากับกาต้มน้ำ

หลังจากออกแบบลายวงจรของบอร์ดควบคุมไมโครคอนโทรลเลอร์กับการต่อใช้งานอุปกรณ์ต่างๆเรียบร้อยแล้ว จึงทำการต่ออุปกรณ์ทั้งหมดตามลายวงจรและตามวิธีการต่อใช้งานของไอซีแต่ละแบบลงในบอร์ด โดยต่อเชื่อมด้วยสายไฟ ใต้บอร์ดควบคุมดังแสดงในรูปที่ 3.13



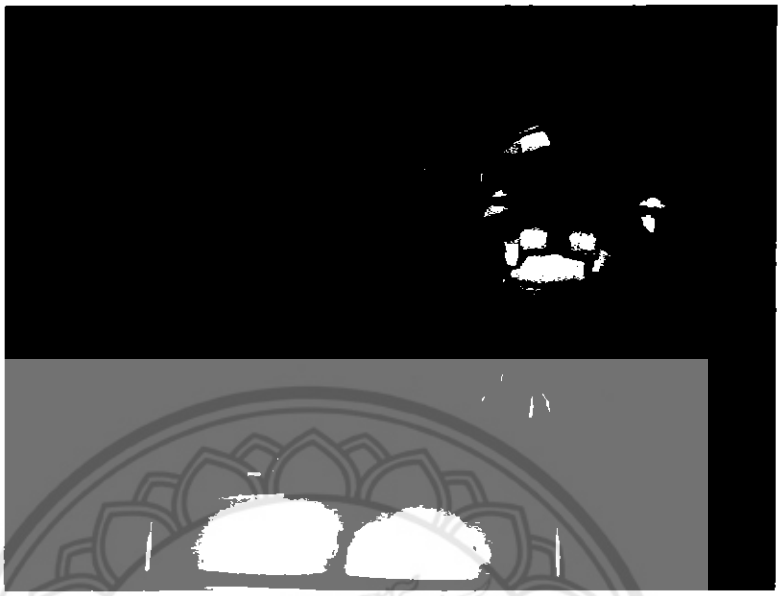
รูปที่ 3.13 บอร์ดควบคุมไมโครคอนโทรลเลอร์

เพื่อความเรียบร้อยของชิ้นงาน ชุดวงจรของตัวควบคุมไมโครคอนโทรลเลอร์ ที่ติดตั้งบนแผง วงจร จะถูกประกอบอยู่ในกล่องอุปกรณ์ที่จัดเตรียมไว้ ค่าอุณหภูมิของน้ำที่ตั้งค่าไว้ และค่าอุณหภูมิของน้ำที่วัดได้ขณะนั้นจะถูกแสดงผลบนหน้าจอแอลซีดี ซึ่งติดตั้งอยู่ด้านหน้าของกล่อง ดังแสดงในรูปที่ 3.14



รูปที่ 3.14 กล่องควบคุมไมโครคอนโทรลเลอร์

เมื่อนำกล่องของบอร์ดควบคุมไมโครคอนโทรลเลอร์ และอุปกรณ์วัดอุณหภูมิมา มาต่อเข้ากับกาต้มน้ำ จะได้เครื่องต้มน้ำควบคุมอุณหภูมิแบบพีไอดี ดังแสดงในรูปที่ 3. 15



รูปที่ 3.15 เครื่องต้มน้ำควบคุมอุณหภูมิแบบพีไอดี

จากการดำเนินการสร้างชิ้นงานใน บทที่ 3 โดยเริ่มจากการสร้างบอร์ดควบคุมไมโครคอนโทรลเลอร์เพื่อนำไปใช้ในการควบคุมระบบการทำงานของกาต้มน้ำ หลังจากได้บอร์ดควบคุมไมโครคอนโทรลเลอร์แล้ว ก็ทำการเขียนโปรแกรมภาษาซีควบคุมการทำงานของกาต้มน้ำ โดยใช้โปรแกรม (Code Vision AVR C Compiler) ช่วยในการเขียนโปรแกรมภาษาซี และใช้ทฤษฎีของการควบคุมแบบพีไอดีมาประกอบการเขียนโปรแกรม และเมื่อได้ค่า U(t) เรียบร้อยแล้ว ก็ส่งออกไปควบคุมการทำงานของขดลวดความร้อน โดยค่าเอาท์พุตที่ส่งออกไปเป็นแบบ PWM ดังที่ได้อธิบายไว้ หลังจากการดำเนินการทั้งหมดจนทำให้ได้ เครื่องต้มน้ำควบคุมแบบพีไอดี แล้ว ก็จะนำไปสู่การทดลองการปรับค่า Kp, Ki, Kd ใน บทที่ 4 ต่อไป

บทที่ 4
ผลการทดลองและผลการวิเคราะห์

บทนี้เป็นการดำเนินการปฏิบัติงานต่อจากบทที่ 3 เพื่อพิสูจน์ผลการทดลองว่าอุปกรณ์ดังกล่าวที่ทำขึ้นสามารถใช้งานได้จริง

4.1 การทดลองการทำงานของกาต้มน้ำควบคุมอุณหภูมิแบบพีไอดี

1) กำหนดค่า Kp, Ki และ Kd ที่ต้องการทำการทดลองโดยเริ่มจากการเปรียบเทียบค่า Kp ก่อน กำหนดให้ Ki และ Kd มีค่าเท่ากับศูนย์

2) นำน้ำใส่ในกาต้มน้ำ วัดระดับน้ำและอุณหภูมิขณะนั้นว่ามีค่ากี่องศาเซลเซียส

3) บันทึกผลค่าอุณหภูมิเริ่มต้นของน้ำ ก่อนทำการทดลองการทำงาน

4) เปิดเครื่องควบคุมอุณหภูมิแบบพีไอดี หลังจากนั้นกำหนดค่าอุณหภูมิที่ต้องการ และปล่อยให้ต้มน้ำไปจนครบ 1นาที แล้วทำการบันทึกค่าอุณหภูมิที่วัดได้

5) ทำการบันทึกค่าอุณหภูมิทุกนาทีไปจนครบ 55 นาที

6) ทำการทดลองซ้ำโดยกำหนดให้ เปรียบเทียบค่า Ki โดยที่ Kp และ Kd มีค่าคงที่ และเปรียบเทียบค่า Kd โดยที่ Kp และ Ki มีค่าคงที่ ตามลำดับ

4.2 ผลการทดลอง

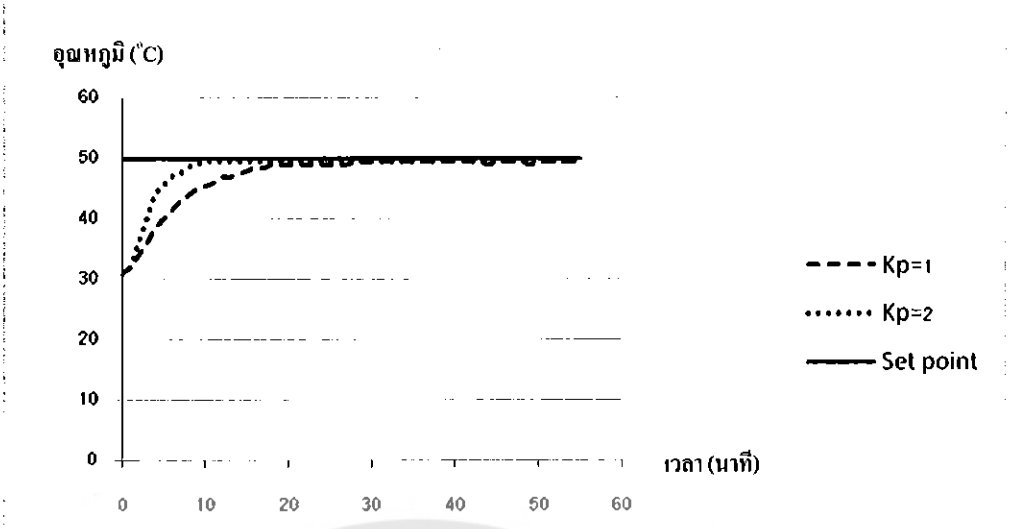
จากการทดลองโปรแกรมสามารถติดต่อกับไมโครคอนโทรลเลอร์ได้ตามปกติโดยผลการทดลองจะเป็นตามตารางที่ 4.1, 4.2, 4.3, 4.4 และ 4.5

ตารางที่ 4.1 ผลการทดลองเมื่อปรับค่า Kp=1 เทียบกับ Kp=2 เมื่อกำหนดให้ Ki=0, Kd=0

เวลา(นาที)	Kp=1 (°C)	Kp=2 (°C)	Set point (°C)
0	31	31	50
1	32	32	50
2	34	36	50
3	36	40	50

เวลา(นาที)	Kp=1 (°C)	Kp=2 (°C)	Set point (°C)
4	38.5	44	50
5	40	45.5	50
6	41.5	47	50
7	43	47.5	50
8	44	48.5	50
9	45	49	50
10	45.5	49.5	50
11	46	49.5	50
12	47	49.5	50
13	47	49.5	50
14	47.5	49.5	50
15	48	49.5	50
16	48.5	49.5	50
17	48.5	49.5	50
18	49	49.5	50
19	49	49.5	50
20	49	49.5	50
21	49	49.5	50
22	49	49.5	50
23	49	49.5	50
24	49	49.5	50
25	49	49.5	50
26	49	49.5	50
27	49	49.5	50
28	49.5	49.5	50
29	49.5	49.5	50
30	49.5	49.5	50
31	49.5	49.5	50
32	49.5	49.5	50

เวลา(นาที)	Kp=1 (°C)	Kp=2 (°C)	Set point (°C)
33	49.5	49.5	50
34	49.5	49.5	50
35	49.5	49.5	50
36	49.5	49.5	50
37	49.5	49.5	50
38	49.5	49.5	50
39	49.5	50	50
40	49.5	50	50
41	49.5	50	50
42	49.5	49.5	50
43	49.5	49.5	50
44	49	49.5	50
45	49.5	49.5	50
46	49.5	49.5	50
47	49.5	49.5	50
48	49.5	49.5	50
49	49	49.5	50
50	49.5	49.5	50
51	49.5	49.5	50
52	49.5	49.5	50
53	49.5	49.5	50
54	49.5	49.5	50
55	49.5	49.5	50



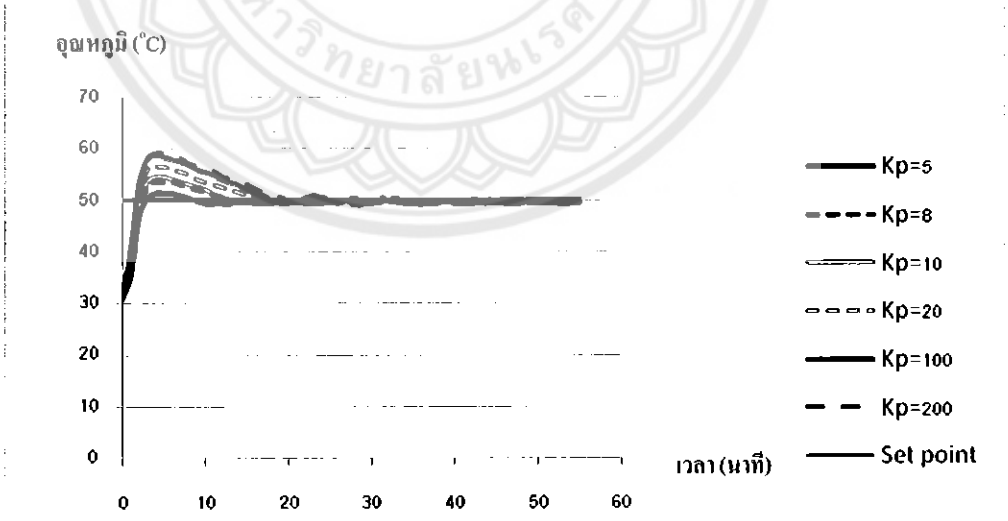
รูปที่ 4.1 กราฟแสดงผลการเปรียบเทียบผลอัตราขยายของ Kp

ตารางที่ 4.2 ผลการทดลองเปรียบเทียบ Kp=5, Kp=8, Kp=10, Kp=20, Kp=100, Kp=200 เมื่อ Ki=0, Kd=0

เวลา (นาที)	Kp=5 (°C)	Kp=8 (°C)	Kp=10 (°C)	Kp=20 (°C)	Kp=100 (°C)	Kp=200 (°C)	Set point (°C)
0	31	31	31	31	31	31	50
1	35.5	36	37	38	40	39.5	50
2	47	48	49	51	53	52.5	50
3	50.5	52.5	53.5	56	58	58	50
4	51.5	53.5	54.5	56.5	59	59	50
5	51.5	53.5	54.5	56.5	58.5	59	50
6	51.5	53.5	54	56	58	58.5	50
7	51	53	53.5	55.5	57.5	58	50
8	50.5	52.5	53	55	57	57	50
9	50	52	52.5	54	56	56.5	50
10	49.5	51.5	52	53.5	55.5	56	50
11	49.5	51	51	53	55	55.5	50
12	49.5	50.5	50.5	52.5	54	54.5	50

เวลา (นาที)	Kp=5 (°C)	Kp=8 (°C)	Kp=10 (°C)	Kp=20 (°C)	Kp=100 (°C)	Kp=200 (°C)	Set point (°C)
13	49.5	50	50	52	53.5	54	50
14	50	49.5	49.5	51.5	53	53.5	50
15	50	49.5	49.5	51	52	53	50
16	49.5	49.5	49.5	50.5	51.5	52	50
17	49.5	50	50	50	51	51.5	50
18	49.5	50	50	49.5	50	51	50
19	50	49.5	49.5	50	49.5	50.5	50
20	50	49.5	49.5	50	49.5	50	50
21	50	49.5	49.5	50	50	50	50
22	49.5	49.5	50	49.5	50.5	49.5	50
23	49.5	50	50	49.5	51	50	50
24	49.5	50	49.5	50	50.5	50.5	50
25	50	49.5	49.5	50	50	50	50
26	50	49.5	49.5	50	49.5	50	50
27	50	49.5	50	49.5	50	50	50
28	49.5	49.5	50	49	50.5	50	50
29	49.5	50	49.5	49	49.5	49.5	50
30	49.5	50	49.5	49.5	49.5	50	50
31	50	49.5	49.5	49.5	50	50.5	50
32	50	49.5	50	49.5	50	50	50
33	50	49.5	50	49.5	50.5	50	50
34	49.5	49.5	49.5	49.5	49.5	49.5	50
35	49.5	50	49.5	49.5	49.5	49.5	50
36	49.5	50	49.5	49	49.5	49.5	50
37	50	49.5	50	49	49.5	49.5	50
38	50	49.5	50	49.5	49.5	50	50
39	50	49.5	49.5	49.5	50	50	50
40	49.5	49.5	49.5	49.5	50	50	50
41	49.5	50	49.5	49.5	50	49.5	50

เวลา (นาที)	Kp=5 (°C)	Kp=8 (°C)	Kp=10 (°C)	Kp=20 (°C)	Kp=100 (°C)	Kp=200 (°C)	Set point (°C)
42	49.5	50	50	49.5	49.5	49.5	50
43	50	50	50	49.5	49.5	49.5	50
44	50	49.5	49.5	49.5	49.5	50	50
45	50	49.5	49.5	49.5	49.5	50	50
46	49.5	49.5	49.5	49.5	49.5	50	50
47	49.5	49.5	50	49.5	50	50	50
48	49.5	50	50	49.5	50	49.5	50
49	50	50	49.5	49.5	49.5	49.5	50
50	50	50	49.5	49.5	49.5	49.5	50
51	50	49.5	49.5	49.5	49.5	49.5	50
52	49.5	49.5	50	50	49.5	50	50
53	49.5	49.5	50	50	50	50	50
54	49.5	49.5	49.5	49.5	49.5	50	50
55	50	50	49.5	49.5	49.5	49.5	50

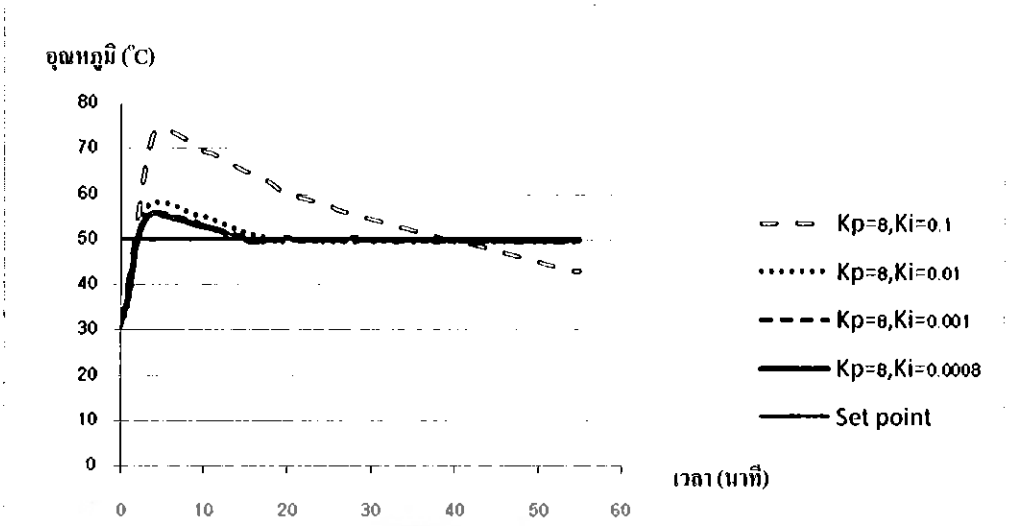


รูปที่ 4.2 กราฟแสดงผลการเปรียบเทียบผลอัตราขยายของ Kp

ตารางที่ 4.3 ผลการทดลองการเปรียบเทียบค่า Ki =0.1, Ki =0.01, Ki =0.001, Ki =0.0008 เมื่อกำหนดให้ Kp=8, Kd=0

เวลา (นาที)	Ki=0.1 (°C)	Ki=0.01 (°C)	Ki=0.001 (°C)	Ki=0.0008 (°C)	Set point (°C)
0	31	31	31	31	50
1	41.5	36	38	38	50
2	53.5	49	49	50	50
3	66	56.5	55	54.5	50
4	73.5	58	56	56	50
5	74.5	58	56	55.5	50
6	74	57.5	55.5	55	50
7	73	57	55	54.5	50
8	72	56	54.5	54	50
9	71	55.5	54	53.3	50
10	69.5	55	53	53	50
11	69	54.5	52.5	52.5	50
12	68	53.5	52	52	50
13	67	53	51.5	51	50
14	66	52	51	50.5	50
15	65	51.5	50.5	50	50
16	64.5	51	50	49.5	50
17	63.5	50.5	49.5	50	50
18	62.5	50	49.5	50	50
19	61	49.5	50	50	50
20	60	49.5	50	50.5	50
21	59.5	50	50	50	50
22	59	50	49.5	50	50
23	58.5	50	49.5	50	50
24	58	49.5	50	50	50
25	57.5	49.5	50	50	50
26	56.5	49.5	50	50	50

เวลา (นาที)	Ki=0.1 (°C)	Ki=0.01 (°C)	Ki=0.001 (°C)	Ki=0.0008 (°C)	Set point (°C)
27	56	50	49.5	50	50
28	55.5	50	49.5	50.5	50
29	55	50	50	50	50
30	54.5	49.5	50	50	50
31	54	49.5	50	50	50
32	53.5	50	49.5	50	50
33	53	50	49.5	50	50
34	52.5	50	50	50	50
35	52	49.5	50	50	50
36	51.5	49.5	50	50	50
37	51	50	49.5	50	50
38	50.5	50	49.5	50	50
39	50	49.5	50	50	50
40	49.5	49.5	50	50	50
41	49	49.5	50	50	50
42	49	50	49.5	49.5	50
43	48.5	50	49.5	49.5	50
44	48	49.5	50	50	50
45	47.5	49.5	50	50	50
46	47	49.5	50	50	50
47	46.5	50	49.5	50	50
48	46	50	49.5	50	50
49	45.5	49.5	50	50	50
50	45	49.5	50	50	50
51	44.5	49.5	50	50	50
52	44	50	49.5	50	50
53	43.5	50	49.5	50	50
54	43	49.5	50	50	50
55	43	49.5	50	50	50



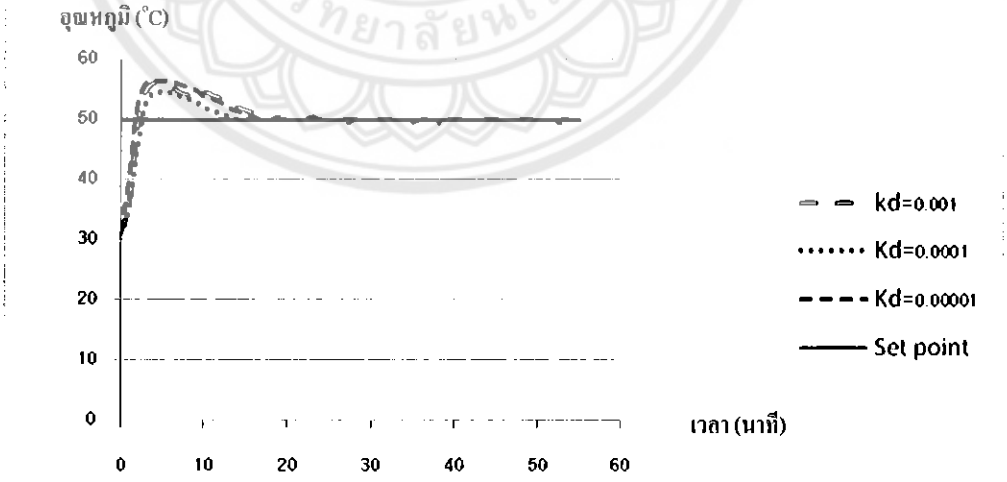
รูปที่ 4.3 กราฟแสดงผลการเปรียบเทียบผลอัตราขยายของ Ki

ตารางที่ 4.4 ผลการทดลองการเปรียบเทียบค่า Kd=0.001, Kd=0.0001, Kd=0.00001 เมื่อกำหนดค่า Kp=8, Ki=0.0008

เวลา (นาที)	kd=0.001 (°C)	Kd=0.0001 (°C)	Kd=0.00001 (°C)	Set point (°C)
0	31	31	31	50
1	35.5	34.5	39.5	50
2	48.5	44.5	52	50
3	54.5	52.5	56	50
4	56	54.5	56.5	50
5	56	54.5	56.5	50
6	55.5	54.5	56.5	50
7	54.5	54	56	50
8	54	53.5	55.5	50
9	53.5	53	55	50
10	54.5	52	54	50
11	54	51.5	53.5	50
12	53.5	51	53	50

เวลา (นาที)	kd=0.001 (°C)	Kd=0.0001 (°C)	Kd=0.00001 (°C)	Set point (°C)
13	53	50.5	52	50
14	52	50	51.5	50
15	51.5	50	51	50
16	51	50	50.5	50
17	50.5	50	50	50
18	50	50	50	50
19	50	50	50.5	50
20	50	50	50	50
21	50	50	50	50
22	50	50	50	50
23	50	50	50.5	50
24	50	50	50	50
25	50	50	50	50
26	50	50	50	50
27	50	50	49.5	50
28	50	50	50	50
29	50	50	50	50
30	50	50	50	50
31	50	50	50	50
32	50	50	50	50
33	50	50	50	50
34	49.5	50	50	50
35	49.5	50	50	50
36	50	50	50	50
37	50	50	50	50
38	50	49.5	50	50
39	50	50	50	50
40	50	50	50	50
41	50	50	50	50

เวลา (นาที)	kd=0.001 (°C)	Kd=0.0001 (°C)	Kd=0.00001 (°C)	Set point (°C)
42	50	50	50	50
43	50	50	50	50
44	50	50	50	50
45	50	50	50	50
46	50	50	50	50
47	50	50	50	50
48	50	50	50	50
49	50	50	50	50
50	50	50	50	50
51	50	50	50	50
52	50	50	50	50
53	50	50	49.5	50
54	50	50	50	50
55	50	50	50	50

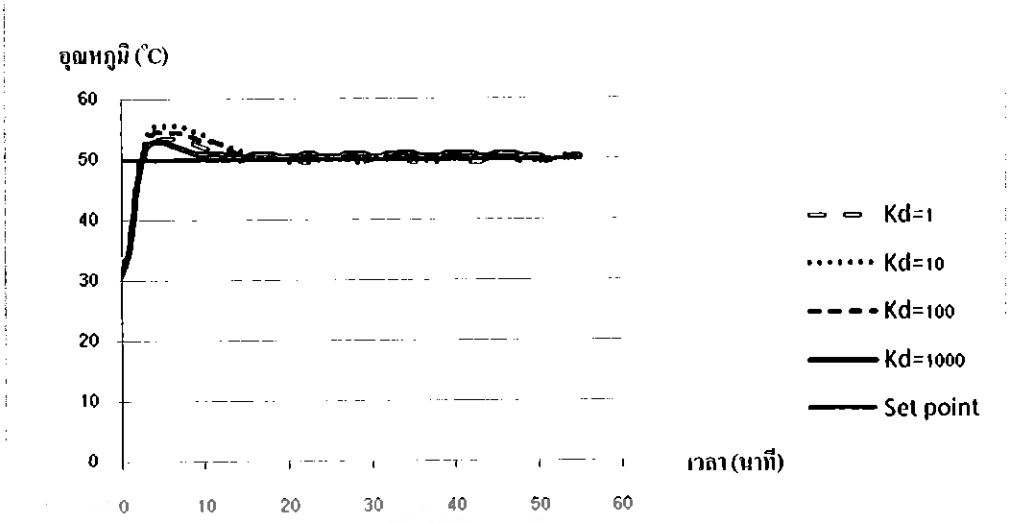


รูปที่ 4.4 กราฟแสดงผลการเปรียบเทียบผลอัตราขยายของ Kd

ตารางที่ 4.5 ผลการทดลองเปรียบเทียบ Kd=1, Kd=10, Kd=100, Kd=1000 เมื่อกำหนดค่า Kp=8, Ki=0.0008

เวลา (นาที)	Kd=1 (°C)	Kd=10 (°C)	Kd=100 (°C)	Kd=1000 (°C)	Set point (°C)
0	31	31	31	31	50
1	37	34.5	35	35.5	50
2	48.5	46.5	46	46.5	50
3	51	53.5	52.5	52	50
4	53	55.5	54.5	53	50
5	53.5	55.5	54.5	53	50
6	53.5	55.5	54.5	52.5	50
7	53.5	55.5	54.5	52	50
8	53	55	54	51.5	50
9	52.5	54.5	53.5	51	50
10	52	54	53	51	50
11	51.5	53	53	51	50
12	51	52.5	52.5	51	50
13	50.5	52	51.5	50.5	50
14	50	51.5	51	50.5	50
15	49.5	51	50.5	50.5	50
16	49.5	50.5	50	51	50
17	50	50	50	51	50
18	50	50	50	51	50
19	50	50	50	50.5	50
20	50	49.5	50	50.5	50
21	50	50	50.5	50.5	50
22	49.5	49.5	50.5	51	50
23	50	49.5	50	51	50
24	50	50	50	50.5	50
25	50	50.5	50	50.5	50
26	50	50	50	50.5	50

เวลา (นาที)	Kd=1 (°C)	Kd=10 (°C)	Kd=100 (°C)	Kd=1000 (°C)	Set point (°C)
27	50	50	50.5	51	50
28	49.5	49.5	50.5	51	50
29	49.5	49.5	50.5	51	50
30	50	50	50	50.5	50
31	50	50	50	50.5	50
32	50	50.5	50	51	50
33	50	50.5	50	51	50
34	50	50	50	51	50
35	49.5	50	50.5	51	50
36	49.5	50	50	50.5	50
37	50	49.5	50.5	50.5	50
38	50	49.5	50.5	50.5	50
39	50	50	50	51	50
40	50	50	50	51	50
41	49.5	50.5	50	51	50
42	49.5	50	50	51	50
43	49.5	50	50.5	50.5	50
44	50	50.5	50.5	50.5	50
45	50	50.5	50	51	50
46	50	50	50	51	50
47	50	50	50	51	50
48	49.5	50	50	50.5	50
49	49.5	50	50	50.5	50
50	49.5	50	49.5	50.5	50
51	50	50	49.5	50	50
52	50	50	50	50	50
53	50	50.5	50	50	50
54	50	50.5	50	50.5	50
55	50	50	50	50.5	50



รูปที่ 4.5 กราฟแสดงผลการเปรียบเทียบผลอัตราขยายของ Kd

4.3 สรุปผลการทดลอง

จากการทดลองที่ 4.1 ผลอัตราขยายของ Kp ที่กำหนดคือ Kp=1 และ Kp=2 ค่าอุณหภูมิจะค่อยๆขึ้นไปสู่จุดเซ็ตพ้อยท์ (Setpoint) และไม่เกิดค่าโอเวอร์ชูต (Overshoot) เนื่องจากอัตราขยาย Kp ที่มีค่าน้อยและค่าอุณหภูมิที่ได้ก็จะมีค่าในสถานะอยู่ตัวอยู่ที่ 49.5 องศาเซลเซียส และอัตราขยาย Kp=1 ที่มีค่าน้อยกว่า ก็จะเข้าสู่จุดเซ็ตพ้อยท์ ได้ช้ากว่า แต่มีค่าอุณหภูมิในสถานะอยู่ตัวเท่ากัน

จากการทดลองที่ 4.2 ผลอัตราขยายของ Kp ที่กำหนดคือ Kp=8, Kp=10, Kp=20 และ Kp=100 จากกราฟจะเห็นว่าเมื่อเรากำหนดค่าอัตราขยายมากๆคือ Kp=100 ค่าอุณหภูมิที่ได้ก็จะมีโอเวอร์ชูตใกล้เคียงกับ Kp ค่าอื่นๆที่กำหนดคือ Kp=8, Kp=10, Kp=20 และค่าอุณหภูมิในสถานะอยู่ตัวก็มีค่าใกล้เคียงกันจะแกว่งขึ้นลงอยู่ที่ 49.5-50 องศาเซลเซียส

จากการทดลองที่ 4.3 การเปรียบเทียบค่า Ki=0.1, Ki=0.01, Ki=0.001, Ki=0.0008 เมื่อกำหนดให้ Kp=8, Kd=0 จะเห็นได้ว่าที่ค่า Ki=0.1 ค่าอุณหภูมิที่ได้มีโอเวอร์ชูตสูงจนไม่สามารถเข้าสู่จุดเซ็ตพ้อยท์ได้ แต่ค่า Ki=0.01, Ki=0.001, Ki=0.0008 มีค่าอุณหภูมิที่ได้ใกล้เคียงกัน เราจึงเลือกจาก Ki ที่ทำให้โอเวอร์ชูต น้อยที่สุด คือ Ki=0.0008

จากการทดลองที่ 4.4 การเปรียบเทียบค่า Kd=0.001, Kd=0.0001, Kd=0.00001 เมื่อกำหนดค่า Kp=8, Ki=0.0008 ผลที่ได้คือค่า Kd=0.001, Kd=0.00001 มีค่าโอเวอร์ชูตใกล้เคียงกัน แต่ค่า Kd=0.0001 มีค่าโอเวอร์ชูตน้อยที่สุดทำให้เข้าสู่จุดเซ็ตพ้อยท์ได้เร็วกว่า

จากการทดลองที่ 4.5 การเปรียบเทียบค่า Kd=1, Kd=10, Kd=100, Kd=1000 เมื่อกำหนดค่า Kp=8, Ki=0.0008 ผลที่ได้จากการทดลองจะเห็นได้ว่าที่ Kd=1000 อัตราการแกว่งของอุณหภูมิในช่วงสถานะอยู่ตัวอยู่ที่ 49.5-51 องศาเซลเซียส ซึ่งมากกว่าการกำหนดค่า Kd ค่าอื่นๆ

จากการทดลองทั้งหมดทำให้สามารถกำหนดค่า Kp, Ki, Kd ได้คือ ค่า Kp=8, Ki=0.0008, Kd=0.0001 เลือกจากค่าที่เหมาะสมที่สุด โดยดูจากการเกิดโอเวอร์ชูตที่ไม่มากเกินไป คือโอเวอร์ชูตไม่เกิน 5 องศาเซลเซียส และดูจากค่าการแกว่งของอุณหภูมิในสถานะอยู่ตัวที่น้อยที่สุด อยู่ในช่วง 49.5-50 องศาเซลเซียส จากการทดลองทั้ง 5 ผลการทดลอง

บทที่ 5
สรุปผลของโครงงาน

บทที่ 5 เป็นเรื่องเกี่ยวกับการสรุปผลของโครงงาน รวมถึงปัญหาและข้อเสนอแนะเพิ่มเติมเพื่อเป็นแนวทางในการพัฒนาโครงงานต่อไป

5.1 สรุปผลของโครงงาน

จากการทดลองการควบคุมอุณหภูมิด้วยพีไอดีคอนโทรลเลอร์สามารถทำงานได้จริงอย่างมีประสิทธิภาพ โดยมีไมโครคอนโทรลเลอร์ไปควบคุมอุปกรณ์ต่างๆได้ดี แต่อาจจะพบปัญหาการทำงานบ้าง เนื่องจากการเวลาจะเริ่มทำการทดลองใหม่อีกครั้งจะต้องทำการปรับค่า Kp, Ki และ Kd ใหม่เพื่อให้เกิดประสิทธิ์ภาพมากขึ้นกว่าการทดลองครั้งก่อนหน้านี้จึงทำให้ต้องรอให้น้ำเย็นก่อนถึงจะทำการทดลองใหม่ได้ วิธีแก้ไขคือการเปลี่ยนน้ำใหม่ในกาโดยไม่ต้องรอให้น้ำเย็นตัวลง เพราะทำให้เสียเวลาในการทดลอง

ปัญหาที่พบจากการทดลองคือ การทำงานของขดลวดความร้อนมีการสะสมความร้อนจนทำให้เกิดโอเวอร์ชูต และหลังจากเกิดโอเวอร์ชูต จะต้องรอให้น้ำคายความร้อนเอง ซึ่งอุณหภูมิจะเริ่มลดลงหลังจากผ่านไปประมาณ 3 นาที ที่โอเวอร์ชูตสูงสุด และจะลดลงอย่างต่อเนื่องในอัตราส่วนโดยประมาณ 2 นาทีต่อ 1 องศาเซลเซียส เป็นผลให้อุณหภูมิของน้ำเข้าสู่จุดเซ็ตพ้อยต์ได้ช้าหลังจากเกิดโอเวอร์ชูต

ในโครงงานนี้การควบคุมอุณหภูมิด้วยพีไอดีคอนโทรลเลอร์สามารถทำงานได้ตามเป้าหมายที่วางไว้ในโครงงาน แต่ยังต้องการพัฒนาคุณภาพอีกหลายอย่างเพื่อการควบคุมอุณหภูมิด้วย ระบบควบคุมแบบพีไอดีได้ดียิ่งขึ้น

5.2 ปัญหาที่เกิดขึ้นในระหว่างทำโครงงาน

1. ปัญหาด้านการเชื่อมต่อสายเป็นลายวงจร อาจจะลัดวงจรได้ เนื่องจากไม่ได้ทำการกดลายวงจร และเมื่อเจอการกระแทกมากอาจทำให้ลัดวงจรได้

2. การตั้งค่า Kp, Ki และ Kd จะต้องต้องตั้งค่าให้เหมาะสมกับระบบควบคุมการทำงาน ทำให้โอเวอร์ชูตมีค่าสูงเกินไป

5.3 แนวทางในการแก้ไขปัญหา

1. วิธีการออกแบบลายวงจรแล้วทำการกัดลายวงจรเป็นวิธีที่ดีที่สุด แต่ถ้าต้องการต่อเพื่อทดสอบการทำงานของระบบว่าทำงานได้จริง ควรใช้สายขนาดเล็กเชื่อมเข้ากับโหนดต่างๆ และการต่อโหนดต่างๆไม่ควรจะมีการพ่วงสายมากเกินไป เนื่องจากจะทำให้โหนดนั้นเกิดความร้อนสะสม จนทำให้บอร์ดมีปัญหาได้ในภายหลัง

2. หาวิธีการตั้งค่า Kp, Ki และ Kd ให้เหมาะสมกับระบบควบคุมของกาต้มน้ำเพื่อที่จะต้มน้ำให้ได้อุณหภูมิที่กำหนดค่าไว้และได้การทำงานที่มีประสิทธิภาพมากสุดในการแก้ปัญหาโอเวอร์ชูตของอัตราขยายที่มีค่าสูงเกินไป

5.4 แนวทางในการพัฒนา

พัฒนาแนวทางในการแก้ไขค่าความผิดพลาดในการควบคุมอุณหภูมิด้วยการควบคุมแบบพีไอดี ให้มีค่าโอเวอร์ชูตไม่มากจนเกินไปและแก้ไขค่าความผิดพลาดในสถานะอยู่ตัวคงที่ให้มีความเสถียรจนอยู่ในระดับอุณหภูมิที่ต้องการอย่างเหมาะสม และลดค่าความหน่วงเวลาของระบบเมื่อทำการการควบคุมอุณหภูมิด้วยการควบคุมแบบพีไอดีและควบคุมอุณหภูมิไม่ผิดพลาด ต้องเพิ่มเติมการทดลองและศึกษาการตั้งค่าของ Kp, Ki และ Kd ให้มีความเหมาะสม ยกตัวอย่างเช่นวิธีการ Ziegler–Nichols [3] วิธีการนี้นำเสนอโดย John G. Ziegler และ Nathaniel B. Nichols ในคริสต์ทศวรรษที่ 1940 เป็นวิธีการปรับค่าตัวแปรแก้ไขระบบที่ดีอีกวิธีหนึ่งนอกจากวิธีที่ได้นำมาใช้คือการปรับจูนค่าเกนของตัวควบคุมพีไอดีสำหรับระบบที่ไม่ทราบแบบจำลองทางคณิตศาสตร์ หรือ ไม่ทราบสมการระบบ ซึ่งจะทำให้สร้างเครื่องควบคุมอุณหภูมิได้มีประสิทธิภาพ สามารถใช้งานได้อย่างกว้างขวางและสามารถพัฒนาต่อเนื่องควบคู่กับอุปกรณ์อย่างอื่นได้อย่างหลากหลายตามความเหมาะสม

เอกสารอ้างอิง

[1] ประจิน พลังสันติกุล. (2549) . C Programming for AVR Microcontroller and WinAVR (C Compiler).กรุงเทพฯ : จัดพิมพ์โดยบริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด

[2] ประจิน พลังสันติกุล. (2551) . Apply C Programming for AVR Microcontroller Book 2. กรุงเทพฯ : จัดพิมพ์โดยบริษัท อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด.

[3] อนุชา หิรัญวัฒน์, นฤพนธ์ พนากุลชัยวิทย์, และสมชัย ตรีรัตนจารุ. (2551) . การควบคุมอัตโนมัติ และการประยุกต์ใช้งานพีแอลซี (ชั้นกลาง). นนทบุรี : จัดพิมพ์โดย ห้างหุ้นส่วนจำกัด ธนินชัย.

[4] สราวุฒิ สุจิตจร. (2546) . การควบคุมอัตโนมัติ. กรุงเทพฯ : จัดพิมพ์โดยบริษัท เพียร์สัน เอ็ดดูเคชั่น อินโดไชน่า จำกัด.

[5] นิรุธ อำนวยศิลป์. (2550) . คู่มือการเขียนโปรแกรมภาษา C. กรุงเทพฯ : จัดพิมพ์โดยบริษัท โปรวิชั่น จำกัด.

[6] ธานี โกสุม, เภรี โตเหี้ยม, พิชัย ศรีนพเกล้า. (2553) . การควบคุมลิฟต์ด้วยไมโครคอนโทรลเลอร์. ระดับปริญญาตรี, สาขาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์. พิษณุโลก : มหาวิทยาลัยนเรศวร.

[7] นิรนาม. (2552) . "BASICLITE." เซนเซอร์.[ออนไลน์]. เข้าถึงได้จาก : http://www.basiclite.com/web/index.php?topic=125.0. (วันที่ค้นข้อมูล : 5 กุมภาพันธ์ 2555)

ภาคผนวก ก

รายละเอียดของ ATmaga 32

Features

- High-performance, Low-power Atmel®AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single-clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory segments
  - 32Kbytes of In-System Self-programmable Flash program memory
  - 1024Bytes EEPROM
  - 2Kbytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C[1]
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
    8 Single-ended Channels
    7 Differential Channels in TQFP Package Only
    2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
  - Byte-oriented Two-wire Serial Interface
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
- Operating Voltages
  - 2.7V - 5.5V for ATmega32L
  - 4.5V - 5.5V for ATmega32
- Speed Grades
  - 0 - 8MHz for ATmega32L
  - 0 - 16MHz for ATmega32
- Power Consumption at 1MHz, 3V, 25°C
  - Active: 1.1mA
  - Idle Mode: 0.35mA
  - Power-down Mode: < 1µA

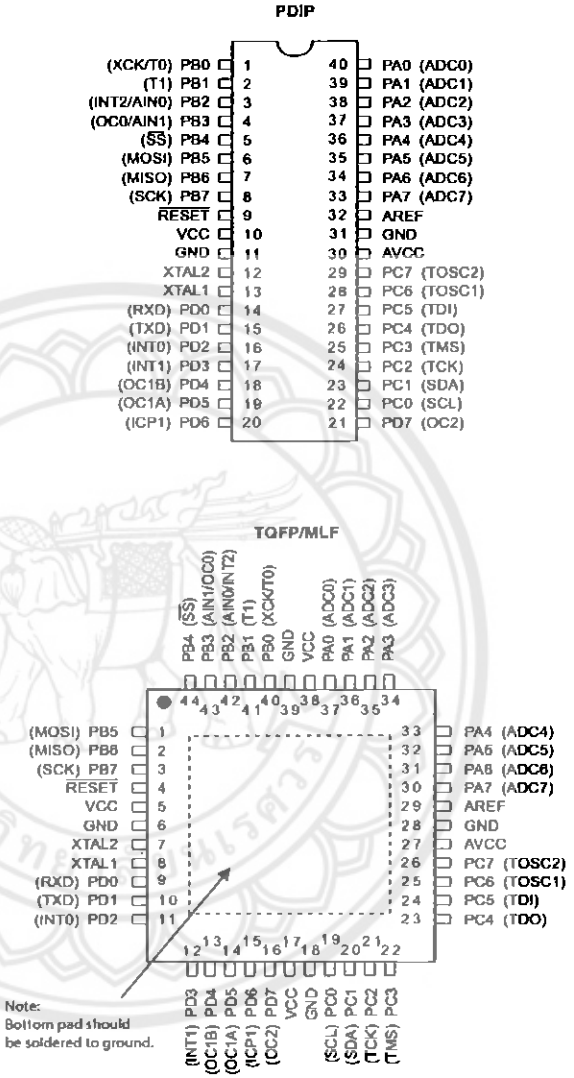
8-bit AVR® Microcontroller with 32KBytes In-System Programmable Flash

ATmega32
ATmega32L

2503Q–AVR–02/11

Pin Configurations

Figure 1. Pinout ATmega32

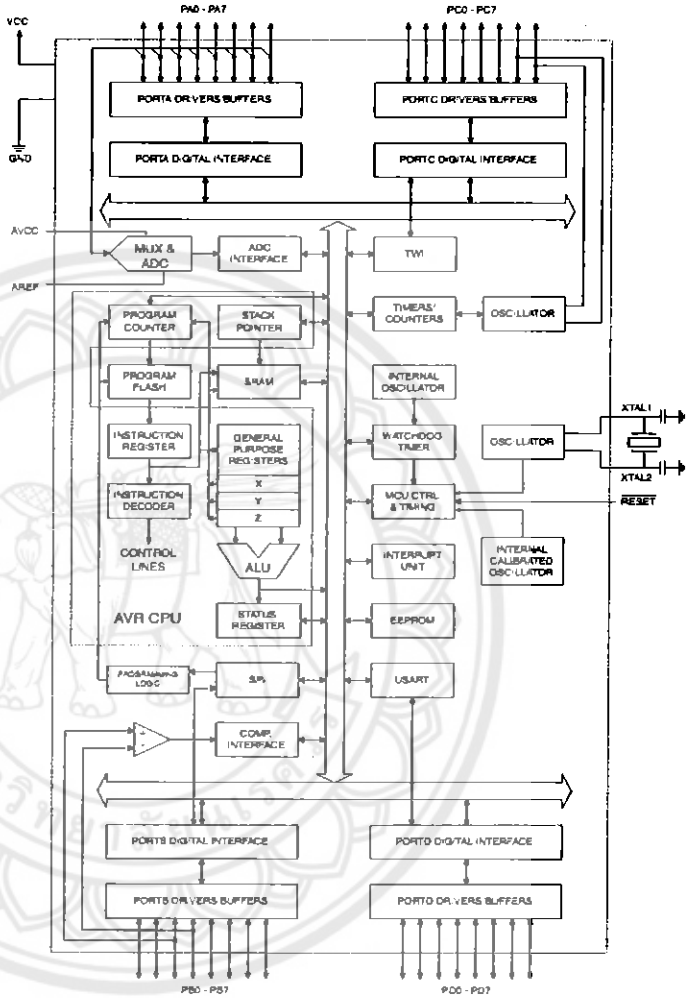


Overview

The Atmel®AVR®ATmega32 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega32 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 2. Block Diagram



The Atmel®AVR®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 ATmega32 provides the following features: 32Kbytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 1024bytes EEPROM, 2Kbyte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain (TQFP package only), a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, 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 External Interrupt or Hardware Reset. In Power-save mode, the Asynchronous Timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

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, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega32 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The Atmel AVR ATmega32 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.

Pin Descriptions

VCC Digital supply voltage.

GND Ground.

Port A (PA7..PA0) Port A serves as the analog inputs to the A/D Converter.

Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. 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. 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 (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega32 as listed on page 57.

Port C (PC7..PC0) Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs.

The TD0 pin is tri-stated unless TAP states that shift out data are entered.

Port C also serves the functions of the JTAG interface and other special features of the ATmega32 as listed on page 60.

Port D (PD7..PD0) Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega32 as listed on page 62.

$\overline{\text{RESET}}$ Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 37. 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 AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

AREF AREF is the analog reference pin for the A/D Converter.

ภาคผนวก ข

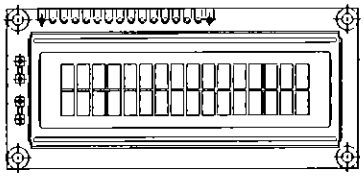
รายละเอียดของ LCD 16×2

VISHAY

LCD-016M002B

Vishay

16 x 2 Character LCD



FEATURES

- 5 x 8 dots with cursor
- Built-in controller (KS 0066 or Equivalent)
- + 5V power supply (Also available for + 3V)
- 1/16 duty cycle
- B/L to be driven by pin 1, pin 2 or pin 15, pin 16 or A.K (LED)
- N.V. optional for + 3V power supply

MECHANICAL DATA

ITEM	STANDARD VALUE	UNIT
Module Dimension	80.0 x 36.0	mm
Viewing Area	66.0 x 16.0	mm
Dot Size	0.56 x 0.66	mm
Character Size	2.96 x 5.56	mm

ABSOLUTE MAXIMUM RATING

ITEM	SYMBOL	STANDARD VALUE			UNIT
		MIN.	TYP.	MAX.	
Power Supply	VDD-VSS	- 0.3	–	7.0	V
Input Voltage	VI	- 0.3	–	VDD	V

NOTE: VSS = 0 Volt, VDD = 5.0 Volt

ELECTRICAL SPECIFICATIONS

ITEM	SYMBOL	CONDITION		STANDARD VALUE			UNIT
				MIN.	TYP.	MAX.	
Input Voltage	VDD	VDD = + 5V		4.7	5.0	5.3	V
		VDD = + 3V		2.7	3.0	5.3	V
Supply Current	IDD	VDD = 5V		–	1.2	3.0	mA
Recommended LC Driving Voltage for Normal Temp. Version Module	VDD - V0	- 20 °C		–	–	–	V
		0°C		4.2	4.8	5.1	
		25°C		3.8	4.2	4.6	
		50°C		3.6	4.0	4.4	
		70°C		–	–	–	
LED Forward Voltage	VF	25°C		–	4.2	4.6	V
LED Forward Current	IF	25°C	Array	–	130	260	mA
			Edge	–	20	40	
EL Power Supply Current	IEL	Vel = 110VAC:400Hz		–	–	5.0	mA

DISPLAY CHARACTER ADDRESS CODE:

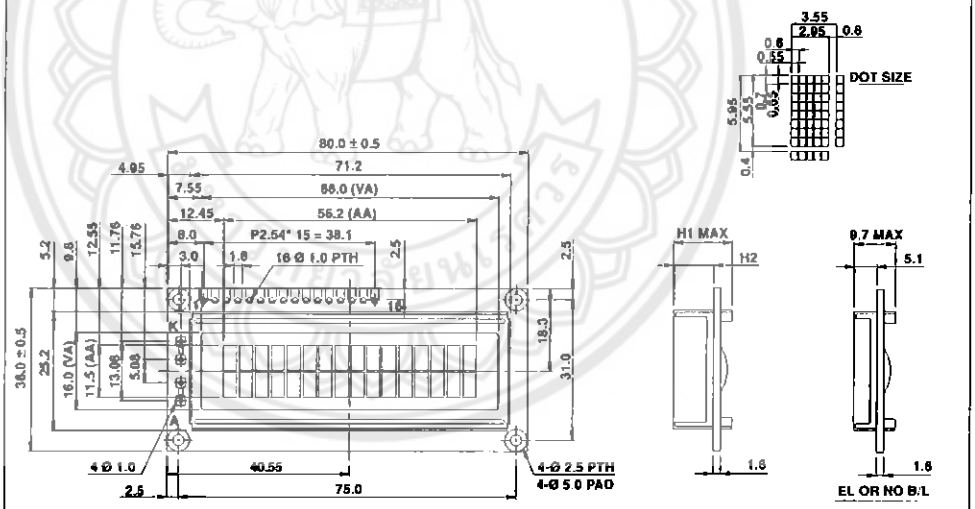
Display Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DD RAM Address	00	01														0F
DD RAM Address	40	41														4F

LCD-016M002B

Vishay 16 x 2 Character LCD

PIN NUMBER	SYMBOL	FUNCTION
1	Vss	GND
2	Vdd	+ 3V or + 5V
3	Vo	Contrast Adjustment
4	RS	H/L Register Select Signal
5	R/$\overline{W}$	H/L Read/Write Signal
6	E	H →L Enable Signal
7	DB0	H/L Data Bus Line
8	DB1	H/L Data Bus Line
9	DB2	H/L Data Bus Line
10	DB3	H/L Data Bus Line
11	DB4	H/L Data Bus Line
12	DB5	H/L Data Bus Line
13	DB6	H/L Data Bus Line
14	DB7	H/L Data Bus Line
15	A/Vee	+ 4.2V for LED/Negative Voltage Output
16	K	Power Supply for B/L (0V)

DIMENSIONS in millimeters



LED - H/L B/L		
	HIGH	LOW
H1	13.2	12.1
H2	8.6	7.5

ภาคผนวก ค รายละเอียดของ LM2575, SOLID STATE RELAY

National Semiconductor

April 2007

LM1575/LM2575/LM2575HV
SIMPLE SWITCHER® 1A Step-Down Voltage Regulator

General Description

The LM2575 series of regulators are monolithic integrated circuits that provide all the active functions for a step-down (buck) switching regulator, capable of driving a 1A load with excellent line and load regulation. These devices are available in fixed output voltages of 3.3V, 5V, 12V, 15V, and an adjustable output version.

Requiring a minimum number of external components, these regulators are simple to use and include internal frequency compensation and a fixed-frequency oscillator.

The LM2575 series offers a high-efficiency replacement for popular three-terminal linear regulators. It substantially reduces the size of the heat sink, and in many cases no heat sink is required.

A standard series of inductors optimized for use with the LM2575 are available from several different manufacturers. This feature greatly simplifies the design of switch-mode power supplies.

Other features include a guaranteed ±4% tolerance on output voltage within specified input voltages and output load conditions, and ±10% on the oscillator frequency. External shutdown is included, featuring 50 µA (typical) standby current. The output switch includes cycle-by-cycle current limiting, as well as thermal shutdown for full protection under fault conditions.

Features

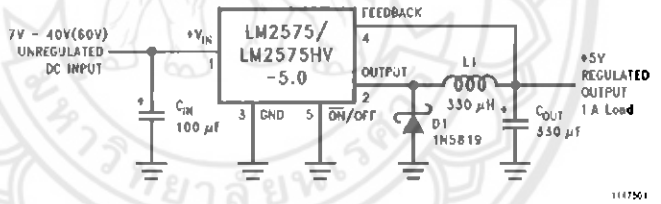
- 3.3V, 5V, 12V, 15V, and adjustable output versions
- Adjustable version output voltage range, 1.23V to 37V (57V for HV version) ±4% max over line and load conditions
- Guaranteed 1A output current
- Wide input voltage range, 40V up to 60V for HV version
- Requires only 4 external components
- 52 kHz fixed frequency internal oscillator
- TTL shutdown capability, low power standby mode
- High efficiency
- Uses readily available standard inductors
- Thermal shutdown and current limit protection
- P+ Product Enhancement tested

Applications

- Simple high-efficiency step-down (buck) regulator
- Efficient pre-regulator for linear regulators
- On-card switching regulators
- Positive to negative converter (Buck-Boost)

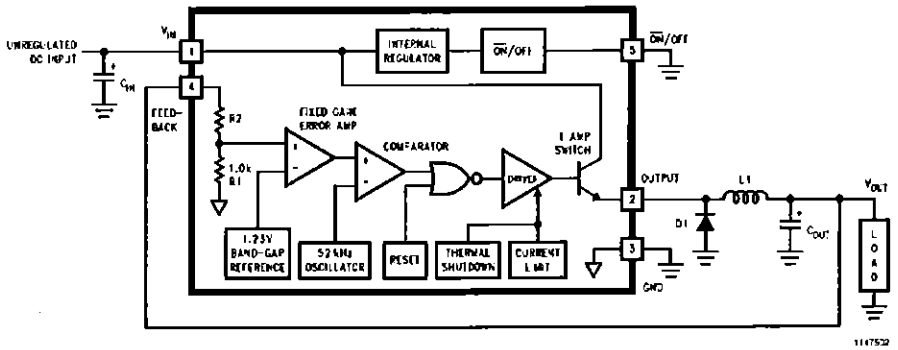
Typical Application

(Fixed Output Voltage Versions)



Note: Pin numbers are for the TO-220 package.

Block Diagram and Typical Application



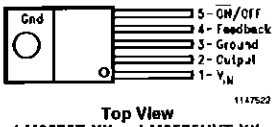
3.3V, R2 = 1.7k
5V, R2 = 3.1k
12V, R2 = 8.84k
15V, R2 = 11.3k
For ADJ. Version
R1 = Open, R2 = 0Ω
Note: Pin numbers are for the TO-220 package.

FIGURE 1.

Connection Diagrams

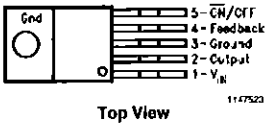
(XX indicates output voltage option. See Ordering Information table for complete part number.)

Straight Leads
5-Lead TO-220 (T)

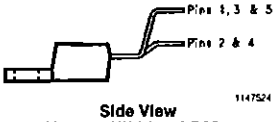


Top View
LM2575T-XX or LM2575HVT-XX
See NS Package Number T05A

Bent, Staggered Leads
5-Lead TO-220 (T)

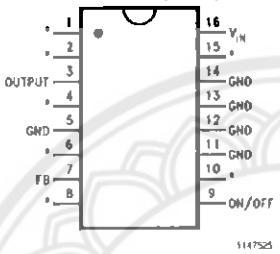


Top View



Side View
LM2575T-XX Flow LB03 or
LM2575HVT-XX Flow LB03
See NS Package Number T05D

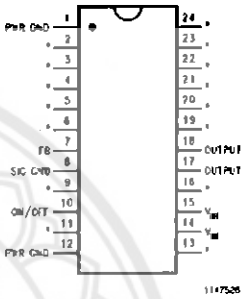
16-Lead DIP (N or J)



*No Internal Connection

Top View
LM2575N-XX or LM2575HVN-XX
See NS Package Number N16A
LM1675J-XX-QML
See NS Package Number J16A

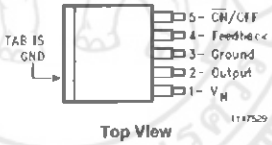
24-Lead Surface Mount (M)



*No Internal Connection

Top View
LM2575M-XX or LM2575HVM-XX
See NS Package Number M24B

TO-263(S)
5-Lead Surface-Mount Package



Top View

Side View
LM2575S-XX or LM2575HVS-XX
See NS Package Number TS5B

AC Power Series Specifications

Opto 22 provides a full range of Power Series relays with a wide variety of voltage (120–575) and current options (3–45 amps). All Power Series relays feature 4,000 volts of optical isolation and have a high PRV rating. Operating temperature is –40 °C to 100 °C

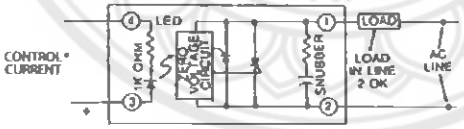
120/240/380 Volt

NOTE: Model numbers ending in -17 are replacement parts only. Their specifications are identical to the same model number without the -17. For example, 240D10-17 is identical to 240D10

Model Number	Nominal AC Line Voltage	Nominal Current Rating (Amps)	1 cycle Surge (Amps) Peak	Nominal Signal Input Resistance (Ohms)	Signal Pickup Voltage	Signal Drop-out Voltage	Peak Repetitive Voltage Maximum	Maximum Output Voltage Drop	Off-State Leakage (mA) Maximum**	Operating Voltage Range (Volts AC)	I^2t Rating (mA)^2 (ms)	Isolation Voltage	θ_{jc}* (°C/Watt)	Dissipation (Watts/ Amp)
120D3	120	3	66	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	2.5 mA	12–140	30	4 000 V_{rms}	11	1.7
120D10	120	10	110	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	7 mA	12–140	50	4 000 V_{rms}	1.3	1.6
120D25	120	25	250	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	7 mA	12–140	260	4 000 V_{rms}	1.2	1.3
120D45	120	45	660	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	7 mA	12–140	1750	4 000 V_{rms}	0.67	0.9
240D3	240	3	66	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	6 mA	24–280	30	4 000 V_{rms}	11	1.7
240D10	240	10	110	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	14 mA	24–280	50	4 000 V_{rms}	1.3	1.6
240D10	240	10	110	730	3VDC (32V allowed)	1 VDC	800	1.6 volts	14 mA	24–280	50	4 000 V_{rms}	1.3	1.6
240D25	240	25	260	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	14 mA	24–280	260	4 000 V_{rms}	1.2	1.3
240D25	240	25	260	730	3VDC (32V allowed)	1 VDC	800	1.6 volts	14 mA	12–280	260	4 000 V_{rms}	1.2	1.3
240D45	240	45	660	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	14 mA	24–280	1750	4 000 V_{rms}	0.67	0.9
240D45	240	45	660	730	3VDC (32V allowed)	1 VDC	800	1.6 volts	14 mA	24–280	1750	4 000 V_{rms}	0.67	0.9
380D25	380	25	260	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	12 mA	24–420	260	4 000 V_{rms}	1.2	1.3
380D45	380	45	660	1000	3VDC (32V allowed)	1 VDC	800	1.6 volts	12 mA	24–425	1750	4 000 V_{rms}	0.67	0.9
120A10	120	10	110	33K	85VAC (28CV allowed)	10 VAC	800	1.6 volts	7 mA	12–140	50	4 000 V_{rms}	1.3	1.6
120A25	120	25	250	33K	85VAC (28CV allowed)	10 VAC	800	1.6 volts	7 mA	12–140	260	4 000 V_{rms}	1.2	1.3
240A10	240	10	110	33K	85VAC (28CV allowed)	10 VAC	800	1.6 volts	14 mA	24–280	50	4 000 V_{rms}	1.3	1.6
240A25	240	25	250	33K	85VAC (28CV allowed)	10 VAC	800	1.6 volts	14 mA	24–280	260	4 000 V_{rms}	1.2	1.3
240A45	240	45	660	33K	85VAC (28CV allowed)	10 VAC	800	1.6 volts	14 mA	24–280	1750	4 000 V_{rms}	0.67	0.9

Note: θ_{jc} = Thermal resistance from internal junction to base. Maximum internal junction temperature is 110 °C
** Operating Frequency: 25 to 66 Hz (operates at 400 Hz with 8 times the off-state leakage)

Connection Diagram, DC Power Series



*Control Current varies with control voltage. See "Control Current Calculation" on page 17 for information.

Opto 22 • 43044 Business Park Drive • Temecula, CA 92590-3614 • www.opto22.com
SALES 800-321-6786 • 951-695-3000 • FAX 951-695-3095 • sales@opto22.com • SUPPORT 800-835-6786 • 951-695-3080 • FAX 951-695-3017 • support@opto22.com

480/575 Volt

Model Number	Nominal AC Line Voltage	Nominal Current Rating (Amps)	1 cycle Surge (Amps) Peak	Nominal Signal Input Resistance (Ohms)	Signal Pick-up Voltage	Signal Drop-out Voltage	Peak Repetitive Voltage Maximum	Maximum Output Voltage Drop	Off-State Leakage (mA) Maximum**	Operating Voltage Range (Volts AC)	I^2t Rating t=8.3 (ms)	Isolation Voltage	θ_{jc}* (°C/Watt)	Dissipation (Watts/Amp)
480D10-12	480	10	110	1000	3 VDC (32V allowed)	1 VDC	1200	2.2 volts	11 mA	100–530	50	4,000 V_{rms}	1.2	2.5
480D15-12	480	15	150	1000	3 VDC (32V allowed)	1 VDC	1200	2.2 volts	11 mA	100–530	50	4,000 V_{rms}	1.2	2.5
480D25-12	480	25	250	1000	3 VDC (32V allowed)	1 VDC	1200	1.6 volts	11 mA	100–530	250	4,000 V_{rms}	1.5	1.9
480D45-12	480	45	650	1000	3 VDC (32V allowed)	1 VDC	1200	1.6 volts	11 mA	100–530	1750	4,000 V_{rms}	0.67	0.9
575D15-12	575	15	150	1000	3 VDC (32V allowed)	1 VDC	1200	2.2 volts	15 mA	100–600	50	4,000 V_{rms}	1.2	2.5
575D45-12	575	45	650	1000	3 VDC (32V allowed)	1 VDC	1200	1.6 volts	15 mA	100–600	1750	4,000 V_{rms}	0.67	0.9
575D45-12	575	45	650	730	3 VDC (32V allowed)	1 VDC	1200	1.6 volts	15 mA	100–600	1750	4,000 V_{rms}	0.67	0.9

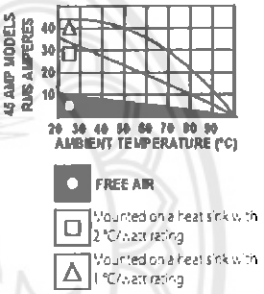
Note: θ_{jc}* = Thermal resistance from internal junction to base. Maximum internal junction temperature = 110 °C
** Operating Frequency: 25 to 65 Hz (operates at 400 Hz with 5 times the off-state leakage)

Surge Current Data

Time Second	Time*** (Cycles)	10-Amp Peak Amps	15-Amp Peak Amps	25-Amp Peak Amps	45-Amp Peak Amps
0.017	1	110	150	250	650
0.050	3	85	140	175	420
0.100	6	70	110	140	320
0.200	12	60	90	112	245
0.500	30	50	70	80	175
1	60	40	55	67	134
2	120	33	49	53	119
3	180	32	47	49	98
4	240	31	43	47	95
5	300	30	40	45	91
10	600	28	35	42	84

Note: ***60 Hz

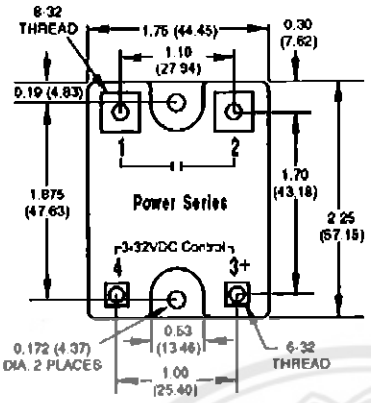
Thermal Ratings

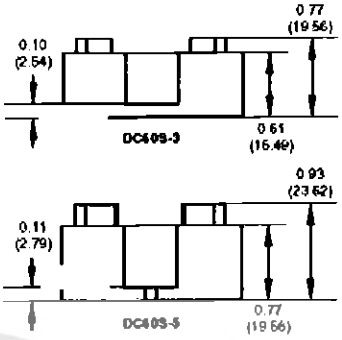


Solid-State Relays

480/575 Volt (cont)

Dimensional Drawings





Side view: Part numbers DC60S3, 120D3, and 240D3 only

Side view: All other part numbers

ภาคผนวก ง รายละเอียดของไอซี DS1820

DALLAS SEMICONDUCTOR

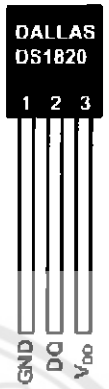
DS18S20
High Precision 1-Wire® Digital Thermometer

www.dalsemi.com

FEATURES

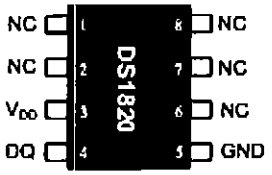
- Unique 1-wire interface requires only one port pin for communication
- Each device has a unique 64-bit serial code stored in an on-board ROM
- Multi-drop capability simplifies distributed temperature sensing applications
- Requires no external components
- Can be powered from data line. Power supply range is 3.0V to 5.5V
- Measures temperatures from –55°C to +125°C (–67°F to +257°F)
- ±0.5°C accuracy from –10°C to +85°C
- 9-bit thermometer resolution
- Converts temperature in 750 ms (max.)
- User-definable nonvolatile alarm settings
- Alarm search command identifies and addresses devices whose temperature is outside of programmed limits (temperature alarm condition)
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system

PIN ASSIGNMENT



(BOTTOM VIEW)

TO-92
(DS18S20)



8-pin 150-mil SOIC
(DS18S20Z)

PIN DESCRIPTION

GND - Ground
DQ - Data In/Out
V_{DD} - Power Supply Voltage
NC - No Connect

DESCRIPTION

The DS18S20 Digital Thermometer provides 9–bit centigrade temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18S20 communicates over a 1-wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. It has an operating temperature range of –55°C to +125°C and is accurate to ±0.5°C over the range of –10°C to +85°C. In addition, the DS18S20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18S20 has a unique 64-bit serial code, which allows multiple DS18S20s to function on the same 1–wire bus; thus, it is simple to use one microprocessor to control many DS18S20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment or machinery, and process monitoring and control systems.

DETAILED PIN DESCRIPTIONS Table 1

8-PIN SOIC*	TO-92	SYMBOL	DESCRIPTION
5	1	GND	**Ground.**
4	2	DQ	**Data Input/Output pin.** Open-drain 1-wire interface pin. Also provides power to the device when used in parasite power mode (see "Parasite Power" section.)
3	3	V_{DD}	**Optional V_{DD} pin.** V_{DD} must be grounded for operation in parasite power mode.

*All pins not specified in this table are "No Connect" pins.

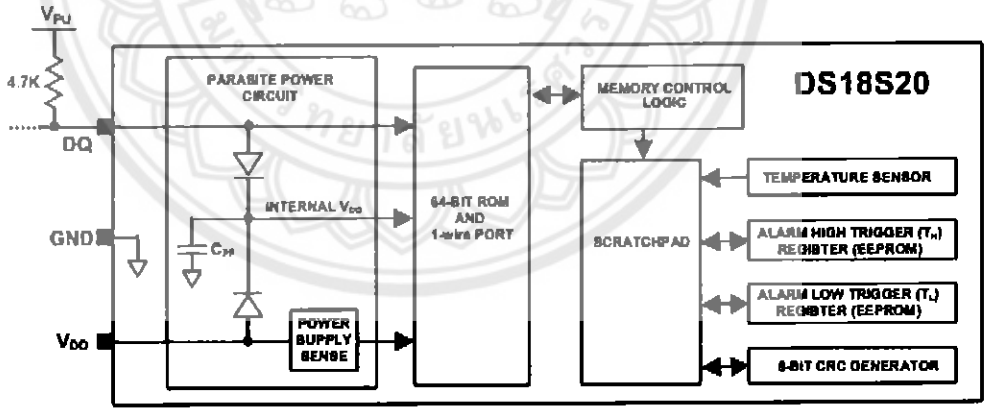
OVERVIEW

Figure 1 shows a block diagram of the DS18S20, and pin descriptions are given in Table 1. The 64-bit ROM stores the device's unique serial code. The scratchpad memory contains the 2-byte temperature register that stores the digital output from the temperature sensor. In addition, the scratchpad provides access to the 1-byte upper and lower alarm trigger registers (T_H and T_L). The T_H and T_L registers are nonvolatile (EEPROM), so they will retain data when the device is powered down.

The DS18S20 uses Dallas' exclusive 1-wire bus protocol that implements bus communication using one control signal. The control line requires a weak pullup resistor since all devices are linked to the bus via a 3-state or open-drain port (the DQ pin in the case of the DS18S20). In this bus system, the microprocessor (the master device) identifies and addresses devices on the bus using each device's unique 64-bit code. Because each device has a unique code, the number of devices that can be addressed on one bus is virtually unlimited. The 1-wire bus protocol, including detailed explanations of the commands and "time slots," is covered in the 1-WIRE BUS SYSTEM section of this datasheet.

Another feature of the DS18S20 is the ability to operate without an external power supply. Power is instead supplied through the 1-wire pullup resistor via the DQ pin when the bus is high. The high bus signal also charges an internal capacitor (C_{PP}), which then supplies power to the device when the bus is low. This method of deriving power from the 1-wire bus is referred to as "parasite power." As an alternative, the DS18S20 may also be powered by an external supply on V_{DD}.

DS18S20 BLOCK DIAGRAM Figure 1



OPERATION – MEASURING TEMPERATURE

The core functionality of the DS18S20 is its direct-to-digital temperature sensor. The temperature sensor output has 9-bit resolution, which corresponds to 0.5°C steps. The DS18S20 powers-up in a low-power idle state; to initiate a temperature measurement and A-to-D conversion, the master must issue a Convert T [44h] command. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18S20 returns to its idle state. If the DS18S20 is powered by an external supply, the master can issue "read time slots" (see the 1-WIRE BUS SYSTEM section) after the Convert T command and the DS18S20 will respond by transmitting 0 while the temperature conversion is in progress and 1 when the conversion is done. If the DS18S20 is powered with parasite power, this notification technique cannot be used since the bus must be pulled high by a strong pullup during the entire temperature conversion. The bus requirements for parasite power are explained in detail in the POWERING THE DS18S20 section of this datasheet.

The DS18S20 output data is calibrated in degrees centigrade; for Fahrenheit applications, a lookup table or conversion routine must be used. The temperature data is stored as a 16-bit sign-extended two's complement number in the temperature register (see Figure 2). The sign bits (S) indicate if the temperature is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. Table 2 gives examples of digital output data and the corresponding temperature reading.

Resolutions greater than 9 bits can be calculated using the data from the temperature, COUNT REMAIN and COUNT PER °C registers in the scratchpad. Note that the COUNT PER °C register is hard-wired to 16 (10h). After reading the scratchpad, the TEMP_READ value is obtained by truncating the 0.5°C bit (bit 0) from the temperature data (see Figure 2). The extended resolution temperature can then be calculated using the following equation:

$$TEMPERATURE = TEMP\_READ - 0.25 + \frac{COUNT\_PER\_C - COUNT\_REMAIN}{COUNT\_PER\_C}$$

Additional information about high-resolution temperature calculations can be found in Application Note 105: "High Resolution Temperature Measurement with Dallas Direct-to-Digital Temperature Sensors".

TEMPERATURE REGISTER FORMAT Figure 2

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
LS Byte	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}
	bit 15	**bit 14**	**bit 13**	**bit 12**	**bit 11**	**bit 10**	**bit 9**	**bit 8**
MS Byte	S	S	S	S	S	S	S	S

TEMPERATURE/DATA RELATIONSHIP Table 2

TEMPERATURE	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+85.0°C*	0000 0000 1010 1010	00AAh
+25.0°C	0000 0000 0011 0010	0032h
+0.5°C	0000 0000 0000 0001	0001h
0°C	0000 0000 0000 0000	0000h
-0.5°C	1111 1111 1111 1111	FFFFh
-25.0°C	1111 1111 1100 1110	FFCEh
-55.0°C	1111 1111 1001 0010	FF92h

*The power-on reset value of the temperature register is +85°C

OPERATION – ALARM SIGNALING

After the DS18S20 performs a temperature conversion, the temperature value is compared to the user-defined two's complement alarm trigger values stored in the 1-byte T_H and T_L registers (see Figure 3). The sign bit (S) indicates if the value is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. The T_H and T_L registers are nonvolatile (EEPROM) so they will retain data when the device is powered down. T_H and T_L can be accessed through bytes 2 and 3 of the scratchpad as explained in the MEMORY section of this datasheet.

T_H AND T_L REGISTER FORMAT Figure 3

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
S	2^6	2^5	2^5	2^5	2^2	2^1	2^0

Only bits 8 through 1 of the temperature register are used in the T_H and T_L comparison since T_H and T_L are 8-bit registers. If the result of a temperature measurement is higher than T_H or lower than T_L, an alarm condition exists and an alarm flag is set inside the DS18S20. This flag is updated after every temperature measurement; therefore, if the alarm condition goes away, the flag will be turned off after the next temperature conversion.

The master device can check the alarm flag status of all DS18S20s on the bus by issuing an Alarm Search [ECh] command. Any DS18S20s with a set alarm flag will respond to the command, so the master can determine exactly which DS18S20s have experienced an alarm condition. If an alarm condition exists and the T_H or T_L settings have changed, another temperature conversion should be done to validate the alarm condition.

POWERING THE DS18S20

The DS18S20 can be powered by an external supply on the V_{DD} pin, or it can operate in "parasite power" mode, which allows the DS18S20 to function without a local external supply. Parasite power is very useful for applications that require remote temperature sensing or that are very space constrained. Figure 1 shows the DS18S20's parasite-power control circuitry, which "steals" power from the 1-wire bus via the DQ pin when the bus is high. The stolen charge powers the DS18S20 while the bus is high, and some of the charge is stored on the parasite power capacitor (C_{PP}) to provide power when the bus is low. When the DS18S20 is used in parasite power mode, the V_{DD} pin must be connected to ground.

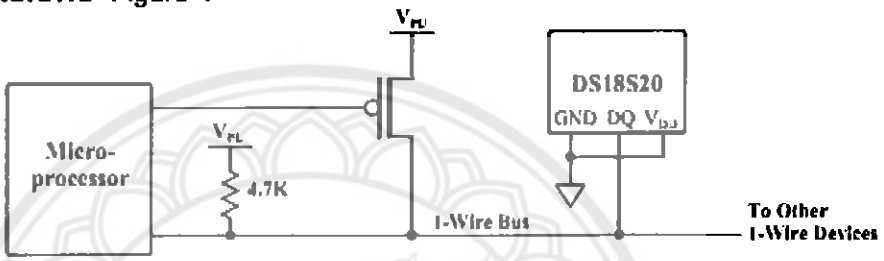
In parasite power mode, the 1-wire bus and C_{PP} can provide sufficient current to the DS18S20 for most operations as long as the specified timing and voltage requirements are met (refer to the DC ELECTRICAL CHARACTERISTICS and the AC ELECTRICAL CHARACTERISTICS sections of this data sheet). However, when the DS18S20 is performing temperature conversions or copying data from the scratchpad memory to EEPROM, the operating current can be as high as 1.5 mA. This current can cause an unacceptable voltage drop across the weak 1-wire pullup resistor and is more current than can be supplied by C_{PP}. To assure that the DS18S20 has sufficient supply current, it is necessary to provide a strong pullup on the 1-wire bus whenever temperature conversions are taking place or data is being copied from the scratchpad to EEPROM. This can be accomplished by using a MOSFET to pull the bus directly to the rail as shown in Figure 4. The 1-wire bus must be switched to the strong pullup within 10 µs (max) after a Convert T [44h] or Copy Scratchpad [48h] command is issued, and the bus must be held high by the pullup for the duration of the conversion (t_{CONV}) or data transfer (t_{WR} = 10 ms). No other activity can take place on the 1-wire bus while the pullup is enabled.

The DS18S20 can also be powered by the conventional method of connecting an external power supply to the V_{DD} pin, as shown in Figure 5. The advantage of this method is that the MOSFET pullup is not required, and the 1–wire bus is free to carry other traffic during the temperature conversion time.

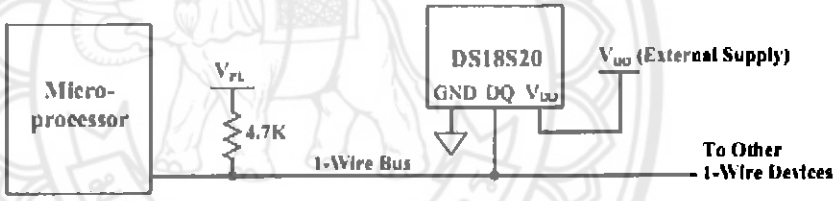
The use of parasite power is not recommended for temperatures above 100°C since the DS18S20 may not be able to sustain communications due to the higher leakage currents that can exist at these temperatures. For applications in which such temperatures are likely, it is strongly recommended that the DS18S20 be powered by an external power supply.

In some situations the bus master may not know whether the DS18S20s on the bus are parasite powered or powered by external supplies. The master needs this information to determine if the strong bus pullup should be used during temperature conversions. To get this information, the master can issue a Skip ROM [CCh] command followed by a Read Power Supply [B4h] command followed by a "read time slot". During the read time slot, parasite powered DS18S20s will pull the bus low, and externally powered DS18S20s will let the bus remain high. If the bus is pulled low, the master knows that it must supply the strong pullup on the 1-wire bus during temperature conversions.

SUPPLYING THE PARASITE-POWERED DS18S20 DURING TEMPERATURE CONVERSIONS Figure 4



POWERING THE DS18S20 WITH AN EXTERNAL SUPPLY Figure 5



64-BIT LASERED ROM CODE

Each DS18S20 contains a unique 64-bit code (see Figure 6) stored in ROM. The least significant 8 bits of the ROM code contain the DS18S20's 1-wire family code: 10h. The next 48 bits contain a unique serial number. The most significant 8 bits contain a cyclic redundancy check (CRC) byte that is calculated from the first 56 bits of the ROM code. A detailed explanation of the CRC bits is provided in the CRC GENERATION section. The 64-bit ROM code and associated ROM function control logic allow the DS18S20 to operate as a 1-wire device using the protocol detailed in the 1-WIRE BUS SYSTEM section of this datasheet.

64-BIT LASERED ROM CODE Figure 6

8-BIT CRC		48-BIT SERIAL NUMBER		8-BIT FAMILY CODE (10h)	
MSB	LSB	MSB	LSB	MSB	LSB

MEMORY

The DS18S20's memory is organized as shown in Figure 7. The memory consists of an SRAM scratchpad with nonvolatile EEPROM storage for the high and low alarm trigger registers (T_H and T_L). Note that if the DS18S20 alarm function is not used, the T_H and T_L registers can serve as general-purpose memory. All memory commands are described in detail in the DS18S20 FUNCTION COMMANDS section.

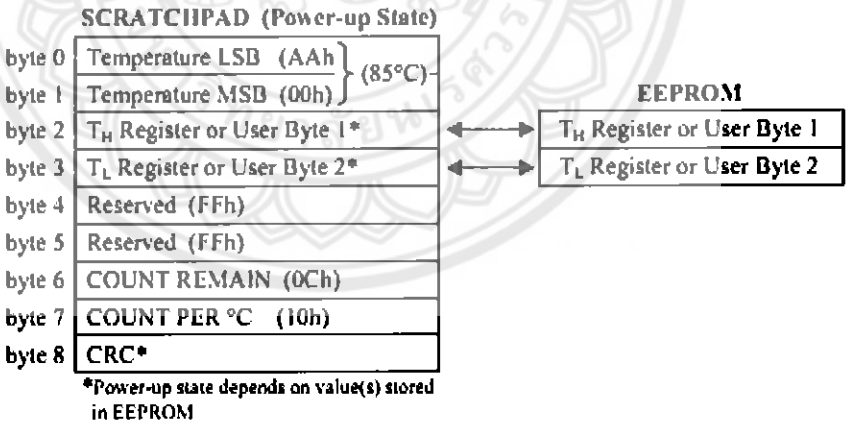
Byte 0 and byte 1 of the scratchpad contain the LSB and the MSB of the temperature register, respectively. These bytes are read-only. Bytes 2 and 3 provide access to T_H and T_L registers. Bytes 4 and 5 are reserved for internal use by the device and cannot be overwritten; these bytes will return all 1s when read. Bytes 6 and 7 contain the COUNT REMAIN and COUNT PER °C registers, which can be used to calculate extended resolution results as explained in the OPERATION – MEASURING TEMPERATURE section.

Byte 8 of the scratchpad is read-only and contains the cyclic redundancy check (CRC) code for bytes 0 through 7 of the scratchpad. The DS18S20 generates this CRC using the method described in the CRC GENERATION section.

Data is written to bytes 2 and 3 of the scratchpad using the Write Scratchpad [4Eh] command; the data must be transmitted to the DS18S20 starting with the least significant bit of byte 2. To verify data integrity, the scratchpad can be read (using the Read Scratchpad [BEh] command) after the data is written. When reading the scratchpad, data is transferred over the 1-wire bus starting with the least significant bit of byte 0. To transfer the T_H and T_L data from the scratchpad to EEPROM, the master must issue the Copy Scratchpad [48h] command.

Data in the EEPROM registers is retained when the device is powered down; at power-up the EEPROM data is reloaded into the corresponding scratchpad locations. Data can also be reloaded from EEPROM to the scratchpad at any time using the Recall E^2 [B8h] command. The master can issue "read time slots" (see the 1-WIRE BUS SYSTEM section) following the Recall E^2 command and the DS18S20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done.

DS18S20 MEMORY MAP



*Power-up state depends on value(s) stored in EEPROM

CRC GENERATION

CRC bytes are provided as part of the DS18S20's 64-bit ROM code and in the 9th byte of the scratchpad memory. The ROM code CRC is calculated from the first 56 bits of the ROM code and is contained in the most significant byte of the ROM. The scratchpad CRC is calculated from the data stored in the scratchpad, and therefore it changes when the data in the scratchpad changes. The CRCs provide the bus master with a method of data validation when data is read from the DS18S20. To verify that data has been read correctly, the bus master must re-calculate the CRC from the received data and then compare this value to either the ROM code CRC (for ROM reads) or to the scratchpad CRC (for scratchpad reads). If the calculated CRC matches the read CRC, the data has been received error free. The comparison of CRC values and the decision to continue with an operation are determined entirely by the bus master. There is no circuitry inside the DS18S20 that prevents a command sequence from proceeding if the DS18S20 CRC (ROM or scratchpad) does not match the value generated by the bus master.

The equivalent polynomial function of the CRC (ROM or scratchpad) is:

$$CRC = X^8 + X^5 + X^4 + 1$$

The bus master can re-calculate the CRC and compare it to the CRC values from the DS18S20 using the polynomial generator shown in Figure 8. This circuit consists of a shift register and XOR gates, and the shift register bits are initialized to 0. Starting with the least significant bit of the ROM code or the least significant bit of byte 0 in the scratchpad, one bit at a time should shifted into the shift register. After shifting in the 56th bit from the ROM or the most significant bit of byte 7 from the scratchpad, the polynomial generator will contain the re-calculated CRC. Next, the 8-bit ROM code or scratchpad CRC from the DS18S20 must be shifted into the circuit. At this point, if the re-calculated CRC was correct, the shift register will contain all 0s. Additional information about the Dallas 1-wire cyclic redundancy check is available in Application Note 27 entitled "Understanding and Using Cyclic Redundancy Checks with Dallas Semiconductor Touch Memory Products."

CRC GENERATOR Figure 8

1-WIRE BUS SYSTEM

The 1-wire bus system uses a single bus master to control one or more slave devices. The DS18S20 is always a slave. When there is only one slave on the bus, the system is referred to as a "single-drop" system; the system is "multi-drop" if there are multiple slaves on the bus.

All data and commands are transmitted least significant bit first over the 1-wire bus.

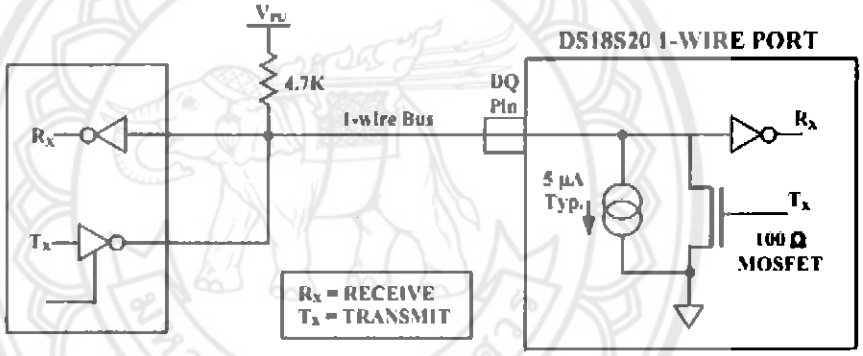
The following discussion of the 1-wire bus system is broken down into three topics: hardware configuration, transaction sequence, and 1-wire signaling (signal types and timing).

HARDWARE CONFIGURATION

The 1-wire bus has by definition only a single data line. Each device (master or slave) interfaces to the data line via an open drain or 3-state port. This allows each device to "release" the data line when the device is not transmitting data so the bus is available for use by another device. The 1-wire port of the DS18S20 (the DQ pin) is open drain with an internal circuit equivalent to that shown in Figure 9.

The 1-wire bus requires an external pullup resistor of approximately 5 kΩ; thus, the idle state for the 1-wire bus is high. If for any reason a transaction needs to be suspended, the bus MUST be left in the idle state if the transaction is to resume. Infinite recovery time can occur between bits so long as the 1-wire bus is in the inactive (high) state during the recovery period. If the bus is held low for more than 480 µs, all components on the bus will be reset.

HARDWARE CONFIGURATION Figure



TRANSACTION SEQUENCE

The transaction sequence for accessing the DS18S20 is as follows:

Step 1. Initialization

Step 2. ROM Command (followed by any required data exchange)

Step 3. DS18S20 Function Command (followed by any required data exchange)

It is very important to follow this sequence every time the DS18S20 is accessed, as the DS18S20 will not respond if any steps in the sequence are missing or out of order. Exceptions to this rule are the Search ROM [F0h] and Alarm Search [ECh] commands. After issuing either of these ROM commands, the master must return to Step 1 in the sequence.

INITIALIZATION

All transactions on the 1-wire bus begin with an initialization sequence. The initialization sequence consists of a reset pulse transmitted by the bus master followed by presence pulse(s) transmitted by the slave(s). The presence pulse lets the bus master know that slave devices (such as the DS18S20) are on the bus and are ready to operate. Timing for the reset and presence pulses is detailed in the 1-WIRE SIGNALING section.

ROM COMMANDS

After the bus master has detected a presence pulse, it can issue a ROM command. These commands operate on the unique 64–bit ROM codes of each slave device and allow the master to single out a specific device if many are present on the 1-wire bus. These commands also allow the master to determine how many and what types of devices are present on the bus or if any device has experienced an alarm condition. There are five ROM commands, and each command is 8 bits long. The master device must issue an appropriate ROM command before issuing a DS18S20 function command. A flowchart for operation of the ROM commands is shown in Figure 14.

SEARCH ROM [F0h]

When a system is initially powered up, the master must identify the ROM codes of all slave devices on the bus, which allows the master to determine the number of slaves and their device types. The master learns the ROM codes through a process of elimination that requires the master to perform a Search ROM cycle (i.e., Search ROM command followed by data exchange) as many times as necessary to identify all of the slave devices. If there is only one slave on the bus, the simpler Read ROM command (see below) can be used in place of the Search ROM process. For a detailed explanation of the Search ROM procedure, refer to the iButton Book of Standards at www.ibutton.com/ibuttons/standard.pdf. After every Search ROM cycle, the bus master must return to Step 1 (Initialization) in the transaction sequence.

READ ROM [33h]

This command can only be used when there is one slave on the bus. It allows the bus master to read the slave's 64-bit ROM code without using the Search ROM procedure. If this command is used when there is more than one slave present on the bus, a data collision will occur when all the slaves attempt to respond at the same time.

MATCH ROM [55h]

The match ROM command followed by a 64–bit ROM code sequence allows the bus master to address a specific slave device on a multi-drop or single-drop bus. Only the slave that exactly matches the 64–bit ROM code sequence will respond to the function command issued by the master; all other slaves on the bus will wait for a reset pulse.

SKIP ROM [CCh]

The master can use this command to address all devices on the bus simultaneously without sending out any ROM code information. For example, the master can make all DS18S20s on the bus perform simultaneous temperature conversions by issuing a Skip ROM command followed by a Convert T [44h] command. Note, however, that the Skip ROM command can only be followed by the Read Scratchpad [BEh] command when there is one slave on the bus. This sequence saves time by allowing the master to read from the device without sending its 64–bit ROM code. This sequence will cause a data collision on the bus if there is more than one slave since multiple devices will attempt to transmit data simultaneously.

ALARM SEARCH [ECh]

The operation of this command is identical to the operation of the Search ROM command except that only slaves with a set alarm flag will respond. This command allows the master device to determine if any DS18S20s experienced an alarm condition during the most recent temperature conversion. After

every Alarm Search cycle (i.e., Alarm Search command followed by data exchange), the bus master must return to Step 1 (Initialization) in the transaction sequence. Refer to the OPERATION – ALARM SIGNALING section for an explanation of alarm flag operation.

DS18S20 FUNCTION COMMANDS

After the bus master has used a ROM command to address the DS18S20 with which it wishes to communicate, the master can issue one of the DS18S20 function commands. These commands allow the master to write to and read from the DS18S20's scratchpad memory, initiate temperature conversions and determine the power supply mode. The DS18S20 function commands, which are described below, are summarized in Table 4 and illustrated by the flowchart in Figure 15.

CONVERT T [44h]

This command initiates a single temperature conversion. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18S20 returns to its low-power idle state. If the device is being used in parasite power mode, within 10 µs (max) after this command is issued the master must enable a strong pullup on the 1-wire bus for the duration of the conversion (t_{conv}) as described in the POWERING THE DS18S20 section. If the DS18S20 is powered by an external supply, the master can issue read time slots after the Convert T command and the DS18S20 will respond by transmitting 0 while the temperature conversion is in progress and 1 when the conversion is done. In parasite power mode this notification technique cannot be used since the bus is pulled high by the strong pullup during the conversion.

WRITE SCRATCHPAD [4Eh]

This command allows the master to write 2 bytes of data to the DS18S20's scratchpad. The first byte is written into the T_H register (byte 2 of the scratchpad), and the second byte is written into the T_L register (byte 3 of the scratchpad). Data must be transmitted least significant bit first. Both bytes MUST be written before the master issues a reset, or the data may be corrupted.

READ SCRATCHPAD [BEh]

This command allows the master to read the contents of the scratchpad. The data transfer starts with the least significant bit of byte 0 and continues through the scratchpad until the 9th byte (byte 8 – CRC) is read. The master may issue a reset to terminate reading at any time if only part of the scratchpad data is needed.

COPY SCRATCHPAD [48h]

This command copies the contents of the scratchpad T_H and T_L registers (bytes 2 and 3) to EEPROM. If the device is being used in parasite power mode, within 10 µs (max) after this command is issued the master must enable a strong pullup on the 1-wire bus for at least 10 ms as described in the POWERING THE DS18S20 section.

RECALL E^2 [B8h]

This command recalls the alarm trigger values (T_H and T_L) from EEPROM and places the data in bytes 2 and 3, respectively, in the scratchpad memory. The master device can issue read time slots following the Recall E^2 command and the DS18S20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done. The recall operation happens automatically at power-up, so valid data is available in the scratchpad as soon as power is applied to the device.

READ POWER SUPPLY [B4h]

The master device issues this command followed by a read time slot to determine if any DS18S20s on the bus are using parasite power. During the read time slot, parasite powered DS18S20s will pull the bus low, and externally powered DS18S20s will let the bus remain high. Refer to the POWERING THE DS18S20 section for usage information for this command.

DS18S20 FUNCTION COMMAND SET Table 4

Command	Description	Protocol	1-Wire Bus Activity After Command Is Issued	Notes
TEMPERATURE CONVERSION COMMANDS				
Convert T	Initiates temperature conversion.	44h	DS18S20 transmits conversion status to master (not applicable for parasite-powered DS18S20s).	1
MEMORY COMMANDS				
Read Scratchpad	Reads the entire scratchpad including the CRC byte.	BEh	DS18S20 transmits up to 9 data bytes to master.	2
Write Scratchpad	Writes data into scratchpad bytes 2 and 3 (T_H and T_L).	4Eh	Master transmits 2 data bytes to DS18S20.	3
Copy Scratchpad	Copies T_H and T_L data from the scratchpad to EEPROM.	48h	None	1
Recall E^2	Recalls T_H and T_L data from EEPROM to the scratchpad.	B8h	DS18S20 transmits recall status to master.	
Read Power Supply	Signals DS18S20 power supply mode to the master.	B4h	DS18S20 transmits supply status to master.	

NOTES:

1. For parasite-powered DS18S20s, the master must enable a strong pullup on the 1-wire bus during temperature conversions and copies from the scratchpad to EEPROM. No other bus activity may take place during this time.
2. The master can interrupt the transmission of data at any time by issuing a reset.
3. Both bytes must be written before a reset is issued.

1-WIRE SIGNALING

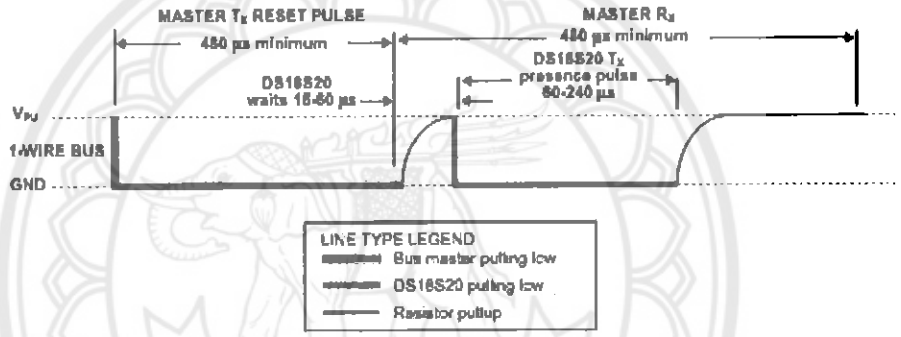
The DS18S20 uses a strict 1-wire communication protocol to insure data integrity. Several signal types are defined by this protocol: reset pulse, presence pulse, write 0, write 1, read 0, and read 1. All of these signals, with the exception of the presence pulse, are initiated by the bus master.

INITIALIZATION PROCEDURE: RESET AND PRESENCE PULSES

All communication with the DS18S20 begins with an initialization sequence that consists of a reset pulse from the master followed by a presence pulse from the DS18S20. This is illustrated in Figure 10. When the DS18S20 sends the presence pulse in response to the reset, it is indicating to the master that it is on the bus and ready to operate.

During the initialization sequence the bus master transmits (T_X) the reset pulse by pulling the 1-wire bus low for a minimum of 480 µs. The bus master then releases the bus and goes into receive mode (R_X). When the bus is released, the 5k pullup resistor pulls the 1-wire bus high. When the DS18S20 detects this rising edge, it waits 15–60 µs and then transmits a presence pulse by pulling the 1-wire bus low for 60–240 µs.

INITIALIZATION TIMING Figure 10



READ/WRITE TIME SLOTS

The bus master writes data to the DS18S20 during write time slots and reads data from the DS18S20 during read time slots. One bit of data is transmitted over the 1-wire bus per time slot.

WRITE TIME SLOTS

There are two types of write time slots: "Write 1" time slots and "Write 0" time slots. The bus master uses a Write 1 time slot to write a logic 1 to the DS18S20 and a Write 0 time slot to write a logic 0 to the DS18S20. All write time slots must be a minimum of 60 µs in duration with a minimum of a 1 µs recovery time between individual write slots. Both types of write time slots are initiated by the master pulling the 1-wire bus low (see Figure 11).

To generate a Write 1 time slot, after pulling the 1-wire bus low, the bus master must release the 1-wire bus within 15 µs. When the bus is released, the 5k pullup resistor will pull the bus high. To generate a Write 0 time slot, after pulling the 1-wire bus low, the bus master must continue to hold the bus low for the duration of the time slot (at least 60 µs). The DS18S20 samples the 1-wire bus during a window that lasts from 15 µs to 60 µs after the master initiates the write time slot. If the bus is high during the sampling window, a 1 is written to the DS18S20. If the line is low, a 0 is written to the DS18S20.

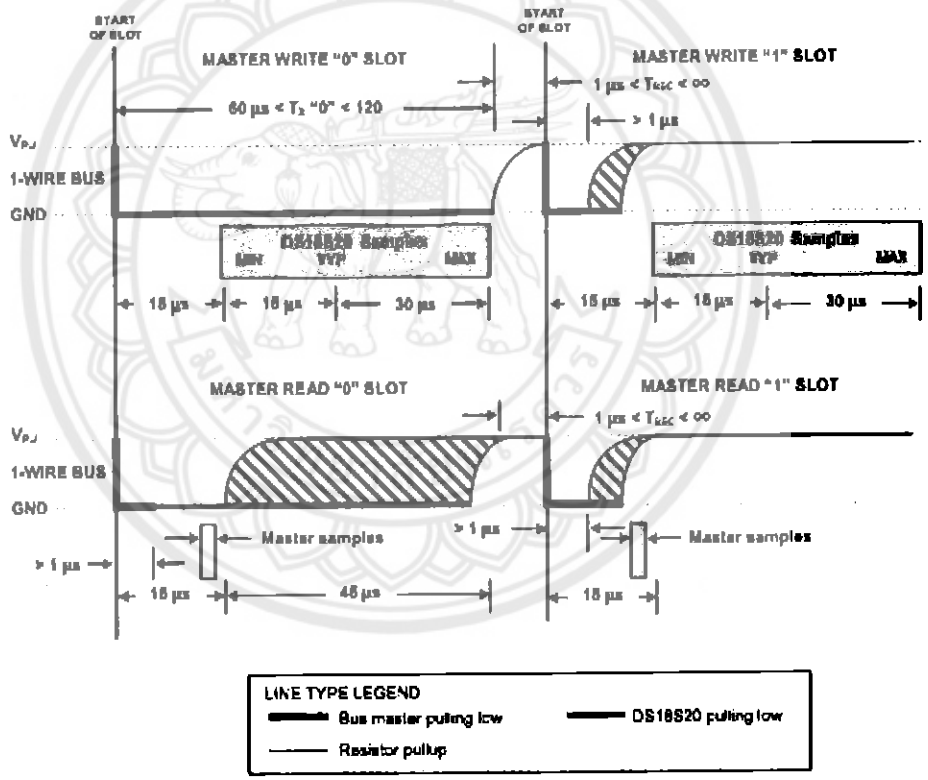
READ TIME SLOTS

The DS18S20 can only transmit data to the master when the master issues read time slots. Therefore, the master must generate read time slots immediately after issuing a Read Scratchpad [BEh] or Read Power Supply [B4h] command, so that the DS18S20 can provide the requested data. In addition, the master can generate read time slots after issuing Convert T [44h] or Recall E^2 [B8h] commands to find out the status of the operation as explained in the DS18S20 FUNCTION COMMAND section.

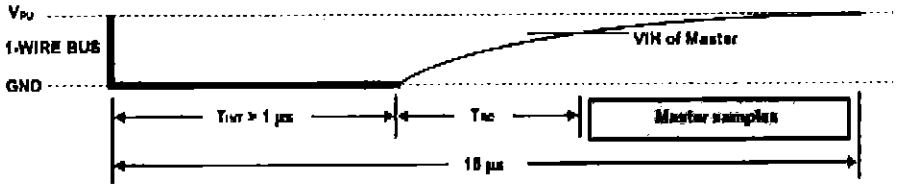
All read time slots must be a minimum of 60 µs in duration with a minimum of a 1 µs recovery time between slots. A read time slot is initiated by the master device pulling the 1-wire bus low for a minimum of 1 µs and then releasing the bus (see Figure 11). After the master initiates the read time slot, the DS18S20 will begin transmitting a 1 or 0 on bus. The DS18S20 transmits a 1 by leaving the bus high and transmits a 0 by pulling the bus low. When transmitting a 0, the DS18S20 will release the bus by the end of the time slot, and the bus will be pulled back to its high idle state by the pullup resister. Output data from the DS18S20 is valid for 15 µs after the falling edge that initiated the read time slot. Therefore, the master must release the bus and then sample the bus state within 15 µs from the start of the slot.

Figure 12 illustrates that the sum of T_{INIT}, T_{RC}, and T_{SAMPLE} must be less than 15 µs for a read time slot. Figure 13 shows that system timing margin is maximized by keeping T_{INIT} and T_{RC} as short as possible and by locating the master sample time during read time slots towards the end of the 15 µs period.

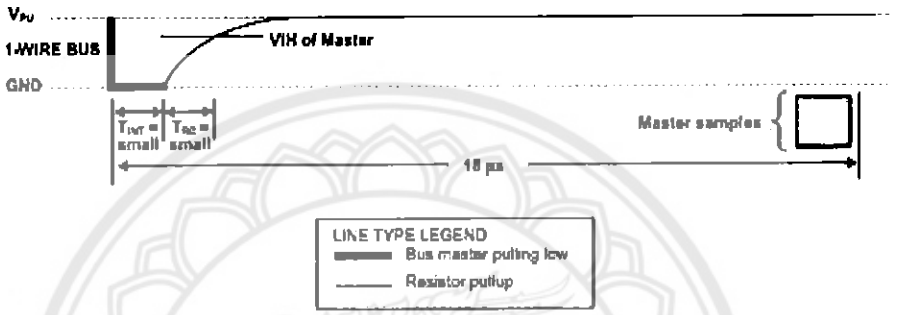
READ/WRITE TIME SLOT TIMING DIAGRAM Figure 11



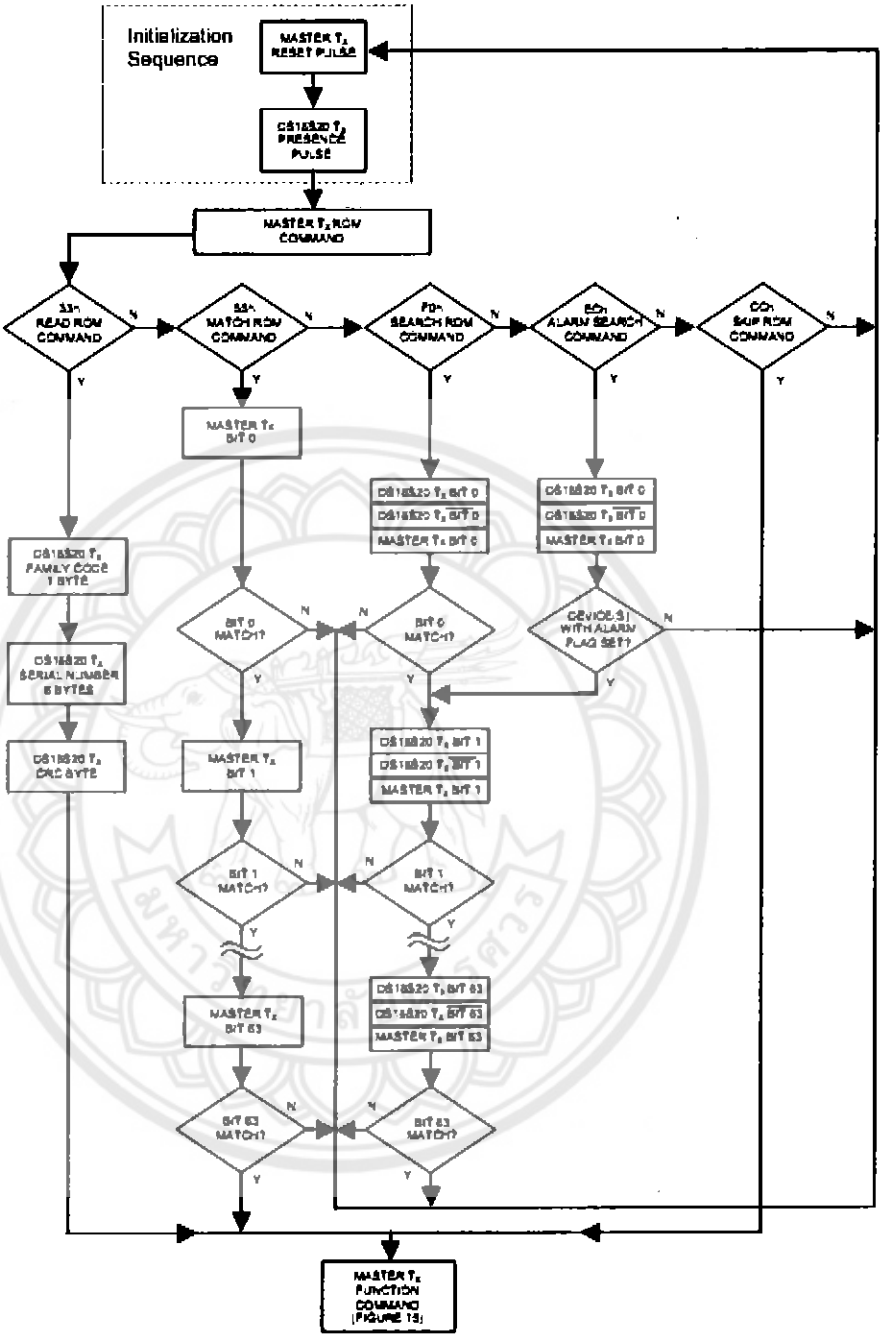
DETAILED MASTER READ 1 TIMING Figure 12



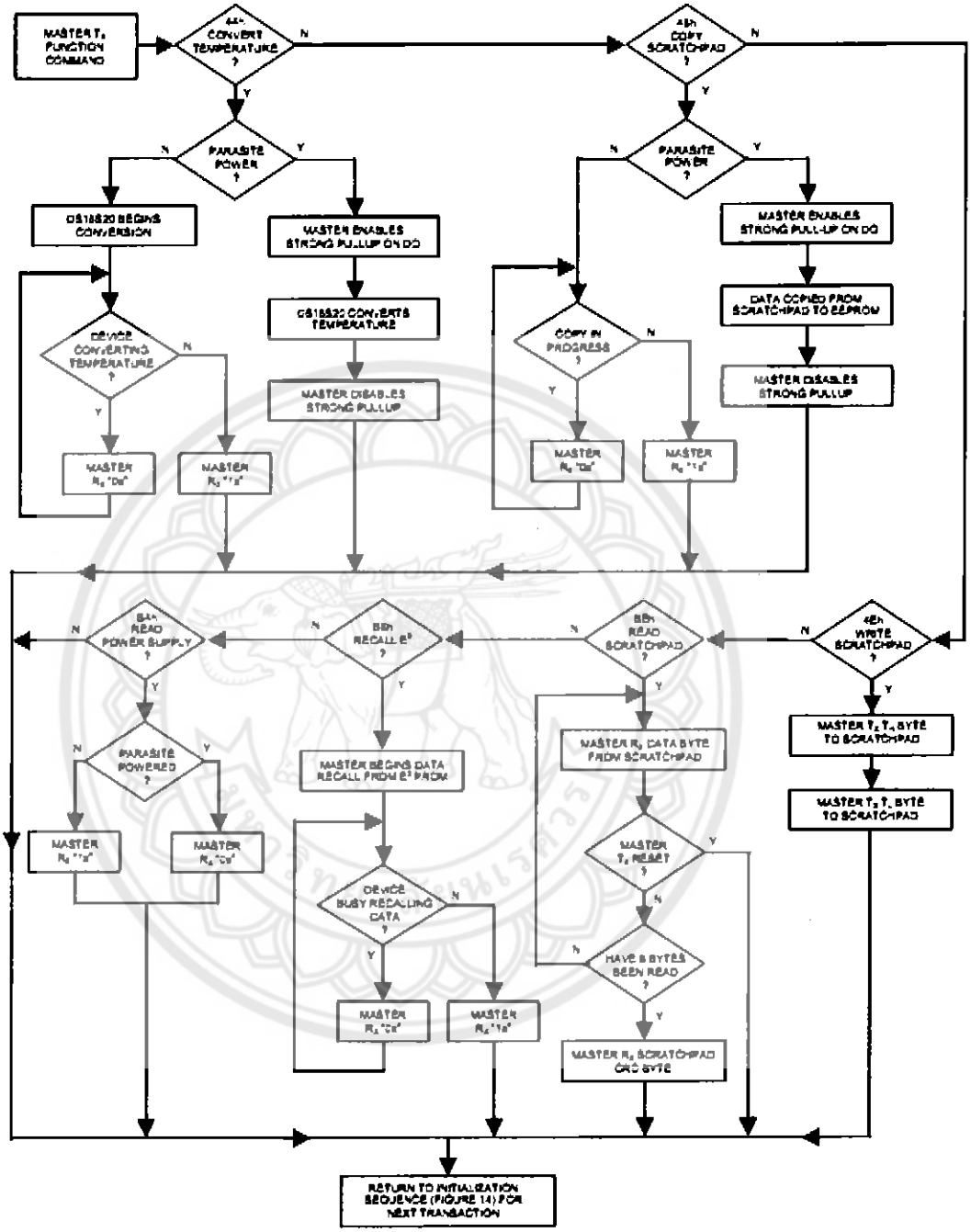
RECOMMENDED MASTER READ 1 TIMING Figure 13



ROM COMMANDS FLOW CHART Figure 14



DS18S20 FUNCTION COMMANDS FLOW CHART Figure 15



DS18S20 OPERATION EXAMPLE 1

In this example there are multiple DS18S20s on the bus and they are using parasite power. The bus master initiates a temperature conversion in a specific DS18S20 and then reads its scratchpad and recalculates the CRC to verify the data.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
TX	Reset	Master issues reset pulse.
RX	Presence	DS18S20s respond with presence pulse.
TX	55h	Master issues Match ROM command.
TX	64-bit ROM code	Master sends DS18S20 ROM code.
TX	44h	Master issues Convert T command.
TX	DQ line held high by strong pullup	Master applies strong pullup to DQ for the duration of the conversion (t_{conv}).
TX	Reset	Master issues reset pulse.
RX	Presence	DS18S20s respond with presence pulse.
TX	55h	Master issues Match ROM command.
TX	64-bit ROM code	Master sends DS18S20 ROM code.
TX	BEh	Master issues Read Scratchpad command.
RX	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.

DS18S20 OPERATION EXAMPLE 2

In this example there is only one DS18S20 on the bus and it is using parasite power. The master writes to the T_H and T_L registers in the DS18S20 scratchpad and then reads the scratchpad and recalculates the CRC to verify the data. The master then copies the scratchpad contents to EEPROM.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
TX	Reset	Master issues reset pulse.
RX	Presence	DS18S20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	4Eh	Master issues Write Scratchpad command.
TX	2 data bytes	Master sends two data bytes to scratchpad (T_H and T_L)
TX	Reset	Master issues reset pulse.
RX	Presence	DS18S20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	BEh	Master issues Read Scratchpad command.
RX	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.
TX	Reset	Master issues reset pulse.
RX	Presence	DS18S20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	48h	Master issues Copy Scratchpad command.
TX	DQ line held high by strong pullup	Master applies strong pullup to DQ for at least 10 ms while copy operation is in progress.

DS18S20 OPERATION EXAMPLE 3

In this example there is only one DS18S20 on the bus and it is using parasite power. The bus master initiates a temperature conversion then reads the DS18S20 scratchpad and calculates a higher resolution result using the data from the temperature, COUNT REMAIN and COUNT PER °C registers.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
TX	Reset	Master issues reset pulse.
TR	Presence	DS18S20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	44h	Master issues Convert T command.
TX	DQ line held high by strong pullup	Master applies strong pullup to DQ for the duration of the conversion (t_{conv}).
TX	Reset	Master issues reset pulse.
RX	Presence	DS18S20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	BEh	Master issues Read Scratchpad command.
RX	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated. The master also calculates the TEMP_READ value and stores the contents of the COUNT REMAIN and COUNT PER °C registers.
TX	Reset	Master issues reset pulse.
RX	Presence	DS18S20 responds with presence pulse.
-	-	CPU calculates extended resolution temperature using the equation in the OPERATION - MEASURING TEMPERATURE section of this datasheet.

RELATED APPLICATION NOTES

The following Application Notes can be applied to the DS18S20. These notes can be obtained from the Dallas Semiconductor "Application Note Book," via the Dallas website at http://www.dalsemi.com/ or through our faxback service at (214) 450–0441.

Application Note 27: "Understanding and Using Cyclic Redundancy Checks with Dallas Semiconductor Touch Memory Product"
Application Note 55: "Extending the Contact Range of Touch Memories"
Application Note 74: "Reading and Writing Touch Memories via Serial Interfaces"
Application Note 104: "Minimalist Temperature Control Demo"
Application Note 105: "High Resolution Temperature Measurement with Dallas Direct-to-Digital Temperature Sensors"
Application Note 106: "Complex MicroLANs"
Application Note 108: "MicroLAN – In the Long Run"

Sample 1-wire subroutines that can be used in conjunction with AN74 can be downloaded from the Dallas website or anonymous FTP Site.

AC ELECTRICAL CHARACTERISTICS: NV MEMORY

(-55°C to +100°C; V_{DD}=3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITION	MIN	TYP	MAX	UNITS
NV Write Cycle Time	t_{WR}			2	10	ms
EEPROM Writes	N_{EEWR}	-55°C to +55°C	50k			writes
EEPROM Data Retention	t_{EEDR}	-55°C to +55°C	10			years

AC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD}=3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITION	MIN	TYP	MAX	UNITS	NOTES
Temperature Conversion Time	t_{CONV}				750	ms	1
Time to Strong Pullup On	t_{SPON}	Start Convert T Command Issued			10	µs	
Time Slot	t_{SLOT}		60		120	µs	1
Recovery Time	t_{REC}		1			µs	1
Write 0 Low Time	t_{LOW0}		60		120	µs	1
Write 1 Low Time	t_{LOW1}		1		15	µs	1
Read Data Valid	t_{RDV}				15	µs	1
Reset Time High	t_{RSTH}		480			µs	1
Reset Time Low	t_{RSTL}		480			µs	1,2
Presence Detect High	t_{PDHIGH}		15		60	µs	1
Presence Detect Low	t_{PDLOW}		60		240	µs	1
Capacitance	$C_{IN/OUT}$				25	pF	

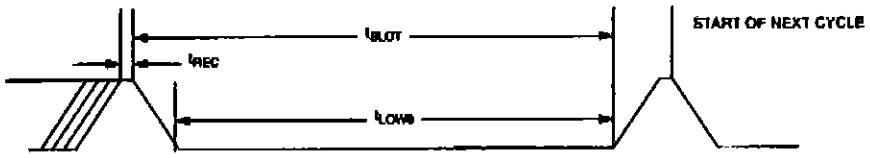
NOTES:

1. Refer to timing diagrams in Figure 17.
2. Under parasite power, if t_{RSTL} > 960 µs, a power on reset may occur.

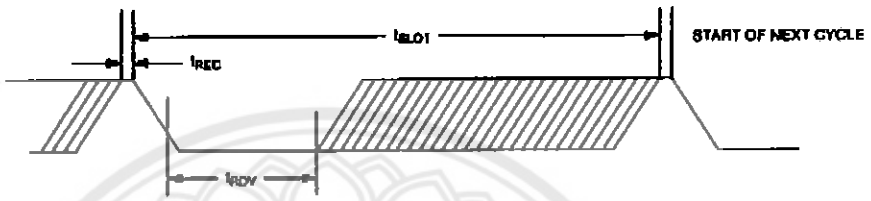
TYPICAL PERFORMANCE CURVE Figure 16

TIMING DIAGRAMS Figure 17

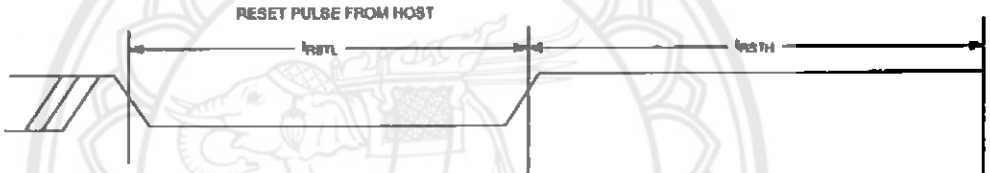
1-WIRE WRITE ZERO TIME SLOT



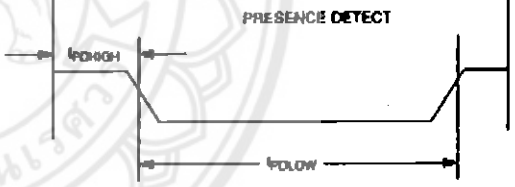
1-WIRE READ ZERO TIME SLOT



1-WIRE RESET PULSE



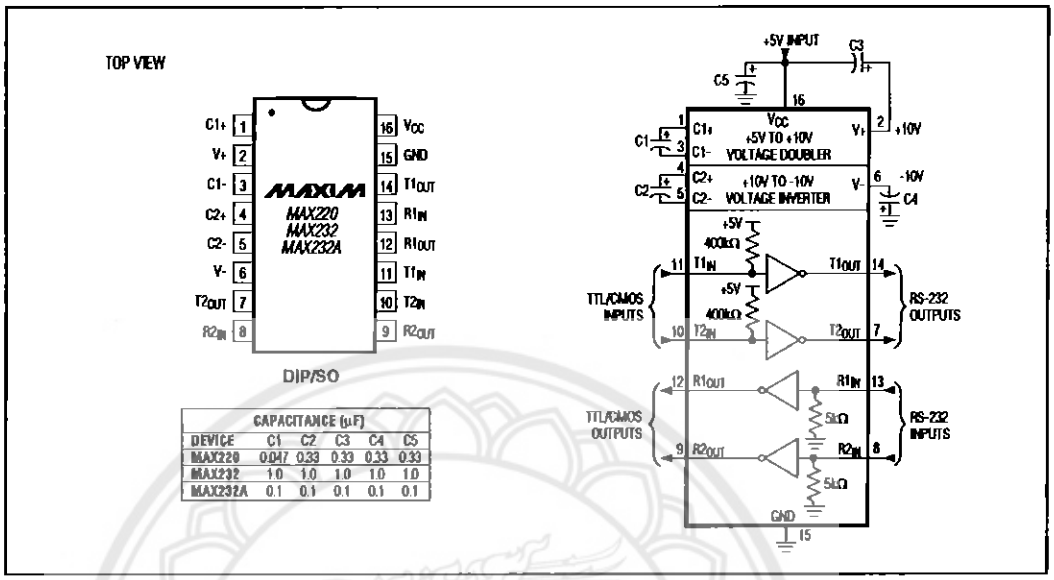
1-WIRE PRESENCE DETECT



ภาคผนวก จ

รายระเอียดของ ไอซีเบอร์ MAX 232

+5V-Powered, Multichannel RS-232 Drivers/Receivers



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	0.047	0.33	0.33	0.33	0.33
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit

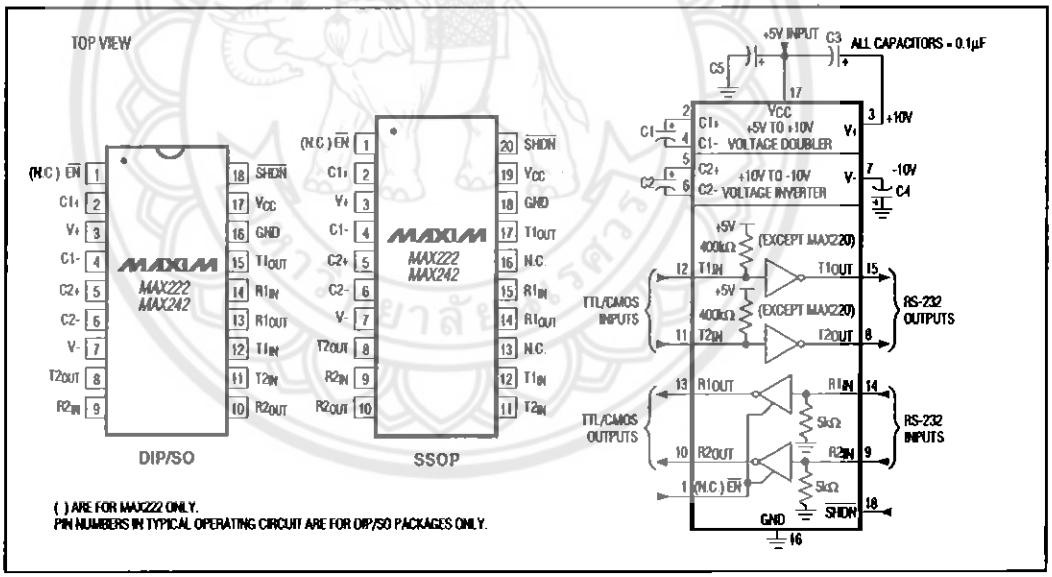


Figure 6. MAX222/MAX242 Pin Configurations and Typical Operating Circuit