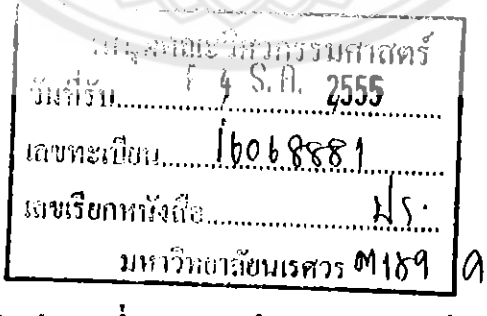
เครื่องผสมปุ๋ยน้ำอัตโนมัติ

AUTOMATIC LIQUID FERTILIZER MIXER

นางสาวตรีระนุช สังข์บัวแก้ว รหัส 47363866



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา 2552

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน เครื่องผสมปุ๋ยน้ำอัตโนมัติ

ผู้ดำเนินโครงงาน นางสาวตรีระนุช สังข์บัวแก้ว รหัส 47363866

ที่ปรึกษาโครงงาน ดร.แคทรียา สุวรรณศรี

สาขาวิชา วิศวกรรมไฟฟ้า

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2552

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

.....ที่ปรึกษาโครงงาน

(ดร.แคทรียา สุวรรณศรี)

.....กรรมการ

(ดร.นิพัทธ์ จันทรมินทร์)

.....กรรมการ

(ดร.ศุภวรรณ พลพิทักษ์ชัย)

ชื่อหัวข้อโครงงาน	เครื่องผสมปุ๋ยน้ำอัตโนมัติ
ผู้ดำเนินโครงงาน	นางสาวตรีระนุช สังข์บัวแก้ว รหัส 47363866
ที่ปรึกษาโครงงาน	ดร.แคทรียา สุวรรณศรี
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2552

.....

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้นำเสนอโครงงานเกี่ยวกับการออกแบบและสร้างเครื่องผสมปุ๋ยน้ำเพื่อการฉีดพ่นโดยตั้งเวลาการทำงานอัตโนมัติการขับเคลื่อนมอเตอร์จะดูดน้ำและปุ๋ยมายังถังผสมโดยใช้ไมโครคอนโทรลเลอร์ในการควบคุมการทำงานซึ่งทำงานร่วมกับ Real Time Clock (RTC) ผลที่ได้จากการทดสอบแสดงให้เห็นว่าเครื่องผสมปุ๋ยน้ำอัตโนมัติสามารถที่จะผสมน้ำและปุ๋ยในอัตราส่วนที่ต้องการได้ โดยน้ำที่ใช้ได้ถึง 20 ลิตร หลังจากผสมน้ำและปุ๋ยแล้วสามารถฉีดพ่นในเวลาที่กำหนดได้ ระดับน้ำและปุ๋ยน้ำ รวมถึงระยะเวลาในการผสมและฉีดพ่นสามารถเปลี่ยนได้ตามความต้องการของผู้ใช้งาน

Project title	Automatic Liquid Fertilizer Mixer
Name	Ms. Triranoot Sangbuakaew ID. 47262866
Project advisor	Mrs. Cattareeya Suwanasri, D.Eng.
Major	Electrical Engineering
Department	Electrical and Computer Engineering
Academic year	2009

.....

Abstract

This thesis presents a project on the design and construction of liquid fertilizer to the spray mixture. The motor will absorb water and fertilizer to the tank by using a microcontroller to control the operation that works with Real Time Clock (RTC). The results show that the automatic mixer can mix the water and fertilizer in the ratio desired up to 20 liters of water usage. After the water and fertilizer are mixed, the liquid fertilizer can be sprayed on-time manner. The levels of the liquid fertilizer as well as the spraying and mixing time can be changed depending on the user's requirement.

กิตติกรรมประกาศ

ในการทำปริญญานิพนธ์ฉบับนี้ ผู้จัดทำขอกราบขอบพระคุณ ดร.แคทรียา สุวรรณศรี เป็นอย่างสูงที่ได้กรุณาเสียสละเวลามาเป็นอาจารย์ที่ปรึกษา ได้ให้ความช่วยเหลือในการให้คำแนะนำและตรวจแก้ไขปริญญานิพนธ์ฉบับนี้ให้สำเร็จลงได้ด้วยดี และขอกราบขอบพระคุณ ผศ.ดร.ยงยุทธ ชนบดีเฉลิมรุ่ง อย่างสูงที่ได้ให้ความรู้รวมทั้งข้อเสนอแนะที่เป็นประโยชน์ซึ่งทำให้เนื้อหาของปริญญานิพนธ์ฉบับนี้สมบูรณ์ยิ่งขึ้น

ผู้จัดทำขอกราบขอบพระคุณ ดร.นิพัทธ์ จันทรมินทร์ และ ดร.สุภวรรณ พลพิทักษ์ชัย ที่ได้ให้ข้อเสนอแนะรวมทั้งความรู้ที่เป็นประโยชน์และกรุณาเสียสละเวลามาเป็นกรรมการสอบปริญญานิพนธ์ฉบับนี้

ท้ายที่สุดผู้จัดทำขอกราบขอบพระคุณ คุณพ่อ คุณแม่ และญาติพี่น้อง ที่คอยให้กำลังใจสนับสนุนด้านการศึกษาตลอดจนอาจารย์ทุกท่านที่ได้ประสิทธิ์ประสาทความรู้ให้กับผู้ดำเนินงานด้วยความรักตลอดมา

ตรีระนุช สังข์บัวแก้ว

สารบัญ

หน้า

ใบรับรองปริญญานิพนธ์.....ก
บทคัดย่อภาษาไทย.....ข
บทคัดย่อภาษาอังกฤษ.....ค
กิตติกรรมประกาศ.....ง
สารบัญ.....จ
สารบัญตาราง.....ช
สารบัญรูป.....ซ

บทที่ 1 บทนำ.....1
1.1 ความเป็นมาและความสำคัญของโครงงาน.....1
1.2 วัตถุประสงค์ของโครงงาน.....1
1.3 ขอบเขตของโครงงาน.....1
1.4 ขั้นตอนและแผนการดำเนินงาน.....2
1.5 ประโยชน์ที่ได้รับจากโครงงาน.....2
1.6 งบประมาณ.....3

บทที่ 2 ทฤษฎีและหลักการทำงาน.....4
2.1 ระบบการทำงานของเครื่องผสมปุ๋ยอัตโนมัติ.....4
2.2 ระบบควบคุมการทำงานของเครื่องผสมปุ๋ย.....6
2.3 วงจรแปลงไฟและคงค่าแรงดัน.....9
2.4 วงจรการตั้งเวลา.....9
2.5 วงจรจอแสดงผลแอลซีดี.....14
2.6 วงจรแผงปุ่มกด.....15
2.7 ภาษาสั่งงานคอมพิวเตอร์.....15

สารบัญ (ต่อ)

หน้า

บทที่ 3 การสร้างเครื่องผสมปุ๋ยอัตโนมัติ....18

3.1 ขั้นตอนการสร้างเครื่องผสมปุ๋ยอัตโนมัติ....18

3.2 ส่วนซอฟต์แวร์ของเครื่องผสมปุ๋ยอัตโนมัติ....18

3.3 ส่วนฮาร์ดแวร์ของเครื่องผสมปุ๋ยอัตโนมัติ....21

บทที่ 4 ผลการทดสอบ....26

4.1 ขั้นตอนการใช้งานเครื่องผสมปุ๋ยอัตโนมัติ....26

4.2 ผลการทดสอบเครื่องผสมปุ๋ยอัตโนมัติ....29

บทที่ 5 สรุปผลและข้อเสนอแนะ....35

5.1 สรุปผลการดำเนินโครงงาน....35

5.2 ปัญหาและแนวทางแก้ไข....35

5.3 แนวทางการพัฒนาต่อ....36

เอกสารอ้างอิง....37

ภาคผนวก ก รายละเอียดของไมโครคอนโทรลเลอร์รุ่น ATmega8515....38

ภาคผนวก ข รายละเอียดของไอซีหมายเลข ULN2803A....44

ภาคผนวก ค รายละเอียดของไอซีหมายเลข LM2575....48

ภาคผนวก ง รหัสต้นฉบับของโปรแกรมไมโครคอนโทรลเลอร์....53

ประวัติผู้ดำเนินโครงงาน....73

สารบัญตาราง

ตารางที่ หน้า

2.1 รายละเอียดขั้นต้นของขาต่อใช้งานของไอซีหมายเลข DS 1307.....10

2.2 การควบคุมความถี่ออสซิเลเตอร์ด้วยการตั้งค่าบิต RS1, RS0.....14

4.1 ผลการคำนวณระยะเวลาการผสมปุ๋ยน้ำ 1 รอบ.....32

4.2 ผลการทดสอบการผสมปุ๋ยน้ำจริง 1 รอบ.....32

4.3 ผลการคำนวณระยะเวลาการผสมปุ๋ยน้ำ 2 รอบ.....33

4.4 ผลการทดสอบการผสมปุ๋ยน้ำจริง 2 รอบ.....33

4.5 ค่าความคลาดเคลื่อนของเวลาระหว่างการคำนวณกับการทดสอบ.....34

สารบัญรูป

รูปที่ หน้า

2.1 สัญลักษณ์ของรีเลย์....5
2.2 ตำแหน่งขาของไอซีหมายเลข ULN2803A....5
2.3 การทำงานภายในตัวของไอซีหมายเลข ULN2803A....6
2.4 วงจรไมโครคอนโทรลเลอร์รุ่น ATmega8515....8
2.5 วงจรแปลงไฟจากกระแสสลับให้เป็นกระแสตรง....9
2.6 ตำแหน่งขาไอซี DS1307....10
2.7 การเชื่อมต่อ DS1307 เข้ากับไมโครคอนโทรลเลอร์ด้วยระบบบัส I2C....11
2.8 การรับส่งข้อมูลผ่านบัส I2C....12
2.9 การเขียนข้อมูลอุปกรณ์รองผ่านบัส I2C....12
2.10 การอ่านข้อมูลอุปกรณ์รองผ่านบัส I2C....13
2.11 รีจิสเตอร์ภายในไอซีฐานเวลา DS1307....13
2.12 การต่อใช้งานจอแสดงผลแอลซีดี....14
2.13 การต่อใช้งานแผงปุ่มกด....15
2.14 ผู้คิดค้นภาษาซี....16
3.1 แผนผังขั้นตอนการทำงานของกล่องควบคุม....18
3.2 โปรแกรมโค๊ดวิชั่นเอวีอาร์....19
3.3 โปรแกรมเอวีอาร์สตูดิโอ 4....20
3.4 เครื่องโปรแกรมไมโครคอนโทรลเลอร์ PX-400....20
3.5 กระบวนการทำงานของกล่องควบคุม....21
3.6 วงจรแหล่งจ่ายพลังงาน....22
3.7 การต่อใช้งานการควบคุมของไมโครคอนโทรลเลอร์รุ่น ATmega8515....23
3.8 ปั๊มน้ำ Hai Yang กำลังไฟ 15 วัตต์....23
3.9 มอเตอร์ไฟฟ้ากระแสตรงแบบทดรอบขนาด 12V รุ่น ZAG 37....24
3.10 วงจรตั้งเวลาการทำงาน....24

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.11 วงจรจอแสดงผล....24

3.12 วงจรแผงปุ่มกด....25

4.1 ลักษณะด้านบนของกล่องควบคุม....26

4.2 การประกอบสายของกล่องควบคุมพร้อมสวิตช์เปิดปิด....26

4.3 เวลาปัจจุบันบนหน้าจอแอลซีดี....27

4.4 จอแสดงผลแสดงข้อความให้ตั้งค่าปริมาณน้ำ....27

4.5 จอแสดงผลแสดงข้อความให้ตั้งค่าปริมาณปุ๋ย....28

4.6 จอแสดงผลแสดงข้อความให้ตั้งเวลาการทำงาน....28

4.7 จอแสดงผลแสดงข้อความให้ตั้งจำนวนรอบการทำงาน....28

4.8 จอแสดงผลแสดงข้อความตามสถานะการทำงานของมอเตอร์....29

4.9 จอแสดงผลแสดงข้อความเมื่อปริมาณน้ำหรือปุ๋ยไม่เพียงพอ....29

4.10 สายน้ำและปุ๋ยที่ต่อเข้ากับถังผสม....30

4.11 มอเตอร์ปั๊มดูดน้ำและปุ๋ยเข้าสู่ถังผสม....30

4.12 มอเตอร์กระแสตรงที่มีใบพัดติดอยู่กับฝาถังผสม....31

4.13 การต่อตัวฉีดพ่นกับถังผสม....31

4.14 การฉีดพ่นหลังการผสม....31

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

อาชีพเกษตรกรรมเป็นอาชีพพื้นฐานของคนไทย ดังนั้นการพัฒนาการเกษตรให้ทันสมัยทันต่อยุคโลกาภิวัตน์นั้น จำเป็นต้องอาศัยความรู้หลายด้านประกอบกัน เช่น ความรู้ด้านวิศวกรรม ด้านสถาปัตยกรรม ด้านเศรษฐศาสตร์ การตลาด การบัญชี เป็นต้น ซึ่งในด้านวิศวกรรมได้เข้ามามีบทบาทสำคัญในการพัฒนาด้านอุตสาหกรรมโลจิสติกส์ รวมถึงเกษตรกรรม ดังนั้นเราจำเป็นต้องพัฒนาด้านเกษตรกรรมให้เกิดความสะดวกสบายมากขึ้น ในขั้นตอนการผลิต และบริหารจัดการการประยุกต์ใช้ไมโครคอนโทรลเลอร์มาทำการสั่งงานควบคุมเครื่องจักรจะช่วยให้ประหยัดทรัพยากรบุคคลด้วย

การทำเกษตรกรรม ย่อมต้องมีการใส่ปุ๋ยหรือสารอินทรีย์เพื่อให้อาหารและป้องกันศัตรูพืช เช่นในสวนไม้ดอก สวนไม้ผล หรือแปลงผัก โดยปกติแล้วก็ต้องอาศัยแรงคนในการทำงานเพื่อเป็นการประหยัดรายจ่าย และในโครงงานนี้เราจะทำการสร้างเครื่องผสมปุ๋ยน้ำอัตโนมัติขึ้น ทำให้ได้ปุ๋ยในอัตราส่วนที่เหมาะสมเพิ่มความปลอดภัยให้กับผู้ใช้ ประหยัดต้นทุนและเวลาในการทำงาน

ในโครงงานฉบับนี้มุ่งเน้นศึกษาการทำงานของมอเตอร์ปั๊มร่วมกับ ไมโครคอนโทรลเลอร์ตระกูล MCS-51 สื่อสารกับ RTC – Real Time clock [1] เพื่อกำหนดตั้งเวลาผสมปุ๋ยและฉีดพ่น

1.2 วัตถุประสงค์

เพื่อออกแบบและสร้างเครื่องผสมปุ๋ยอัตโนมัติ ควบคุมการทำงานโดยใช้ไมโครคอนโทรลเลอร์

1.3 ขอบเขตของโครงงาน

1) สร้างเครื่องผสมปุ๋ยอัตโนมัติที่ใช้ไมโครคอนโทรลเลอร์ในการควบคุม
2) สร้างเครื่องผสมปุ๋ยอัตโนมัติที่มีประสิทธิภาพสามารถตั้งปริมาณน้ำและปุ๋ยได้
3) สร้างเครื่องผสมปุ๋ยอัตโนมัติที่สามารถตั้งเวลาการทำงานที่สอดคล้องกับฐานเวลาจริง

1.4 ขั้นตอนและแผนการดำเนินงาน

รายละเอียด	พ.ศ. 2554							พ.ศ. 2555		
	มิ.ย.	ก.ค.	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.
1) ศึกษาเรื่องการทำเครื่องผสมปุ๋ยอัตโนมัติและศึกษาการเขียนโปรแกรมไมโครคอนโทรลเลอร์	←	—	→							
2) ศึกษาการเลือกอุปกรณ์เพื่อใช้ในโครงงาน		←	—	→						
3) ออกแบบและสร้างวงจรควบคุม				←	—	→				
4) ทดสอบและปรับปรุงชิ้นงาน					←	—	→			
5) สรุปการดำเนินโครงงานและจัดทำรูปเล่มปริญญานิพนธ์								←	—	→

หมายเหตุ ระยะเวลาการทำกิจกรรมอาจมีการเปลี่ยนแปลงได้ตามความเหมาะสม

1.5 ประโยชน์ที่ได้รับจากโครงงาน

เครื่องผสมปุ๋ยอัตโนมัติที่สร้างขึ้นในโครงงานนี้ สามารถตั้งปริมาณน้ำและปุ๋ยได้ อีกทั้งยังสามารถตั้งเวลาการผสมและฉีดพ่นได้อีกด้วย อันเป็นการช่วยเหลือเกษตรกรในด้านการประหยัดเวลาและทุ่นแรงในการผสมหรือเดินฉีดพ่น รวมทั้งยังป้องกันสารเคมีจากปุ๋ยและยาฆ่าแมลงให้กับเกษตรกร

1.6 งบประมาณ

1. โครงสร้างของกล่องควบคุม	700	บาท
2. แผงวงจรไมโครคอนโทรลเลอร์	300	บาท
3. วงจรการตั้งเวลาการทำงาน	170	บาท
4. มอเตอร์ปั๊มน้ำ 3 ตัว	360	บาท
5. มอเตอร์กระแสตรง 1 ตัว	260	บาท
6. ค่าถ่ายเอกสารและค่าเข้าเล่มปริญญานิพนธ์	550	บาท
รวมเป็นเงินทั้งสิ้น (สองพันสามร้อยสี่สิบบาท)	2,340	บาท

หมายเหตุ: ถัวเฉลี่ยทุกรายการ

บทที่ 2
ทฤษฎีและหลักการทำงาน

ไมโครคอนโทรลเลอร์ได้ถูกนำมาประยุกต์ใช้งานกับการทำงานในหลายสาขาอาชีพ เช่น งานวิศวกรรม งานการแพทย์ รวมไปถึงงานเกษตร ซึ่งเทคโนโลยีดังกล่าวสามารถนำมาใช้งานเพื่อเพิ่มประสิทธิภาพ และขีดความสามารถได้เป็นอย่างดี ในโครงงานเล่มนี้เน้นการศึกษาการพัฒนาเทคโนโลยีที่ใช้ในงานด้านการเกษตร ในเรื่องของการผสมปุ๋ยในอัตราส่วนที่เหมาะสมถูกต้องและการตั้งเวลาฉีดพ่นให้กับผลผลิตทางการเกษตร เช่น พืช ผัก สวน ไร่ รวมไปถึงแปลงเพาะชำ เป็นต้น

ในบทนี้จะกล่าวถึงหลักการและทฤษฎีต่างๆ ที่เกี่ยวข้องกับการทำงานของระบบเครื่องผสมปุ๋ยน้ำอัตโนมัติ ความสำคัญที่เกิดจากการทำโครงงาน ซึ่งมีความจำเป็นอย่างยิ่งในการศึกษาเพื่อทำการทดลองและจะเป็นพื้นฐานองค์ประกอบทางความรู้ให้ผู้อ่านเกิดความเข้าใจในขั้นตอนการปฏิบัติต่อไป

2.1 ระบบการทำงานของเครื่องผสมปุ๋ยอัตโนมัติ

ระบบการทำงานของเครื่องผสมปุ๋ยอัตโนมัตินั้นเป็นการทำงานของมอเตอร์ปั๊ม เพื่อทำการดูด ผสม และฉีดพ่น ซึ่งควบคุมด้วยไมโครคอนโทรลเลอร์ ในการใช้งานมอเตอร์ปั๊มนั้นมีการใช้ควบคู่กับรีเลย์และไอซีหมายเลข ULN2803A

2.1.1 หลักการทำงานของมอเตอร์

มอเตอร์ไฟฟ้า เป็นเครื่องกลไฟฟ้าชนิดหนึ่งที่เปลี่ยนพลังงานไฟฟ้ามาเป็นพลังงานกล ประกอบด้วยขดลวดที่พันรอบแกนโลหะที่วางอยู่ระหว่างขั้วแม่เหล็ก โดยเมื่อผ่านกระแสไฟฟ้าเข้าไปยังขดลวดที่อยู่ระหว่างขั้วแม่เหล็ก จะทำให้ขดลวดหมุนไปรอบแกน และเมื่อสลับขั้วไฟฟ้าการหมุนของขดลวดจะหมุนกลับทิศทางเดิมมอเตอร์

มอเตอร์ไฟฟ้าแบ่งออกเป็น 2 ประเภท

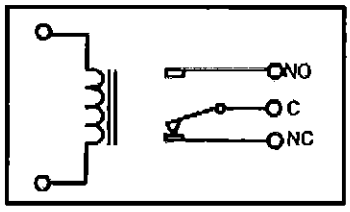
ก) มอเตอร์กระแสตรง เป็นมอเตอร์ที่ต้องใช้ไฟฟ้ากระแสตรงผ่านเข้าไปในขดลวดอาร์เมเจอร์ เพื่อทำให้เกิดการดูดและผลักกันของแม่เหล็กถาวรกับแม่เหล็กไฟฟ้าที่เกิดจากขดลวดมอเตอร์จึงหมุนได้

ข) ส่วนมอเตอร์กระแสสลับ เป็นมอเตอร์ที่ต้องใช้กับไฟฟ้ากระแสสลับ โดยใช้หลักการดูดและผลักกันของแม่เหล็กถาวรกับแม่เหล็กไฟฟ้าจากขดลวดมาทำให้เกิดการหมุนของมอเตอร์

2.1.2 หลักการทำงานของรีเลย์

รีเลย์ (Relay) คือ อุปกรณ์ทำหน้าที่เป็นสวิตช์มีหลักการทำงานคล้ายกับ ขดลวดแม่เหล็กไฟฟ้าหรือโซลินอยด์ (Solenoid) รีเลย์ใช้ในการควบคุมไฟฟ้าได้อย่างหลากหลาย รีเลย์เป็น

สวิตช์ควบคุมที่ทำงานด้วยไฟฟ้าทำหน้าที่ ตัด-ต่อวงจร คล้ายกับสวิตช์ โดยใช้หลักการหน้าสัมผัส

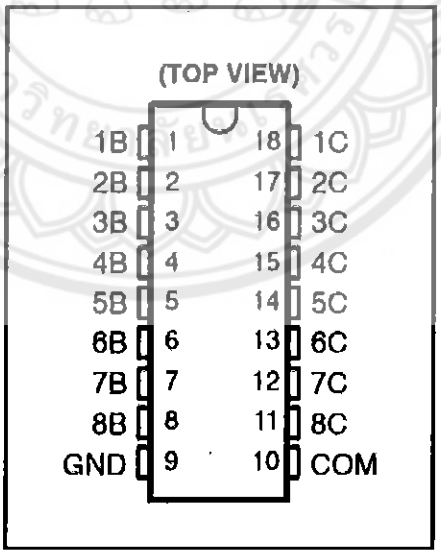


รูปที่ 2.1 สัญลักษณ์ของรีเลย์

การที่จะให้รีเลย์ทำงานก็ต้องจ่ายไฟให้รีเลย์ตามที่กำหนด เพราะเมื่อจ่ายไฟให้กับตัวรีเลย์ รีเลย์จะทำให้หน้าสัมผัสติดกัน กลายเป็นวงจรปิดและตรงข้ามทันทีที่ไม่ได้จ่ายไฟให้รีเลย์ก็จะกลายเป็นวงจรเปิดไฟที่เราใช้ป้อนให้กับตัวรีเลย์ก็จะเป็นไฟที่มาจากแหล่งจ่ายไฟของกล่องควบคุม ดังนั้นทันทีมีคำสั่งจ่ายไฟให้กับรีเลย์ก็จะทำให้รีเลย์ทำงาน

2.1.3 หลักการทำงานของไอซีหมายเลข ULN2803A [1]

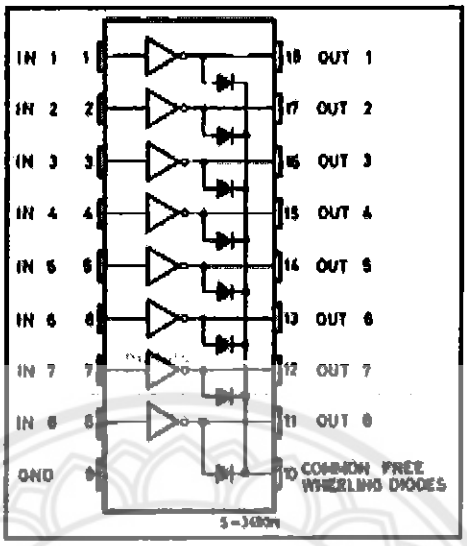
ไอซีหมายเลข ULN2803A เป็นไอซีไดฟ์เวอร์สามารถขับอุปกรณ์เอาท์พุตขนาดไม่เกิน 500 มิลลิแอมป์ เช่น มอเตอร์ รีเลย์ หรือสเต็ปปิ้งมอเตอร์ โดยเราสามารถจะนำมาให้ทำหน้าที่เป็นบับเฟอร์ (Buffer) ข้อมูลขนาด 8 บิต ในกรณีที่ต้องการเชื่อมต่อพอร์ตกับอุปกรณ์ภายนอก



รูปที่ 2.2 ตำแหน่งขาของไอซีหมายเลข ULN2803A [1]

ไอซี ULN2803A โดยภายในจะบรรจุอินเวอร์เตอร์เกต แบบคอลเล็กเตอร์เปิด จำนวน 7 ตัว โดยสามารถใช้กับแรงดันสูงสุดได้ถึง 50 โวลต์ และให้กระแสเอาท์พุตได้สูงสุดในแต่ละขา 500 มิลลิแอมป์ นอกจากนั้นยังมีไดโอดป้องกันไว้ที่ทุกขาของเอาท์พุต ทำให้สามารถต่อกับโหลดที่เป็น

ขดลวดได้โดยจะทำหน้าที่ขับรีเลย์ได้โดยตรง



รูปที่ 2.3 การทำงานภายในตัวของไอซีหมายเลข ULN2803A [1]

2.2 ระบบควบคุมการทำงานของเครื่องผสมปุ๋ยน้ำอัตโนมัติ

ในระบบการควบคุมการทำงานโดยการสร้างกล่องควบคุมซึ่งในโครงงานนี้ได้เลือกใช้ไมโครคอนโทรลเลอร์ตระกูล AVR ในการป้อนคำสั่งและประมวลผล ไมโครคอนโทรลเลอร์ตระกูล AVR เป็นไอซีไมโครคอนโทรลเลอร์ของบริษัท Atmel ใช้พลังงานต่ำ ทนความร้อนได้สูง โดยใช้สัญญาณนาฬิกาเพียง 1 ลูกในการปฏิบัติงานใน 1 คำสั่ง โดยจะประกอบด้วยหน่วยความจำโปรแกรมภายในที่เป็นแบบแฟลช์โปรแกรมข้อมูลได้แบบ In-System programmable และในบางหมายเลขยังสามารถมีการกำหนดตำแหน่งของหน่วยความจำที่สร้างเป็นบูตโหลดเดอร์ (เขียนโปรแกรมเพื่อติดต่อกับเครื่องคอมพิวเตอร์พีซี (PC) หรือไอซีตัวอื่นๆ และยังสามารถโปรแกรมให้กับตัวเองได้) มีขนาดของหน่วยความจำตามหมายเลขของไอซีแต่ละตัว

2.2.1 คุณสมบัติของไมโครคอนโทรลเลอร์ตระกูล AVR [2]

ก) เป็นไมโครคอนโทรลเลอร์ขนาด 8 บิต

ข) มีโครงสร้างภายในแบบ RISC (Reduced Instruction Set Computer)

ค) คำสั่งควบคุมการทำงานไมโครคอนโทรลเลอร์ 130 คำสั่ง คำสั่งส่วนมากจะทำสำเร็จในรอบสัญญาณนาฬิกาเดี่ยว

ง) มีจำนวนรีจิสเตอร์ทั่วไปขนาด 32 x 8

จ) มีหน่วยความจำโปรแกรมภายในแบบแฟลตขนาด 8 กิโลไบต์

ฉ) มีหน่วยความจำภายในแบบ EEPROM ขนาด 512 ไบต์

ช) หน่วยความจำภายในแบบ SRAM ขนาด 1 กิโลไบต์

ซ) เขียนและลบได้ถึง 10,000 ครั้ง สำหรับหน่วยความจำแบบ แฟลตและ 100,000 สำหรับหน่วยความจำแบบ EEPROM

ฌ) สามารถป้องการข้อมูลได้ (Programming Lock for Software Security)

ญ) มีไทม์เมอร์เคาท์เตอร์ 2 ตัว ขนาด 8 บิต

2.2.2 ส่วนประกอบของไมโครคอนโทรลเลอร์

ก) หน่วยประมวลผล (Central Processor Unit) เป็นส่วนที่ตัดสินใจเกี่ยวกับการทำงานต่างๆ

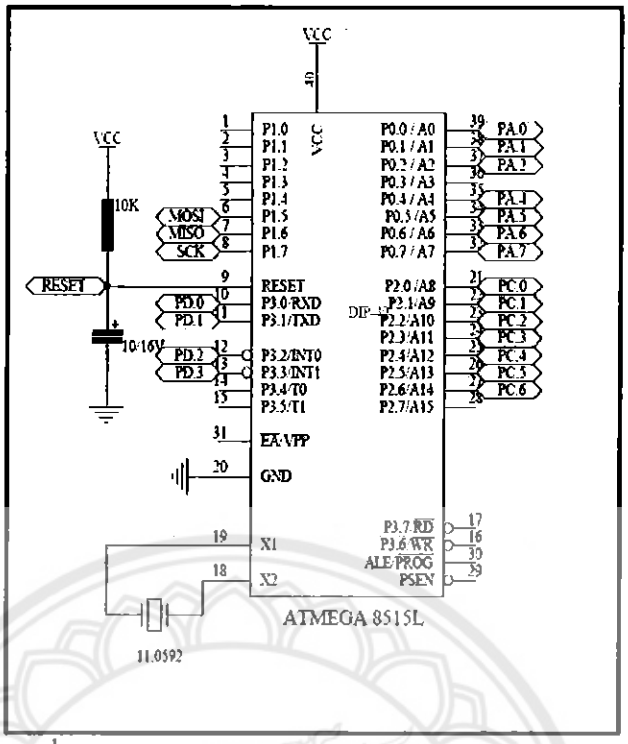
ข) หน่วยความจำ (Memory) เป็นเก็บข้อมูลต่าง ๆ ที่ต้องใช้ในไมโครคอนโทรลเลอร์ไม่ว่าจะเป็นหน่วยความจำโปรแกรมหรือหน่วยความจำข้อมูล โดยหน่วยความจำที่ใช้ ได้แก่ ROM, EPROM, EEPROM, RAM รวมทั้งหน่วยความจำแบบแฟลต

ค) พอร์ตสัญญาณเข้าและสัญญาณออก (Input & Output Port) เป็นส่วนที่ใช้ติดต่อกับอุปกรณ์ภายนอก

ง) คุณสมบัติอื่น ๆ เพิ่มเติม เช่น Separate Prescaler โหมด Compare อีก 1 ตัว, มี PWM 3 ช่อง, มีการติดต่อแบบ Master/Slave SPI Serial Interface, ทำงานที่แรงดัน 2.7 - 5.5V และทำงานที่ความถี่0 - 16 MHz

2.2.3 รายละเอียดขาสัญญาณและการใช้งาน

ในส่วนของวงจรไมโครคอนโทรลเลอร์ก่อนจะนำไปใช้งาน ต้องต่อวงจรพื้นฐานเพิ่มดังรูปที่ 2.4 คือวงจรรีเซต ประกอบด้วยตัวต้านทานขนาด 10 กิโลโอห์ม และตัวเก็บประจุขนาด 10/16V ฟารัด ต่อกับขา 9 ของไมโครคอนโทรลเลอร์ เพื่อทำการรีเซตไมโครคอนโทรลเลอร์ให้พร้อมใช้งานทุกครั้ง หลังจากเปิดไฟเข้าเครื่อง และวงจรสร้างสัญญาณนาฬิกา ใช้คริสตัล (Crystal) ขนาด 11.0592 MHz เพื่อทำการสร้างสัญญาณนาฬิกาส่งไปยังไมโครคอนโทรลเลอร์ เพื่อให้ไมโครคอนโทรลเลอร์ทำงานตามจังหวะของสัญญาณนาฬิกา



รูปที่ 2.4 วงจรไมโครคอนโทรลเลอร์รุ่น ATmega8515

รายละเอียดขาสัญญาณของไมโครคอนโทรลเลอร์รุ่น ATmega8515

P0.0-P0.7 (ขา 39-32) เป็นพอร์ตอินพุตเอาท์พุต กำหนดการพลูอัพภายในขาพอร์ตและใช้งานเป็นอินพุตอนาลอก

P1.0-P1.7 (ขา 1-8) เป็นพอร์ตอินพุตเอาท์พุต ขาสำหรับโปรแกรมชิพ ขาป้อนสัญญาณภายนอก

P2.0-P2.7 (ขา 21-28) เป็นพอร์ตอินพุตเอาท์พุต ขาเชื่อมต่อดีบักและโปรแกรมด้วยการเชื่อมต่อแบบ JTAG

P3.0-P3.7 (ขา10-17) เป็นพอร์ตอินพุตเอาท์พุต เชื่อมต่ออนุกรม อินเตอร์รัพต์เนื่องจากสัญญาณภายนอก

ขา EA (External Access Enable) หรือขาที่ 31 เป็นขาที่ทำหน้าที่เลือกการทำงานของไมโครคอนโทรลเลอร์ว่าจะใช้หน่วยความจำโปรแกรมภายนอกหรือความจำโปรแกรมภายใน กรณีเลือกใช้ความจำโปรแกรมภายนอกขา EA ต้องเป็นลอจิก 0 ส่วนในกรณีที่ใช้หน่วยความจำโปรแกรมภายในขา EA ต้องเป็นลอจิก 1 สำหนับไมโครคอนโทรลเลอร์รุ่น ATmega8515 นั้นมีความจำโปรแกรมแบบแฟลตขนาด 64 กิโลไบท์

ขา RESET หรือขาที่ 9 ทำหน้าที่รีเซ็ตการทำงานของไมโครคอนโทรลเลอร์

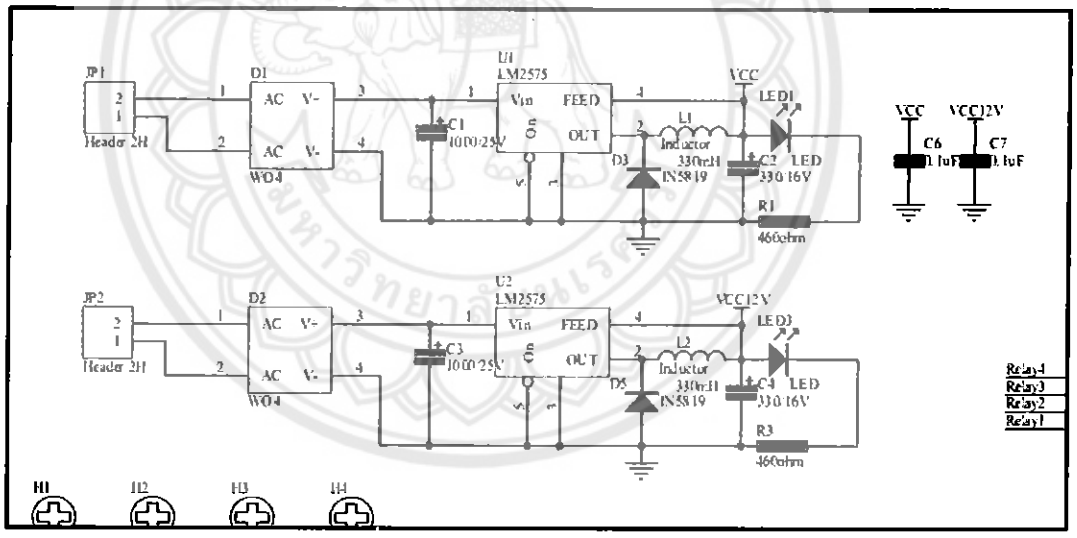
ขา XTAL1 และ XTAL2 หรือขาที่ 19 และ 18 ตามลำดับเป็นขาที่ทำหน้าที่สำหรับต่อกับคริสตอลเพื่อสร้างสัญญาณนาฬิกาให้กับไมโครคอนโทรลเลอร์

ขา ALE (Address Latch Enable) หรือขาที่ 30 เป็นขาที่ทำหน้าที่ควบคุมการแลตช์ของขาสายเชื่อม 0 เมื่อมีการต่อใช้งานกับหน่วยความจำภายนอก

ขา PSEN (Program Store Enable) หรือขาที่ 29 ทำหน้าที่ส่งสัญญาณเพื่อร้องขอการติดต่อกับหน่วยความจำโปรแกรมภายนอก สำหรับในไมโครคอนโทรลเลอร์บางรุ่นขา PSEN นี้ใช้ในการควบคุมสภาวะการทำงานของไมโครคอนโทรลเลอร์สำหรับการโหลดโปรแกรมลงบนไมโครคอนโทรลเลอร์ด้วย

2.3 วงจรแปลงไฟและคงค่าแรงดัน

สำหรับวงจรแปลงไฟและคงค่าแรงดันแสดงได้ดังรูปที่ 2.5 มีวงจรสองวงจรในวงจรคงค่าแรงดันวงจรแรกใช้ตัวคงค่าแรงดันหมายเลข LM2575t ทำหน้าที่แปลงไฟกระแสสลับขนาด 9 โวลต์ ให้เป็นไฟกระแสตรงขนาด 5 โวลต์ ส่วนวงจรที่สองเป็นวงจรคงค่าแรงดันใช้ตัวคงค่าแรงดันหมายเลข LM2575t ทำหน้าที่แปลงไฟกระแสสลับขนาด 12โวลต์ ให้เป็นไฟกระแสตรงขนาด 12 โวลต์

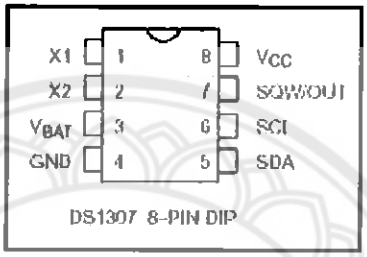


รูปที่ 2.5 วงจรแปลงไฟจากกระแสสลับให้เป็นกระแสตรง

2.4 วงจรตั้งเวลาการทำงาน [3]

ในวงจรการตั้งเวลาการทำงานในโครงงานนี้เราจะใช้ Real Time Clock (RTC) [3] ระบบฐานเวลาเป็นสิ่งสำคัญที่สามารถไปใช้ในอุปกรณ์อิเล็กทรอนิกส์ได้หลากหลาย ภายในไมโครคอนโทรลเลอร์เองก็มีไทเมอร์เพื่อใช้ในการจับเวลาหรือนำไปใช้เป็นฐานเวลาจริงได้เช่นกัน แต่เนื่องจากไมโครคอนโทรลเลอร์สามารถทำงานได้ต่อเมื่อมีไฟเลี้ยงเท่านั้น ดังนั้นการใช้ไทเมอร์ของไมโครคอนโทรลเลอร์สร้างฐานเวลาจริงจึงไม่เหมาะในบางแอฟพลิเคชัน DS 1307 เป็นไอซี

ฐานเวลาของดัลลัสเซมิคอนดักเตอร์ (Dallas Semiconductor) มีบัสรับส่งข้อมูลแบบ I2C (Inter-IC Communication) ซึ่งเป็นแบบที่สามารถสื่อสารได้สองทิศทาง (Bi-direction bus) ฐานเวลาของ DS 1307 นั้นสามารถเก็บข้อมูลวินาที นาที ชั่วโมง วัน วันที่ เดือนและปีได้ ระบบเวลาสามารถทำงานโหมดรูปแบบ 24 ชั่วโมง หรือ 12 ชั่วโมง ภายในมีระบบตรวจสอบแหล่งจ่ายไฟ โดยถ้าแหล่งจ่ายไฟหลักถูกตัดไป DS1307 สามารถสวิตช์ไปใช้ไฟจากแบตเตอร์รี่และทำงานต่อไป โดยที่ยังสามารถรักษาข้อมูลไว้ได้ โครงสร้างมีขาทั้งหมด 8 ขาดังแสดงในรูปที่ 2.6 และมีรายละเอียดการทำงานของขาต่างๆ ดังนี้



รูปที่ 2.6 ตำแหน่งขาไอซี DS1307 [3]

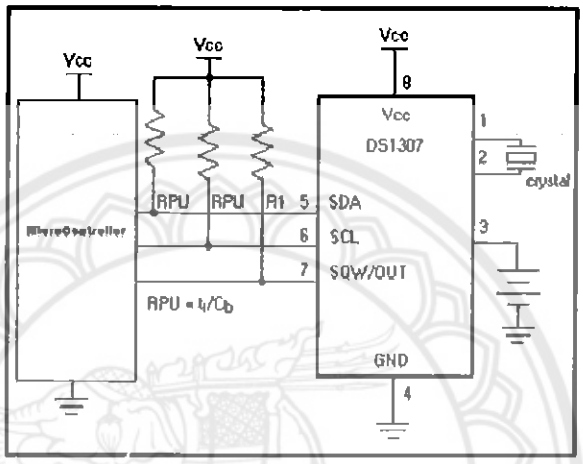
ตารางที่ 2.1 รายละเอียดขั้นต้นของขาต่อใช้งานของไอซีหมายเลข DS1307

ขา	รายละเอียด
1 - 2	X1, X2: ใช้ต่อกับคริสตอลความถี่ 32.768 kHz เพื่อสร้างฐานเวลาจริงให้กับไอซี
3	VBAT: ใช้ต่อกับแบตเตอรี่ 3 โวลต์เพื่อรักษาการทำงาน ในกรณีที่ไม่มีไฟเลี้ยงจ่าย
4	GND: ใช้ต่อกราวด์
5	SDA: ขารับส่งข้อมูลด้วยระบบบัส I2C
6	SCL: ขาสัญญาณนาฬิกาสำหรับการรับส่งข้อมูลด้วยระบบบัส I2C
7	SQW/OUT: ขาเอาท์พุตสัญญาณสแคร์วเวฟ (Square Wave) สามารถเลือกความถี่ได้
8	VCC: ใช้ต่อไฟเลี้ยง +5V

2.4.1 ระบบการรับส่งข้อมูลแบบI2C (Inter-IC Communication)

ระบบบัสข้อมูลแบบ I2C (Inter-IC Communication) ได้ถูกพัฒนาขึ้นโดยบริษัทฟิลิปส์ (Phillips) การรับส่งข้อมูลใช้สายสัญญาณเพียงแค่ 2 เส้น คือสายสัญญาณข้อมูล SDA (Serial Data Line) และสายสัญญาณนาฬิกา SCL (Serial Clock Line) มีการทำงานเป็นแบบมาสเตอร์และสเลป (Master, Slave) โดยอุปกรณ์ที่ทำหน้าที่เป็นมาสเตอร์ (Master) คือ ไมโครคอนโทรลเลอร์จะควบคุมการรับส่งข้อมูล และควบคุมสัญญาณนาฬิกาบน SCL ส่วนอุปกรณ์สเลป (Slave) คือ DS1307 นั้นจะทำงานภายใต้การควบคุมของอุปกรณ์มาสเตอร์ (Master)

การต่อใช้งานร่วมกับไมโครคอนโทรลเลอร์ด้วยระบบบัส I2C นั้นสามารถทำได้โดยต่อตัวต้านทานพลูอัพดังแสดงในรูปที่ 2.7 ในกรณีที่ต้องการต่อร่วมกับอุปกรณ์สแลป (Slave) หลายตัว ก็สามารถทำได้โดยต่ออุปกรณ์สแลป (Slave) ขนานกันไป การติดต่อสื่อสารระหว่างอุปกรณ์มาสเตอร์ (Master) กับสแลป (Slave) แต่ละตัวนั้น จะถูกแยกโดยแอดเดรส (Address) ของอุปกรณ์สแลป (Slave) ซึ่งจะถูกส่งจากอุปกรณ์มาสเตอร์ (Master) ไปยังอุปกรณ์สแลป (Slave) ก่อนเริ่มการรับส่งข้อมูล



รูปที่ 2.7 การเชื่อมต่อ DS1307 เข้ากับไมโครคอนโทรลเลอร์ด้วยระบบบัสแบบ I2C

การรับส่งข้อมูลแบบ I2C นั้นมีข้อกำหนดอยู่ 2 ประการคือ

ก) การรับส่งข้อมูลจะเริ่มขึ้นได้เมื่อบัสมีสถานะว่างเท่านั้น

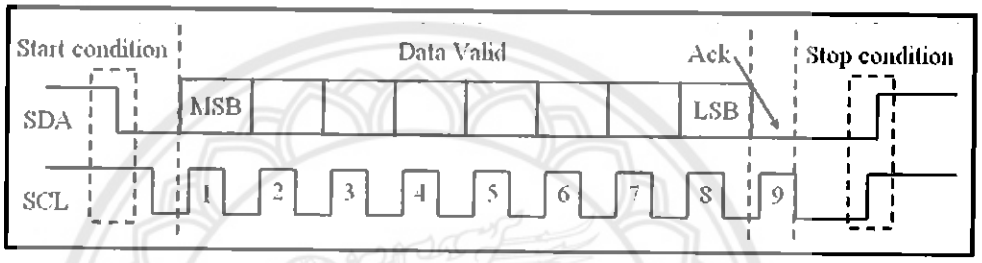
ข) ในช่วงที่ทำการรับส่งข้อมูลอยู่ สายสัญญาณ SDA ต้องไม่เปลี่ยนสถานะในช่วงที่ SCL มีสถานะเป็นลอจิก "1" ถ้า SDA มีการเปลี่ยนสถานะในช่วงที่ SCL เป็นลอจิก "1" จะถือว่าเป็นสัญญาณควบคุมการรับส่งข้อมูล

สถานะของการรับส่งข้อมูลแบบ I2C สามารถแบ่งออกได้เป็น 5 สถานะด้วยกันดังแสดงในรูปที่ 2.8 และมีรายละเอียดดังนี้

1. สถานะว่าง (Bus not busy) สัญญาณ SDA และ SCL มีระดับสัญญาณเป็น High
2. เริ่มส่งข้อมูล (Start data transfer) มีการเปลี่ยนระดับสัญญาณของ SDA จาก High เป็น Low ในขณะที่ SCL มีระดับสัญญาณเป็น High ค้างไว้
3. หยุดส่งข้อมูล (Stop data transfer) มีการเปลี่ยนระดับสัญญาณของ SDA จาก Low เป็น High ในขณะที่ SCL มีระดับสัญญาณเป็น High ค้างไว้
4. รับส่งข้อมูล (Data valid) มีการรับส่งข้อมูลผ่านสายสัญญาณ SDA โดยข้อมูลแต่ละบิตจะถูกส่งในช่วงที่ SCL มีระดับเป็น High โดยในช่วงที่ SCL มีสถานะเป็น High อยู่นั้น SDA

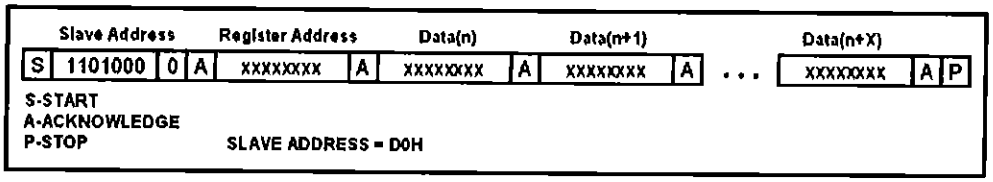
จะต้องไม่เกิดการเปลี่ยนระดับสัญญาณ SDA (Serial Data Line) จะเปลี่ยนระดับของสัญญาณ ในช่วงที่ SCL มีระดับสัญญาณเป็น Low เท่านั้น ตามมาตรฐานการส่งข้อมูล แบบ I2C นี้สามารถส่งข้อมูลด้วยความถี่สัญญาณนาฬิกาสูงสุด 100 kHz ที่โหมดการทำงานธรรมดา และ 400 kHz ที่โหมดการทำงานแบบเร็ว แต่สำหรับ DS1307 สามารถทำงานได้ในโหมดธรรมดาเท่านั้น

5. ตอบรับ (Acknowledge) เกิดขึ้นหลังจากที่มีการรับส่งข้อมูลครบแล้ว โดยอุปกรณ์ Master ต้องสร้างสัญญาณ Clock บน SCL เพิ่มอีกลูก อุปกรณ์ที่เป็นตัวรับข้อมูลจะดึงระดับสัญญาณบน SDA ให้เป็น Low เพื่อให้ตัวส่งรับรู้ว่าตัวรับได้รับข้อมูลครบแล้ว



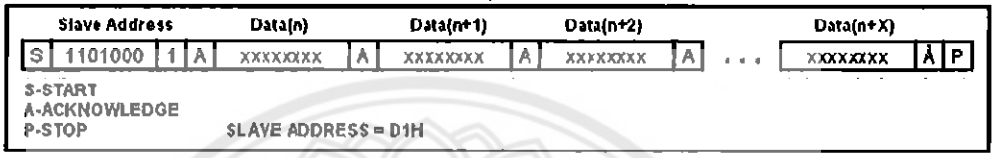
รูปที่ 2.8 การรับส่งข้อมูลผ่านบัส I2C

ในการรับส่งข้อมูลผ่านบัส I2C อุปกรณ์ Master จะเป็นผู้สร้างสัญญาณนาฬิกาบน SDA และเป็นตัวควบคุมสถานะเริ่มและหยุด เพื่อควบคุมการรับส่งข้อมูลทั้งหมดการส่งข้อมูลไปยังอุปกรณ์ DS1307 ดังแสดงในรูปที่ 2.9 ไมโครคอนโทรลเลอร์ต้องสร้างสภาวะเริ่มก่อน จากนั้นต้องส่งข้อมูลที่อยู่ของ DS1307 ขนาด 7 บิตซึ่งมีค่าเป็น 1101000 และตามด้วยบิตระบุทิศทางของข้อมูล ในกรณีที่เป็นการเขียนข้อมูลลง DS1307 จะต้องเป็น "0" จากนั้นไมโครคอนโทรเลอร์จะต้องส่งตำแหน่งที่อยู่ภายในรีจิสเตอร์ของ DS1307 ที่ต้องการเขียนข้อมูลลง แล้วจึงค่อยเขียนข้อมูลลง โดยในการส่งข้อมูลแต่ละไบต์จะต้องรอบิต Ack จาก DS1307 ทุกไบต์ เมื่อส่งจนครบแล้ว ถึงจะสร้างสภาวะหยุดเพื่อกลับสู่สถานะว่าง



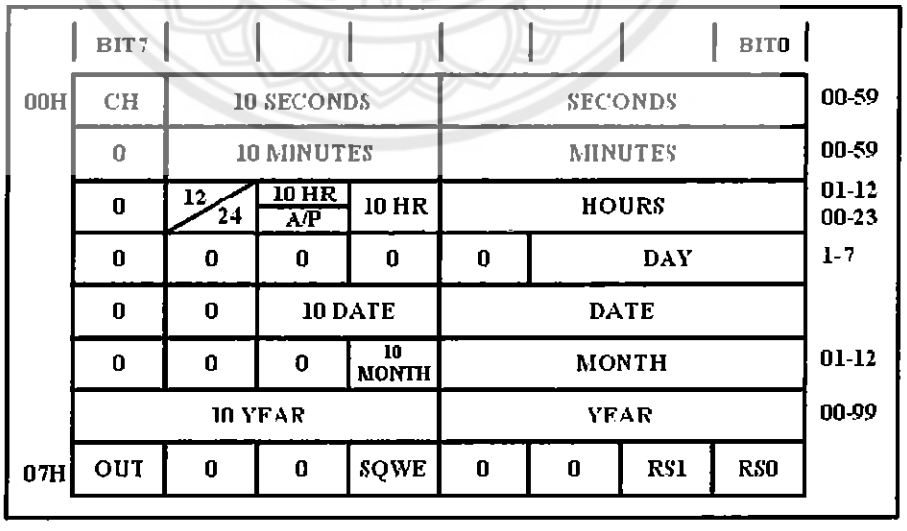
รูปที่ 2.9 การเขียนข้อมูลอุปกรณ์ Slave ผ่านบัส I2C

การรับข้อมูลจากอุปกรณ์ Slave ดังแสดงในรูปที่ 2.10 เริ่มแรกไมโครคอนโทรลเลอร์ต้องสร้างสภาวะเริ่มก่อน จากนั้นต้องส่งที่อยู่ของ DS1307 ขนาด 7 บิตซึ่งมีค่าเป็น 1101000 และตามด้วยบิตระทิศทางของข้อมูล ในกรณีที่เป็นการอ่านข้อมูลจาก DS1307 จะต้องเป็น "1" จากนั้นจึงค่อยรับข้อมูลจากอุปกรณ์รอง (Slave) ทีละไบต์ โดยตำแหน่งที่อ่านเข้ามาจะขึ้นอยู่กับตำแหน่งรีจิสเตอร์พอยท์เตอร์ ซึ่งจะเป็นตำแหน่งท้ายสุดที่ได้ทำการเขียนข้อมูลไว้ เมื่ออ่านข้อมูลครบแต่ละไบต์อุปกรณ์หลัก (Master) ต้องส่งบิตรับทราบ (Acknowledge bit) กลับไปให้อุปกรณ์รอง (Slave) ด้วย ในกรณีที่เป็นไบต์สุดท้าย อุปกรณ์หลัก (Master) ต้องส่ง "not acknowledge" กลับไป



รูปที่ 2.10 การอ่านข้อมูลจากอุปกรณ์รอง (Slave) ผ่านบัส I2C

ภายใน DS1307 มีรีจิสเตอร์ภายในใช้เก็บข้อมูลเวลาขนาด 7 ไบต์ 00H-06H ดังแสดงในรูปที่2.11 ข้อมูลค่าเวลา และวันที่จะถูกเก็บอยู่ในรูปของเลขฐาน 10 สามารถเลือกได้ว่าให้ทำงานแบบ 12 ชั่วโมง หรือ 24 ชั่วโมง โดยกำหนดที่บิตที่ 6 ที่แอดเดรส 02H โดยถ้าเป็น "1" จะเป็นการทำงานในโหมด 12 ชั่วโมง และเมื่อเลือกแบบ 12 ชั่วโมง ที่บิต 5 ในแอดเดรส 02H นั้นจะใช้แสดงค่า AM/PM โดยถ้าบิตนี้เป็น "1" จะเป็น PM ในกรณีที่แสดงแบบ 24 ชั่วโมง บิตนี้จะใช้ในการแสดงค่าของหลักสิบในของหน่วยชั่วโมงด้วย



รูปที่ 2.11 รีจิสเตอร์ภายในไอซีฐานเวลา DS1307

ที่แอดเดรส 07H เป็นรีจิสเตอร์ควบคุมการทำงานของ SQW/OUT โดยมีรายละเอียดดังนี้

ก) OUT (Out control) ใช้ควบคุมเอาท์พุต

ข) SQWE (Square Wave Enable) ใช้ควบคุมออสซิลเลเตอร์ภายใน DS1307 โดยถ้าบิตนี้เป็น "1" จะเป็นการเปิดออสซิลเลเตอร์

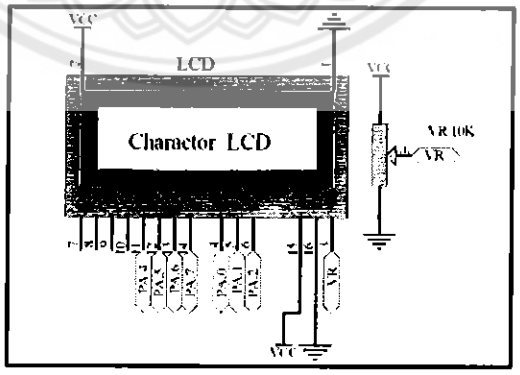
ค) RS (Rate Select): ใช้ควบคุมความถี่ของ Square Wave เมื่อเปิดการทำงานของออสซิลเลเตอร์ โดยสามารถปรับเปลี่ยนความถี่ได้ 4 ความถี่ด้วยกัน ดังแสดงในตารางที่ 2.2

ตารางที่ 2.2 การควบคุมความถี่ออสซิลเลเตอร์ด้วยการเซตบิต RS1, RS0

RS1	RS0	SQW OUTPUT FREQUENCY
0	0	1 Hz
0	1	4.096 kHz
1	0	8.192 kHz
1	1	32.768 kHz

2.5 วงจรจอแสดงผลแอลซีดี

จอแสดงผลแอลซีดีที่ใช้ในโครงงานนี้ เป็นจอแอลซีดีขนาดสิบหกตัวอักษรสองบรรทัด ทำการติดต่อแบบ 8 บิต



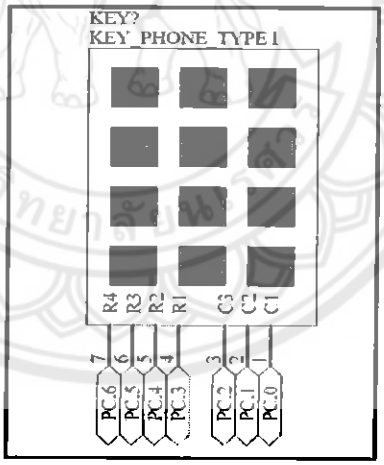
รูปที่ 2.12 การต่อใช้งานจอแสดงผลแอลซีดี

จากรูปที่ 2.12 เป็นวงจรแสดงผลผ่านจอแอลซีดี (LCD) ขา 1 ต่อเข้ากับกราวด์ (GND) ขา 2 ต่อเข้ากับไฟเลี้ยง 5 โวลต์ ขา 3 เป็นขาปรับความสว่างของจอแอลซีดี (LCD) ส่วนขา 4 5 6 11 12 13 14 เป็นขาควบคุมที่ส่งมาจากไมโครคอนโทรลเลอร์

ดูได้จากโปรแกรม Code Vision AVR เมื่อป้อนรุ่นไมโครคอนโทรลเลอร์ข้าไปแล้วกำหนดพอร์ตโปรแกรมจะทำการบอกการต่อขาแอลซีดีจากไมโครคอนโทรลเลอร์มาให้ เพื่อทำการเขียนตัวอักษรบนตัวแอลซีดีขา 15 ต่อไฟเลี้ยง 5 โวลต์ขา 16 ต่อกราวด์ (GND) มีตัวต้านทานต่อกับขาที่ 3 เนื่องจากรองรับกระแสก่อนต่อเข้าจอแอลซีดี(LCD)

2.6 วงจรแผงปุ่มกด (Keypad)

ใช้หลักการสแกน เพื่อรับค่าข้อมูลมายังไมโครคอนโทรลเลอร์ โดยกำหนดพอร์ตซี(port -c) ของไมโครคอนโทรลเลอร์ (ATmaga8515L) เป็นพอร์ตสำหรับรับข้อมูลมาประมวลผล จะเห็นว่าตรงพอร์ตที่เชื่อมต่อของคอนเน็ตเตอร์ IDC2 ไม่ต้องต่อไฟเลี้ยงและกราวด์ ที่ขาจะมีอยู่พอร์ต 7 พอร์ต เนื่องจากมีสี่ แถวกับสามหลัก โดยกำหนดให้ pc0 pc1 pc2 เป็นขาเอาท์พุต และ pc3 pc4 pc5 pc6 เป็นขาอินพุต



รูปที่ 2.13 การต่อใช้งานแผงปุ่มกด (Keypad)

2.7 ภาษาสั่งงานคอมพิวเตอร์ (Computer Languages) [4]

เนื่องจากซีพียู (CPU) จะถูกสั่งงานด้วยคำสั่งที่อยู่ในรูปของเลขฐานสองเท่านั้น แต่การสั่งงานด้วยเลขฐานสองเป็นเรื่องยากที่คนจะเข้าใจ ดังนั้นจึงจำเป็นต้องมีตัวกลางเข้ามาช่วยในการแปลงภาษาสั่งงานให้อยู่ในเลขฐานสอง ทำให้ผู้สั่งงานเขียนคำสั่งในรูปของภาษาที่ตัวเองเข้าใจแล้วตัวกลางจะแปลงให้เป็นคำสั่งในรูปของเลขฐานสองให้คอมพิวเตอร์เข้าใจต่อไป การแปลภาษาของคอมพิวเตอร์แบ่งได้เป็น 3 แบบคือ

1. แอสแซมเบลอ (Assembler) เป็นตัวแปลภาษาแอสแซมบลี ซึ่งเป็นภาษาระดับต่ำให้เป็นภาษาเครื่อง
2. อินเทอพรีเตอร์ (Interpreter) เป็นตัวแปลภาษาระดับสูง เป็นการแปลภาษาระดับสูงให้เป็นภาษาเครื่อง โดยใช้หลักการแปลทีละประโยคหรือคำสั่งหากเจอที่ผิดในตัวโปรแกรม ตัวแปลภาษาจะยุติการแปลภาษาและจะแสดงตำแหน่งข้อผิดพลาดเพื่อให้ผู้เขียนโปรแกรมแก้ไขโปรแกรมให้เรียบร้อย เมื่อผู้เขียนโปรแกรมแก้ไขเรียบร้อยเสร็จแล้ว ผู้เขียนโปรแกรมจะต้องนำโปรแกรมมาแปลภาษาอีกครั้งจนกว่าจะไม่พบที่ผิด
3. คอมไพเลอร์ (Compiler) เป็นตัวแปลภาษาระดับสูง เป็นการแปลภาษาระดับสูงให้เป็นภาษาเครื่องโดยทำการแปลทั้งโปรแกรม และแสดงรายละเอียดข้อผิดพลาดที่เกิดขึ้นทั้งหมด เพื่อให้ผู้เขียนโปรแกรมดำเนินการแก้ไขโปรแกรมให้เรียบร้อย เมื่อผู้เขียนโปรแกรมแก้ไขเสร็จแล้ว ผู้เขียนโปรแกรมจะต้องนำโปรแกรมมาแปลภาษาอีกครั้งจนกว่าจะไม่พบที่ผิด

2.7.1 โปรแกรมภาษาซี (C Programming) [4]

ภาษาที่ได้รับความนิยมสูงสุดตั้งแต่อดีตจนถึงปัจจุบันคือภาษาซี ซึ่งถูกพัฒนาโดย Ken Thompson และ Dennis Ritchie ในปี 1972 โดยทั้งสองได้รับแนวคิดจากภาษา BCPL ซึ่งมีชื่อเรียกเล่น ๆ ว่าภาษาบี (B) ดังนั้นภาษาที่ต่อจากภาษาบีก็คือภาษาซี นอกจากนั้นภาษาซียังมีความเกี่ยวข้องกับระบบปฏิบัติการยูนิกซ์ (Unix) ตั้งแต่เริ่มต้น ดังนั้นความนิยมในตัวระบบปฏิบัติการ ยูนิกซ์ (Unix) จึงส่งผลกับความนิยมในตัวภาษาซีด้วยเช่นกัน จุดแข็งสำคัญของภาษาซีคือ โครงสร้างที่เหมาะสมกับโปรแกรมขนาดใหญ่ โปรแกรมเล็ก ๆ ที่เขียนจากภาษาซีสามารถนำมารวมกันให้เป็นโปรแกรมขนาดใหญ่ได้ และนอกจากนี้ภาษาซียังสามารถเข้าถึงอุปกรณ์ต่าง (Hardware) ได้ไม่ต่างจากภาษาแอสแซมบลี (Assembly) ทำให้ภาษาซีถูกจัดให้เป็นภาษาที่มีความสามารถไร้ขอบเขต

รูปที่ 2.14 Ken Thompson และ Dennis Ritchie ผู้คิดค้นภาษาซี

ก่อนที่จะลงมือพัฒนาโปรแกรมคอมพิวเตอร์ ขั้นแรก เราต้องศึกษารูปแบบความต้องการของโปรแกรมที่จะพัฒนา จากนั้นก็วิเคราะห์ถึงปัญหาตลอดจนวิธีแก้ปัญหา จากนั้นจึงนำเอาความคิดในการแก้ปัญหาอย่างเป็นขั้นตอน ไปเขียนในรูปแบบโปรแกรมภาษาในระดับสูง ซึ่งจะ

เป็นรูปแบบของโค๊ดโปรแกรม (Source code) จากนั้นเราจะใช้คอมไพเลอร์ (Compiler) มาทำการแปลงโค๊ดโปรแกรม (Source code) ของเราให้เป็นภาษาเครื่องนั้นเอง ซึ่งในขั้นตอนนี้ผลที่ได้เราจะเรียกว่า Object code จากนั้นคอมไพเลอร์ (compiler) ก็จะทำการเชื่อม (Link) Object code เข้ากับฟังก์ชั่นการทำงานในหน่วยเก็บข้อมูล (Libraries) ต่าง ๆ ที่จำเป็นต่อการใช้งานแล้วนำไปไว้ในหน่วยความจำแล้วเราก็สามารถประมวลผลโปรแกรม (Run) เพื่อดูผลการทำงานของโปรแกรมได้ หากโปรแกรมมีข้อผิดพลาด เราก็จะทำการแก้ไข หรือที่เรียกกันในภาษาคอมพิวเตอร์ว่า การดีบั๊ก (Debug) นั่นเอง

2.7.2 ข้อดีของภาษาซี

1. ภาษาซีใช้ได้ในไมโครคอมพิวเตอร์ ตั้งแต่ขนาด 8 บิต 16 บิต 32 บิต มินิคอมพิวเตอร์ หรือคอมพิวเตอร์ระดับเมนเฟรม มีการพัฒนาการใช้งานเพื่อให้เป็นมาตรฐาน ไม่ขึ้นอยู่กับโปรแกรมจัดระบบงาน หรืออุปกรณ์อิเล็กทรอนิกส์ (ฮาร์ดแวร์)

2. ภาษาซีมีผู้ผลิตต่างบริษัทแต่มีโครงสร้างคล้ายกัน และสามารถใช้งานร่วมกันได้

3. ภาษาซีมีความอ่อนตัว สามารถเจาะลงระดับลึกให้เข้ากับฮาร์ดแวร์ ทำงานได้รวดเร็วและที่สำคัญ ภาษาซีเป็นคอมไพเลอร์

4. ภาษาซีเป็นภาษาที่มีโครงสร้าง

2.7.3 โครงสร้างภาษาซี

```
# header
main()
{- กำหนดตัวแปร
 - กำหนดค่าตัวแปร
 - ฟังก์ชัน ในรูปสเตตเม้นท์
 - การควบคุม
 - คอมเมนท์
}
function a()
{
...(มีส่วนประกอบเช่นเดียวกับ ฟังก์ชัน main)
}
function b()
```

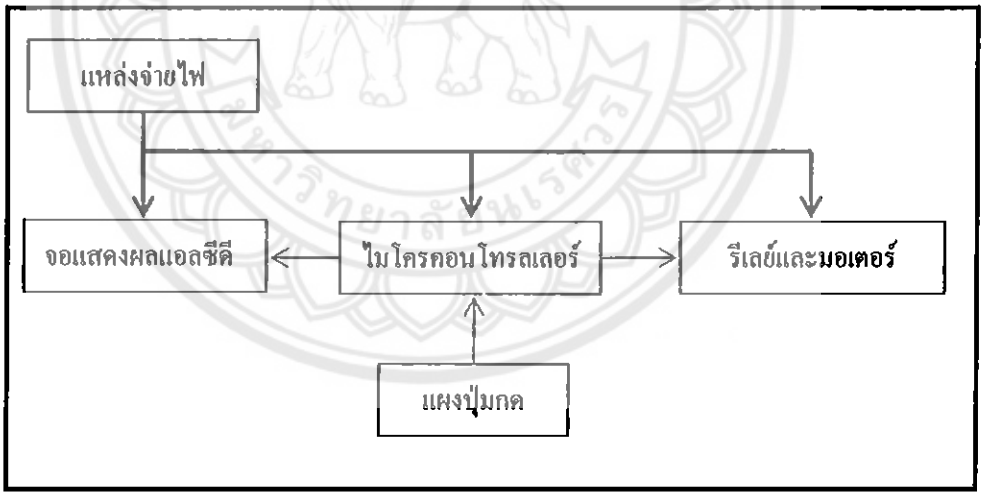
หลักการเบื้องต้นในการเขียนดังแสดงด้านบน ส่วนฟังก์ชันการกำหนดพารามิเตอร์ต่าง ๆ จะมีรายละเอียดที่มาก ท่านผู้อ่านสามารถศึกษาได้จากหนังสือ การเขียนโปรแกรมภาษาซีได้ต่อไป

บทที่ 3
การสร้างเครื่องผสมปุ๋ยน้ำอัตโนมัติ

ในการสร้างเครื่องผสมปุ๋ยน้ำอัตโนมัติ ได้มีการศึกษาข้อมูลและทำการออกแบบจนได้เป็นกล่องควบคุมที่สามารถตั้งเวลาการทำงานของมอเตอร์ได้ โดยมีโครงสร้างไม่ซับซ้อนและง่ายต่อการใช้งาน

3.1 ขั้นตอนการสร้างเครื่องผสมปุ๋ยน้ำอัตโนมัติ

ในการสร้างกล่องควบคุมการทำงานของเครื่องผสมปุ๋ยอัตโนมัติมีขั้นตอนการทำงาน โดยเริ่มจากการออกแบบวงจรที่เป็นส่วนประกอบต่าง ๆ ของเครื่องผสมปุ๋ยน้ำอัตโนมัติอันได้แก่ วงจรแหล่งจ่ายไฟ (Power Supply) โดยการใช้วงจรแปลงไฟเพื่อจ่ายไฟให้กับแผงไมโครคอนโทรลเลอร์ จอแสดงผลแอลซีดี (LCD) และมอเตอร์ปั๊มน้ำ ดังแสดงในรูปที่ 3.1 ส่วนแผงปุ่มกดเลข (Keypad) ไม่ต้องใช้ไฟเลี้ยงหลังจากที่ได้วงจรทั้งหมดแล้วทำการทดสอบการทำงานของเครื่องผสมปุ๋ย



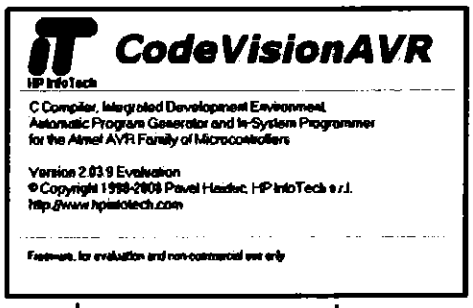
รูปที่ 3.1 แผนผังขั้นตอนการทำงานของกล่องควบคุม

3.2 ส่วนซอฟแวร์ของเครื่องผสมปุ๋ยอัตโนมัติ

ส่วนประกอบที่เป็นซอฟต์แวร์ (Software) ของเครื่องผสมปุ๋ยน้ำอัตโนมัติ ได้แก่

3.2.1 โปรแกรมโค๊ดวิชั่นเอวีอาร์

โปรแกรมโค๊ดวิชั่นเอวีอาร์ (Code Vision AVR) ดังรูปที่ 3.2 ที่ใช้เขียนโปรแกรมภาษาซีและแปลงรหัสคำสั่งเลขฐานสิบหก



รูปที่ 3.2 โปรมแกรมโค๊ดวิชั่นเอวีอาร์

โปรแกรมโค๊ดวิชั่นเอวีอาร์ (Code Vision AVR) เป็นคอมไพล์เลอร์ที่รวมเอาคุณสมบัติการพัฒนาโปรแกรมและยังสามารถโปรแกรมได้โดยอัตโนมัติหลังจากทำการคอมไพล์โปรแกรมผ่านแล้ว โดยสนับสนุนไมโครคอนโทรลเลอร์ ที่อยู่ในตระกูล AVR Microcontroller โปรแกรมได้ออกแบบให้ใช้งานได้บนระบบปฏิบัติการที่เป็น Windows 95, 98, ME, NT, 2000 และ Windows XP

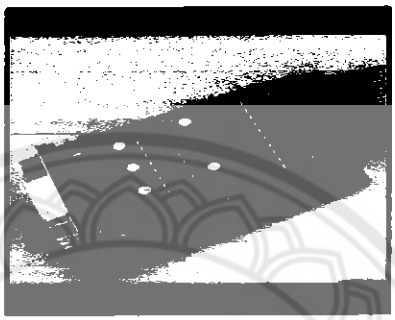
ซีคอมไพล์เลอร์มีวิธีการเขียนที่ใกล้เคียงกับมาตรฐานที่เป็น ANSI C แต่ก็ต้องเป็นไปตามโครงสร้างคำสั่งที่อ้างอิงถึงไมโครคอนโทรลเลอร์ตระกูล AVR ซึ่งบางลักษณะสามารถเพิ่มเติมได้โดยตามโครงสร้างสถาปัตยกรรมของ ไมโครคอนโทรลเลอร์ตระกูล AVR และความต้องการทางด้านงานระบบฝังตัวการคอมไพล์ที่ได้เป็น COFF จาก Source file ที่เป็น C จะได้ไฟล์ที่มีนามสกุลเป็น COFF ที่ใช้ในระดับการตรวจสอบความผิดพลาดของโปรแกรมได้โดยสามารถทำการดูค่าตัวแปรที่เก็บในรีจิสเตอร์ได้โดยใช้ซอฟท์แวร์ของ Atmel AVR Studio

การพัฒนาโปรแกรมตามสภาวะแวดล้อมนั้น สามารถที่จะพัฒนาโปรแกรมแล้วดูผลของโปรแกรมโดยสามารถถ่ายโอนตัวเฮ็กซ์ไฟล์ที่ผ่านการคอมไพล์แล้วลงบนตัวชิพได้โดยอัตโนมัติหลังจากการลงรหัสโปรแกรมแล้ว ได้ทำการคอมไพล์โปรแกรมผ่านแล้วก็จะสามารถสั่งโปรแกรมชิพได้การโปรแกรมนั้นจะต้องเลือกแบบในการพัฒนาโปรแกรม คือ จะต้องเลือกชนิดของบอร์ดในการพัฒนาว่าจะต่ออยู่กับพอร์ตใดของคอมพิวเตอร์ แต่สำหรับการดีบักโปรแกรมนั้นก็สามารถกระทำได้โดยผ่านทางซีเรียลพอร์ต (Serial port) โดยการพัฒนาโปรแกรมจะมีซอฟต์แวร์ (Software) ในการทำทอมินอลกับเอ็มซียู (MCU)

3.2.2 โปรแกรมเอวีอาร์สตูดิโอ 4

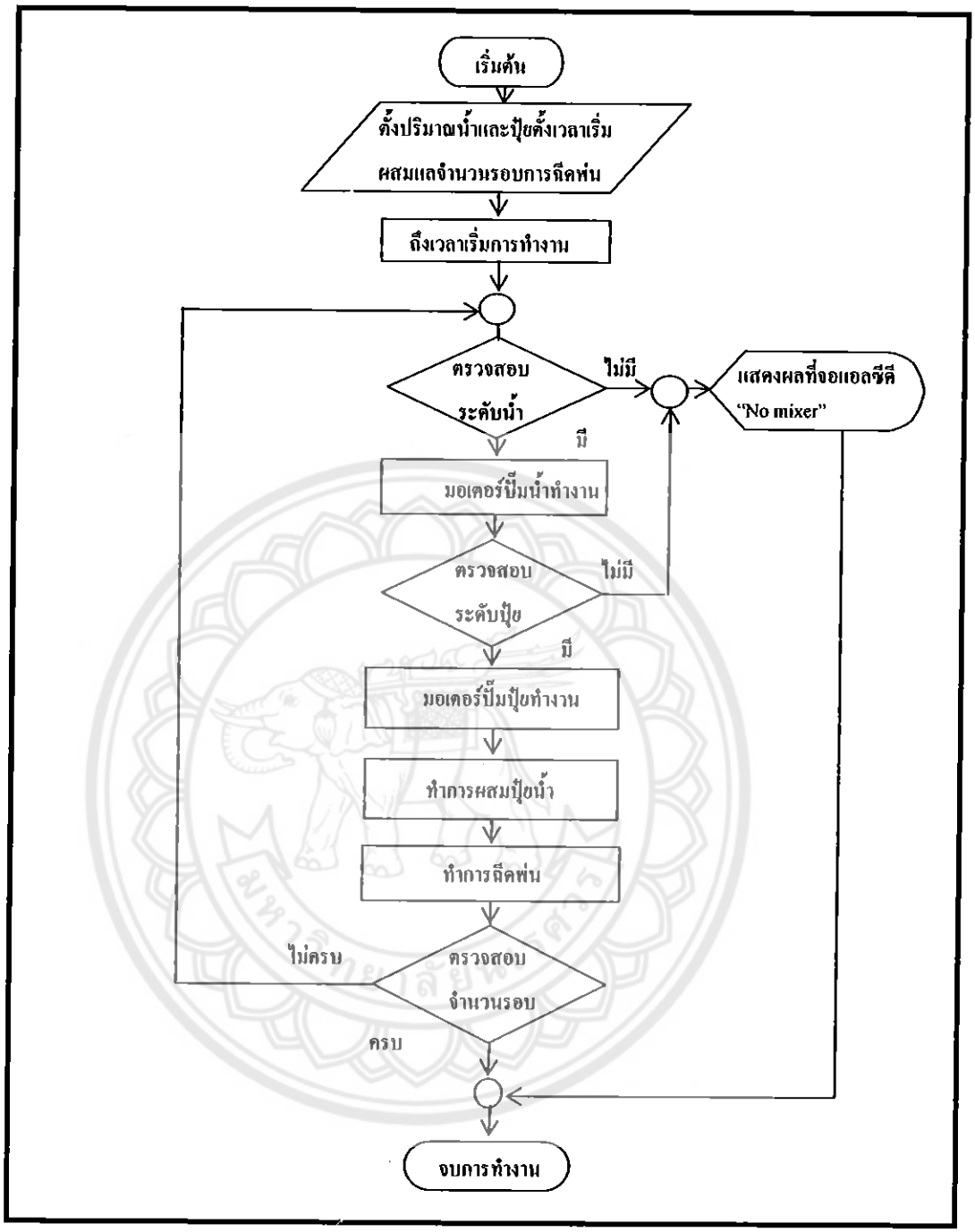
โปรแกรมเอวีอาร์สตูดิโอ 4 (AVR Studio 4) ดังรูปที่ 3.3 เป็นโปรแกรมที่ใช้สำหรับการโปรแกรมรหัสควบคุมลงในไมโครคอนโทรลเลอร์โดยผ่าน px-400 เป็นเครื่องโปรแกรมไมโครคอนโทรลเลอร์ AVR แบบ ISP ดังรูปที่ 3.4

รูปที่ 3.3 โปรแกรมเอวีอาร์สตูดิโอ 4



รูปที่ 3.4 เครื่องโปรแกรมไมโครคอนโทรลเลอร์ PX-400

กระบวนการทำงานของเครื่องผสมปุ๋ยน้ำอัตโนมัติมีการทำงานตามเงื่อนไขดังต่อไปนี้สรุปได้ดังรูปที่ 3.5 เป็นการเริ่มการทำงานโดยการกดเครื่องหมายดอกจันทร์บนแผงปุ่มกดเพื่อตั้งค่าปริมาณน้ำ กดเครื่องหมายสี่เหลี่ยมเมื่อตั้งค่าเสร็จ จากนั้นที่หน้าจอแสดงผลจะปรากฏข้อความให้ตั้งค่าปริมาณปุ๋ย เวลาที่ต้องการให้เครื่องผสมปุ๋ยน้ำอัตโนมัติเริ่มทำงานและจำนวนรอบการทำงาน เมื่อถึงเวลาที่ตั้งค่าเอาไว้เครื่องผสมปุ๋ยน้ำอัตโนมัติจะเริ่มทำงานโดยการตรวจสอบปริมาณน้ำว่าพอที่จะทำการผสมหรือไม่ถ้าพอกล่องควบคุมจะสั่งให้มอเตอร์ปั๊มน้ำทำงานโดยการดูดน้ำเข้ามาในถังผสม จากนั้นทำการตรวจสอบปริมาณปุ๋ยว่าพอที่จะทำการผสมหรือไม่ถ้าพอกล่องควบคุมจะสั่งให้มอเตอร์ปั๊มปุ๋ยทำงานโดยการดูดปุ๋ยเข้ามาในถังผสม จากนั้นมอเตอร์ใบพัดทำการงานเพื่อผสมน้ำและปุ๋ย มอเตอร์ปั๊มที่อยู่ในถังผสมดูดปุ๋ยน้ำที่ทำการผสมเรียบร้อยแล้วออกไปทำการฉีดพ่น



รูปที่ 3.5 กระบวนการทำงานของกล่องควบคุม

3.3 ส่วนฮาร์ดแวร์ของเครื่องผสมปุ๋ยอัตโนมัติ

ส่วนประกอบที่เป็นฮาร์ดแวร์ (Hardware) ของเครื่องผสมปุ๋ยอัตโนมัติมีดังนี้

ก) ชุดแหล่งจ่ายพลังงานไฟฟ้า

ข) ส่วนควบคุมการทำงานของเครื่องผสมปุ๋ยอัตโนมัติ

ค) ส่วนของการทำงานของเครื่องผสมปุ๋ย

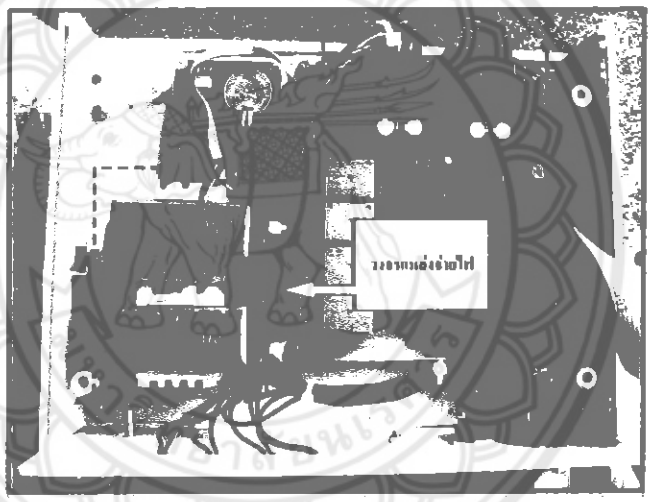
ง) ส่วนของวงจรตั้งเวลาการทำงานของเครื่องผสมปุ๋ย

จ) ส่วนของวงจรจอแสดงผลแอลซีดี

ฉ) ส่วนของวงจรแผงปุ่มกด

3.3.1 ชุดแหล่งจ่ายพลังงานไฟฟ้า

ภายในกล่องควบคุมมีวงจรสองวงจรด้วยกันวงจรแรกสำหรับไมโครคอนโทรลเลอร์และอีกวงจรเป็นไฟฟ้ากระแสตรงขนาด 12 โวลต์ สำหรับมอเตอร์ใบพัด มีลักษณะการต่อภายในกล่องควบคุมดังรูปที่ 3.6 เนื่องจากแผงวงจรไมโครคอนโทรลเลอร์และจอแอลซีดีต้องการแรงดันไฟฟ้ากระแสตรง 5 โวลต์ ดังนั้นเราจึงต้องทำวงจรแปลงแรงดันไฟฟ้าเพื่อให้ได้แรงดันที่เราต้องการ



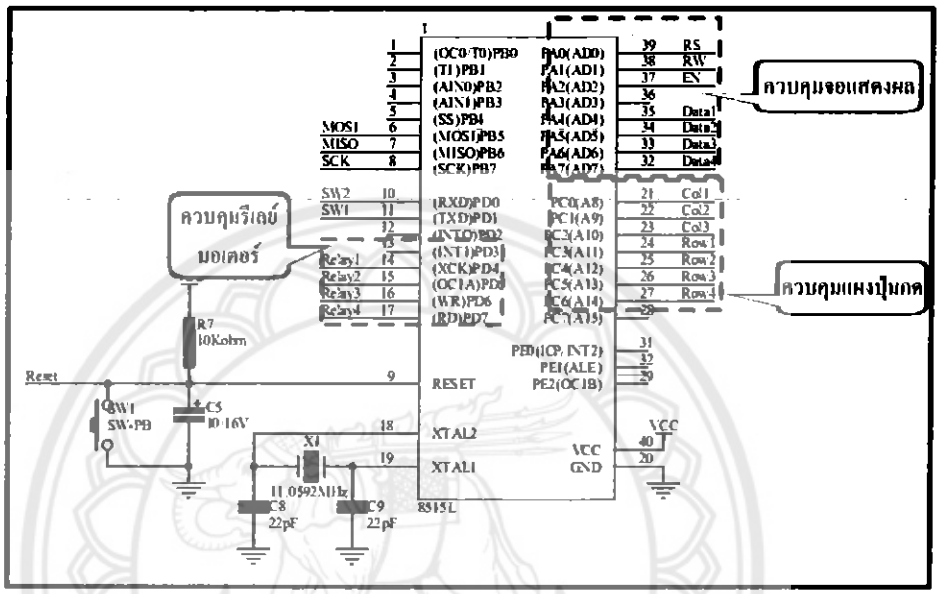
รูปที่ 3.6 วงจรแหล่งจ่ายพลังงานไฟฟ้า

หลักการทำงานของวงจร เมื่อแรงดันไฟฟ้ากระแสสลับ 220 โวลต์ ผ่านหม้อแปลงเหลือ 12 โวลต์ จากนั้นเข้ามาสู่บริดจ์ (Bridge) แปลงไฟออกจาก 12 โวลต์ให้เป็นไฟกระแสตรง 12โวลต์ เพื่อทำการจ่ายไฟให้กลับรีเลย์มอเตอร์ดีซี (Motor DC) โดยมีตัวเก็บประจุขนาด 2200/25 V ฟารัด ทำหน้าที่กรองกระแสและปรับระดับแรงดันให้มากขึ้น

และอีกวงจรหนึ่งเป็นการแรงดันไฟฟ้ากระแสสลับ 220 โวลต์ ผ่านหม้อแปลงเหลือ 9 โวลต์ จากนั้นเข้ามาสู่บริดจ์ (Bridge) แปลงไฟออกจาก 9โวลต์ให้เป็นไฟกระแสตรง 9โวลต์ จากนั้นทำการลดระดับแรงดันให้เหลือ 5โวลต์โดยใช้ไอซีหมายเลข *LM2575* เพื่อเป็นไฟเลี้ยงเข้าสู่บอร์ดไมโครคอนโทรลเลอร์ จอแอลซีดี และรีเลย์

3.3.2 ส่วนควบคุมการทำงานของเครื่องผสมปุ๋ยอัตโนมัติ

ในส่วนควบคุมการทำงานทั้งควบคุมรีเลย์มอเตอร์ จอแสดงผลและแผงปุ่มกดคือไมโครคอนโทรลเลอร์รุ่น ATmega8515L ดังรูปที่ 2.4 โดยพอร์ตที่ถูกใช้งานคือ พอร์ต 0 พอร์ต 2 และพอร์ต 3 ซึ่งทำหน้าที่เป็นพอร์ตอินพุตเอาท์พุตดังแสดงในรูปที่ 3.7



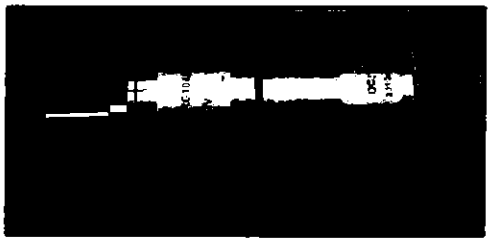
รูปที่ 3.7 การต่อใช้งานการควบคุมของไมโครคอนโทรเลอร์รุ่น ATmega8515L

3.3.3 ส่วนการทำงานของเครื่องผสมปุ๋ยอัตโนมัติ

ในส่วนการทำงานของเครื่องผสมปุ๋ย ใช้มอเตอร์ปั๊มน้ำขนาด 1000 ลิตร/ชั่วโมง กำลังไฟ 15 วัตต์ จำนวน 3 ตัวเป็นตัวดูดน้ำและปุ๋ยเข้าถังผสมรวมทั้งดูดปุ๋ยที่ผสมเรียบร้อยแล้วออกมาฉีดพ่น ดังรูปที่ 3.8

รูปที่ 3.8 ปั๊มน้ำ HaiYang กำลังไฟ 15 วัตต์

ในส่วนของการผสมใช้มอเตอร์ไฟฟ้ากระแสตรงความเร็วรอบ 150:1 นาที จำนวน 1 ตัว ในการควบคุมการทำงาน ดังรูปที่ 3.9



รูปที่ 3.9 มอเตอร์ไฟฟ้ากระแสตรงแบบทดรอบขนาด 12 V รุ่น ZAG37

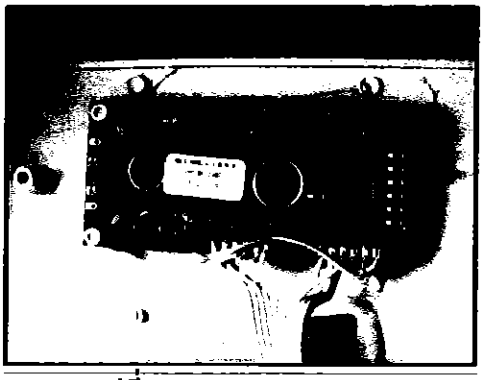
3.3.4 ส่วนของวงจรตั้งเวลาการทำงานของเครื่องผสมปุ๋ยอัตโนมัติ

วงจรการตั้งเวลาเป็นการทำงานร่วมกับ Real Time Clock (RTC) ดังรูปที่ 3.10

รูปที่ 3.10 วงจรการตั้งเวลาการทำงาน

3.3.5 ส่วนของวงจรจอแสดงผลแอลซีดี

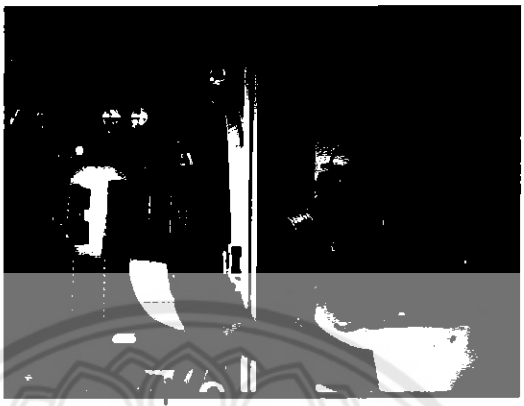
วงจรจอแสดงผลแอลซีดี เป็นส่วนที่มีไว้สำหรับแสดงผลการทำงานของมอเตอร์และการตั้งเวลาดังรูปที่ 3.11



รูปที่ 3.11 วงจรจอแสดงผล

3.3.6 ส่วนของวงจรแผงปุ่มกด

วงจรแผงปุ่มกด (Keypad) เป็นส่วนมีไว้สำหรับกดตั้งค่าปริมาณน้ำ ปริมาณปุ๋ย ตั้งเวลาเริ่มการทำงาน และจำนวนรอบการทำงานดังรูปที่ 3.12

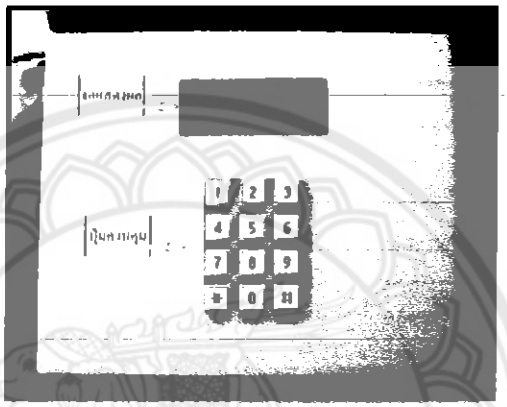


รูปที่ 3.12 วงจรแผงปุ่มกด

บทที่ 4

ผลการทดสอบ

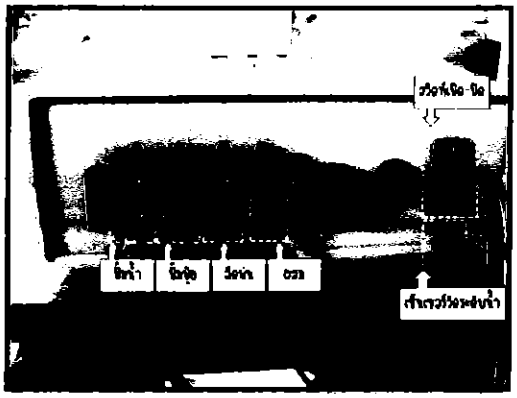
ลักษณะของกล่องควบคุมด้านบนจะมีจอแสดงผลแอลซีดีและปุ่มสั่งการดังรูปที่ 4.1 จะมีปุ่มสวิตช์ปิดเปิด และเซ็นเซอร์วัดระดับน้ำ



รูปที่ 4.1 ลักษณะด้านบนของกล่องควบคุม

4.1 ขั้นตอนการใช้งานของเครื่องผสมปุ๋ยน้ำอัตโนมัติ

การประกอบสายมอเตอร์ปั๊มเข้ากับกล่องควบคุมโดยสองช่องแรกต่อเข้ากับปั๊มน้ำ ช่องสามและสี่ต่อเข้ากับปั๊มปุ๋ย ช่องที่ห้าและหกต่อเข้ากับปั๊มที่ออกไปฉีดพ่น และสองช่องสุดท้ายต่อเข้ากับมอเตอร์ที่ทำหน้าที่ผสมดังรูปที่ 4.2



รูปที่ 4.2 การประกอบสายกับกล่องควบคุมพร้อมสวิตช์เปิดปิด

4.1.1 การบอกเวลาของกล่องควบคุม

เมื่อมีการเสียบปลั๊กกล่องควบคุมจะแสดงเวลาที่เป็นปัจจุบันที่ทำงานโดย RTC ถึงแม้ว่าจะมีการถอดปลั๊กแต่เวลาก็ยังเดินอยู่ เมื่อมีการเสียบปลั๊กครั้งใหม่เวลายังคงต่อเนื่องอยู่ ไม่ต้องเสียเวลาในการตั้งเวลาใหม่ทำให้การตั้งเวลาผสมและฉีดพ่นทำได้รวดเร็วยิ่งขึ้น ดังรูปที่ 4.3

รูปที่ 4.3 เวลาปัจจุบันบนหน้าจอแอลซีดี

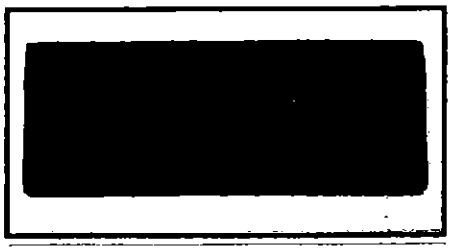
4.1.2 การตั้งค่าปริมาณน้ำ

การตั้งปริมาณเริ่มการทำงานโดยการกดเครื่องหมายคอกจันทร์ จากนั้นหน้าจอแสดงผลจะปรากฏข้อความว่า "Set water" เป็นการใส่ค่าเพื่อกำหนดปริมาณน้ำที่ต้องการเป็นจำนวนลิตร สามารถใส่เป็นค่าทศนิยมได้ ดังรูปที่ 4.4

รูปที่ 4.4 จอแสดงผลแสดงข้อความให้ตั้งค่าปริมาณน้ำ

4.1.3 การตั้งค่าปริมาณปุ๋ย

เมื่อทำการป้อนตัวเลขเพื่อตั้งค่าปริมาณน้ำเรียบร้อยแล้ว แล้วกดตกลงโดยกดเครื่องหมายสี่เหลี่ยมหน้าจอแสดงผลจะปรากฏข้อความ "Set fertilizer" เพื่อเป็นการใส่ค่าปริมาณปุ๋ยที่ต้องการเป็นจำนวนลิตร โดยสามารถกำหนดให้เป็นทศนิยมได้ ดังรูปที่ 4.5



รูปที่ 4.5 จอแสดงผลแสดงข้อความให้ตั้งค่าปริมาณปุ๋ย

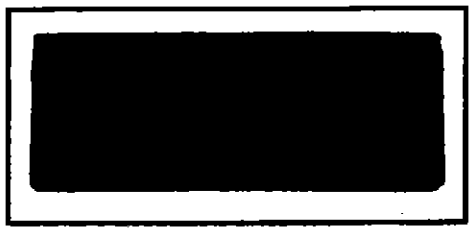
4.1.4 การตั้งเวลาเริ่มการผสมปุ๋ยน้ำ

เมื่อทำการป้อนตัวเลขเพื่อตั้งค่าปริมาณปุ๋ยแล้ว กดตกลง โดยกดเครื่องหมายสี่เหลี่ยมหน้าจอจะแสดงข้อความ "Time start" เป็นการตั้งค่าเวลาที่จะให้เครื่องเริ่มทำงานที่เวลาเมื่อใด ดังรูปที่ 4.6

รูปที่ 4.6 จอแสดงผลแสดงข้อความให้ตั้งเวลาเริ่มการทำงาน

4.1.5 การตั้งจำนวนรอบการทำงาน

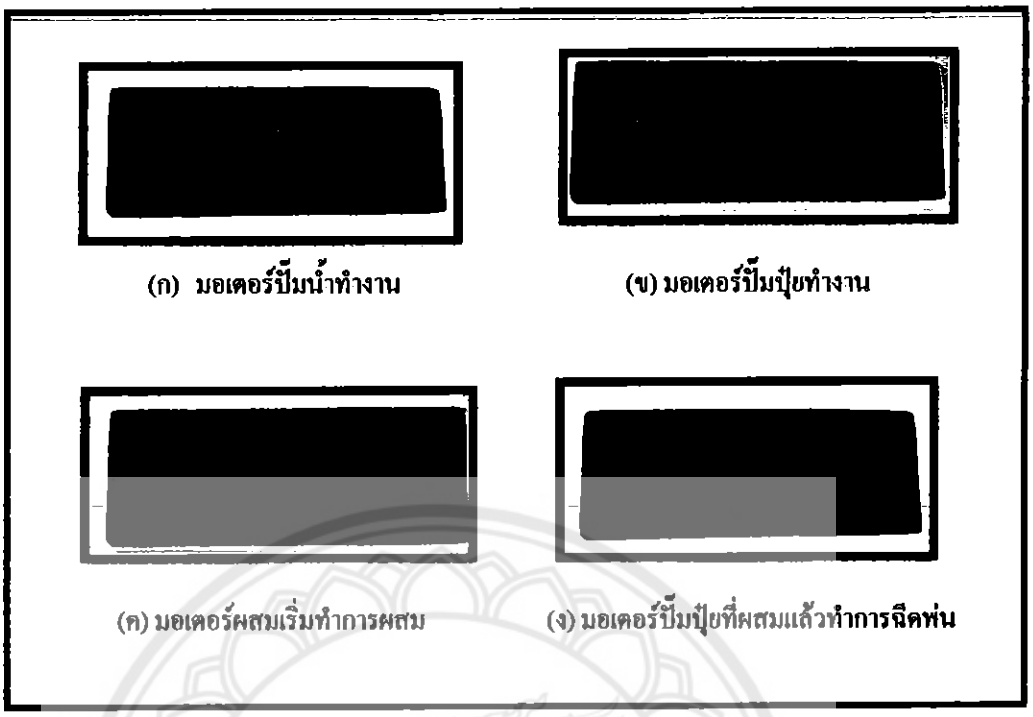
เมื่อทำการป้อนตัวเลขเพื่อตั้งเวลาเริ่มการทำงานของเครื่องผสมปุ๋ยอัตโนมัติแล้ว กดตกลง โดยกดเครื่องหมายสี่เหลี่ยมหน้าจอแสดงผลจะปรากฏข้อความ "Play off" เป็นการใส่จำนวนรอบการทำงานดังรูปที่ 4.7 เรียบร้อยแล้วกดเครื่องหมายสี่เหลี่ยมเป็นการตกลง



รูปที่ 4.7 จอแสดงผลแสดงข้อความให้ตั้งจำนวนรอบการทำงาน

4.1.6 การเริ่มต้นการทำงานของเครื่องผสมปุ๋ยอัตโนมัติ

เมื่อถึงเวลาที่เราตั้งค่าเริ่มการทำงานไว้ มอเตอร์ปั๊มต่าง ๆ จะเริ่มทำงานตามลำดับและหน้าจอจอแสดงผลจะแสดงข้อความเพื่อบอกว่าปั๊มใดและปริมาณเท่าใดทำงานอยู่ ดังรูปที่ 4.8



รูปที่ 4.8 จอแสดงผลแสดงข้อความตามสถานการณ์ทำงานของมอเตอร์

4.2 ผลการทดสอบเครื่องผสมปุ๋ยอัตโนมัติ

จากการทดสอบเครื่องผสมปุ๋ยอัตโนมัติมีการทำงานตรงตามที่ได้กำหนดเอาไว้

4.2.1 ตรวจสอบระดับน้ำและปุ๋ย

เมื่อถึงเวลาการทำงานตามที่กำหนดไว้ เครื่องผสมปุ๋ยอัตโนมัติทำการตรวจสอบระดับน้ำและปุ๋ยโดยเซ็นเซอร์ที่อยู่ในถังน้ำและปุ๋ยว่าเพียงพอที่จะทำการผสมหรือไม่ ถ้าปริมาณน้ำหรือปุ๋ยมีไม่เพียงพอการการผสมจะแสดง "No Mixer" ที่จอแสดงผลและเครื่องผสมปุ๋ยจะหยุดทำงานอัตโนมัติดังรูปที่ 4.9

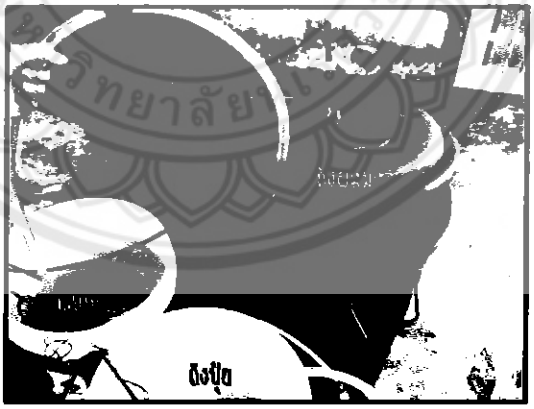
รูปที่ 4.9 จอแสดงผลแสดงข้อความเมื่อปริมาณน้ำหรือปุ๋ยไม่เพียงพอ

4.2.2 การผสมน้ำและปุ๋ย

การผสมน้ำและปุ๋ยโดยสายที่ติดเทปกาวสีฟ้าเป็นสายน้ำและสายที่ติดเทปกาวสีแดงเป็นสายปุ๋ยดังรูปที่ 4.10

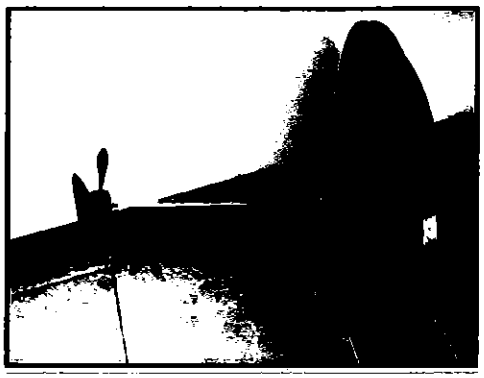
รูปที่ 4.10 สายน้ำและปุ๋ยที่ต่อเข้าถังผสม

หลังจากที่กล่องควบคุมทำการตรวจสอบระดับน้ำและปุ๋ยเรียบร้อยแล้ว เมื่อมีน้ำเพียงพอที่จะทำการผสม มอเตอร์ปั๊ม ทำงานโดยการดูดน้ำหรือปุ๋ยเข้ามาในถังผสมดังรูปที่ 4.11

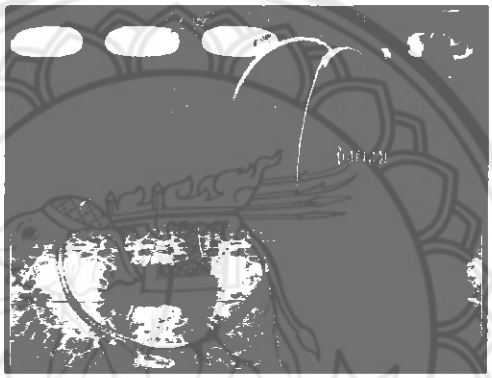


รูปที่ 4.11 มอเตอร์ปั๊มน้ำดูดน้ำและปุ๋ยเข้ามาในถังผสม

ในส่วนของถังผสม บนฝาถังผสมมีมอเตอร์ไฟฟ้ากระแสตรง (DC Motor) ติดใบพัดมีหน้าที่ปั่นผสมน้ำและปุ๋ยให้เข้ากันดังรูปที่ 4.12 ภายในถังยังมีมอเตอร์ปั๊มที่ทำหน้าดูดปุ๋ยที่ผสมเรียบร้อยแล้วนำออกไปฉีดพ่นดังรูปที่ 4.13 โดยลักษณะการฉีดพ่นจะขึ้นอยู่กับหัวที่ใช้ ในโครงงานนี้ใช้หัวแบบสปริงเกอร์รดน้ำดังรูปที่ 4.14



รูปที่ 4.12 มอเตอร์กระแสตรงมีใบพัดที่ติดอยู่กับฝาถังผสม



รูปที่ 4.13 การต่อตัวฉีดพ่นกับถังผสม

รูปที่ 4.14 การฉีดพ่นหลังการผสม

4.2.3 ระยะเวลาที่ใช้ในการทำงาน

เครื่องผสมปุ๋ยน้ำอัตโนมัติในโครงงานนี้ ได้ทำการทดสอบการทำงานของกล่องควบคุม เพื่อแสดงประสิทธิภาพของการทำงานจริงเปรียบเทียบกับการคำนวณจากค่าพิกัดของมอเตอร์ และความคลาดเคลื่อนโดยจะแสดงเป็นตารางออกมาเพื่อเปรียบเทียบความแตกต่าง

ตารางที่ 4.1 ผลการคำนวณระยะเวลาการผสมปุ๋ยน้ำ 1 รอบ

อัตราส่วน		ระยะเวลาการทำงาน (วินาที)				เวลาที่ใช้ทั้งหมด (วินาที)
น้ำ (ลิตร)	ปุ๋ย (ลิตร)	ปั๊มน้ำ	ปั๊มปุ๋ย	ผสม	ฉีดพ่น	
3	1	25.5	8.5	34	34	102
5	2	42.5	17	59.5	59.5	178.5
7	3	59.5	25.5	85	85	232
10	5	85	42.5	127.5	127.5	382.5

จากตารางที่ 4.1 แสดงการคำนวณระยะเวลาการทำงานของมอเตอร์ปั๊มน้ำ ปั๊มปุ๋ย มอเตอร์ผสมและปั๊มฉีดพ่นใน 1 รอบการทำงาน โดยใช้อัตราส่วนของน้ำและปุ๋ยที่ต่างกัน 4 ครั้งได้แก่ 3:1, 5:2, 7:3 และ 10:5

ตารางที่ 4.2 ผลการทดสอบการผสมปุ๋ยน้ำจริง 1 รอบ

อัตราส่วน		ระยะเวลาการทำงาน (วินาที)				เวลาที่ใช้ทั้งหมด (วินาที)
น้ำ (ลิตร)	ปุ๋ย (ลิตร)	ปั๊มน้ำ	ปั๊มปุ๋ย	ผสม	ฉีดพ่น	
3	1	25.82	8.74	34.49	35.94	104.99
5	2	42.97	17.06	60.37	60.2	180.12
7	3	60.17	26.00	85.36	86.09	257.62
10	5	85.8	44.21	127.42	129.96	387.39

จากตารางที่ 4.2 แสดงระยะเวลาการทำงานของมอเตอร์ปั๊มน้ำ ปั๊มปุ๋ย มอเตอร์ผสมและปั๊มฉีดพ่นใน 1 รอบการทำงานจริง โดยใช้อัตราส่วนของน้ำและปุ๋ยที่ต่างกัน 4 ครั้งได้แก่ 3:1, 5:2, 7:3 และ 10:5

ตารางที่ 4.3 ผลการคำนวณระยะเวลาการผสมปุ๋ยน้ำ 2 รอบ

อัตราส่วน		ระยะเวลาการทำงาน (วินาที)								เวลาที่ใช้ทั้งหมด (วินาที)
น้ำ (ลิตร)	ปุ๋ย (ลิตร)	ปั๊มน้ำ		ปั๊มปุ๋ย		ผสม		ฉีดพ่น		
		1	2	1	2	1	2	1	2	
3	1	25.5	25.5	8.5	8.5	34	34	34	34	204
		51		17		68		68		
5	2	42.5	42.5	17	17	59.5	59.5	59.5	59.5	357
		85		34		119		119		
7	3	59.5	59.5	25.5	25.5	85	85	85	85	510
		119		51		170		170		
10	5	85	85	42.5	42.5	127.5	127.5	127.5	127.5	765
		170		85		255		255		

จากตารางที่ 4.3 แสดงการคำนวณระยะเวลาการทำงานของมอเตอร์ปั๊มน้ำ ปั๊มปุ๋ย มอเตอร์ผสมและปั๊มฉีดพ่นใน 2 รอบการทำงาน โดยใช้อัตราส่วนของน้ำและปุ๋ยที่ต่างกัน 4 ครั้งได้แก่ 3:1, 5:2, 7:3 และ 10:5 ซึ่งเวลาที่ใส่ไว้ในตารางแยกออกเป็นรอบที่ 1 และ 2 และตัวเลขสีแดงเป็นเวลารวมที่มอเตอร์ทำงาน

ตารางที่ 4.4 ผลการทดสอบโดยการปฏิบัติจริง 2 รอบ

อัตราส่วน		ระยะเวลาการทำงาน (วินาที)								เวลาที่ใช้ทั้งหมด (วินาที)
น้ำ (ลิตร)	ปุ๋ย (ลิตร)	ปั๊มน้ำ		ปั๊มปุ๋ย		ผสม		ฉีดพ่น		
		1	2	1	2	1	2	1	2	
3	1	25.71	25.77	8.36	8.68	34.18	34.8	34.22	35.96	207.68
		51.48		17.04		68.98		70.18		
5	2	43.05	42.63	16.98	17.15	60.39	59.84	59.61	59.87	359.52
		85.68		34.13		120.23		119.48		
7	3	60.22	59.72	25.71	26.62	85.41	85.31	85.83	85.60	514.42
		119.94		52.33		170.72		171.43		
10	5	85.63	86.25	42.86	42.08	128.66	128.49	128.04	128.7	770.71
		171.88		84.94		257.15		256.74		

จากตารางที่ 4.4 แสดงระยะเวลาการทำงานของมอเตอร์ปั๊มน้ำ ปั๊มปุ๋ย มอเตอร์ผสมและปั๊มฉีดพ่นใน 2 รอบการทำงาน โดยใช้อัตราส่วนของน้ำและปุ๋ยที่ต่างกัน 4 ครั้งได้แก่ 3:1, 5:2, 7:3 และ 10:5 ซึ่งเวลาที่ใส่ไว้ในตารางแยกออกเป็นรอบที่ 1 และ 2 และตัวเลขสีแดงเป็นเวลารวมที่มอเตอร์ทำงาน

ตารางที่ 4.5 ความคลาดเคลื่อนของเวลาระหว่างการคำนวณกับการทดสอบจริง

อัตราส่วน		ค่าความคลาดเคลื่อน (วินาที)				เวลาที่ใช้ทั้งหมด (วินาที)
น้ำ (ลิตร)	ปุ๋ย (ลิตร)	ปั๊มน้ำ	ปั๊มปุ๋ย	ผสม	ฉีดพ่น	
3	1	-1.25	-2.82	-1.44	-5.70	-2.93
5	2	-1.10	-0.35	-1.47	-1.17	-0.90
7	3	-1.13	-1.96	-0.42	-1.28	-11.0
10	5	-0.94	-4.02	0.06	-1.92	-1.27

จากตารางที่ 4.4 เป็นการแสดงค่าความคลาดเคลื่อนของเวลาเมื่อเปรียบเทียบกับการคำนวณกับเวลาจริงจะเห็นว่าตัวเลขในตารางมีค่าเป็นลบ แสดงให้เห็นว่าเวลาที่ทดสอบจริงใช้เวลานานกว่าที่คำนวณ

<u>หมายเหตุ</u>

- มอเตอร์ที่ทำการผสมและมอเตอร์ปั๊มที่ฉีดพ่นปุ๋ยที่ผสมเสร็จแล้ว จะทำงานโดยการคำนวณปริมาณน้ำและปริมาณปุ๋ยรวมกันเป็นจำนวนลิตร
- ค่าความคลาดเคลื่อนคิดจาก

$$\text{ค่าความคลาดเคลื่อน} = \frac{\text{เวลาจากการคำนวณ} - \text{เวลาจริง}}{\text{เวลาจากการคำนวณ}} \times 100$$

บทที่ 5

สรุปผลและข้อเสนอแนะ

จากการดำเนินงานสามารถมาสรุปผลและเสนอแนวทางแก้ไข พร้อมทั้งให้ข้อเสนอแนะเพื่อการนำโครงงานไปพัฒนาต่อ

5.1 สรุปผลการดำเนินโครงงาน

จากการดำเนินงานในการทดสอบระบบการทำงานของเครื่องผสมปุ๋ยน้ำอัตโนมัติ ในส่วนของการควบคุมขั้นตอนการผสมปุ๋ยด้วยโปรแกรมไมโครคอนโทรลเลอร์ ผลการทดสอบที่ได้เป็นไปตามที่กำหนดคือ เมื่อกดสวิตช์เปิดของกล่องควบคุม หน้าจอแสดงผลจะแสดงเวลาปัจจุบันและสถานะการทำงาน เมื่อป้อนค่าเพื่อตั้งปริมาณน้ำ ปริมาณปุ๋ย เวลาเริ่มการทำงานและจำนวนรอบการทำงานแล้ว ทำให้ระบบสามารถทำงานตามขั้นตอน หรือตามข้อมูลที่ป้อนไว้ และสามารถเริ่มและหยุดการทำงานได้โดยอัตโนมัติตามเวลาที่ตั้งไว้หรือแม้กระทั่งน้ำหรือปุ๋ยหมดก็สามารถหยุดการทำงานได้โดยอัตโนมัติเช่นกัน

5.2 ปัญหาและแนวทางแก้ไข

จากการดำเนินงานในการทดสอบระบบการทำงานของเครื่องผสมปุ๋ยน้ำอัตโนมัติ ในส่วนของขั้นตอนการฉีดพ่นปุ๋ยที่ผสมเรียบร้อยแล้วพบว่า

ก) เมื่อปริมาณปุ๋ยในถังผสมมีระดับต่ำกว่ามอเตอร์ทำให้ไม่สามารถฉีดพ่นปุ๋ยออกจากถังผสมได้หมด แนวทางแก้ไขคือสามารถใช้มอเตอร์ปั๊มที่มีลักษณะราบเรียบแต่พลังการดูดสูง จะสามารถดูดน้ำออกได้หมด

ข) ความคลาดเคลื่อนของเวลาที่ใช้ในการดูดของมอเตอร์ปั๊มใช้เวลามากกว่าที่คำนวณไว้ ค่าความคลาดเคลื่อนนี้อาจเกิดจากการผิดพลาดของการจับเวลา โดยค่าความคลาดเคลื่อนส่วนมากไม่เกินสามวินาที

5.3 แนวทางการพัฒนาต่อไป

เนื่องจากการออกแบบเครื่องผสมปุ๋ยน้ำอัตโนมัติในโครงงานนี้เป็นเพียงแบบจำลองเท่านั้น เซ็นเซอร์ที่ใช้จึงมีข้อจำกัดคือใช้ได้จนกว่าสนิทจะขึ้น แต่ถ้ามีการนำเอาเครื่องผสมปุ๋ยน้ำอัตโนมัตินี้ไปใช้งานจริงจึงต้องมีการเปลี่ยนเซ็นเซอร์ใหม่โดยใช้ลูกลอยซึ่งมีความเที่ยงตรงและทนทานสูงกว่าสายไฟ

ในส่วนของอัตราการผสมของน้ำกับปุ๋ยอาจจะออกแบบให้ทำงานอัตโนมัติในอนาคต ให้มีการเก็บข้อมูลโดยไม่ต้องทำการป้อนปริมาณน้ำและปุ๋ยใหม่ทุกครั้ง เมื่อได้อัตราของส่วนผสมที่แน่นอน เพื่อความสะดวกในการทำงานต่อไป และเครื่องผสมปุ๋ยน้ำอัตโนมัตินี้ยังสามารถใช้เป็นเครื่องรดน้ำอัตโนมัติได้อีกด้วย

มหาวิทยาลัยนเรศวร

เอกสารอ้างอิง

[1] ULN 2803A Description. จาก www.wvshare.com/produck/ULN22003A%20(TI).html สืบค้นเมื่อวันที่ 10 ตุลาคม 2554,

[2] ประจิน พลังสันติกุล. (2549). การเขียนโปรแกรมควบคุมไมโครคอนโทรลเลอร์ AVR ด้วยภาษา C กับ win AVR.กรุงเทพฯ:แอพซอฟต์เทค

[3] การใช้งาน RTC-Real time clock ด้วย DS1307. (2008-06-01). Mindtek Online Articles. จาก www.mind-tek.net/ds1307.php สืบค้นเมื่อวันที่ 10 ตุลาคม 2554,

[4] ประจิน พลังสันติกุล. (2551) .การประยุกต์ใช้งานด้วยภาษา C สำหรับไมโครคอนโทรลเลอร์ AVR. กรุงเทพฯ: แอพซอฟต์เทค

ภาคผนวก ก
รายละเอียดของไมโครคอนโทรลเลอร์รุ่น ATmega8515

Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- RISC Architecture
  - 130 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
  - 8K Bytes of In-System Self-programmable Flash
    Endurance: 10,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - 512 Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 512 Bytes Internal SRAM
  - Up to 64K Bytes Optional External Memory Space
  - Programming Lock for Software Security
- Peripheral Features
  - One 8-bit Timer/Counter with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Three PWM Channels
  - 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
  - Three Sleep Modes: Idle, Power-down and Standby
- I/O and Packages
  - 35 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, and 44-pad QFN/MLF
- Operating Voltages
  - 2.7 - 5.5V for ATmega8515L
  - 4.5 - 5.5V for ATmega8515
- Speed Grades
  - 0 - 8 MHz for ATmega8515L
  - 0 - 16 MHz for ATmega8515

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

ATmega8515
ATmega8515L

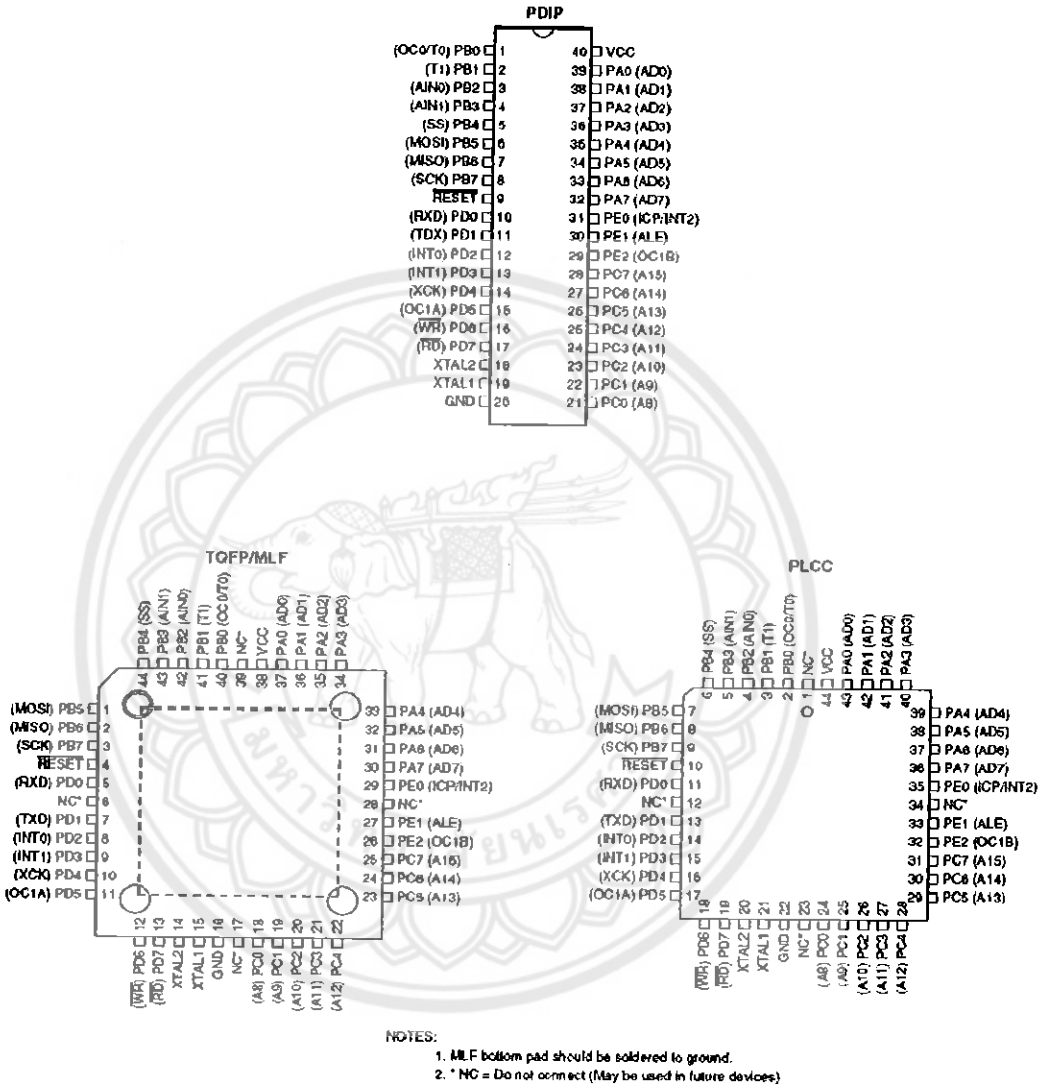
Summary

2512JS–AVR–10/06

Note: This is a summary document. A complete document is available on our Web site at www.atmel.com.

Pin Configurations

Figure 1. Pinout ATmega8515



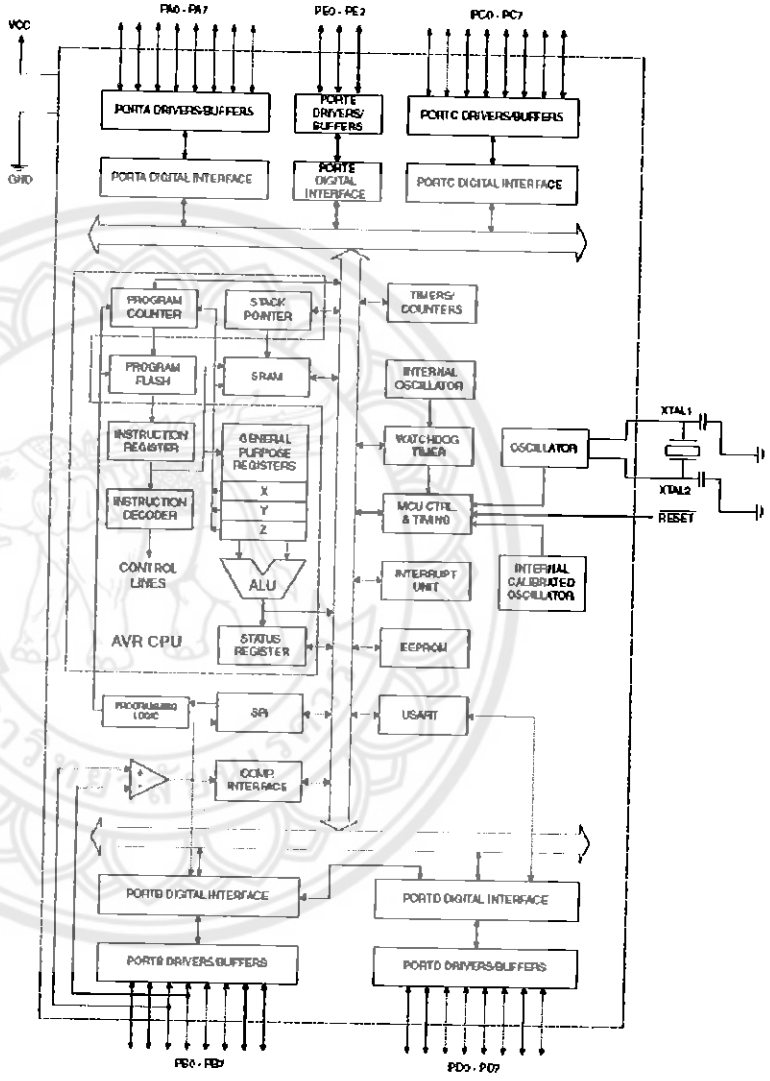
NOTES:
1. MLF bottom pad should be soldered to ground.
2. * NC = Do not connect (May be used in future devices)

Overview

The ATmega8515 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 ATmega8515 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 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 ATmega8515 provides the following features: 8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes EEPROM, 512 bytes SRAM, an External memory interface, 35 general purpose I/O lines, 32 general purpose working registers, two flexible Timer/Counters with compare modes, Internal and External interrupts, a Serial Programmable USART, a programmable Watchdog Timer with internal Oscillator, a SPI serial port, and three software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and Interrupt system to continue functioning. The Power-down mode saves the Register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In 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.

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 ATmega8515 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega8515 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.

Disclaimer

Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

AT90S4414/8515 and ATmega8515 Compatibility

The ATmega8515 provides all the features of the AT90S4414/8515. In addition, several new features are added. The ATmega8515 is backward compatible with AT90S4414/8515 in most cases. However, some incompatibilities between the two microcontrollers exist. To solve this problem, an AT90S4414/8515 compatibility mode can be selected by programming the S8515C Fuse. ATmega8515 is 100% pin compatible with AT90S4414/8515, and can replace the AT90S4414/8515 on current printed circuit boards. However, the location of Fuse bits and the electrical characteristics differs between the two devices.

AT90S4414/8515 Compatibility Mode

Programming the S8515C Fuse will change the following functionality:

- The timed sequence for changing the Watchdog Time-out period is disabled. See "Timed Sequences for Changing the Configuration of the Watchdog Timer" on page 53 for details.
- The double buffering of the USART Receive Registers is disabled. See "AVR USART vs. AVR UART – Compatibility" on page 137 for details.
- PORTE(2:1) will be set as output, and PORTE0 will be set as input.

Pin Descriptions

VCC

Digital supply voltage.

GND

Ground.

Port A (PA7..PA0)

Port A is an 8-bit bi-directional I/O port with 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 A also serves the functions of various special features of the ATmega8515 as listed on page 67.

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 ATmega8515 as listed on page 67.

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.

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 ATmega8515 as listed on page 72.

Port E(PE2..PE0)

Port E is an 3-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port E output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated. The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port E also serves the functions of various special features of the ATmega8515 as listed on page 74.

$\overline{\text{RESET}}$

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 18 on page 46. 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.

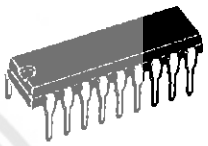
ภาคผนวก ข

รายละเอียดไอซีหมายเลข ULN2803A

ULN2801A
ULN2802A - ULN2803A
ULN2804A - ULN2805A

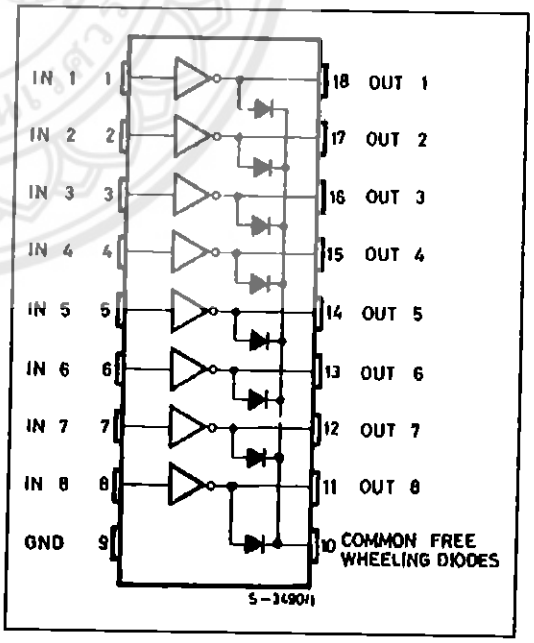
EIGHT DARLINGTON ARRAYS

- EIGHT DARLINGTONS WITH COMMON EMITTERS
- OUTPUT CURRENT TO 500 mA
- OUTPUT VOLTAGE TO 50 V
- INTEGRAL SUPPRESSION DIODES
- VERSIONS FOR ALL POPULAR LOGIC FAMILIES
- OUTPUT CAN BE PARALLELED
- INPUTS PINNED OPPOSITE OUTPUTS TO SIMPLIFY BOARD LAYOUT



DIP18

PIN CONNECTION (top view)



DESCRIPTION

The ULN2801A-ULN2805A each contain eight darlington transistors with common emitters and integral suppression diodes for inductive loads. Each darlington features a peak load current rating of 600mA (500mA continuous) and can withstand at least 50V in the off state. Outputs may be paralleled for higher current capability.

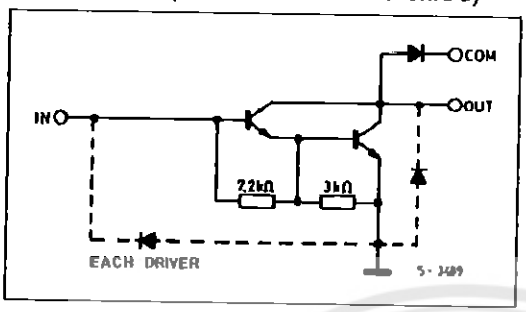
Five versions are available to simplify interfacing to standard logic families : the ULN2801A is designed for general purpose applications with a current limit resistor ; the ULN2802A has a 10.5kΩ input resistor and zener for 14-25V PMOS ; the ULN2803A has a 2.7kΩ input resistor for 5V TTL and CMOS ; the ULN2804A has a 10.5kΩ input resistor for 6-15V CMOS and the ULN2805A is designed to sink a minimum of 350mA for standard and Schottky TTL where higher output current is required.

All types are supplied in a 18-lead plastic DIP with a copper lead from and feature the convenient input-opposite-output pinout to simplify board layout.

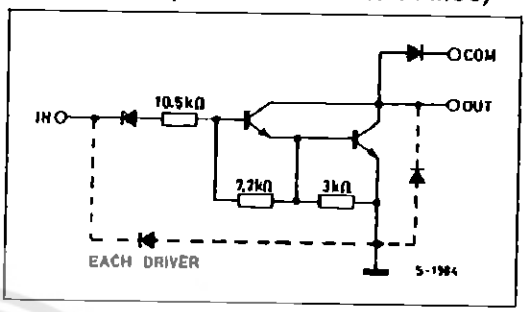
ULN2801A - ULN2802A - ULN2803A - ULN2804A - ULN2805A

SCHEMATIC DIAGRAM AND ORDER CODES

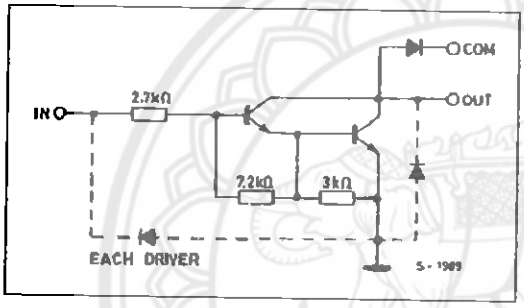
For ULN2801A (each driver for PMOS-CMOS)



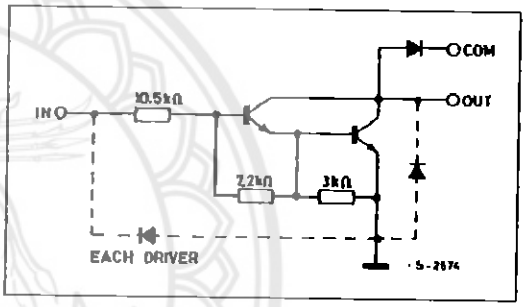
For ULN2802A (each driver for 14-15 V PMOS)



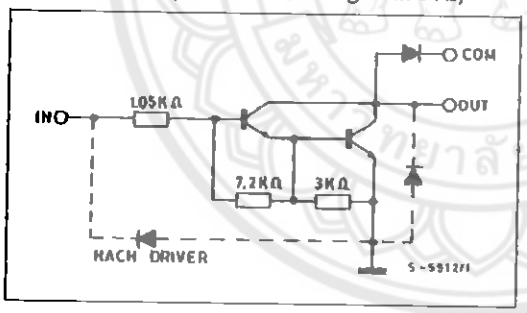
For ULN2803A (each driver for 5 V, TTL/CMOS)



For ULN2804A (each driver for 6-15 V CMOS/PMOS



For ULN2805A (each driver for high out TTL)



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_o	Output Voltage	50	V
V_i	Input Voltage for ULN2802A, UL2803A, ULN2804A for ULN2805A	 30 15	V
I_C	Continuous Collector Current	500	mA
I_B	Continuous Base Current	25	mA
P_{tot}	Power Dissipation (one Darlington pair) (total package)	 1.0 2.25	W
T_{amb}	Operating Ambient Temperature Range	– 20 to 85	°C
T_{stg}	Storage Temperature Range	– 55 to 150	°C
Tj	Junction Temperature Range	– 20 to 150	°C

THERMAL DATA

Symbol	Parameter		Value	Unit
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	55	°C/W

ELECTRICAL CHARACTERISTICS (T_{amb} = 25°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit	Fig.
I_{CEX}	Output Leakage Current	V_{CE} = 50V			50	µA	1a
		T_{amb} = 70°C, V_{CE} = 50V			100	µA	1a
		T_{amb} = 70°C for ULN2802A V_{CE} = 50V, V_i = 6V			500	µA	1b
		for ULN2804A V_{CE} = 50V, V_i = 1V			500	µA	1b
$V_{CE(sat)}$	Collector-emitter Saturation Voltage	I_C = 100mA, I_B = 250µA		0.9	1.1	V	2
		I_C = 200mA, I_B = 350µA		1.1	1.3	V	
		I_C = 350mA, I_B = 500µA		1.3	1.6	V	
$I_{i(on)}$	Input Current	for ULN2802A V_i = 17V		0.82	1.25	mA	3
		for ULN2803A V_i = 3.85V		0.93	1.35	mA	
		for ULN2804A V_i = 5V		0.35	0.5	mA	
		V_i = 12V		1	1.45	mA	
		for ULN2805A V_i = 3V		1.5	2.4	mA	
$I_{i(off)}$	Input Current	T_{amb} = 70°C, I_C = 500µA	50	65		µA	4
$V_{i(on)}$	Input Voltage	V_{CE} = 2 V for ULN2802A I_C = 300mA			13	V	5
		for ULN2803A I_C = 200mA			2.4	V	
		I_C = 250mA			2.7	V	
		I_C = 300mA			3	V	
		for ULN2804A I_C = 125mA			5	V	
		I_C = 200mA			6	V	
		I_C = 275mA			7	V	
		I_C = 350mA			8	V	
		for ULN2805A I_C = 350mA			2.4	V	
h_{FE}	DC Forward Current Gain	for ULN2801A V_{CE} = 2V, I_C = 350mA	1000			–	2
C_I	Input Capacitance			15	25	pF	–
t_{PLH}	Turn-on Delay Time	0.5 V_i to 0.5 V_o		0.25	1	µs	–
t_{PHL}	Turn-off Delay Time	0.5 V_i to 0.5 V_o		0.25	1	µs	–
I_R	Clamp Diode Leakage Current	V_R = 50V			50	µA	6
		T_{amb} = 70°C, V_R = 50V			100	µA	6
V_F	Clamp Diode Forward Voltage	I_F = 350mA		1.7	2	V	7

ภาคผนวก ค

รายละเอียดไอซีหมายเลข LM2575

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

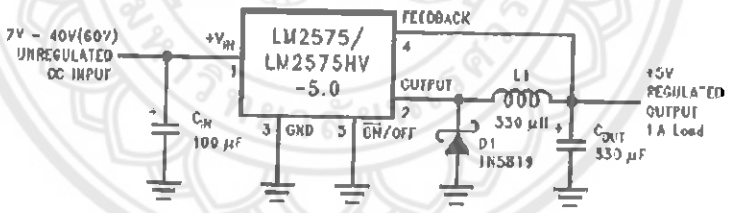
- 3.3V, 5V, 12V, 15V, and adjustable output versions
- Adjustable version output voltage range, 1.23V to 37V (57V for HV version) ±4% max over line and load conditions
- Guaranteed 1A output current
- Wide input voltage range, 40V up to 60V for HV version
- Requires only 4 external components
- 52 kHz 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

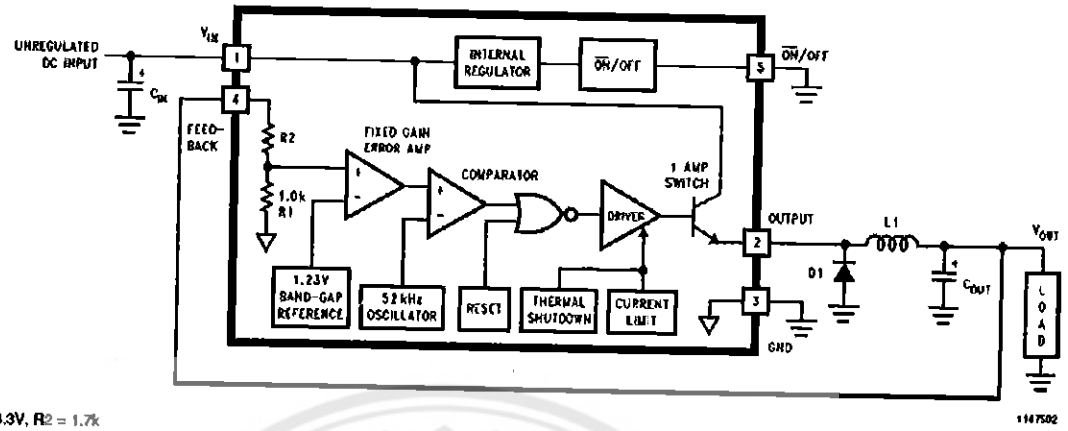
(Fixed Output Voltage Versions)



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

114594

Block Diagram and Typical Application



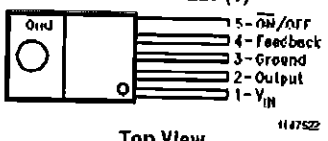
3.3V, R2 = 1.7k
5V, R2 = 3.1k
12V, R2 = 8.84k
15V, R2 = 11.3k
For ADJ. Version
R1 = Open, R2 = 0Ω
Note: Pin numbers are for the TO-220 package.

FIGURE 1.

Connection Diagrams

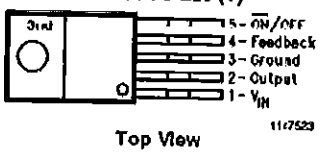
(XX indicates output voltage option. See Ordering Information table for complete part number.)

Straight Leads
5-Lead TO-220 (T)

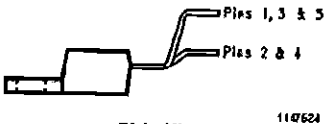


Top View
LM2575T-XX or LM2575HVT-XX
See NS Package Number T05A

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

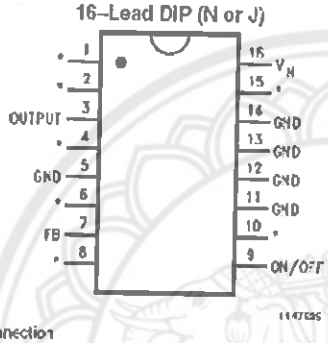


Top View



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

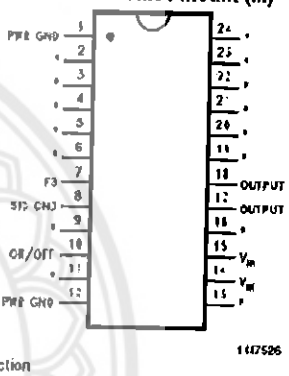
16-Lead DIP (N or J)



*No Internal Connection

Top View
LM2575N-XX or LM2575HVN-XX
See NS Package Number N16A
LM1575J-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

Ordering Information

Package Type	NSC Package Number	Standard Voltage Rating (40V)	High Voltage Rating (60V)	Temperature Range
5-Lead TO-220 Straight Leads	T05A	LM2575T-3.3 LM2575T-5.0 LM2575T-12 LM2575T-15 LM2575T-ADJ	LM2575HVT-3.3 LM2575HVT-5.0 LM2575HVT-12 LM2575HVT-15 LM2575HVT-ADJ	
5-Lead TO-220 Bent and Staggered Leads	T05D	LM2575T-3.3 Flow LB03 LM2575T-5.0 Flow LB03 LM2575T-12 Flow LB03 LM2575T-15 Flow LB03 LM2575T-ADJ Flow LB03	LM2575HVT-3.3 Flow LB03 LM2575HVT-5.0 Flow LB03 LM2575HVT-12 Flow LB03 LM2575HVT-15 Flow LB03 LM2575HVT-ADJ Flow LB03	
16-Pin Molded DIP	N16A	LM2575N-5.0 LM2575N-12 LM2575N-15 LM2575N-ADJ	LM2575HVN-5.0 LM2575HVN-12 LM2575HVN-15 LM2575HVN-ADJ	$-40°C \leq T_J \leq +125°C$
24-Pin Surface Mount	M24B	LM2575M-5.0 LM2575M-12 LM2575M-15 LM2575M-ADJ	LM2575HVM-5.0 LM2575HVM-12 LM2575HVM-15 LM2575HVM-ADJ	
5-Lead TO-263 Surface Mount	TS5B	LM2575S-3.3 LM2575S-5.0 LM2575S-12 LM2575S-15 LM2575S-ADJ	LM2575HVS-3.3 LM2575HVS-5.0 LM2575HVS-12 LM2575HVS-15 LM2575HVS-ADJ	
16-Pin Ceramic DIP	J16A	LM1575J-3.3-QML LM1575J-5.0-QML LM1575J-12-QML LM1575J-15-QML LM1575J-ADJ-QML		$-55°C \leq T_J \leq +150°C$

ภาคผนวก ง
รหัสต้นฉบับของโปรแกรมไมโครคอนโทรลเลอร์

```
/*****************************************************
This program was produced by the
CodeWizardAVR V1.24.8b Professional
Automatic Program Generator
Copyright 1998-2006 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com
Date    : 27/3/2012
Author  : F4CG
Company : F4CG
Comments:
Chip type           : ATmega8515L
Program type        : Application
Clock frequency     : 11.059200 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 128
*****************************************************/
#include <mega8515.h>
#include <stdio.h>
#include <string.h>
#include <delay.h>
#include <stdlib.h>
#include <math.h>
#define col_1   PORTC.0
#define col_2   PORTC.1
#define col_3   PORTC.2
#define row_1   PINC.3
```

```
#define row_2   PINC.4
#define row_3   PINC.5
#define row_4   PINC.6
#define Relay_water      PORTD.4
#define Relay_fertilizer  PORTD.5
#define Relay_Mixer      PORTD.6
#define Relay_Pay        PORTD.7
#define SW_water      PIND.0
#define SW_fertilizer  PIND.1
int num;
char hour_,min_,sec_;
eeprom char hour_strat,min_start,sec_start;
char Show_lcd[16];
Int Time1,Time2,count;
int loop_set;
int loop;
int loop1;
eeprom char round;
int water_pump,fertilizer_pump,mixer,ready ;
eeprom int LV_water,LV_fertilizer;
int Set1,Set2,Set3;
long Timer1,Timer2,Timer3,Timer4;
int m;
eeprom long Time_li = 1000;
long Time_li1,Time_li2,Time_li3,Time_li4;
// I2C Bus functions
#asm
```

```
  .equ __i2c_port=0x18 ;PORTB
  .equ __sda_bit=0
  .equ __scl_bit=1
#endasm
#include <i2c.h>
// DS1307 Real Time Clock functions
#include <ds1307.h>
// Alphanumeric LCD Module functions
#asm
  .equ __lcd_port=0x1B ;PORTA
#endasm
#include <lcd.h>
// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
// Reinitialize Timer 0 value
TCNT0=0x50;
// Place your code here
 if(water_pump==1)
 {
   Relay_water = 1;
   if(Timer1++>((long)LV_water*Time_li)){ water_pump = 0;  Relay_water = 0;  }
 }else if(fertilizer_pump==1)
 {
   Relay_fertilizer = 1;
   if(Timer2++>((long)LV_fertilizer*Time_li)){ fertilizer_pump = 0;  Relay_fertilizer = 0;  }
 }else if(mixer==1)
```

```
 {
   Relay_Mixer = 1;
   if(Timer3++>(((long)LV_fertilizer+(long)LV_water)*Time_li)){ mixer = 0;  Relay_Mixer = 0;
} }else if(ready==1)
 {
   Relay_Pay = 1;
   if(Timer4++>(((long)LV_fertilizer+(long)LV_water)*Time_li)){ ready = 0;  Relay_Pay = 0;  }
 }}
// Declare your global variables here
void Scan_Keypad (void)
{
      PORTC = 0xFF;
           col_1 = 0; col_2 = 1; col_3 = 1;   delay_ms(3);
      if(!row_1)     { num = 1;    while(!row_1); }
      else if(!row_2) { num = 4;    while(!row_2); }
      else if(!row_3) { num = 7;    while(!row_3); }
      else if(!row_4) { num = '*';  while(!row_4); }
      col_1 = 1; col_2 = 0; col_3 = 1;    delay_ms(3);
      if(!row_1)     { num = 2;    while(!row_1); }
      else if(!row_2) { num = 5;    while(!row_2); }
      else if(!row_3) { num = 8;    while(!row_3); }
      else if(!row_4) { num = 0;    while(!row_4); }
      col_1 = 1; col_2 = 1; col_3 = 0;     delay_ms(3);
      if(!row_1)     { num = 3;    while(!row_1); }
      else if(!row_2) { num = 6;    while(!row_2); }
      else if(!row_3) { num = 9;    while(!row_3); }
      else if(!row_4) { num = '#';  while(!row_4); }
```

```
    }
void main(void)
{
// Declare your local variables here
// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;
// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0x00;
// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0xFF;
DDRC=0b00000111;
// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x0F;
DDRD=0xF0;
// Port E initialization
// Func2=In Func1=In Func0=In
```

```
// State2=T State1=T State0=T
PORTE=0x00;
DDRE=0x00;
// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: 172.800 kHz
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x03;
TCNT0=0x50;
OCR0=0x00;
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
```

```
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
EMCUCR=0x00;
// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x02;
// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
// I2C Bus initialization
i2c_init();
// DS1307 Real Time Clock initialization
// Square wave output on pin SQW/OUT: Off
// SQW/OUT pin state: 0
rtc_init(0,0,0);
// LCD module initialization
lcd_init(16);
// Global enable interrupts
```

```
#asm("sei")
sec_ = rtc_read(0);
rtc_write(0,sec_ & 0x7f);
//rtc_set_time(4,17,0);
Scan_Keypad();
 if(num == '#')
     {      loop1 = 1;
          num = 'C';
          lcd_clear();
          while(loop1)
          {  lcd_gotoxy(0,0);
              sprintf(Show_lcd,"Set Timer liter ");
              lcd_puts(Show_lcd);
              lcd_gotoxy(0,1);
              sprintf(Show_lcd,"=======> %i     ",Time_li);
              lcd_puts(Show_lcd);
              Scan_Keypad();
              if((num>=0)&&(num<=9))
              {
                  count++;
                //  lcd_clear();
                  if(count == 1)     { Time_li1 = num;                Time_li  = Time_li1; }
                  else if(count == 2) { Time_li2 = (Time_li1*10)+num;      Time_li  =
Time_li2; }
                  else if(count == 3) { Time_li3 = (Time_li2*10)+num;      Time_li  =
Time_li3; }
```

```
                else if(count == 4) { Time_li4 = (Time_li3*10)+num;      Time_li  =
Time_li4; }
                if(count >= 5) count = 5;
                num = 'C';
                   }
              if(num == '#')  {
                count = 0;
                Time_li = 0;
                lcd_clear();
                num = 'C';
              }
            else if(num == '*')
            {
                loop1 = 0;
                count = 0;
                num = 'C';
                lcd_gotoxy(0,0);
                sprintf(Show_lcd,"Set Timer liter ");
                lcd_puts(Show_lcd);
                lcd_gotoxy(0,1);
                sprintf(Show_lcd,"=====> %i Save   ",Time_li);
                lcd_puts(Show_lcd);
                delay_ms(1000);
            }
            delay_ms(50);
                    }      }
if(num == '*')
```

```
{
    loop = 1;
    num = 'C';
    lcd_clear();
    hour_ = 0; min_ = 0; sec_ = 0;
    while(loop)
    {
        lcd_gotoxy(0,0);
        sprintf(Show_lcd,"Set Initial Time");
        lcd_puts(Show_lcd);
        lcd_gotoxy(0,1);
        sprintf(Show_lcd,"%02i:%02i:%02i    ",hour_,min_,sec_);
        lcd_puts(Show_lcd);
        Scan_Keypad();
        if((num>=0)&&(num<=9))
        {
            count++;
          //  lcd_clear();
            if(count == 1)    { Time1 = num;              hour_ = Time1; }
            else if(count == 2) { Time2 = (Time1*10)+num;     hour_ = Time2; }
            else if(count == 3) { Time1 = num;              min_ = Time1; }
            else if(count == 4) { Time2 = (Time1*10)+num;     min_ = Time2; }
            else if(count == 5) { Time1 = num;              sec_ = Time1; }
            else if(count == 6) { Time2 = (Time1*10)+num;     sec_ = Time2; }
             if( hour_ >= 24 )       hour_ = 0;
            if( min_ >= 60 )       min_ = 0;
            if( sec_ >= 60 )       sec_ = 0;
```

```
            if(count >= 6) count = 7;
                num = 'C';
        }
        if(num == '*')  {
            count = 0;
            hour_ = 0;
            min_  = 0;
            sec_ = 0;
            lcd_clear();
            num = 'C';
        }
        else if(num == '#')
        {
            loop = 0;
            count = 0;
            num = 'C';
            rtc_set_time(hour_,min_,sec_);
        //    rtc_set_date(date,month,year);
            lcd_gotoxy(0,0);
            sprintf(Show_lcd,"Set Initial Time");
            lcd_puts(Show_lcd);
            lcd_gotoxy(0,1);
            sprintf(Show_lcd,"%02i:%02i:%02i SAVE  ",hour_,min_,sec_);
            lcd_puts(Show_lcd);
            delay_ms(1000);
        }
        delay_ms(50);
```

```
        }     }
while (1)
    {
    // Place your code here
        rtc_get_time(&hour_,&min_,&sec_);
        Scan_Keypad();
        count = 0;
        if(num == '*')
        {
          loop = 1;
          num = 'C';
          lcd_clear();
          while(loop)
          {
              lcd_gotoxy(0,0);
              sprintf(Show_lcd,"  Set water    ");
              lcd_puts(Show_lcd);
              lcd_gotoxy(0,1);
              sprintf(Show_lcd,"  %i.%i L      ",(LV_water/10),(LV_water%10));
              lcd_puts(Show_lcd);
              Scan_Keypad();
              if((num>=0)&&(num<=9))
              {
                  count++;
                  if(count == 1)     { Set1 = num;              LV_water  = Set1; }
                  else if(count == 2) { Set2 = (Set1*10)+num;     LV_water  = Set2; }
                  else if(count == 3) { Set3 = (Set2*10)+num;     LV_water  = Set3; }
```

```
            if( LV_water >= 200 )        LV_water = 200;
          if(count >= 4) count = 4;
            num = 'C';
        }
         if(num == '*') {
            count = 0;
            LV_water = 0;
            lcd_clear();
            num = 'C';
         }
        else if(num == '#')
        {
            loop = 0;
            count = 0;
            num = 'C';
        }          }
  loop = 1;
num = 'C';
lcd_clear();
while(loop)
{
     lcd_gotoxy(0,0);
     sprintf(Show_lcd,"  Set fertilizer");
     lcd_puts(Show_lcd);
     lcd_gotoxy(0,1);
     sprintf(Show_lcd,"  %i.%i L      ",(LV_fertilizer/10),(LV_fertilizer%10));
     lcd_puts(Show_lcd);
```

```
        Scan_Keypad();
        if((num>=0)&&(num<=9))
        {
            count++;
            if(count == 1)     { Set1 = num;              LV_fertilizer  = Set1; }
            else if(count == 2) { Set2 = (Set1*10)+num;     LV_fertilizer  = Set2; }
            else if(count == 3) { Set3 = (Set2*10)+num;     LV_fertilizer  = Set3; }
            if( LV_water >= 200 )       LV_water = 200;
             if(count >= 4) count = 4;
            num = 'C';
          }
        if(num == '*')  {
            count = 0;
            LV_fertilizer = 0;
            lcd_clear();
            num = 'C';
         }
        else if(num == '#')
        {
            loop = 0;
            count = 0;
            num = 'C';
        }              }
    loop = 1;
    num = 'C';
    lcd_clear();
    hour_ = 0; min_ = 0; sec_ = 0;
```

```
while(loop)
{
    lcd_gotoxy(0,0);
    sprintf(Show_lcd,"  Time start  ");
    lcd_puts(Show_lcd);
    lcd_gotoxy(0,1);
    sprintf(Show_lcd,"  %02i:%02i:%02i  ",hour_strat,min_start,sec_start);
    lcd_puts(Show_lcd);
    Scan_Keypad();
    if((num>=0)&&(num<=9))
    {
      count++;
     //  lcd_clear();
      if(count == 1)    { Time1 = num;              hour_strat  = Time1; }
      else if(count == 2) { Time2 = (Time1*10)+num;     hour_strat = Time2; }
      else if(count == 3) { Time1 = num;              min_start = Time1; }
      else if(count == 4) { Time2 = (Time1*10)+num;     min_start = Time2; }
      else if(count == 5) { Time1 = num;              sec_start = Time1; }
      else if(count == 6) { Time2 = (Time1*10)+num;     sec_start = Time2; }
      if( hour_strat >= 24 )       hour_strat = 0;
      if( min_start  >= 60 )       min_start = 0;
      if( sec_start  >= 60 )       sec_start = 0;
       if(count >= 6) count = 7;
      num = 'C';
    }
    if(num == '*')  {
      count = 0;
```

```
            hour_strat = 0;
            min_start  = 0;
            sec_start = 0;
            lcd_clear();
            num = 'C';
        }
        else if(num == '#')
        {
            loop = 0;
            count = 0;
            num = 'C';
        }           }
loop = 1;
num = 'C';
lcd_clear();
while(loop)
{
    lcd_gotoxy(0,0);
    sprintf(Show_lcd,"  Play off     ");
    lcd_puts(Show_lcd);
    lcd_gotoxy(0,1);
    sprintf(Show_lcd,"  %i round     ",round);
    lcd_puts(Show_lcd);
    Scan_Keypad();
    if((num>=0)&&(num<=9))
    {
        count++;
```

```
            if(count == 1)     { Set1 = num;              round  = Set1; }
             if( round >= 100 )        round = 100;
              if(count >= 2) count = 2;
            num = 'C';
          }                    if(num == '*')  {
            count = 0;
            round = 0;
            lcd_clear();
            num = 'C';
          }                    else if(num == '#')
          {                          loop = 0;
            count = 0;
            num = 'C';
          }              }           }
        if( (hour_strat == hour_) && (min_start == min_) && (sec_start == sec_) )
{             for(m=1;m<=round;m++)
  {                        if((SW_water) || (SW_fertilizer) )
        {                    lcd_gotoxy(0,0);
          sprintf(Show_lcd,"  No Mixer     ");
          lcd_puts(Show_lcd);
          lcd_gotoxy(0,1);
          sprintf(Show_lcd,"               ");
          lcd_puts(Show_lcd);
          delay_ms(2000);
        }else
        {          Timer1 = 0;
          water_pump = 1;
```

```
lcd_gotoxy(0,0);
sprintf(Show_lcd,"  Water Pump    ");
lcd_puts(Show_lcd);
lcd_gotoxy(0,1);
sprintf(Show_lcd,"  %i.%i L       ",(LV_water/10),(LV_water%10));
lcd_puts(Show_lcd);
delay_ms(100);
while(water_pump);
 Timer2 = 0;
fertilizer_pump = 1;
lcd_gotoxy(0,0);
sprintf(Show_lcd," fertilizer Pump");
lcd_puts(Show_lcd);
lcd_gotoxy(0,1);
sprintf(Show_lcd,"  %i.%i L       ",(LV_fertilizer/10),(LV_fertilizer%10));
lcd_puts(Show_lcd);
delay_ms(100);
while(fertilizer_pump);
  Timer3 = 0;
mixer = 1;
lcd_gotoxy(0,0);
sprintf(Show_lcd," Mixer         ");
lcd_puts(Show_lcd);
lcd_gotoxy(0,1);
sprintf(Show_lcd," %i.%i L
",((LV_water+LV_fertilizer)/10),((LV_water+LV_fertilizer)%10));
lcd_puts(Show_lcd);
```

```
                delay_ms(100);
                while(mixer);
                  Timer4 = 0;
                ready = 1;
                lcd_gotoxy(0,0);
                sprintf(Show_lcd," ready %i      ",m);
                lcd_puts(Show_lcd);
                lcd_gotoxy(0,1);
                sprintf(Show_lcd,"               ");
                lcd_puts(Show_lcd);
                delay_ms(100);
                while(ready);
                  }          }          }
        lcd_gotoxy(0,0);
        sprintf(Show_lcd,"   %02i:%02i:%02i   ",hour_,min_,sec_);
        lcd_puts(Show_lcd);
        lcd_gotoxy(0,1);
        if((SW_water) || (SW_fertilizer) )   sprintf(Show_lcd,"   No Mixer   ");
        else sprintf(Show_lcd,"               ",hour_,min_,sec_);
        lcd_puts(Show_lcd);
        delay_ms(100);
    };
}
```

ประวัติผู้ดำเนินโครงงาน

ชื่อ นางสาวตรีระนุช สังข์บัวแก้ว

ภูมิลำเนา 204/62 หมู่ 8 ต.ท่าทอง อ.เมือง จ.พิษณุโลก

ประวัติการศึกษา

- จบระดับมัธยมศึกษาจากโรงเรียนพุทธชินราชพิทยา
- ปัจจุบันกำลังศึกษาระดับปริญญาตรี ชั้นปีที่ 8

สาขาวิชาวิศวกรรมไฟฟ้า คณะวิศวกรรมศาสตร์

มหาวิทยาลัยนเรศวร

E-mail: redgirl_blueman@hotmail.com