สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ระบบบำบัดน้ำเสีย

WASTE WATER TREATMENT SYSTEM

โดย

นายเวทประสิทธิ์ บุญทอง

นายสนธยา กลิ่นนาค

นายอภิรัตน์ มลเทียร

ปภ.
ว8995
2550

เลขหมู่.....
เลขทะเบียน 83297
วัน,เดือน,ปี 11 ส.ค. 2551

b 11965725
i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบบำบัดน้ำเสีย

WASTE WATER TREATMENT SYSTEM

โดย

นายเวทประสิทธิ์	บุญทอง	48015034
นายสนธยา	กลิ่นนาค	48015037
นายอภิรัตน์	มลเทียร	48015043

อาจารย์ที่ปรึกษา

ผศ. สุรพล บุญจันทร์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2550

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง **ระบบบำบัดน้ำเสีย**

WASTE WATER TREATMENT SYSTEM

ผู้จัดทำ

1. **นายเวทประสิทธิ์ บุญทอง 48015034**
2. **นายสนธยา กลิ่นนาค 48015037**
3. **นายอภิรัตน์ มลเทียร 48015043**

..... อาจารย์ที่ปรึกษา

(ผศ. สุรพล บุญจันทร์)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบบำบัดน้ำเสีย

WASTE WATER TREATMENT SYSTEM

โดย นายเวทประสิทธิ์ บุญทอง 48015034
นายสนธยา กลิ่นนาค 48015037
นายอภิรัตน์ มลเทียร 48015043

อาจารย์ที่ปรึกษา ผศ. สุรพล บุญจันทร์

บทคัดย่อ

โครงงานนี้เป็นการศึกษาชุดควบคุมค่าออกซิเจนในน้ำเพื่อใช้ควบคุมปริมาณออกซิเจนของน้ำให้อยู่ในปริมาณที่เหมาะสม โดยส่งสัญญาณควบคุมผ่านอุปกรณ์ไร้สายสั่งการให้ปั๊มออกซิเจนทำงานเพื่อเพิ่มค่าออกซิเจนของน้ำ และจะหยุดทำงานเมื่อปริมาณออกซิเจนอยู่ในระดับที่ตั้งไว้อย่างเหมาะสม เพื่อนำไปใช้งานได้จริงในอุตสาหกรรมการเกษตร

ABSTRACT

This project is education an oxygen value. It can controls set in a water, to control for an appropriate quantity of oxygen value in a water, to transfer control signal via wireless component, Then it command the oxygen pump to increase an oxygen value in a water.When an oxygen value in suitable level.This system will finish.In this application use for agriculture industry.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลงได้ด้วยดี เนื่องด้วยได้รับความอนุเคราะห์และได้รับคำปรึกษาด้วยดี จาก ผศ.สุรพล บุญจันทร์ และขอขอบคุณผู้ให้การสนับสนุนและคำปรึกษาในส่วนของเนื้อหาของวิชาต่างๆ อีกทั้งในส่วนของเนื้อหาที่สำคัญในปริญญานิพนธ์ของรุ่นพี่ เพื่อเป็นฐานอ้างอิงในการออกแบบและการประยุกต์ที่เกียวกับผลงานชิ้นนี้ และขอขอบคุณรุ่นพี่และเพื่อนๆร่วมรุ่นทุกคนที่คอยให้กำลังใจและให้คำปรึกษาในทุกๆ ด้านด้วยดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้า
บทที่ 1 บทนำ	1
1.1 ความเป็นมาและความสำคัญของโครงงาน	1
1.2 วัตถุประสงค์ของโครงงาน	1
1.3 ขอบเขตของโครงงาน	1
1.4 ประโยชน์ที่คาดว่าจะได้รับ	1
บทที่ 2 ทฤษฎีและหลักการที่เกี่ยวข้อง	2
2.1 คุณสมบัติออกซิเจนของน้ำ	2
2.1.1ปริมาณออกซิเจนที่ละลายในน้ำ	5
2.2 ตัวตรวจวัดค่าออกซิเจนในน้ำ	5
2.3 วงจรแปลงสัญญาณอนาล็อกเป็นดิจิตอล	6
2.3.1 วงจรแปลงสัญญาณอนาล็อกเป็นดิจิตอลใช้ไอซีเบอร์ ADC0832	10
2.4 ประเภทการรับและส่งสัญญาณด้วยคลื่นวิทยุ	10
2.4.1 การส่งแบบทางเดียว (Simplex Mode)	10
2.4.2 การรับ-ส่งสองทางแต่สลับกันส่ง (Half Duplex Mode)	11
2.4.3 การรับ-ส่งสองทางแบบสมบูรณ์ (Full Duplex)	11
2.5 วงจรขยายสัญญาณ	12
2.6 คู่มือการใช้งาน TRW-2.4GHz	13
2.6.1 ShockBurst	13
2.6.2 หลักการของ ShockBurst	13
2.6.3 ShockBurst ขณะทำการส่ง	14
2.6.4 ShockBurst ขณะทำการรับ	14
2.6.5 DuoCeiver โหมดรับข้อมูล 2 ช่องทาง	15
2.6.6 การตั้งค่าให้กับอุปกรณ์	16
2.6.7 Configuration Word Overviews	17
2.6.8 รายละเอียดของ Configuration Word	17
2.6.9 การตั้งค่าสำหรับการทำงานแบบ ShockBurst	18
2.6.10 การตั้งค่าสำหรับการทำงานแบบ Direct	20
2.6.11 คำอธิบาย DATA package	22
2.6.12 Timing Data ที่สำคัญ	23
2.6.13 Configuration mode timing	25
2.6.14 ShockBurst mode Timing	26
2.6.15 Output Power adjustment	27

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

	หน้า
2.7 ไมโครคอนโทรลเลอร์ MCS – 51 แบบแฟลช (Flash)	27
2.7.1 คุณสมบัติของไมโครคอนโทรลเลอร์ตระกูล MCS – 51 อนุกรม P89LV51RD2BN	28
2.7.2 การจัดขาของไมโครคอนโทรลเลอร์ MCS-51	29
2.7.3 การอ่านค่าลอจิกจากพอร์ต	31
2.7.4 จังหวะการทำงานของไมโครคอนโทรลเลอร์ MCS-51	31
2.7.5 โครงสร้างและการทำงานของพอร์ต	32
2.7.6 การใช้งานเป็นพอร์ตอินพุต	32
2.7.7 การใช้งานเป็นพอร์ตเอาต์พุต	32
2.8 โมดูลแอลซีดี (Module LCD)	32
2.8.1 โครงสร้างภายในของตัวควบคุมโมดูลแอลซีดี	33
2.8.2 โมดูลแอลซีดี ขนาด 16 ตัวอักษร 1 บรรทัด (LCD 16×1)	33
2.8.3 คำสั่งควบคุมโมดูลแอลซีดี	34
2.8.4 การเขียนคำสั่งและข้อมูลให้แก่โมดูลแอลซีดี	38
2.8.5 จังหวะการทำงานของแอลซีดีโมดูล	38
2.9 การใช้งานพอร์ตอนุกรม RS232	38
2.9.1 การทำงานของขาสัญญาณ DB9	40
2.9.2 ระดับสัญญาณของ RS232	40
2.10 ออฟโต้ไทรแอค	43
บทที่ 3 การออกแบบวงจร	45
3.1 การอกแบบโครงงาน	45
3.2 การอกแบบวงจรทางด้าน Slave	46
3.2.1 การเชื่อมต่อไมโครคอนโทรลเลอร์กับตัวไอซีแปลงอนาล็อกเป็นดิจิตอล	46
3.2.2 การเชื่อมต่อไมโครคอนโทรลเลอร์กับ วงจรควบคุมปั๊ม	47
3.2.3 การเชื่อมต่อของไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz ทางด้าน Slave	49
3.3การอออกแบบวงจรทางด้าน Master	51
3.3.1 การเชื่อมต่อไมโครคอนโทรลเลอร์กับดัวแอลซีดีโมดูล	51
3.3.2 การเชื่อมต่อไมโครคอนโทรลเลอร์กับวงจร RS 232	51
3.3.3 การเชื่อมต่อไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz	53
บทที่ 4 ผลการทดลอง	55
4.1 การทดลองวัดค่าแรงดันไฟฟ้าผ่านตัวไอซีแปลงสัญญาณอนาล็อกเป็นดิจิตอล	55
4.1.1 วัตถุประสงค์	55
4.1.2 วิธีการทดลอง	55

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

4.2 การวัดเปรียบเทียบค่า Dissolved Oxygen Sensor KDS25B กับเครื่อง Dissolved Oxygen meter 57

4.2.1 วัตถุประสงค์ในการทดลอง 57

4.2.2 ขั้นตอนการทดลอง 57

4.2.3 ผลการทดลอง 57

4.3 การทดลองแสดงค่าปริมาณออกซิเจนผ่านทางโปรแกรม Visual basic 59

4.3.1 วัตถุประสงค์ในการทดลอง 59

4.3.2 ขั้นตอนการทดลอง 59

4.3.3 ผลการทดลอง 60

บทที่ 5 บทสรุปและวิจารณ์ 63

5.1 สรุปผลการทดลอง 63

5.1.1 สรุปผลการทดลองที่ 4.1 63

5.1.2 สรุปผลการทดลองที่ 4.2 63

5.1.3 สรุปผลการทดลองที่ 4.3 63

5.2 ปัญหาและแนวทางแก้ไข 64

5.3 แนวทางการพัฒนาโครงงาน 64

กิตติกรรมประกาศ

บรรณานุกรม

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

หน้า

ตารางที่ 2.1 แสดง Configuration Word 17

ตารางที่ 2.2 แสดง Configuration data word 18

ตารางที่ 2.3 แสดงการตั้งค่า PLL 19

ตารางที่ 2.4 แสดงจำนวนบิทของ payload 19

ตารางที่ 2.5 Address ของตัวรับที่ 2 และตัวรับที่ 1 19

ตารางที่ 2.6 แสดงจำนวนบิทสำรองของ RX Address + CRC 20

ตารางที่ 2.7 แสดงการตั้งค่าการทำงานของ RF 20

ตารางที่ 2.8 แสดงการตั้งค่าของ Crystal 21

ตารางที่ 2.9 แสดงการตั้งค่าของ RF output power 21

ตารางที่ 2.10 แสดงการตั้งค่าช่องสัญญาณความถี่ และ RX/TX 21

ตารางที่ 2.11 อธิบาย Data package 22

ตารางที่ 2.12 Timing data 22

ตารางที่ 2.13 แสดง การปรับแต่ง Output Power 27

ตารางที่ 2.14 แสดงความสัมพันธ์ในการทำงานของ RS, RW และ E ของโมดูลแอลซีดีแบบอักขระ 34

ตารางที่ 2.15 ตารางแสดงรายละเอียดของรูปแบบข้อมูลคำสั่ง 35

ตารางที่ 2.16 ตารางคำสั่งควบคุมการแสดงผลของแอลซีดี 35

ตารางที่ 2.17 แสดงคำสั่งควบคุมการเลื่อนเคอร์เซอร์และข้อมูลตัวอักษร 36

ตารางที่ 2.18 แสดงลักษณะทิศทางการเลื่อนของเคอร์เซอร์ 36

ตารางที่ 2.19 แสดงรูปแบบคำสั่งที่กำหนดฟังก์ชันการทำงานของแอลซีดี 36

ตารางที่ 2.20 แสดงรูปแบบข้อมูลคำสั่งในการอ่านแฟลก Busy และแอดเดรส 37

ตารางที่ 2.21 การแสดงขาของ DB9 39

ตารางที่ 2.22 อัตราขนาดสูงสุดของออฟโต้ไทรแอค 43

ตารางที่ 2.23 แสดงคุณลักษณะต่างๆของตัวออฟโต้ไทรแอค 44

ตารางที่ 4.1 ผลการทดลองการวัดเปรียบเทียบค่า Dissolved Oxygen Sensor KDS25B กับเครื่อง Dissolved Oxygen meter 57

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

	หน้า
รูปที่2.1 ตัวตรวจวัดออกซิเจนในน้ำรุ่น KDS25B	6
รูปที่ 2.2 การพิจารณา Quantization error	6
รูปที่ 2.3 ระบบการได้มาของข้อมูลคอมพิวเตอร์	8
รูปที่ 2.4 (a)แปลงสัญญาณอนาลอกเป็นดิจิตอล; (b)การ reconstructing ให้กับสัญญาณ.	9
รูปที่ 2.5 แสดงกาจัดขาของไอซีเบอร์ ADC0832	10
รูปที่ 2.6 Timing Diagram ของไอซีเบอร์ ADC0832	10
รูปที่ 2.7 การส่งวิทยุแบบทางเดียว	11
รูปที่ 2.8 การรับ-ส่งสัญญาณวิทยุแบบสองทางแต่สลับกันพูด	11
รูปที่ 2.9 การรับส่งวิทยุสองทางแบบสมบูรณ์	12
รูปที่ 2.10 ไอซี LM 358	13
รูปที่ 2.11 แสดงการทำงานของ ShockBurst	13
รูปที่ 2.12 กระแสที่ใช้ในการทำงานขณะใช้ ShockBurst และขณะที่ไม่ได้ใช้	14
รูปที่ 2.13 การรับข้อมูล 2 ช่องสัญญาณของ TRW-2.4GHz	15
รูปที่ 2.14 การรับข้อมูล 2 ช่องสัญญาณพร้อมกัน	16
รูปที่ 2.15 ส่วนประกอบของ Data package	16
รูปที่ 2.16 DATA package	22
รูปที่ 2.17 แสดง Timing diagram ของ TRW-2.4GHz ขณะอยู่ในโหมด stand-by	24
รูปที่ 2.18 ค่าแรงดัน VDD ของ active mode	24
รูปที่ 2.19 Timing diagram สำหรับการตั้งค่าการทำงาน	25
รูปที่ 2.20 Timing ของ ShockBurst TX	26
รูปที่ 2.21 Timing ของ ShockBurst RX	27
รูปที่ 2.22 การจัดขามาตรฐานของไมโครคอนโทรเลอร์ MCS-51 P89LV51RD2BN	29
รูปที่ 2.23 รูปร่างและการจัดขาโมดูลแอลซีดีแบบอักขระ	34
รูปที่ 2.24 การเชื่อมต่อ COMPUTER กับ MCU BOARD โดย RS232	38
รูปที่ 2.25 พอร์ตอนุกรมของ DB9	39
รูปที่ 2.26 การเชื่อมต่ออุปกรณ์อุกรณ์ภายนอกเข้ากับคอมพิวเตอร์ด้วยสาย DB9	40
รูปที่ 2.27 ระดับสัญญาณของ RS232C และระดับสัญญาณของ TTL	40
รูปที่ 2.28 Diagram ของ Clock และ Data ของการสื่อสารแบบซิงโครนัส	41
รูปที่ 2.29 รูปแบบข้อมูลแบบอะซิงโครนัส	42
รูปที่ 3.1 Block Diagram ของโครงงานทั้งหมด	45
รูปที่ 3.2 การเชื่อมต่อไมโครคอนโทรลเลอร์กับตัวไอซีแปลงอนาล็อกเป็นดิจิตอล	47
รูปที่ 3.3 การเชื่อมต่อของไมโครคอนโทรลเลอร์กับ วงจรควบคุมปั๊ม	48

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ (ต่อ)

หน้า

รูปที่ 3.4 การเชื่อมต่อของไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz ทางด้าน Slave 49

รูปที่ 3.5 รูปแบบผังงานแสดงการเขียนโปรแกรมควบคุมโมดูลความถี่ TRW-2.4 GHz ทางด้าน Slave 50

รูปที่ 3.6 การเชื่อมต่อไมโครคอนโทรลเลอร์กับตัวแอลซีดีโมดูล 51

รูปที่ 3.7 การเชื่อมต่อไมโครคอนโทรลเลอร์กับวงจร RS-232 52

รูปที่ 3.8 การเชื่อมต่อไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz ทางด้าน Master 53

รูปที่ 3.9 รูปแบบผังงานแสดงการเขียนโปรแกรมควบคุมโมดูลความถี่ TRW-2.4 GHz ทางด้าน Master 54

รูปที่ 4.1 การทดลองวัดค่าแรงดันไฟฟ้าผ่านตัวไอซีแปลงสัญญาณอนาล็อกเป็นดิจิตอล 55

รูปที่ 4.2 ค่าผลการทดองที่ 4.1 56

รูปที่ 4.3 ขาสัญญาณ CLK และ Data Out 56

รูปที่ 4.4 เครื่อง Dissolved Oxygen meter 58

รูปที่ 4.5 สาร Sodium Sulphite 58

รูปที่ 4.6วัดเปรียบเทียบค่า Dissolved Oxygen Sensor KDS25B กับเครื่อง Dissolved Oxygen meter 58

รูปที่ 4.7 การเปิด Port ที่แสดงบนหน้าจอ Visual Basic 59

รูปที่ 4.8 การตั้งค่า Set point 60

รูปที่ 4.9 ปริมาณออกซิเจนในน้ำ ในขณะที่ปั๊มไม่ทำงาน ทาง Visual Basic 60

รูปที่ 4.10 ปริมาณออกซิเจนในน้ำ ในขณะที่ปั๊มไม่ทำงาน ทาง LCD 61

รูปที่ 4.11 ปั๊มไม่ทำงานเมื่อออกซิเจนสูงค่ากว่าค่าการทำงาน 61

รูปที่ 4.12 ปริมาณออกซิเจนในน้ำ ในขณะที่ปั๊มไม่ทำงาน ทาง Visual Basic 62

รูปที่ 4.13 ปริมาณออกซิเจนในน้ำ ในขณะที่ปั๊มทำงาน ทาง LCD 62

รูปที่ 4.14 ปั๊มทำงานเมื่อออกซิเจนต่ำกว่าที่ตั้งไว้ 62

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงงาน

ในโลกยุคปัจจุบันเทคโนโลยีได้เข้ามามีบทบาทในการดำรงชีวิตในแต่ละวันของมนุษย์ จึงมีการคิดค้นหาวิธีการที่จะประยุกต์เทคโนโลยีเหล่านั้นมาสร้างความสะดวกสบายในแต่ละวัน ซึ่งก็เป็นที่รู้จักกันว่าอาชีพหลักของประชาชนในประเทศก็คือ การประกอบอาชีพเกษตรกรรม อันได้แก่ การเพาะปลูกเลี้ยงสัตว์ โดยเฉพาะการเลี้ยงสัตว์น้ำนั้น องค์ประกอบสำคัญก็คงจะหนีไม่พ้นน้ำ ซึ่งน้ำที่ใช้ในการอยู่อาศัยของสัตว์น้ำ ไม่ว่าจะเป็นกุ้ง หอย ปู ปลา สัตว์เหล่านี้มีความต้องการออกซิเจนในการดำรงชีวิตที่ไม่เท่ากัน จึงได้มีการคิดค้นอุปกรณ์อำนวยความสะดวกขึ้นมาชุดหนึ่ง ซึ่งเป็นการประยุกต์เทคโนโลยีในด้านการสื่อสารไร้สาย มาช่วยแก้ไขบำบัดน้ำเสีย โดยจะมีการรับส่งข้อมูลผ่านช่องสัญญาณ(อากาศ) ข้อมูลที่ใช้ในการส่งไปยังตัวรับคือค่าออกซิเจน ที่ต่ำกว่ามาตรฐาน ทำให้ลดอัตราการตายของสัตว์น้ำลง ผลที่ตามมาคือช่วยลดภาวะของเกษตรกรลงด้วยส่วนหนึ่ง ซึ่งถ้าสัตว์น้ำตาย ลงในปริมาณที่มาก ก็จะทำให้เกษตรกรขาดทุนในปริมาณที่มาก

1.2 วัตถุประสงค์ของโครงงาน

1.2.1 เพื่อควบคุมปริมาณออกซิเจนในน้ำให้อยู่ในระดับที่ต้องการ

1.2.2 เพื่อจำลองระบบบำบัดน้ำเสียแบบไร้สาย

1.2.3 เพื่อทำการออกแบบวงจรการทำงานของระบบน้ำเสีย

1.3 ขอบเขตของโครงงาน

1.3.1 สร้างวงจรเครื่องรับเครื่องส่ง

1.3.2 ขอบเขตอุณหภูมิอยู่ในช่วง 5-35 องศาเซลเซียส

1.3.3 ค่าแรงดันภายในน้ำ -2 ถึง 1.0 Kgf/cm^2 gage

1.3.4 สามารถกำหนดค่าออกซิเจนให้เหมาะสมกับความต้องการของสัตว์น้ำชนิดนั้นๆ

1.4 ประโยชน์ที่คาดว่าจะได้รับจากการทำโครงงาน

1.4.1 รู้และเข้าใจถึงหลักการของการรับส่งข้อมูลผ่านตัว TRW-2.4 GHz

1.4.2 รู้และเข้าใจถึงหลักการทำงานของไมโครคอนโทรลเลอร์

1.4.3 ได้เครื่องมือที่มีประสิทธิภาพในการเพิ่มค่าออกซิเจนให้กับน้ำ

1.4.4 สามารถพัฒนาเครื่องมือให้มีมูลค่าถูกลงในโอกาสต่อไปได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 คุณสมบัติออกซิเจนของน้ำ

การบำบัดน้ำเสีย น้ำเสีย ตาม พ.ร.บ. ส่งเสริมและรักษาคุณภาพ สิ่งแวดล้อมแห่งชาติ พ.ศ. 2550 หมายถึง ของเสียที่อยู่ในสภาพ เป็นของเหลวรวมทั้งมวลสาร ที่ปะปนหรือปนเปื้อน อยู่ในของเหลวนั้น แต่โดยทั่วไปนั้น หมายถึง น้ำที่เสื่อมคุณภาพ หรือมีคุณสมบัติ ที่เปลี่ยนไปจากเดิมตามธรรมชาติ มักจะผ่านการใช้งานมาแล้ว โดยมีมลสาร หรือสิ่งปฏิกูล ที่ละลายน้ำและไม่ละลายน้ำเจือปนอยู่ เช่น สารอนินทรีย์ สารอินทรีย์ สารเคมีที่เป็นพิษ สารที่ทำให้เกิดฟอง กรด ด่าง น้ำร้อน สารแขวนลอย สี และจุลินทรีย์ เป็นต้น จนไม่สามารถนำน้ำนั้น มาใช้ประโยชน์ได้ดีเท่าที่ควร สารอนินทรีย์ที่ย่อยสลายได้ เมื่อถูกทิ้งลงไปในแหล่งน้ำจะถูกจุลินทรีย์ในน้ำย่อยสลาย ในกระบวนการย่อยสลาย ของจุลินทรีย์ จะต้องใช้ออกซิเจนที่อยู่ในน้ำ หากมีสารอินทรีย์มาก ก็จะใช้ออกซิเจนมาก ทำให้ปริมาณออกซิเจนในน้ำ ลดลงไปเรื่อยๆ (น้ำสะอาดปกติจะมีออกซิเจน ละลายอยู่ประมาณ 7-8 มิลลิกรัมต่อลิตร ถ้าบังเอิญมีมากกว่านี้ ก็จะซึมออกไปในบรรยากาศ ถ้ามีน้อยกว่านี้ ออกซิเจนในบรรยากาศ ก็จะซึมเข้าไปในน้ำ การซึมจะเข้าไปได้เร็วหรือช้าขึ้นอยู่กับการกระเพื่อมของผิวน้ำ) เมื่อออกซิเจนลดลงจนเหลือน้อยอาจทำให้ปลาหรือสัตว์น้ำบางชนิด ไม่สามารถมีชีวิตอยู่ได้เพราะขาดออกซิเจน ยิ่งเมื่อออกซิเจนที่ละลายอยู่ในน้ำหมดไป จุลินทรีย์ไม่ใช้ออกซิเจนในการย่อยสลาย สารอนินทรีย์จะเข้ามาทำหน้าที่ย่อยสลายแทน ซึ่งทำให้เกิดก๊าซมีเทน ก๊าซไฮโดรเจนซัลไฟด์ หรือก๊าซไข่เน่าที่มีกลิ่นเหม็น และทำให้น้ำดำสกปรกเรียกกันติดปากว่า "น้ำเน่า" น้ำเสียที่ถูกทิ้งออกมาสู่แหล่งน้ำอื่นๆ นั้น จะมีอิทธิพลจนทำให้แหล่งน้ำต่างๆ เปลี่ยนแปลงไป หรือทำให้เน่าเสียทันทีหรือไม่ จะขึ้นอยู่กับชนิด และปริมาณของน้ำทิ้ง ที่ระบายลงไป สารต่างๆ ที่สามารถทำให้น้ำเน่าเสียได้ อาจแบ่งได้เป็นกลุ่มใหญ่ๆ ดังนี้ สารอินทรีย์ที่ละลายน้ำได้ เช่น น้ำแกง น้ำล้างจาน-ชาม น้ำเชื่อม น้ำหวาน ปัสสาวะ อุจจาระ เป็นต้น สารอินทรีย์ที่ไม่ละลายน้ำ เช่น กระดาษ พลาสติก โฟม เศษเนื้อ เศษผัก-ใบตอง กอสวะ เป็นต้น บางชนิดก็ลอยอยู่ผิวน้ำ ซึ่งทำให้ไม่น่าดู การย่อยสลายสารอินทรีย์เหล่านี้ อาจทำให้น้ำเน่าเสียได้ สารอนินทรีย์ที่ละลายน้ำได้ เช่น สารปรอท ตะกั่ว สังกะสี แคดเมียม สารเคมีพวกนี้ส่วนใหญ่ถูกทิ้งออกมา จากโรงงานอุตสาหกรรมต่างๆ ที่ไม่มีการบำบัดก่อนทิ้ง แม้จะมีปริมาณเพียงเล็กน้อย เมื่อเข้าสู่ร่างกายคน ก็สามารถ ทำให้เป็นมะเร็ง หรือมีผลกระทบต่อส่วนต่างๆ ของร่างกาย และทำให้เกิด การสะสมสารพิษเหล่านี้ ในสัตว์น้ำได้ สารอนินทรีย์ที่ไม่ละลายน้ำ เช่น ท่อนเหล็ก ท่อนไม้ ตะกรัน โลหะ และของแข็งชนิดต่างๆ เป็นต้น ซึ่งทำให้ไม่น่าดู หรือทำให้กีดขวางทางน้ำไหลได้ นอกจากสารดังกล่าวแล้ว เราอาจพบเห็นสาร ที่ก่อให้เกิดฟองตามแหล่งน้ำ ซึ่งเกิดจากสารซักฟอกต่างๆ หากมีมากอาจกีดกั้น ไม่ให้ออกซิเจน แทรกตัวเข้าไปในน้ำ บางชนิดได้ นอกจากนั้นน้ำร้อน จากการหล่อเย็น ของโรงงานอุตสาหกรรม เมื่อระบายสู่แหล่งน้ำ ก็อาจทำให้ออกซิเจนในน้ำลดลง ซึ่งจะมีผลกระทบ ต่อสิ่งมีชีวิตในน้ำ น้ำเสียเขาวัดกันอย่างไร ในวงสนทนา เรื่องน้ำดี-น้ำเสีย เราอาจได้ยินคำศัพท์ต่อไปนี้บ่อยๆ คือ - BOD (บีโอดี) ย่อมาจากคำว่า Biochemical Oxygen Demand หมายถึง ค่าของปริมาณออกซิเจน ที่จุลินทรีย์ต้องการใช้ ในการย่อยสลายอินทรีย์ ที่อยู่ในน้ำ การวิเคราะห์หาค่า BOD เป็นการวิเคราะห์เพื่อที่จะทราบ ปริมาณความสกปรก ของน้ำโดยทั่วไป เป็นการวัดปริมาณออกซิเจน ที่ถูกจุลินทรีย์ใช้หมด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไป ในเวลา 5 วัน ติดต่อกันในตู้ควบคุม อุณหภูมิที่ 20 องศาเซลเซียส ปัจจุบันนี้ รัฐบาลได้กำหนดมาตรฐานของน้ำทิ้งต่างๆ ไว้ แยกตามประเภทและขนาด ของแหล่งน้ำทิ้ง เช่น น้ำทิ้งที่ระบายออก จากโรงงานอุตสาหกรรม ต้องมีค่า BOD ไม่เกิน 20 มิลลิกรัมต่อลิตร ส่วนน้ำทิ้ง จากแหล่งชุมชน มีค่ามาตรฐานกำหนดไว้ หลายระดับ ขึ้นอยู่กับประเภท และขนาดของอาคาร ตามที่กฎหมายกำหนด - COD (ซีโอดี) ย่อมาจากคำว่า Chemical Oxygen Demand หมายถึง ค่าของปริมาณออกซิเจน ที่ต้องการใช้ ในการทำปฏิกิริยาทางเคมี กับสารอินทรีย์ที่อยู่ในน้ำ - DO (ดีโอ) ย่อมาจากคำว่า Dissolved Oxygen หมายถึงค่าของปริมาณออกซิเจน ที่ละลายอยู่ในน้ำ โดยปกติ ในน้ำบริสุทธิ์ที่อุณหภูมิ 20 องศาเซลเซียส จะมีออกซิเจน ละลายอยู่ประมาณ 5 มิลลิกรัมต่อลูกบาศก์เดซิเมตร ซึ่งเป็นจำนวนจำกัด ส่วนในแหล่งน้ำธรรมชาติที่สะอาด จนมีออกซิเจนละลาย อยู่ประมาณ 7-8 มิลลิกรัมต่อลูกบาศก์เดซิเมตร ถ้ามีค่าต่ำกว่านี้ออกซิเจนในอากาศ จะสามารถละลายเข้าไป ในน้ำได้อีก แต่การที่ออกซิเจน จะละลายน้ำได้ช้าหรือเร็วขึ้นอยู่กับการกระเพื่อมของผิวน้ำ - pH (ค่าพีเอช) เป็นค่าที่แสดงปริมาณ ความเข้มข้นของอนุภาคไฮโดรเจน (H+) ในน้ำ ในทางปฏิบัติค่าพีเอช แสดงความเป็นกรดหรือด่างของน้ำ กล่าวคือ น้ำที่มีความเป็นกรด จะมีค่าพีเอชน้อยกว่า 7 ถ้ามีความเป็นด่าง จะมีค่าพีเอชมากกว่า 7 และถ้ามีความเป็นกลาง จะมีค่าพีเอชเท่ากับ 7 ค่าพีเอชของน้ำ มีความสำคัญต่อสิ่งมีชีวิตในน้ำ และมีความสำคัญ ในการบำบัดน้ำเสีย ด้วยวิธีการทางเคมี ฟิสิกส์ ชีววิทยา ซึ่งจำเป็นต้องควบคุม ค่าพีเอชของน้ำทิ้งให้คงที่ หรือควบคุมให้อยู่ในช่วงที่จำกัดไว้ สำหรับมาตรฐานควบคุมการระบายน้ำทิ้ง จากอาคารบางประเภท และบางขนาด กำหนดค่าพีเอช ต้องมีค่าระหว่าง 5-9 เป็นต้น ค่าต่างๆ ข้างต้นนี้เป็นเพียงค่าที่ควรรู้ ไว้โดยทั่วไป แต่ในการวิเคราะห์เพื่อหาค่าความสกปรก ของน้ำเสียนั้น ยังมีการวัดค่าต่างๆ อีกหลายชนิด ซึ่งไม่ได้นำมากล่าวไว้ ณ ที่นี้ เช่น สารพิษต่างๆ อันได้แก่ สารเคมีกำจัดศัตรูพืช หรือสารโลหะหนักต่างๆ เช่น ตะกั่ว ปรอท แคดเมียม สารหนู เป็นต้น แหล่งกำเนิดน้ำเสีย น้ำเสียอาจเกิดขึ้นได้จากหลายสาเหตุ ซึ่งแต่ละสาเหตุ ล้วนมาจากการกระทำ ของคนเราแทบทั้งสิ้น ซึ่งหาพวกเราทั้งหลาย ไม่มีความสำนึก ในการรักษาแหล่งน้ำ หรือไม่มีกฎหมายคอยกำกับบังคับ เพื่อรักษาแหล่งน้ำไว้แล้ว ประเทศของเราอาจจะอุดมไปด้วย แหล่งน้ำเน่าเสียซึ่งนำมาใช้ประโยชน์อะไรไม่ได้ น้ำเสียที่เราพบเห็น กันอยู่ทั่วไป มีแหล่งกำเนิดมาจากที่ต่างๆ ดังนี้

1.น้ำเสียจากชุมชน เป็นน้ำเสียที่ระบาย ออกจากบ้านเรือน โรงแรม โรงพยาบาล ร้านอาหาร และตลาด เป็นต้น น้ำเสียเกิดจากการนำน้ำ มาใช้ประโยชน์ในกิจกรรมต่างๆ แล้วทิ้งลงสู่แหล่งน้ำ ซึ่งหากแหล่งน้ำนั้น สามารถฟอกตัวเองได้ ตามธรรมชาติก็ดีไป แต่ส่วนใหญ่ น้ำที่ถูกทิ้งมักจะประกอบไปด้วยสิ่งสกปรกหลายชนิด เช่น สารอินทรีย์และอนินทรีย์ ทั้งที่เป็นของแข็ง และสารละลาย นอกจากนั้นอาจมีเชื้อโรค และพยาธิปะปนออกมาอีกด้วย โดยน้ำเสียจากบ้านเรือน ที่พักอาศัยทั้งคอนโดมิเนียม หรือโรงแรม มักจะเป็นน้ำจากส้วม น้ำจากการใช้ในครัว น้ำเสียจากร้านอาหาร ส่วนใหญ่จะเป็นประเภทสารอินทรีย์ต่างๆ และไขมันจากอาหารที่ทิ้ง ส่วนน้ำเสียจากโรงพยาบาล ก็จะเป็นประเภทที่มีเชื้อโรคปะปนอยู่ด้วย เนื่องมาจากการขับถ่าย ของผู้ป่วย และจากโรงซักผ้า เครื่องใช้ของผู้ป่วย เป็นต้น

2.น้ำเสียจากการอุตสาหกรรม น้ำที่ถูกทิ้งมาจากโรงงานอุตสาหกรรมนั้น เป็นน้ำที่มาจากขบวนการต่างๆ ในโรงงาน เช่น น้ำจากขบวนการผลิตโดยตรง น้ำจากขบวนการล้างต่างๆ หรือน้ำจากการหล่อเย็น ซึ่งแต่ละโรงงาน ก็อาจจะมีปริมาณและชนิด ของสารเจือปนแตกต่างกันไป เช่น อุตสาหกรรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทางการเกษตร เช่น โรงงานแปรรูปผลผลิต จากการเกษตร โรงงานผลิตอาหาร ทั้งของคนและของสัตว์ น้ำทิ้งประเภทนี้ จะมีสารอินทรีย์มาก มีปริมาณความสกปรกสูง เมื่อทิ้งลงสู่แหล่งน้ำ จะทำให้ปริมาณออกซิเจนในน้ำลดลง บางโรงงานอาจมีเชื้อโรค ปะปนออกมาอีกต่างหาก อุตสาหกรรมแร่หรือโลหะต่างๆ เช่น โรงงานผลิตโลหะหรือโลหะผสม โรงงานแปรรูปโลหะ โรงถลุงแร่ โรงงานพวกนี้อาจมีน้ำทิ้งปริมาณไม่มากนัก แต่จะมีสารพิษจำพวกโลหะหนัก ปะปนออกมามาก อุตสาหกรรมเคมีภัณฑ์ เช่น โรงงานผลิตสารเคมี โรงงานผลิตภัณฑ์ปิโตรเลียม โรงงานผลิตยาหรือปุ๋ย โรงงานผลิตกระดาษ ผลิตยาง ผลิตสี เป็นต้น น้ำทิ้งจากโรงงานเหล่านี้ จะมีค่าความสกปรกสูง บางชนิดจะมี ค่าความเป็นกรดเป็นด่างสูง และอาจมีสารพิษ ปะปนออกมากับน้ำทิ้ง บางชนิดอาจทำให้สี รส หรือกลิ่นของน้ำ เปลี่ยนแปลงไป อุตสาหกรรมเครื่องกล และเครื่องไฟฟ้า เช่น โรงงานผลิตเครื่องจักรเครื่องยนต์ โรงงานผลิตอิเลคทรอนิค เป็นต้น น้ำทิ้งจากโรงงานเหล่านี้ ส่วนใหญ่เป็นน้ำ ที่ใช้ล้างทำความสะอาด ซึ่งอาจทำให้มีโลหะหนัก และน้ำมันปะปนออกมาด้วย อุตสาหกรรมสิ่งทอ เช่น โรงงานผลิตเส้นใย โรงงานทอผ้า ย้อมผ้า พิมพ์ผ้า น้ำทิ้งส่วนใหญ่ เกิดจากการฟอกย้อมสี ซึ่งมีการใช้สารเคมี ทำให้น้ำเปลี่ยนสี เป็นการทำลายสภาพแหล่งน้ำ และอาจมีโลหะหนักปะปนอยู่ด้วย

3.น้ำเสียจากการเกษตร เกษตรกรรมนั้นเป็นอาชีพหลัก ของคนส่วนใหญ่ของประเทศ ปัจจุบันได้มีการนำเอา เทคโนโลยีที่ทันสมัย มาช่วยเพื่อเพิ่มผลผลิต และนับวันจะสูงขึ้น น้ำเสียที่ระบายออกมา จากพื้นที่การเกษตรนั้น นอกจากจะมีสารต่างๆ ปะปนออกมาแล้ว ยังเป็นตัวการที่ทำให้แหล่งน้ำต่างๆ เน่าเสียอีกด้วย น้ำเสียจากการเกษตร มาจากแหล่งใหญ่ๆ คือ น้ำเสียจากการเพาะปลูก มาจากน้ำใช้แล้ว จากพื้นที่เพาะปลูก ซึ่งประกอบด้วยปุ๋ยส่วนเกิน ซึ่งจะส่งผลให้ พืชที่ขึ้นในแหล่งน้ำ เจริญเติบโตอย่างรวดเร็ว เช่น สาหร่ายต่างๆ และผักตบชวา นอกจากนั้น อาจมีสารเคมี ที่ใช้กำจัดศัตรูพืช ซึ่งหามีความเข้มข้น อาจทำให้สัตว์น้ำต่างๆ ตายได้ น้ำเสียจากกิจกรรมปศุสัตว์ เช่น ฟาร์มเลี้ยงสัตว์ โดยเฉพาะอย่างยิ่ง ฟาร์มสุกร ซึ่งส่วนใหญ่ เกิดจากมูลของสัตว์ เศษอาหารที่เหลือ และน้ำที่ใช้ล้างคอก โดยจะมีค่าความสกปรกสูง และปริมาณมาก น้ำเสียจากการเพาะเลี้ยงสัตว์น้ำ การเพาะเลี้ยงสัตว์น้ำ ส่วนใหญ่มักมีพื้นที่ ติดกับแหล่งน้ำต่างๆ เช่นคลอง แม่น้ำ ทะเล เนื่องจากเป็นธุรกิจที่ได้ผลเร็ว จึงนิยมทำกันมาก น้ำที่ระบายทิ้งจึงมีอินทรีย์วัตถุต่างๆ ปะปนเป็นจำนวนมาก โดยเกิดจากอาหารที่ใช้เลี้ยง และของเสีย ที่ถ่ายออกมาจากสัตว์น้ำเหล่านั้น ซึ่งจะทำให้ค่าออกซิเจน ในน้ำลดค่าลงเรื่อยๆ

ขั้นตอนการบำบัดน้ำเสีย น้ำทิ้งที่มาจากแหล่งต่างๆ ตามที่กล่าวมาแล้วนั้น แต่ละประเภท จะมีสิ่งสกปรกต่างๆ ปะปนอยู่ ทั้งสารอินทรีย์และสารอนินทรีย์ การบำบัดน้ำทิ้งเหล่านั้น ให้กลับมีคุณภาพ ที่ไม่เป็นอันตรายก่อน ที่จะปล่อยลงไปสู่แหล่งน้ำอื่นๆ มีวิธีการอยู่หลายขั้นตอน ซึ่งพอสรุปได้ดังนี้

1.การบำบัดน้ำเสียขั้นต้น การบำบัดน้ำเสียขั้นต้น มีวัตถุประสงค์ เพื่อขจัดสารแขวนลอยต่างๆ หรือปรับสภาพน้ำเสีย ให้เหมาะสม เพื่อเตรียมไปบำบัดในขั้นที่ 2 ต่อไป น้ำเสียบางประเภท ที่มีความสกปรกไม่มากนัก เมื่อผ่านการบำบัดขั้นตอนนี้แล้ว ก็สามารถระบายน้ำทิ้งได้ทันที การบำบัดน้ำเสียขั้นต้นโดยทั่วไป มีวิธีการดังนี้ การปรับความเป็นกรดเป็นด่างให้ใกล้ความเป็นกลาง การกรอง การทำให้ตกตะกอน การแยกน้ำมันและไขมันออก การกักเก็บชั่วคราวเพื่อปรับสภาพน้ำเสียให้มีความสม่ำเสมอกัน ฯลฯ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.การบำบัดน้ำเสียขั้นที่ 2 เป็นขบวนการบำบัด ต่อจากขั้นตอนที่ 1 โดยใช้วิธีการทางชีวภาพ ซึ่งมีอยู่หลายระบบ ที่ใช้กันอยู่ในประเทศไทย ส่วนใหญ่จะมีดังนี้ ; ระบบบ่อแบบไม่เติมอากาศ ; ระบบบ่อแบบเติมอากาศ ; ระบบตะกอนเร่ง ; ระบบแผ่นฟิล์มชีวภาพ ฯลฯ ระบบต่างๆ เหล่านี้ เป็นที่นิยมใช้กันอยู่ในประเทศไทย โดยมีประสิทธิภาพ ในการบำบัด และข้อดีข้อเสียที่แตกต่างกันบ้าง ซึ่งในการนำไปใช้จะต้องศึกษาและเลือกใช้ให้เหมาะสม

2.1.1ปริมาณออกซิเจนที่ละลายในน้ำ

ระดับที่เหมาะสมต่อการเลี้ยงปลาควรมีค่ามากกว่า 5 มิลลิกรัม ต่อ ลิตร ถ้าออกซิเจนอยู่ในช่วง 2 – 5 มิลลิกรัม ต่อ ลิตร จะทำให้การเจริญเติบโตและการสืบพันธุ์ของปลาลดลง ส่วนใหญ่มักมีออกซิเจนละลายอยู่ในช่วง 5 – 7 มิลลิกรัม ต่อ ลิตร ปริมาณออกซิเจนที่ละลายในน้ำเรียกย่อว่า ดีโอ (DO) ค่าดีโอจะน้อยที่สุดในตอนเช้ามืด ถ้ามีปริมาณต่ำปลาจะมีอาการลอยหัวเพื่อขึ้นมาหายใจบนผิวน้ำ การเพิ่มดีโอ ในน้ำสามารถทำได้ดังนี้ ควบคุมค่าความโปร่งใสของน้ำให้อยู่ในช่วง 30 – 60 เซนติเมตร เปลี่ยนถ่ายน้ำแล้วใส่เกลือ 0.5 – 1 เปอร์เซ็นต์ ใช้เครื่องตีน้ำหรือทำให้น้ำได้สัมผัสกับอากาศเพื่อให้ออกซิเจนละลายในน้ำ ออกซิเจนจากอากาศจะละลายในน้ำได้ดีที่อุณหภูมิต่ำมากกว่าที่อุณหภูมิสูง ค่าออกซิเจนที่ละลายในน้ำได้สูงสุดที่สามารถสะสมไว้ในน้ำได้ในเขตอบอุ่นจะมีค่าประมาณ 20 มก./ลิตร (supersaturation) ซึ่งออกซิเจนส่วนเกินจะกลับไปยังชั้นบรรยากาศ หากปลาอยู่ในสภาวะที่ออกซิเจนสูงมากๆ ตั้งแต่ 15-20 มก./ลิตร จะทำให้ออกซิเจนแทรกเข้าไปตามเนื้อเยื่อบางๆ อาทิเช่นครีบ ได้ง่ายทำให้เกิดภาวะที่เรียกว่า gas bubble disease ซึ่งอาจทำให้ปลามีขนาดเล็กและตายได้

2.2 ตัวตรวจวัดค่าออกซิเจนภายในน้ำ

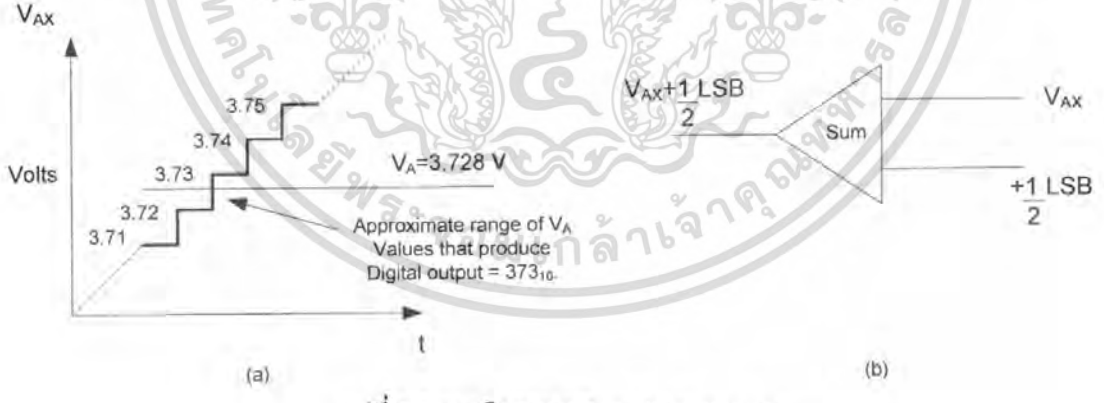
จะใช้ตัวตรวจวัดออกซิเจนในน้ำรุ่น KDS25B โดยจะมีหลักการคือ วัดค่าออกซิเจนน้ำแล้วให้ผลออกมาเป็นแรงดันไฟฟ้า โดยแรงดันไฟฟ้าเอาท์พุทที่ได้จะเป็นอนาล็อกอยู่ในช่วง 5.6 mV – 15.0 mV และเมื่อนำไปวัดค่าออกซิเจนในน้ำจะมีผลตอบสนอง 90% ภายในระยะ 10 นาที โดยสามารถทำงานได้อย่างเต็มที่ ที่อุณหภูมิของน้ำ 24-26 องศาเซลเซียส สำหรับเงื่อนไขการทำงานขอบเขตอุณหภูมิของน้ำจะต้องอยู่ในช่วง5-35 องศาเซลเซียส และแรงดันภายในน้ำ-0.2 ถึง1.0 kgf/cm2 gage (ที่ความลึกของระดับน้ำ=10cm) โดยการเก็บรักษาตัวตรวจวัดนี้จะต้องเก็บในแนวนอน แรงดันไฟฟ้าของตัวตรวจวัดนี้อาจจะเปลี่ยนแปลงเมื่อเกิดการสั่นหรือสะเทือนขึ้น โดยตัวตรวจวัดน้ำสามารถทนอุณหภูมิต่ำสุด-10องศาเซลเซียสสูงสุด 50 องศาเซลเซียสและสามารถทนแรงดัน ต่ำสุด 50 atm สูงสุด 2.5 atm ภายใต้ระยะเวลาการทำงาน 24 ชั่วโมง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่2.1 ตัวตรวจวัดออกซิเจนในน้ำรุ่น KDS25B

2.3วงจรแปลงสัญญาณอนาล็อกเป็นดิจิตอล

Quantization error หมายถึงความผิดพลาดที่เกิดจากการแบ่งระดับของอินพุทที่จะแปลงเป็นดิจิตอล เช่น อินพุท VA ที่ต้องการแปลงเป็นดิจิตอลมีค่าเท่ากับ 3.728V แต่ VAX ที่มาเปรียบเทียบ มีค่าเท่ากับ 3.73V (เลขไบนารี่ 0101110101) ค่าที่ผิดพลาดคือ 0.01V ดังนั้นค่าความผิดพลาดสูงสุดในการแปลงอนาล็อกเป็นดิจิตอลในที่นี้จะมีค่าเท่ากับ + 1 LSB หรือเท่ากับ step size ค่านี้เรียกว่า Quantization error เราสามารถดัดแปลงวงจรให้มี Quantization error เท่ากับ LSB ± 1 ได้โดยการบวกค่าของ VAX = + LSB



รูปที่ 2.2 การพิจารณา Quantization error

Accuracy หมายถึง ความถูกต้องของเครื่องมือวัด จะเป็นค่าความผิดพลาดที่ไม่ขึ้นอยู่กับ Resolution แต่จะขึ้นอยู่กับอุปกรณ์ที่ใช้ในวงจร เช่น Comparator Resistor DAC Supplyและ อื่นๆ ปกติจะบอกเป็นค่าที่เทียบกับ Full scale เช่น วงจรADC มีคุณสมบัติ Accuracy =0.001% of F.S

Conversion time (tc) หมายถึงเวลาในการแปลงสัญญาณจากอานาลอกเป็นดิจิตอลคือช่วงเวลาที่ VAX stepขึ้นจนมีค่าสูงกว่า VA สำหรับวงจร Digital Ramp ADC ค่า Conversion time สูงสุด tc (max) = 2n –1 clock cycles เมื่อ n = จำนวนบิตของดิจิตอลเอาท์พุท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Data Acquisition หมายถึงการแปลงสัญญาณจากอนาล็อกเป็นดิจิตอล และนำค่าดิจิตอลที่ได้เก็บเข้าในคอมพิวเตอร์ ข้อมูลที่เก็บไว้ในหน่วยความจำ อาจจะเป็นข้อมูลสัญญาณเสียง สัญญาณภาพหรือสัญญาณโวลท์เตจใดๆ การเก็บจะเป็นการเก็บค่าความสูงของระดับสัญญาณเป็นจุดๆในรูปแบบตัวเลขดิจิตอล

วิธีการนำข้อมูลเข้าเก็บในคอมพิวเตอร์สามารถอธิบายการทำงานได้ดังนี้

1. ไมโครคอมพิวเตอร์ส่งสัญญาณ Start มายังวงจร ADC

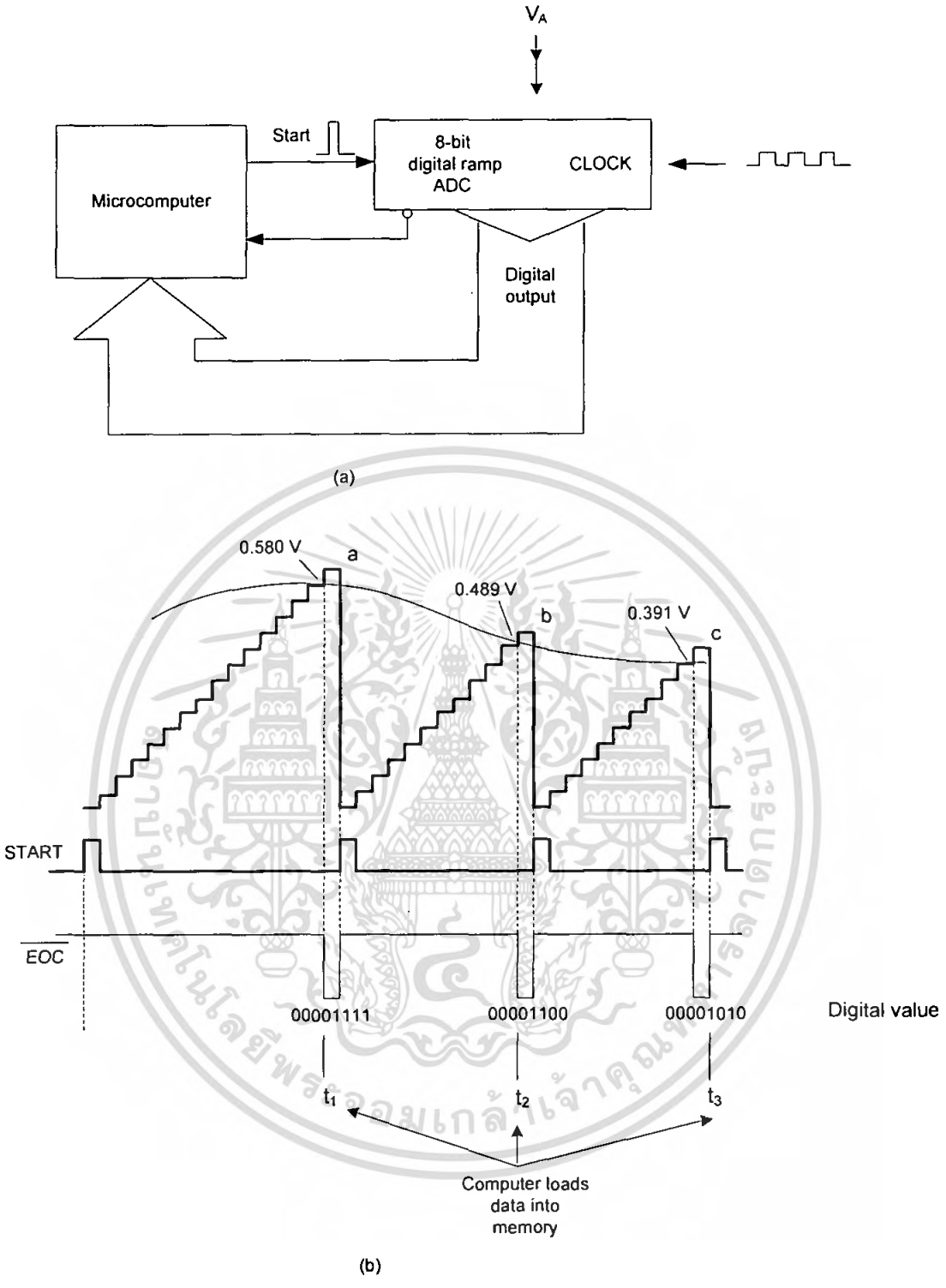
2. วงจร ADC จะเริ่มขบวนการแปลงสัญญาณ VA เป็นดิจิตอล โดย Clock จะทำให้ VAX step เพิ่มขึ้นเรื่อยๆ จนกระทั้ง VAX >VA จึงหยุด

3. เมื่อ VAX >VA สัญญาณ EOC จะตกเป็นโลจิก"0" เพื่อบอกให้ไมโครคอมพิวเตอร์ ว่าทำการแปลงสัญญาณเสร็จแล้ว

4. เมื่อไมโครคอมพิวเตอร์ได้รับสัญญาณEOC ไมโครคอมพิวเตอร์จะทำการอ่านข้อมูลจาก ADC เก็บในหน่วยความจำ ซึ่งเป็นการเก็บค่าความสูงสัญญาณอานาลอกที่จุด a

5. เริ่มทำการแปลงสัญญาณใหม่ ไมโครคอมพิวเตอร์จะทำการเก็บความสูงของสัญญาณที่จุด b , c ,..... ต่อไปเรื่อยๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

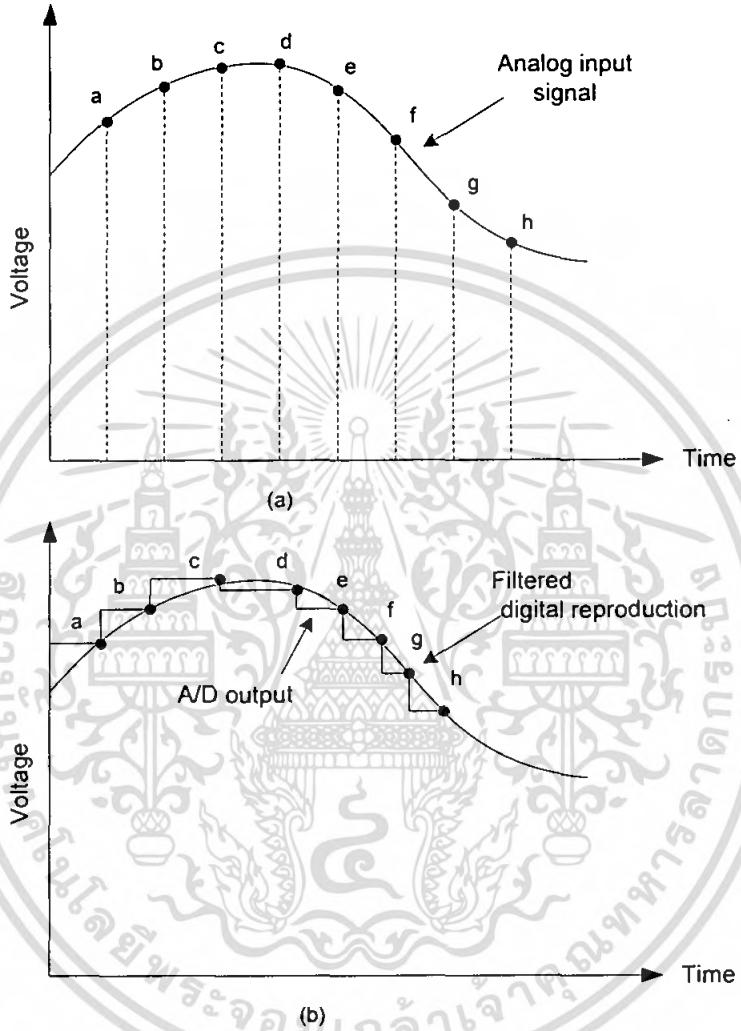


รูปที่ 2.3 ระบบการได้มาของข้อมูลคอมพิวเตอร์

รูปที่ 2.3 แสดงถึงการแปลงสัญญาณจากอนาล็อกเป็นดิจิตอล และนำค่าเก็บไว้ในหน่วยความจำคอมพิวเตอร์ ส่วนรูปที่ 2.4 เป็นการแสดงการนำข้อมูลดิจิตอลที่เก็บในคอมพิวเตอร์ กลับมาเป็นอนาล็อกอีกครั้งเรียกว่า Reconstructing จะทำแบบย้อนกลับคือนำสัญญาณตัวเลขดิจิตอล ให้สัญญาณ step ขึ้นแบบ VAX จะได้ความสูงจุด a , b ,c เท่าเดิม หรือจะทำการแปลงตัวเลขดิจิตอลเป็นอนาล็อกในช่วงเวลาที่เท่ากัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทุกช่วงดังรูปที่ 2.4(b) จากนั้นนำสัญญาณผ่าน Low Pass Filter กรองความถี่สูงออกจะได้สัญญาณเดิมกลับคืน ปัญหาของวงจร Digital Ramp ADC คือ จะเสียเวลามากในการแปลงสัญญาณจากอนาล็อกเป็นดิจิตอล และยังใช้เวลาในการแปลงสัญญาณไม่คงที่ ขึ้นอยู่กับขนาดของสัญญาณอนาล็อก จุดอนาล็อก a , b , c ไม่แน่นอน เมื่อเริ่มสัญญาณ Start แทนที่จะได้โวลท์เตจที่จุด Start (t_0) กลับได้ค่าที่จุดเวลาที่ t1 ดังรูปที่ 2.3

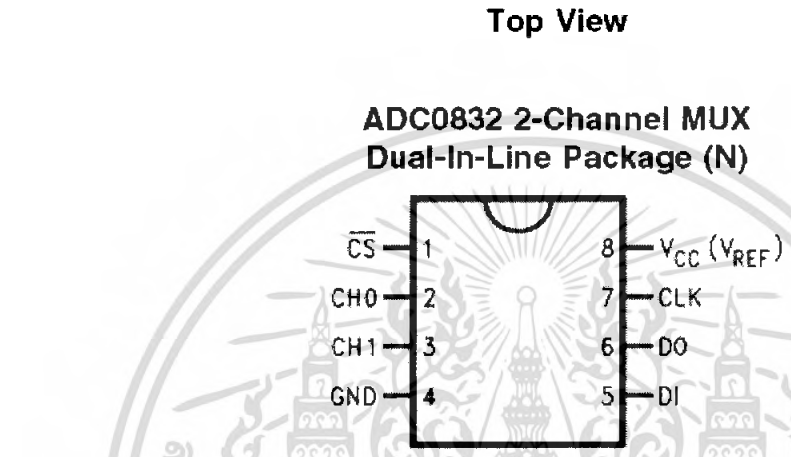


รูปที่ 2.4 (a)แปลงสัญญาณอนาลอกเป็นดิจิตอล; (b)การสร้างสัญญาณอนาลอกกลับคืน.

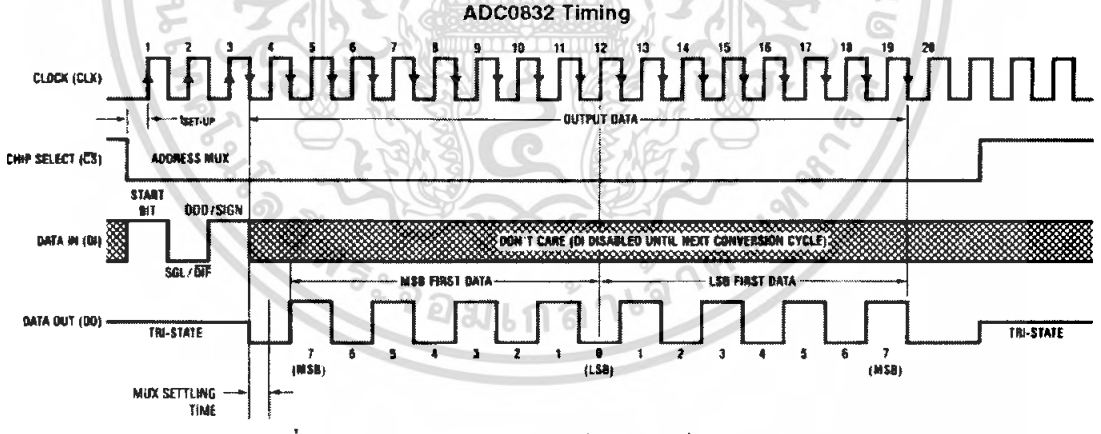
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.1 วงจรแปลงสัญญาณอนาล็อกเป็นดิจิตอลใช้ไอซีเบอร์ ADC0832

เป็นอุปกรณ์ที่แปลงจากอนาล็อกเป็นสัญญาณดิจิตอล 8 บิทโดยอินพุทสามารถทำการมัลติเพล็กซ์ได้สูงถึง 2 ช่องสัญญาณ สามารถทำการอินเตอร์เฟสร่วมกับไมโครโปรเซสเซอร์หรือชิพรีจิสเตอร์ที่ตามมาตรฐานทั่วไปได้ แรงดันไฟฟ้าอินพุทจะถูกเพิ่มขึ้นเมื่อโหมด common mode ในตัวไอซีไม่ทำงานและจะทำงานชดเชยเพิ่มค่าแรงดันไฟฟ้าอนาล็อกอินพุทที่เป็นศูนย์ แรงดันอ้างอิงนั้นสามารถปรับให้เหมาะสมกับการเข้ารหัส (encoding) เพื่อให้แรงดันอื่นๆที่มีค่าน้อยกว่านั้นสามารถทำการแปลงเป็นบิทได้จนครบ 8 บิท



รูปที่ 2.5 แสดงการจัดขาของไอซีเบอร์ ADC0832



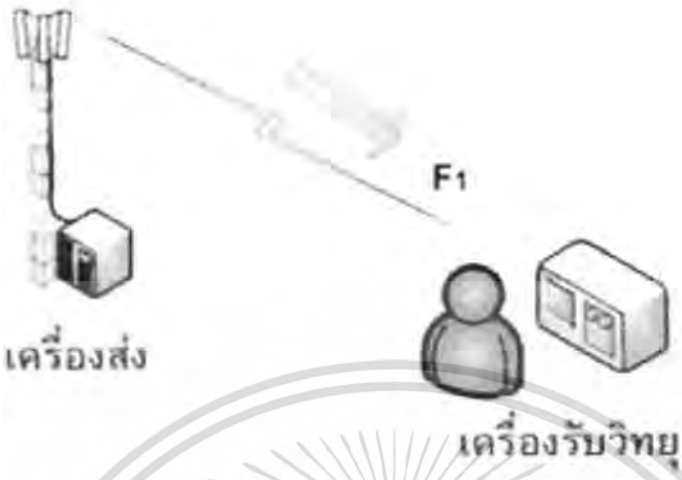
รูปที่ 2.6 Timing Diagram ของไอซีเบอร์ ADC0832

2.4 ประเภทการรับและส่งสัญญาณด้วยคลื่นวิทยุ

2.4.1 การส่งแบบทางเดียว (Simplex Mode) การส่งสัญญาณทางเดียวจะใช้ช่องสัญญาณเดียวในการส่งสัญญาณวิทยุ ในระบบนี้จะมีผู้ส่งผู้เดียว โดยสามารถส่งสัญญาณวิทยุออกไปได้โดยไม่มีการรบกวนจากผู้รับ ผู้รับก็ทำหน้าที่รับสัญญาณอย่างเดียว ไม่สามารถส่งสัญญาณวิทยุออกไปได้ วิธีนี้จะไม่ซับซ้อน มีเป้าหมายในการส่งข้อมูลออกไปทิศทางเดียว พบกันมากในระบบวิทยุกระจายเสียงที่จะมีสถานี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วิทยุส่งสัญญาณออกไป และผู้รับก็จะมีเครื่องรับวิทยุที่สามารถเลือกฟังเพลงจากสถานีช่องใดๆได้ตามอัธยาศัย



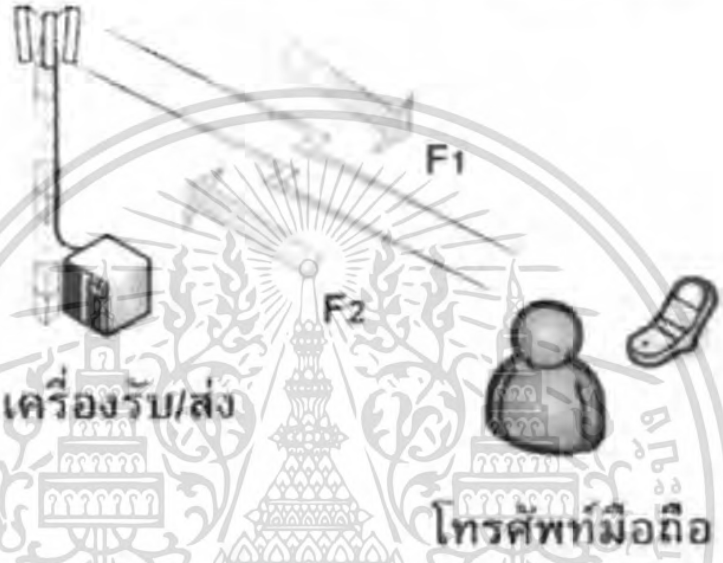
รูปที่ 2.7 การส่งวิทยุแบบทางเดียว

2.4.2 การรับ-ส่งสองทางแต่สลับกันส่ง (Half Duplex Mode) หากให้รับสัญญาณอย่างเดียวก็คงเป็นการสื่อสารทางเดียว ซึ่งผู้รับก็ไม่สามารถตอบกลับใดๆได้ เพียงแต่รับฟัง ดังนั้นจึงมีการปรับปรุงวิธีการติดต่อสื่อสารใหม่โดยผู้รับสามารถส่งสัญญาณกลับไปหาผู้ส่งได้ วิธีนี้เราเรียกว่า ดูเพล็กซ์โหมด (Duplex Mode) วิธีนี้ช่วยให้สามารถสื่อสารระหว่างบุคคลทั้งสองได้เข้าใจกันมากขึ้น เพราะต่างก็สามารถส่งและรับข้อความของฝ่ายตรงกันข้ามได้ และเพื่อให้การใช้ช่องความถี่ที่จำกัดให้มี ประสิทธิภาพจึงต้องมีการสลับกันส่งสัญญาณ โดยจะพูดได้คนละครั้งสลับกันไป เมื่อคุณส่งจะไม่สามารถรับสัญญาณได้ และเมื่อคุณรับจะไม่สามารถส่งสัญญาณได้ เหตุที่เป็นเช่นนี้เพราะเราใช้ช่องสัญญาณเดียวทั้งรับและส่ง หากมีการส่งสัญญาณพร้อมกัน คลื่นก็จะชนกันในอากาศกลายเป็นสัญญาณที่ไม่สามารถตีความได้

รูปที่ 2.8 การรับ-ส่งสัญญาณวิทยุแบบสองทางแต่สลับกันพูด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.3 การรับ-ส่งสองทางแบบสมบูรณ์ (Full Duplex) แม้ว่าการรับ-ส่งสองทางแบบฮาล์ฟดูเพล็กซ์จะสามารถทำให้เราส่งข้อมูลกลับไปหาผู้ส่งได้แต่ก็มีปัญหาคือ การพูดคุยด้วยวิธีนี้ไม่เป็นธรรมชาติ ซึ่งเราก็ต้องคอยจนกว่าผู้หนึ่งพูดจนจบก่อนจะส่งสัญญาณได้ แต่การสื่อสารทั่วๆไป ก็จะมีการพูดแทรกขึ้น ซึ่งวิธีเดิมจะไม่สามารถทำได้ จึงมีการปรับปรุงระบบสื่อสารให้ สามารถรับและส่งได้พร้อมๆกัน เราเรียกวิธีนี้ว่า ฟูลดูเพล็กซ์ (Full Duplex) วิธีนี้จะต้องใช้คลื่นความถี่สองช่องด้วยกัน โดยแต่ละช่องใช้สำหรับส่งข้อมูลไปยังอีกฝั่งหนึ่ง ข้อดีของวิธีนี้ทำให้การสื่อสารเป็นธรรมชาติมากขึ้น สามารถโต้ตอบกันได้ทันที แต่ก็มีข้อเสียคือ เปลืองช่องสัญญาณซึ่งต้องใช้ถึง 2 ช่องสัญญาณด้วยกัน



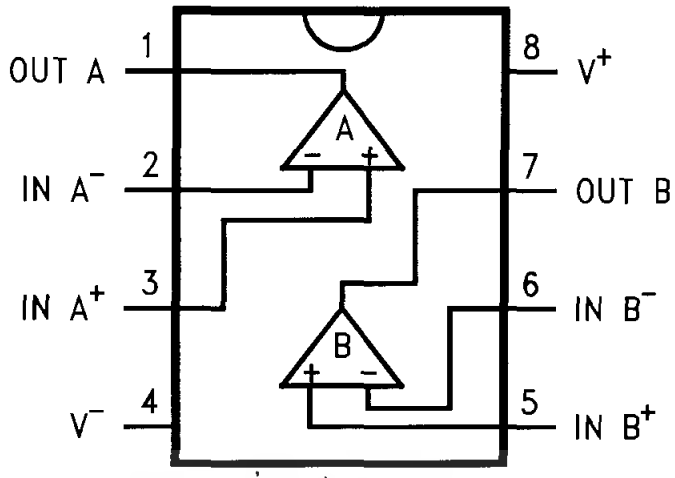
รูปที่ 2.9 การรับส่งวิทยุสองทางแบบสมบูรณ์

2.5 วงจรขยายสัญญาณ

ในโครงงานนี้จะมีวงจรขยายสัญญาณเพื่อที่จะขยายสัญญาณแรงดันจากตัวจับปริมาณออกซิเจนให้มีกำลังสูงขึ้นโดยจะใช้ ไอซีเบอร์ LM385 โดยไอซีเบอร์นี้จะมีคุณสมบัติดังนี้

- มีขนาดที่เหมาะสมกับชิพขนาด 8 บิท
- ความถี่ภายในสามารถชดเชยอัตราได้
- สามารถเพิ่มอัตราขยายแรงดันไฟฟ้ากระแสตรงได้ 100 dB
- ใช้กับแหล่งจ่ายแรงดันหลากหลาย แหล่งจ่ายเดี่ยว 3v-32 v แหล่งจ่ายแบบคู่ ±1.5 v-±16v
- กระแสไหลออกต่ำมาก (500 μA) ทำให้ไม่ต้องขึ้นอยู่กับแหล่งจ่ายแรงดัน
- แรงดันอินพุทแต่ละสาขาต่ำ :2 mV
- แรงดันอินพุทคอมมอนโหมดรวมอยู่กับกราวด์
- ย่านแรงดันอินพุทอ้างอิงเท่ากับแรงดันที่แหล่งจ่าย
- แรงดันเอาต์พุทเปลี่ยนแปลงได้กว้าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.10 ไอซี LM 358

2.6 คู่มือการใช้งาน TRW-2.4GHz

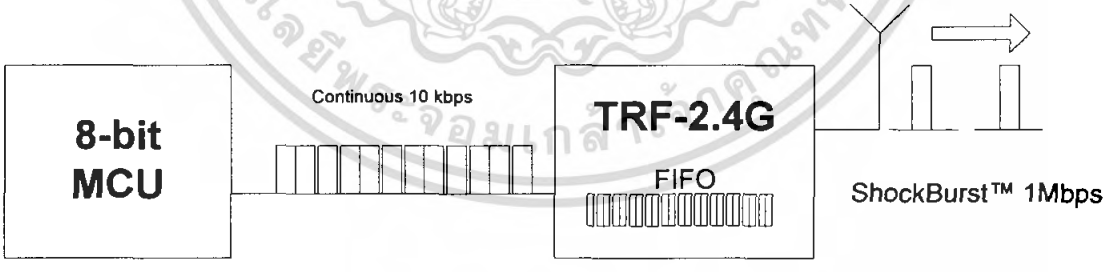
2.6.1 ShockBurst

เทคโนโลยี ShockBurst นั้น ใช้ระบบ FIFO (First-In First-Out) โดยเริ่มการทำงานในอัตราการรับส่งข้อมูลต่ำ จากนั้นจะส่งข้อมูลออกไปในอัตราการรับส่งข้อมูลที่สูงมาก ด้วยเหตุนี้ทำให้สามารถลดการใช้พลังงานลงไปได้อย่างมาก

เมื่อ TRW-2.4GHz ทำงานในโหมด ShockBurst จะทำให้ใช้งานได้ในอัตราการรับส่งข้อมูลสูงถึง (1 Mbps) ในย่านความถี่ 2.4 GHz โดยไม่จำเป็นต้องใช้ไมโครคอนโทรลเลอร์ที่มีความเร็วสูง

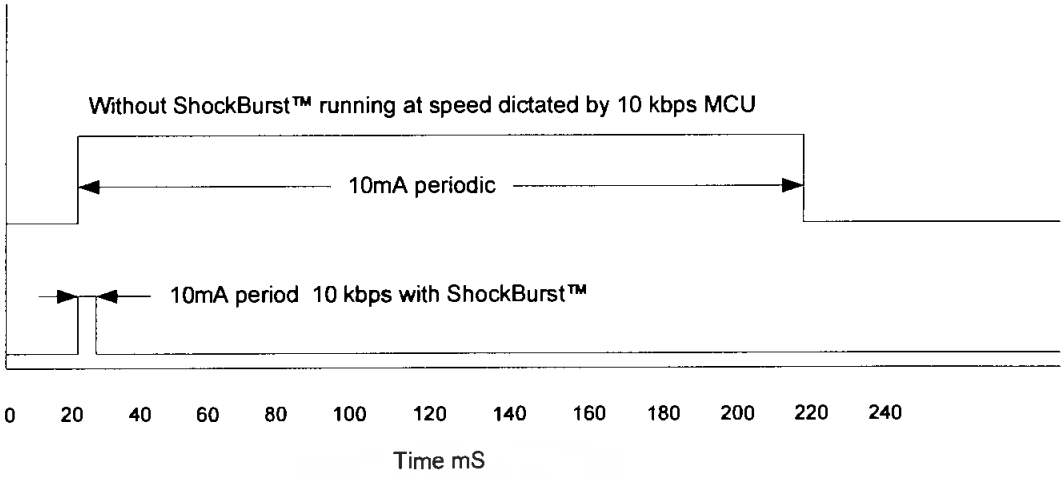
2.6.2 หลักการของ ShockBurst

เมื่อ TRW-2.4GHz ถูกตั้งค่าการทำงานในโหมด ShockBurst , การทำงานของ TX หรือ RX จะเป็นดังรูป (อัตราการรับส่งที่ 10 Kbps นั้นใช้สำหรับเป็นตัวอย่างเท่านั้น)



รูปที่ 2.11 แสดงการทำงานของ ShockBurst

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.12 กระแสที่ใช้ในการทำงานขณะใช้ ShockBurst และขณะที่ไม่ได้ใช้

2.6.3 ShockBurst ขณะทำการส่ง

ขาที่ใช้เชื่อมต่อกับไมโครคอนโทรลเลอร์: CE, CLK1, DATA

2.6.3.1 เมื่อไมโครคอนโทรลเลอร์ มีข้อมูลที่จะส่งออกไป ให้ตั้งค่า CE เป็น high จะทำให้ภายใน TRW-2.4GHz มีการประมวลผลข้อมูลเพื่อเตรียมการส่งข้อมูล

2.6.3.2 address ของการรับ (RX address) และ Payload data นั้นจะเริ่มการทำงานของ TRW-2.4GHz จากนั้นไมโครคอนโทรลเลอร์ทำการกำหนดค่าความเร็วให้ต่ำกว่า 1 Mbps (ในตัวอย่างคือ 10 Kbps)

2.6.3.3 ไมโครคอนโทรลเลอร์ตั้งค่า CE เป็น low ซึ่งเป็นการสั่งให้ TRW-2.4GHz เริ่มทำการส่งในโหมด ShockBurst

2.6.3.4 TRW-2.4GHz ในโหมด ShockBurst

- RF front end (ส่วนที่ป้องกันการเข้าถึงของข้อมูล) นั้นจะมีความสามารถเพิ่มขึ้น
- RF package นั้นจะครบสมบูรณ์ (preamble จะถูกเพิ่ม และ CRC จะถูกสร้างในขั้นตอนนี้)
- ข้อมูลจะถูกส่งออกไปด้วยความเร็วสูง (250 Kbps หรือ 1 Mbps ขึ้นอยู่กับการตั้งค่าของผู้ใช้งาน)
- TRW-2.4GHz จะเข้าสู่สภาวะ stand-by เมื่อทุกขั้นตอนเสร็จสิ้น

2.6.4 ShockBurst ขณะทำการรับ

ขาที่ใช้เชื่อมต่อกับไมโครคอนโทรลเลอร์: CE, DR1, CLK1, และ DATA (การรับข้อมูลช่องทางเดียว)

2.6.4.1 เมื่อตั้งค่า CE เป็น high จะทำให้ภายใน TRW-2.4GHz มีการประมวลผลข้อมูลเพื่อเตรียมการรับข้อมูล

2.6.4.2 มีการตรวจสอบ address และขนาดของ Payload ของ RF package ที่เข้ามา

2.6.4.3 หลังจาก TRW-2.4GHz เริ่มทำงาน 200 uSec TRW-2.4GHz จะตรวจสอบในอากาศว่ามีการติดต่อสื่อสารเข้ามาหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6.4.4 เมื่อได้รับ package ที่ถูกต้อง (address ถูกต้อง และมีการสร้าง CRC) TRW-2.4GHz จะทำการกำจัด preamble, address และบิท CRC

2.6.4.5 TRW-2.4GHz ในเวลานั้นจะแจ้ง(interrupts)ไปที่ไมโครคอนโทรลเลอร์ว่าได้ทำการรับข้อมูลที่ถูกต้อง โดยการทำให้ DR1 เป็น high

2.6.4.6 ไมโครคอนโทรลเลอร์อาจจะทำการตั้งค่า CE ให้เป็น low เพื่อยกเลิกระบบ RF front end (โหมดกินกระแสต่ำ)

2.6.4.7 ไมโครคอนโทรลเลอร์นั้นจะสิ้นสุดการทำงานโดยทำการปรับอัตราการรับส่งข้อมูลของ payload data ให้เหมาะสม (ในตัวอย่างนี้คือ 10 Kbps)

2.6.4.8 เมื่อ payload data ทั้งหมดถูกทำให้กลับสภาพเดิม TRW-2.4GHz จะตั้งค่า DR1 เป็น low อีกครั้ง

- ถ้า CE เป็น high อยู่ตลอดการถ่ายข้อมูล TRW-2.4GHz นั้นจะพร้อมสำหรับการรอรับชุดข้อมูลชุดใหม่

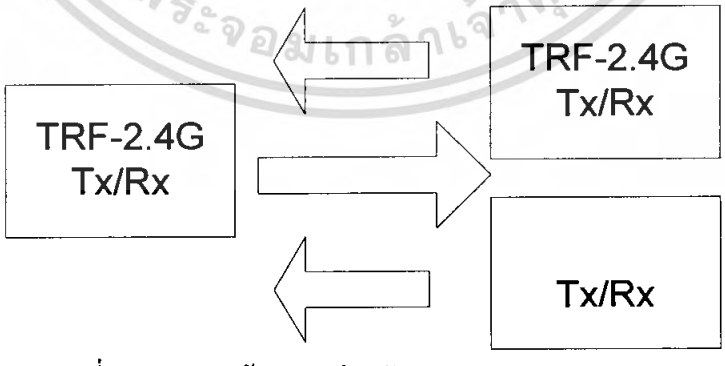
- ถ้า CE เป็น low จะเป็นการเริ่มต้นการทำงานใหม่ตามลำดับ

2.6.5 DuoCeiver โหมดรับข้อมูล 2 ช่องทาง

TRW-2.4GHz สามารถรับข้อมูลจาก ตัวส่ง 1 Mbps สองตัว โดยทั้งสองตัวต้องมีช่องสัญญาณห่างกัน 8 MHz ซึ่ง TRW-2.4GHz ตัวรับจะสามารถรับข้อมูลได้ทั้งสองช่องสัญญาณโดยใช้เสาอากาศของตัวรับเพียงตัวเดียว

การเชื่อมต่อกับไมโครคอนโทรลเลอร์จะใช้ขาสัญญาณดังนี้

- Data Channel 1: CLK1, DATA, and DR1
- Data Channel 2: CLK2, DOUT2, and DR2
- DR1 and DR2 จะใช้ใน ShockBurst mode เท่านั้น



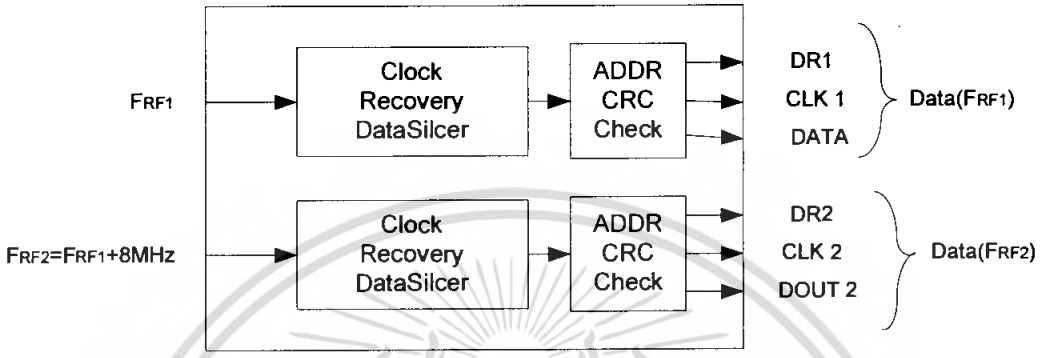
รูปที่ 2.13 การรับข้อมูล 2 ช่องสัญญาณของ TRW-2.4GHz

การจะใช้งานช่องสัญญาณที่ 2 นั้น จะต้องกำหนดให้มีช่องสัญญาณความถี่สูงกว่าช่องสัญญาณที่ 1 อยู่ 8 MHz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใน Direct mode นั้นไมโครคอนโทรลเลอร์สามารถรับข้อมูลของ TRW-2.4GHz ที่เข้ามาจาก 2 ช่องสัญญาณได้พร้อมกัน โดยไม่ต้องทำการสลับรับข้อมูลจาก 2 ช่องสัญญาณ

ใน ShockBurst mode สามารถให้ไมโครคอนโทรลเลอร์หยุดการทำงานของช่องสัญญาณหนึ่งเพื่อไปรับข้อมูลจากอีกช่องสัญญาณหนึ่งได้ โดยปราศจากการสูญเสียของข้อมูล การทำเช่นนี้จะทำให้ลดภาระการทำงานของไมโครคอนโทรลเลอร์ได้



รูปที่ 2.14 การรับข้อมูล 2 ช่องสัญญาณพร้อมกัน

2.6.6 การตั้งค่าให้กับอุปกรณ์

2.6.6.1 การตั้งค่าสำหรับการทำงานแบบ ShockBurst

จะประกอบไปด้วยค่าต่างๆดังนี้

- Payload section width: เป็นการระบุจำนวนของ payload บิทใน RF package ซึ่งจะทำให้ TRW-2.4GHz นั้นสามารถจำแนกความแตกต่างของ payload data และ CRC ไบท์ในแพคเกจที่รับมาได้

- Address width: เป็นการกำหนดจำนวนบิทที่ใช้สำหรับ address ใน RF package ซึ่งจะทำให้ TRW-2.4GHz นั้นสามารถจำแนกความแตกต่างระหว่าง address และ payload data

- Address(RX Channel 1 และ 2): เป็นการกำหนดปลายทางของ address สำหรับการรับข้อมูล

- CRC: ทำให้ TRW-2.4GHz สามารถสร้างและถอดรหัส CRC ได้

PRE-AMBLE	ADDRESS	PAYLOAD	CRC

รูปที่ 2.15 ส่วนประกอบของ Data package

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

2.6.7 Configuration Word Overviews

ตารางที่ 2.1 แสดง Configuration Word

การตั้งค่าการทำงานแบบ ShockBurst	ตำแหน่งบิท	จำนวนของบิท	ชื่อ	หน้าที่ในการทำงาน
	143:120	24	TEST	สำรองไว้เพื่อทดสอบ
	119:112	8	DATA1_W	ความยาวของข้อมูล payload Rx Channel 1
	111:104	8	DATA2_W	ความยาวของข้อมูล payload Rx Channel 2
	103:64	40	ADDR2	สูงสุด 5 ไบท์ address สำหรับ Channel 1
	63:24	40	ADDR1	สูงสุด 5 ไบท์ address สำหรับ Channel 2
	23:18	6	ADDR_W	จำนวนของ address บิท (ทั้ง2 Channel)
	17	1	CRC_L	เลือกระหว่าง 8 บิท หรือ16 บิท CRC
	16	1	CRC_EN	เป็นการเลือกใช้การสร้าง/ ตรวจสอบ CRC
การตั้งค่าการทำงานแบบDirect	15	1	RX2_EN	เป็นการใช้งานในโหมดการรับ 2 ช่องทาง
	14	1	CM	กำหนดโหมด Direct/ShockBurst
	13	1	RFDR_SB	RF data rate (ถ้าใช้ในความเร็ว 1 Mbps จะต้องใช้คริสตอล 16 MHz)
	12:10	3	XO_F	Crystal Frequency
	9:8	2	RF_PWR	RF Output Power
	7:1	7	RF_CH#	ช่องความถี่
	0	1	RXEN	กำหนดการทำงาน TX/RX

2.6.8 รายละเอียดของ Configuration Word

- ในการตั้งค่าการทำงานให้กับ TRW-2.4GHz จะใช้ 144 บิท (บิทที่ 143 เป็น MSB) โดย
- การตั้งค่าการทำงานในแบบ Direct จะใช้งานบิท[15:0]
- การตั้งค่าการทำงานในแบบ ShockBurst จะใช้งานบิท[119:0]
- การตั้งค่าการทำงานสำหรับการทดสอบ จะใช้งานบิท[143:120]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
83297

ตารางที่ 2.2 แสดง Configuration data word

MSB	TEST							
D143	D142	D141	D140	D139	D138	D137	D136	
Reserve for testing								
1	0	0	0	1	1	1	0	Default

MSB	TEST														
D135	D134	D133	D132	D131	D130	D129	D128	D127	D126	D125	D124	D123	D122	D121	D120
Reserve for testing														Close PLL in TX	
0	0	0	0	1	0	0	0	0	0	0	1	1	1	0	0	Default

DATA2_W								
D119	D118	D117	D116	D115	D114	D113	D112	
Data width channel#2 in # of bits excluding addr/crc								
0	0	1	0	0	0	0	0	Default

DATA1_W								
D111	D110	D109	D108	D107	D106	D105	D104	
Data width channel#1 in # of bits excluding addr/crc								
0	0	1	0	0	0	0	0	Default

ADDR2												
D103	D102	D101	...	D71	D70	D69	D68	D67	D66	D65	D64	
Channel#2 Address RX (up to 40 bit)												
0	0	0	...	1	1	1	0	0	1	1	1	Default

ADDR1												
D63	D62	D61	...	D31	D30	D29	D28	D27	D26	D25	D24	
Channel#1 Address RX (up to 40 bit)												
0	0	0	...	1	1	1	0	0	1	1	1	Default

ADDR_W						
D23	D22	D21	D20	D19	D18	
Address width in # of bits (both channels)						
0	0	1	0	0	0	Default

CRC		
D17	D16	
CRC Mode 1 = 16bit, 0 = 8bit	CRC1 = enable, 0 = disable	
0	1	Default

RF-Programming															LED	
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	
Two Ch.	BUF	OD	XO			RF Power		Channel selection							RXEN	
0	0	0	0	1	1	1	1	0	0	0	0	0	1	0	0	Default

2.6.9 การตั้งค่าสำหรับการทำงานแบบ ShockBurst

ในส่วนของบิท[119:16] ประกอบด้วยส่วนของคอนฟิกูเรชั่น ริจิสเตอร์ที่ใช้สำหรับการทำงานแบบ ShockBurst หลังจากที่ VDD เปิดการทำงานของ ShockBurst การตั้งค่าการทำงานจะเสร็จสิ้นทันทีที่ VDD ปรากฏขึ้น และระหว่างการทำงานนั้นที่ไบท์แรกของ frequency channel จะมีการเปลี่ยนเพื่อเป็นการสับเปลี่ยน RX/TX (การรับ/การส่ง)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PLL_CTRL

ตารางที่ 2.3 แสดงการตั้งค่า PLL

PLL_CTRL		
D121	**D120**	**PLL**
0	0	Open TX/Close RX
0	1	Open TX/Open RX
1	0	Close TX/Close RX
1	1	Close TX/Close RX

บิท 121-120:

PLL_CTRL: เพื่อควบคุมการตั้งค่าการทำงานของ PLL สำหรับการทดสอบ พร้อมกันกับปิดการทำงานของ PLL ในขณะที่มีการส่ง (TX) ที่ไม่ถูกต้อง

DATAx_W

ตารางที่ 2.4 แสดงจำนวนบิทของ payload

DATA2_W							
119	118	117	116	115	114	113	112

DATA1_W							
111	110	109	108	107	106	105	104

บิท 119-112:

DATA2_W: ขนาดของ RF package ในส่วนของ payload สำหรับการรับ channel ที่ 2

บิท 111-104:

DATA1_W: ขนาดของ RF package ในส่วนของ payload สำหรับการรับ channel ที่ 1

NOTE:

จำนวนบิททั้งหมดของ RF package ในโหมด ShockBurst จะต้องไม่เกิน 256 ซึ่งจำนวนสูงสุดของส่วน payload จะหาได้จาก

$$DATAx_W(bits) = 256 - ADDR_W - CRC$$

โดย ADDR_W: ความยาวของ RX address ที่จะกำหนดใน configuration word บิท[23:18]

CRC: check sum, 8 บิทหรือ 16 บิท ที่จะกำหนดใน configuration word บิท[17]

PRE: preamble, 4 บิทหรือ 8 บิท ส่วนนี้จะถูกเพิ่มเข้าไปโดยอัตโนมัติ

ADDRx

ตารางที่ 2.5 Address ของตัวรับที่ 2 และตัวรับที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ADDR2											
103	102	101	...	71	70	69	68	67	66	65	64

ADDR1											
63	62	61	...	31	30	29	28	27	26	25	24

บิท 103-64:

ADDR2: Receiver address channel 2, สูงสุด 40 บิท

บิท 63-24:

ADDR1: Receiver address channel 1, สูงสุด 40 บิท

ตารางที่ 2.6 แสดงจำนวนบิทสำรองของ RX Address + CRC

ADDR_W						CRC_L	CRC_EN
23	22	21	20	19	18	17	16

บิท 23-18:

ADDR_W: จำนวนบิทสำรองของ RX address ใน ShockBurst package สูงสุด 40 บิท (5 ไบท์)

บิท 17:

CRC_L: ความยาวของ CRC ที่จะถูกประมวลผลใน TRW-2.4GHz ใน ShockBurst

Logic 0: 8 bit CRC

Logic 1: 16 bit CRC

บิท 16:

CRC_EN: เปิดความสามารถในการสร้าง CRC (TX) และตรวจสอบ CRC (RX)

Logic 0: ปิด On-chip CRC

Logic 1: เปิด On-chip CRC

2.6.10 การตั้งค่าสำหรับการทำงานแบบ Direct Modes

ตารางที่ 2.7 แสดงการตั้งค่าการทำงานของ RF

RX2_EN	CM	RFDR_SB	XO_F			RF_PWR	
15	14	13	12	11	10	9	8

บิท 15:

RX2_EN: Logic 0: 1 channel receive

Logic 1: 2 channel receive

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการใช้ 2 channel receive นั้น การกำหนดความถี่ของ channel 2 จะต้องมากกว่าความถี่ของ channel 1 อยู่ 8 MHz เสมอ

บิท 14:

Communication Mode:

Logic 0: Direct Mode

Logic 1: ShockBurst Mode

บิท 13:

RF Data Rate:

Logic 0: 250 Kbps

Logic 1: 1 Mbps

บิท 12-10:

Crystal Frequency:

ตารางที่ 2.8 แสดงการตั้งค่าของ Crystal

D12	D11	D10
0	1	1

บิท 9-8:

RF_PWR: ตั้งค่ากำลังในการส่ง RF output

ตารางที่ 2.9 แสดงการตั้งค่าของ RF output power

RF OUTPUT POWER		
D9	D8	P[dBm]
0	0	-20
0	1	-10
1	0	-5
1	1	0

บิท 7-1:

RF Channel & Direction

ตารางที่ 2.10 แสดงการตั้งค่าช่องสัญญาณความถี่ และ RX/TX

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

RF_CH#							RXEN
7	6	5	4	3	2	1	0

RF_CH#: ตั้งค่าช่องสัญญาณความถี่ที่จะใช้ในการทำงานให้กับ TRW-2.4GHz

Channel frequency ในการส่ง:

Channel = 2400 MHz + RF_CH#*1.0 MHz

RF_CH#: จะอยู่ระหว่าง 2400MHz และ 2527MHz

Channel frequency ใน data channel 1:

Channel = 2400 MHz + RF_CH#*1.0 MHz

RF_CH#: จะอยู่ระหว่าง 2400MHz และ 2524MHz

Channel frequency ใน data channel 2:

Channel = 2400 MHz + RF_CH#*1.0 MHz + 8 MHz

RF_CH#: จะอยู่ระหว่าง 2400MHz และ 2524MHz

บิท 0:

Set Active Mode: Logic 0: transmit mode

Logic 1: receive mode

2.6.11 คำอธิบาย DATA package

PRE-AMBLE	ADDRESS	PAYLOAD	CRC

รูปที่ 2.16 DATA package

Data package ของทั้ง ShockBurst mode และ Direct mode จะประกอบไปด้วย 4 ส่วนดังนี้

ตารางที่ 2.11 อธิบาย Data package

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.PREAMBLE	-ส่วน preamble นั้นจำเป็นต้องใช้สำหรับ ShockBurst mode
2.ADDRESS	-ส่วน address นั้นจำเป็นต้องใช้สำหรับ ShockBurst mode -มีขนาด 8 ถึง 40 บิท - address นั้นจะถูกถอดออกจากแพคเกจ ที่ได้รับมาโดยอัตโนมัติใน ShockBurst mode
3.PAYLOAD	- เป็นข้อมูลที่จะถูกส่งออกไป -ใน ShockBurst mode ขนาดของ payload คือ 256 บิท โดยไม่รวมค่าต่อไปนี้ (Address: 8 ถึง 40 บิท + CRC 8 หรือ 16 บิท)
4.CRC	- มีขนาด 8 หรือ 16 บิท - CRC จะถูกถอดออกจากข้อมูลที่ได้รับมา

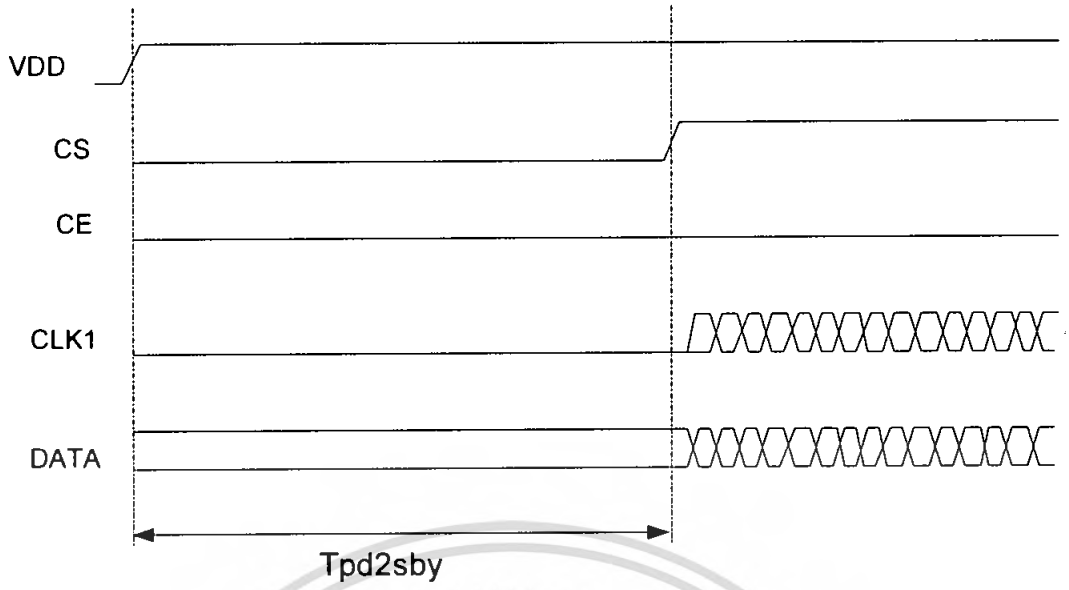
2.6.12 Timing Data ที่สำคัญ

ตารางที่ 2.12 Timing data

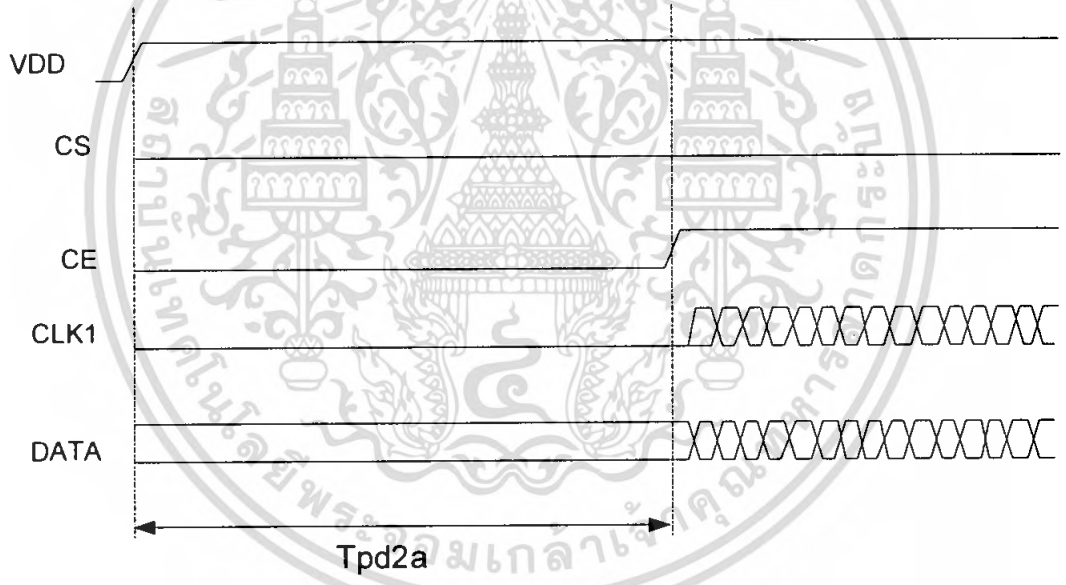
TRW-2.4G	Max.	Min.	Name
V_{DD} off → ST_BY mode	3 mSec		Tpd2sby
V_{DD} off → Active mode (RX/TX)	3 mSec		Tpd2a
ST_BY → TX ShockBurst	195 uSec		Tsby2txSB
ST_BY → TX Direct	202 uSec		Tsby2txDM
ST_BY → RX mode	202 uSec		Tsby2rx
Minimum delay from CS to data		5 uSec	Tcs2data
Minimum delay from CE to data		5 uSec	Tce2data
Minimum delay from DR1 / 2 to clk		50 nSec	Tdr2clk
Minimum delay from clk to data	50 nSec		Tclk2data
Delay between edges		50 nSec	Td
Setup time		500 nSec	Ts
Hold time		500 nSec	Th
Delay to finish internal GFSK data		1/datarate	Tfd
Minimum input clock high		500 nSec	Thmin
Set-up of data in Direct mode	50 nSec		Tsdm
Minimum clock high in Direct mode		300 nSec	Thdm
Minimum clock low in Direct mode		230 nSec	Tldm

เมื่อ TRW-2.4GHz อยู่ในสภาวะ power down จะทำให้เข้าสู่โหมด stand-by (Tpd2sby) ก่อนที่จะเข้าสู่การตั้งค่าการทำงานหรือเข้าสู่โหมดการทำงานโหมดใดโหมดหนึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.17 แสดง Timing diagram ของ TRW-2.4GHz ขณะอยู่ในโหมด stand-by



รูปที่ 2.18 ค่าแรงดัน VDD ของ active mode

สังเกตว่า configuration word จะหายไปเมื่อ VDD นั้นถูกปิดและตัวอุปกรณ์จะต้องถูกตั้งค่าการทำงานก่อนที่จะเข้าสู่โหมดการทำงานโหมดใดโหมดหนึ่ง

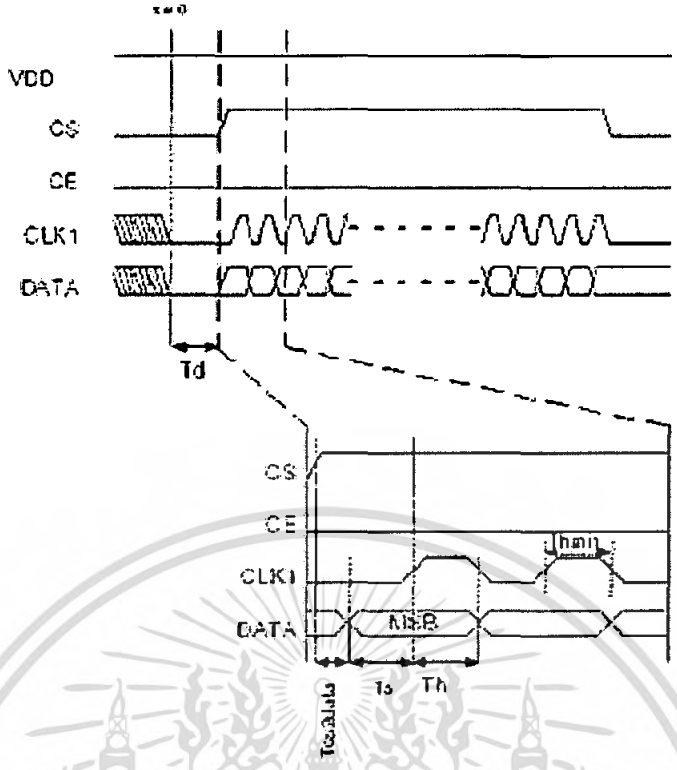
Note:

CE และ CS จะต้องไม่เป็นลอจิก high ในเวลาเดียวกัน

2.6.13 Configuration mode timing

เมื่อมีบิทหนึ่งหรือบิทใดใน configuration word ถูกเปลี่ยนซึ่งจะเปลี่ยนตาม timing diagram

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



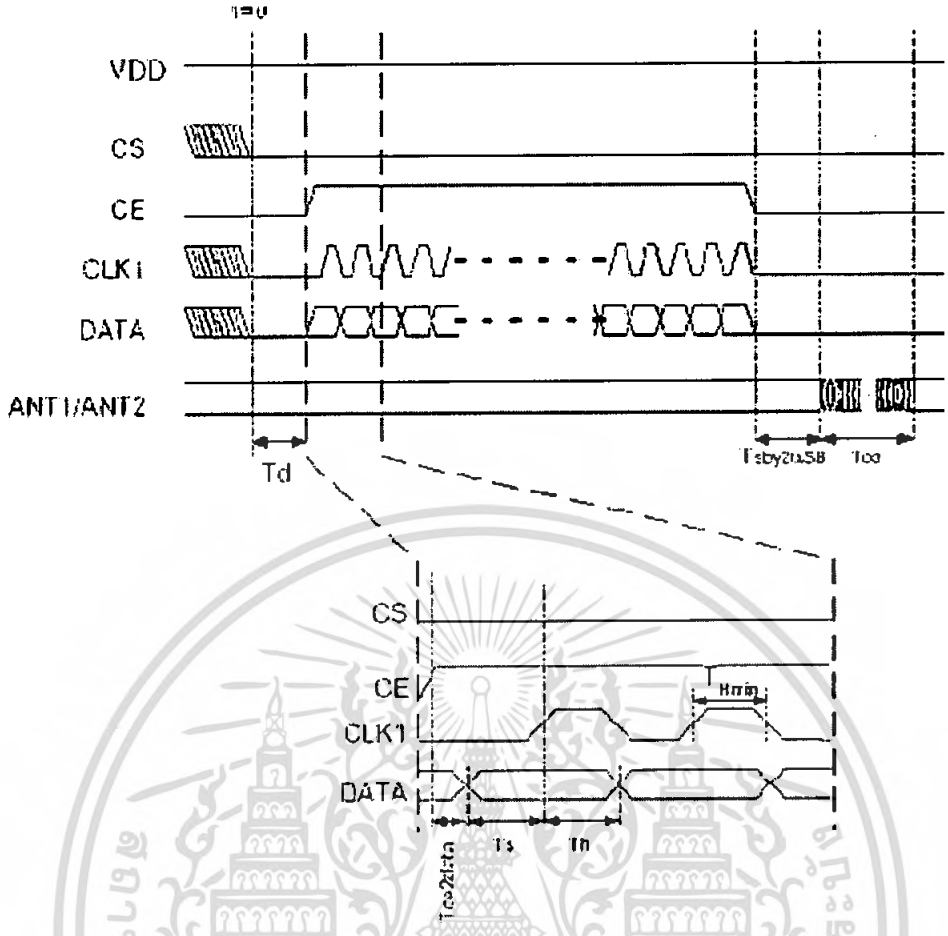
รูปที่ 2.19 Timing diagram สำหรับการตั้งค่าการทำงาน

ถ้าเปลี่ยนจาก power down มาเป็น configuration mode, CS จะต้องถูกตั้งค่าเป็นลอจิก high หลังจาก Tpd2sby ดังรูปที่ 2.18

2.6.14 ShockBurst mode Timing

2.6.14.1 ShockBurst TX:

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



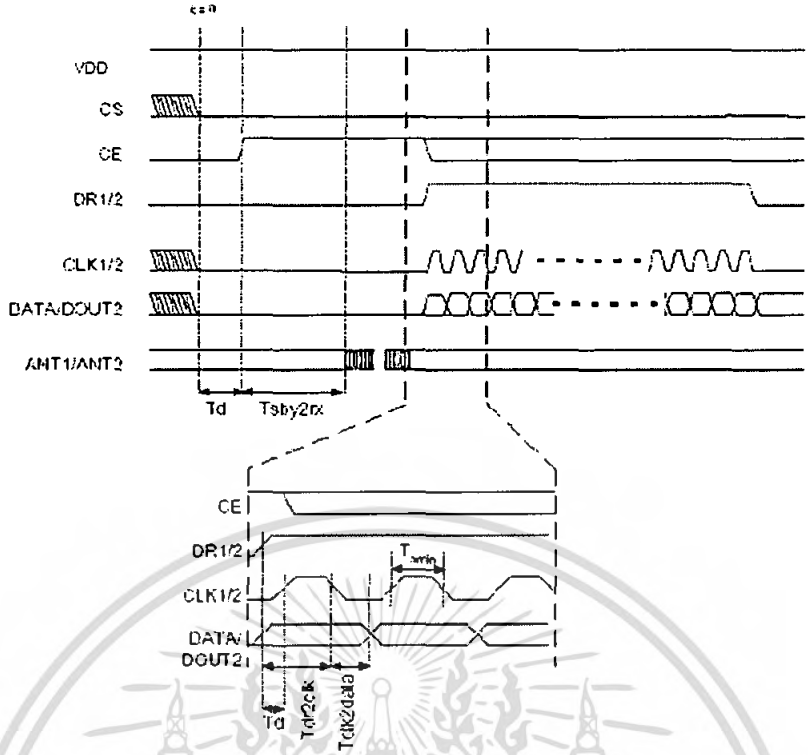
รูปที่ 2.20 Timing ของ ShockBurst TX

ขนาดของ package และ data rate จะแสดงให้เห็นดีเลย์ Toa (Time on air) ดังสมการ

TOA = 1/datarate*(#databits+1)

2.6.14.2 ShockBurst RX:

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.21 Timing ของ ShockBurst RX

CE อาจจะตั้งค่าให้เป็นลอจิก high ตลอดการดาวน์โหลดข้อมูล แต่จะทำให้กินกระแสสูงถึง 18 mA และจะมีข้อดีคือไม่ต้องรอการหน่วงเวลา 200 uSec หลังจากที่ DR1 กลายเป็นลอจิก 1

2.6.15 Output Power adjustment

ตารางที่ 2.13 แสดง การปรับแต่ง Output Power

Power setting bits of configuring word	RF output power	DC current consumption
11	0 dBm ± 3dB	13.0 mA
10	-5 dBm ± 3dB	10.5 mA
01	-10 dBm ± 3dB	9.4 mA
00	-20 dBm ± 3dB	8.8 mA

Condition: VDD = 3.0V, VSS = 0V, T_A = 27°C, Load impedance = 400 ohm

2.7 ไมโครคอนโทรลเลอร์ MCS – 51 แบบแฟลช (Flash)

ไมโครคอนโทรลเลอร์ตระกูล MCS – 51 ที่ใช้ในโครงงานนี้คือ ไมโครคอนโทรลเลอร์ตระกูล MCS – 51 ซึ่งมีหน่วยความจำภายในเป็นแบบแฟลช (Flash memory) เหตุผลที่ใช้ไมโครคอนโทรลเลอร์แบบนี้ในการเรียนรู้เพื่อใช้งานไมโครคอนโทรลเลอร์ตระกูล MCS – 51 มีด้วยกันหลายประการดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

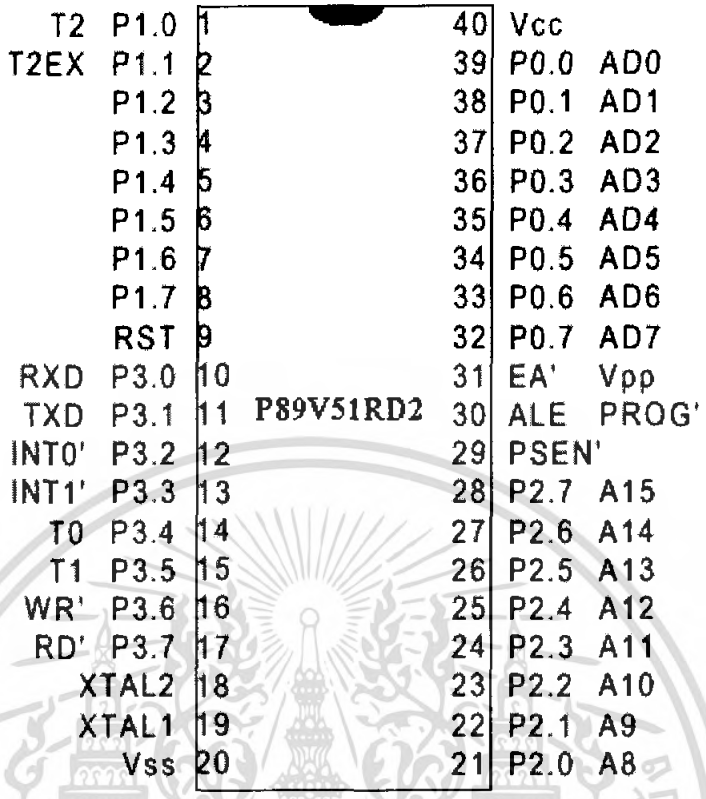
- หน่วยความจำโปรแกรมภายในตัวไมโครคอนโทรลเลอร์เป็นแบบแฟลช ทำให้สามารถลบและเขียนใหม่ได้นับพันครั้ง จึงสามารถใช้งานในรูปแบบของไมโครคอนโทรลเลอร์ชิปเดี่ยวไม่ต้องใช้หน่วยความจำนอกส่งผลให้สามารถใช้งานพอร์ตอินพุตเอาต์พุตของไมโครคอนโทรลเลอร์ได้อย่างเต็มประสิทธิภาพ
- ต้นทุนและเวลาในการพัฒนาระบบไมโครคอนโทรลเลอร์ลดลงอย่างมาก เนื่องจากไม่ต้องใช้เครื่องมือพัฒนาจำพวกอีมูเลตอร์ (Emulator) และเครื่องโปรแกรมอีพรอม (Eprom)
- บริษัทผู้ผลิตได้ทำการผลิตไมโครคอนโทรลเลอร์ตระกูลนี้ออกมาหลายเบอร์ และมีความสามารถแตกต่างกันไปทำให้มีทางเลือกในการใช้งานสูง
- ด้วยการใช้หน่วยความจำภายในตัวไมโครคอนโทรลเลอร์ ทำให้สามารถป้องกันการคัดลอกข้อมูลของหน่วยความจำโปรแกรมได้เป็นอย่างดี

2.7.1 คุณสมบัติของไมโครคอนโทรลเลอร์ตระกูล MCS – 51 อนุกรม P89LV51RD2BN

- หน่วยประมวลผลกลางเป็นแบบ 80C51
- สามารถทำงานที่แรงดัน 3 V และที่ความถี่ 0-33 MHz
- ภายในมีหน่วยความจำโปรแกรมเป็นแบบแฟลชสามารถลบและเขียนใหม่ได้พันครั้ง
- หน่วยความจำข้อมูลพื้นฐาน 64 KBเป็นหน่วยความจำแบบ ISP (In System Programming) และ IAP (In Application Programming)
- รองรับซอฟท์แวร์ ISP แบบที่มีโหมด 12-clock หรือ 6-clock
- การเชื่อมต่อแบบ SPI (Serial Peripheral Interface) และเสริมด้วย UART
- ไทม์เมอร์ (Timer) /เคาน์เตอร์ขนาด 16 บิตอย่างน้อย 3 ตัว
- สามารถรองรับแหล่งกำเนิดอินเตอร์รัปต์ (Interrupt) ได้ 8 ประเภท 4 ระดับ
- ทำการ PCA (Programmable Counter Array) ด้วย PWM และมี Capture/Compare ฟังชั่น
- มีโหมด low EMI (ALE Inhibit)
- มี DPTR register ลำดับสอง
- มีวอตช์ด็อกไทม์เมอร์ (Watch Dog Timer) ในตัว
- TTL- และ CMOS- ในระดับลอจิกสามารถเข้ากันได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.7.2 การจัดขาของไมโครคอนโทรลเลอร์ MCS-51



รูปที่ 2.22 การจัดขามาตรฐานของไมโครคอนโทรเลอร์ MCS-51 P89LV51RD2BN

ไมโครคอนโทรลเลอร์ MCS – 51 ทุกเบอร์จะมีขาใช้งานพื้นฐานเหมือนกัน ดังแสดงในรูปที่ 2.22 โดยมีรายละเอียดขั้นต้น ดังนี้

ขา Vcc ใช้สำหรับต่อไฟเลี้ยง +5 โวลท์

ขา GND เป็นขากราวด์ สำหรับต่อกับกราวด์ของระบบ

ขาพอร์ต 0 (P0.0 – P0.7) มี 8 ขา แต่ละขาสามารถกำหนดให้เป็นได้ทั้งอินพุตและเอาต์พุต สำหรับใช้งานทั่วไป ถ้าหากต้องการกำหนดให้ขาพอร์ต 0 ขาใดขาหนึ่งเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย จึงมีอินพุตอิมพีแดนซ์ (Impedance) สูงสามารถใช้เป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ตนี้ยังถูกใช้งานในการติดต่อกับขาแอดเดรสไบต์ต่ำของหน่วยความจำภายนอก (A0-A7) และขาข้อมูล (D0-D7) โดยใช้กระบวนการมัลติเพล็กซ์ (Multiplex) เข้าช่วย เพื่อสลับการทำงานเป็นได้ทั้งขาติดต่อแอดเดรสและขาข้อมูล

ขาพอร์ต 1 (P1.0 – P1.7) มี 8 ขา แต่ละขาสามารถกำหนดให้เป็นได้ทั้งอินพุตและเอาต์พุต สำหรับใช้งานทั่วไปถ้าหากต้องการ กำหนดให้ขาพอร์ตใดเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย นอกจากนั้นในอนุกรม AT89Sxx จะใช้ขา P1.0 เป็นขา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อินพุตสำหรับนับค่าของไทม์เมอร์ 2 และ P1.1 เป็นขาอินพุตทริกเกอร์ของไทม์เมอร์ 2 ในขณะที่ขา P1.4 ถึง P1.7 เป็นขาสำหรับเชื่อมต่อแบบ SPI เพื่อทำการโปรแกรมข้อมูลในระบบ

ขาพอร์ต 2 (P2.0 – P2.7) มี 8 ขา แต่ละขาสามารถกำหนดให้เป็นได้ทั้งอินพุตและเอาต์พุต สำหรับใช้งานทั่วไป ถ้าหากต้องการกำหนดให้ขาพอร์ตใดเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย จึงมีอินพุตอิมพีแดนซ์สูงสามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ตนี้ยังถูกใช้งานในการติดต่อกับขาแอดเดรสไบต์สูงของหน่วยความจำภายนอก (A8-A15)

ขาพอร์ต 3 (P3.0 – P3.7) มี 8 ขา แต่ละขาสามารถกำหนดให้เป็นได้ทั้งอินพุตและเอาต์พุต สำหรับใช้งานทั่วไป ถ้าหากต้องการกำหนดให้ขาพอร์ตใดเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้

P3.0 ใช้เป็นขาอินพุตสำหรับข้อมูลจากการสื่อสารแบบอนุกรม หรือขา Rx

P3.1 ใช้เป็นขาอินพุตสำหรับส่งข้อมูลจากการสื่อสารแบบอนุกรม หรือขา TxD

P3.2 ใช้เป็นขาอินพุตรับสัญญาณอินเตอร์รัปต์จากภายนอกช่อง 0 หรือขา $\overline{\text{INT0}}$

P3.3 ใช้เป็นขาอินพุตรับสัญญาณอินเตอร์รัปต์จากภายนอกช่อง 1 หรือขา $\overline{\text{INT1}}$

P3.4 ใช้เป็นขาอินพุตสำหรับรับสัญญาณไทม์เมอร์จากภายนอกช่อง 1 หรือขา T0

P3.5 ใช้เป็นขาอินพุตสำหรับรับสัญญาณอินเตอร์รัปต์จากภายนอกช่อง 1 หรือขา T1

P3.6 ใช้เป็นขาสัญญาณ $\overline{\text{WR}}$ ในกรณีที่ใช้เชื่อมต่อกับหน่วยความจำภายนอก

P3.7 ใช้เป็นขาสัญญาณ $\overline{\text{RD}}$ ในกรณีที่ใช้เชื่อมต่อกับหน่วยความจำภายนอก

ขารีเซต (Reset) ใช้ในการรีเซตการทำงานของไมโครคอนโทรลเลอร์ โดยในการป้อนสัญญาณเพื่อรีเซตสถานะที่ขาของต้องอยู่ในระดับรีเซตอย่างน้อย 2 แมชชีนไซเกิล (Machine cycle) โดยที่วงจรกำเนิดสัญญาณนาฬิกายังคงทำงานต่อเนื่องไปอย่างเป็นปกติ

ขา $\overline{\text{ALE}}$/PROG (Address Latch Enable/Program pulse input) เป็นขาที่ใช้ในการควบคุมการแลตช์ (Latch) ของขาพอร์ต 0 เมื่อมีการใช้งานหน่วยความจำภายนอก นอกจากนั้นขานี้ยังใช้เป็นขาสำหรับรับพัลส์ของการโปรแกรมสำหรับโปรแกรมข้อมูลลงในไมโครคอนโทรลเลอร์ MCS – 51 ในรุ่นที่มีหน่วยความจำโปรแกรมเป็นแบบอีพรอม

ขา $\overline{\text{PSEN}}$ (Program Store Enable) ขานี้ใช้ในการส่งสัญญาณเพื่อร้องขอติดต่อกับหน่วยความจำโปรแกรมภายนอก เมื่อไมโครคอนโทรลเลอร์ต้องการอ่านข้อมูลจากหน่วยความจำโปรแกรมภายนอกตัวไมโครคอนโทรลเลอร์จะส่งสัญญาณออกมาที่ขานี้ 2 ครั้งในแต่ละแมชชีนไซเกิล แต่ถ้าหากติดต่อกับหน่วยความจำภายนอกขานี้จะไม่มีการส่งสัญญาณใด ๆ ออกมา

ขา $\overline{\text{EA}}$/Vpp (External access enable/Programming voltage input) ใช้สำหรับเลือกการติดต่อหน่วยความจำโปรแกรมจากภายนอกหรือภายใต้ตัวไมโครคอนโทรลเลอร์ถ้าหากขานี้เป็น " 0 " เป็นการเลือกให้ไมโครคอนโทรลเลอร์ติดต่อกับหน่วยความจำโปรแกรมภายนอก แต่ถ้าหากขานี้เป็น "1 " เป็นการเลือกให้ไมโครคอนโทรลเลอร์ติดต่อกับหน่วยความจำในตัวไมโครคอนโทรลเลอร์ นอกจากนี้ที่ขานี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ยังใช้เป็นขาอินพุตสำหรับรับแรงดันไฟสูงสำหรับการโปรแกรมหน่วยความจำในไมโครคอนโทรลเลอร์ สำหรับไมโครคอนโทรลเลอร์ MCS – 51 แบบแฟลชต้องการแรงดันสำหรับโปรแกรม +12V

ขา XTAL1 และ XTAL2 เป็นขาสำหรับต่อคริสตอลเพื่อสร้างสัญญาณนาฬิกาในการกำหนดจังหวะการทำงานของไมโครคอนโทรลเลอร์

2.7.3 การอ่านค่าลอจิกจากพอร์ต

ในไมโครคอนโทรลเลอร์ MCS-51 แบบแฟลชสามารถอ่านค่าลอจิกจากพอร์ตได้ 2 ลักษณะคือ อ่านจากขาพอร์ตโดยตรงและอ่านจากวงจรแลตช์ของแต่ละพอร์ต ในกรณีที่พอร์ตต่อกับขาเบสทรานซิสเตอร์ (Transistor) ชนิด NPN และขาอิมิตเตอร์ (Emitter) ของทรานซิสเตอร์ตัวนั้นต่อลงกราวด์ หากมีการส่งข้อมูล " 1 " ไปยังทรานซิสเตอร์จะทำให้ทรานซิสเตอร์ทำงานสถานะลอจิกที่ขาพอร์ตจะเป็น " 0 " เนื่องจากเมื่อทรานซิสเตอร์ทำงาน จะเสมือนว่าขาพอร์ตนั้นถูกต่อลงกราวด์ ทำให้หากอ่านค่าลอจิกที่ขาพอร์ตจะได้ผลตรงข้ามกับที่ส่งออกมา แต่ถ้าหากทำการอ่านค่าลอจิกที่วงจรแลตช์ จะได้ค่าที่ตรงกับค่าที่ต้องการส่งจริงดังนั้นในการอ่านค่าลอจิกจากพอร์ตจึงต้องเลือกวิธีการให้เหมาะสมกับอุปกรณ์ที่นำมาต่อด้วย

2.7.4 จังหวะการทำงานของไมโครคอนโทรลเลอร์ MCS-51

ในการใช้งานไมโครคอนโทรลเลอร์ MCS-51 จะต้องทำความเข้าใจถึงจังหวะการทำงานของซีพียูและลำดับขั้นตอนการประมวลผลคำสั่ง ในการประมวลคำสั่งของซีพียูจะมีขั้นตอนหลัก ๆ 2 ขั้นตอน คือ กระบวนการเฟตช์ (Fetch) เป็นการเรียกคำสั่งออกจากหน่วยความจำโปรแกรมแล้วทำการแปลงรหัสคำสั่งนั้นเป็นภาษาเครื่องเพื่อเตรียมการประมวลผล ขั้นตอนต่อมาคือ กระบวนการเอ็กซิคิวต์ **(Execute)** เป็นการกระทำตามคำสั่งที่กำหนดหรือตามที่เฟตช์ขึ้นมาโดยกระบวนการก่อนหน้านี้ เมื่อทำการเอ็กซิคิวต์คำสั่งเรียบร้อยแล้วก็จะไปเริ่มกระบวนการเฟตช์คำสั่งใหม่ต่อไป

เมื่อเริ่มจ่ายไฟให้แก่ไมโครคอนโทรลเลอร์จะเกิดการรีเซตในลักษณะที่เรียกว่า เพาเวอร์ออนรีเซต (Power-on reset) ซีพียูเริ่มต้นการทำงานที่แอดเดรส 0000H ของหน่วยความจำโปรแกรมจังหวะการทำงานของซีพียูจะเป็นไปตามรูปแบบ โดยได้รับการกำหนดมาจากรอบการทำงานหรือแมชชีนไซเกิล (Machine cycle)

2.7.5 โครงสร้างและการทำงานของพอร์ต

ไมโครคอนโทรลเลอร์ MCS – 51 แบบแฟลชมีพอร์ตใช้ในงานทั้งสิ้น 4 พอร์ตคือ พอร์ต 0 ถึงพอร์ต 3 แต่ละพอร์ตมีขนาด 8 บิต เป็นพอร์ตแบบ 2 ทิศทาง กล่าวคือ สามารถเป็นได้ทั้งอินพุตสำหรับรับสัญญาณข้อมูลเข้าและเอาต์พุตสำหรับส่งสัญญาณข้อมูลออก ทุกพอร์ตของไมโครคอนโทรลเลอร์

MCS – 51 แบบแฟลชมีวงจรแลตช์และวงจรขับตลอดจนบัฟเฟอร์อินพุต ที่พอร์ต 0 และพอร์ต 2 จะใช้งานเป็นพอร์ตอินพุตและเอาต์พุตสำหรับทั่วไป และใช้ในการติดต่อกับหน่วยความจำภายนอกสำหรับพอร์ต 3 ทั้งพอร์ตและพอร์ต 1 บางขานอกจากจะใช้เป็นขาพอร์ตอินพุตเอาต์พุตตามปกติแล้วยังสามารถใช้งานในหน้าที่พิเศษได้อีก ขึ้นอยู่กับว่าเป็นไมโครคอนโทรลเลอร์ MCS-51

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.7.6 การใช้งานเป็นพอร์ตอินพุต

เนื่องจากพอร์ตทั้งหมดของไมโครคอนโทรลเลอร์ MCS-51 แบบแฟลชสามารถเป็นได้ทั้งอินพุต และเอาต์พุต ดังนั้นจึงมีความจำเป็นอย่างยิ่งต้องทำความเข้าใจถึงการกำหนดลักษณะการทำงานให้แก่พอร์ตของไมโครคอนโทรลเลอร์ MCS-51 แบบแฟลช ในการกำหนดให้เป็นพอร์ตอินพุต ต้องเริ่มต้นด้วยการเขียนข้อมูล "1" มาแต่ละบิตของพอร์ตที่ต้องการใช้งานเป็นอินพุต เพื่อหยุดการทำงานของเฟต (FET) ที่ใช้ในการขับสัญญาณเอาต์พุตของบิตนั้นๆ ทำให้ขาสัญญาณของพอร์ตเชื่อมต่อเข้ากับวงจร พูลอัพ (Pull - up) ภายในโดยตรงส่งผลให้ขาพอร์ตนั้นมีลอจิกเป็น "1" สามารถรับสัญญาณลอจิก "0" จากอุปกรณ์ภายนอกได้ง่ายสัญญาณข้อมูลจากอุปกรณ์ภายนอกจะถูกส่งเข้ามาแล้วเก็บไว้ในวงจรบัฟเฟอร์ภายในพอร์ตแล้วรอให้ซีพียูมาอ่านค่าเข้าไป เมื่อเป็นเช่นนี้อุปกรณ์ภายนอกที่เชื่อมต่อกับพอร์ตอินพุตของไมโครคอนโทรลเลอร์ MCS - 51 แบบแฟลชควรกำหนดให้ทำงานในสภาวะลอจิก "0" จะดีและสะดวกที่สุด (ซึ่งในปัจจุบันอุปกรณ์อินพุตที่ใช้ต่อไมโครคอนโทรลเลอร์ แทบทั้งหมดทำงานที่ลอจิก "0" แล้ว)

2.7.7 การใช้งานเป็นพอร์ตเอาต์พุต

โดยปกติแล้ว ขาพอร์ตจะกำหนดให้มีลักษณะเป็นเอาต์พุตอยู่แล้ว ดังนั้นจึงสามารถส่งข้อมูลออกไปได้อย่างง่ายดายและตรงไปตรงมา กล่าวคือ เมื่อต้องการส่งข้อมูล "0" ออกไปทางเอาต์พุตก็ให้เขียนข้อมูล "0" ไปยังวงจรแลตช์ ซึ่งก็จะส่งต่อไปขับเฟตทำให้เฟตทำงานที่ขาพอร์ตที่กำหนดให้ทำงานก็จะเกิดลอจิก "0" ขึ้นในทางตรงข้ามต้องการส่งข้อมูล "1" ออกไปก็ให้เขียนข้อมูล "1" ไปยังวงจรแลตช์ วงจรขับก็จะหยุดทำงานทำให้ที่ขาพอร์ตเชื่อมต่อกับวงจรพูลอัพภายในเกิดเป็นลอจิก "1" ที่ขาพอร์ตนั้น ซึ่งจะคล้ายกับการกำหนดให้เป็นขาอินพุตมาก เพียงแต่แตกต่างกันที่กระบวนการในการเคลื่อนย้ายข้อมูล โดยถ้าเป็นอินพุตจะมีสัญญาณมาอ่านข้อมูลที่บัฟเฟอร์ แต่ถ้าเป็นเอาต์พุตจะไม่มีการอ่านข้อมูลที่บัฟเฟอร์แต่อย่างใด เว้นแต่ในกรณีที่ต้องการตรวจสอบข้อมูลที่ส่งออกมาทางเอาต์พุตเมื่อใช้งานเป็นพอร์ตเอาต์พุต แต่ละขา (หรือแต่ละบิต) ของแต่ละพอร์ตมีความสามารถในการจ่ายกระแสหรือที่เรียกว่า กระแสซอร์ส (Source current) ได้สูงสุด 10 mA และทุกขารวมกันในแต่ละพอร์ต (ทั้ง 8 บิต) สูงสุด 26 mA สำหรับพอร์ต 0 และ 15 mA สำหรับพอร์ต 1-3 ในกรณีที่ใช้งานทุกพอร์ตเอาต์พุตจะสามารถจ่ายกระแสได้รวมกันสูงสุด 71 mA ดังนั้นในการใช้งานเป็นพอร์ตเอาต์พุตเพื่อไม่ให้เกิดปัญหาเกี่ยวกับความสามารถในการจ่ายกระแสจึงควรต่อวงจรบัฟเฟอร์ทางเอาต์พุตเพื่อช่วยในการขับกระแสอีกทางหนึ่ง

2.8 โมดูลแอลซีดี (Module LCD)

ในโมดูลแอลซีดีจะมีส่วนประกอบหลัก ๆ 3 ส่วนดังนี้

ตัวแสดงผล (display) ภายในเป็นผลึกเหลวที่สามารถแสดงผลให้เห็นโดยอาศัยแสงจากภายนอก ดังนั้นจึงต้องมีมุมในการมองข้อมูลที่แสดงผลบนจอแอลซีดี

ตัวควบคุม (controller) เป็นตัวรับข้อมูลจากอุปกรณ์ภายนอกมาควบคุมการทำงานของโมดูลแอลซีดี เช่น ลบจอภาพ แสดงอักษร หรือเลื่อนเคอร์เซอร์ (Cursor) เป็นต้น

ตัวขับ (driver) เป็นตัวรับสัญญาณจากตัวควบคุมมาขับให้ตัวแสดงผลแสดงข้อมูลตามที่กำหนด ชิปที่ใช้ทำหน้าที่เป็นตัวขับนี้ได้แก่ เบอร์ HD44100H และ MSM5259 เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.8.1 โครงสร้างภายในของตัวควบคุมโมดูลแอลซีดี

ในการใช้งานโมดูลแอลซีดี จำเป็นต้องทำความเข้าใจเกี่ยวกับโครงสร้างและคำสั่งที่ใช้ในการควบคุมให้ดีเสียก่อน ในหนังสือนี้ขอยกตัวอย่างโมดูลแอลซีดีแบบอักขระ เพราะสามารถเข้าใจได้ง่ายซึ่งใช้ในโมดูลแอลซีดีแบบอักขระ ประกอบด้วย

บัฟเฟอร์อินพุตเอาต์พุต (Buffer input/output) เป็นส่วนที่ใช้ในการติดต่อรับส่งข้อมูลกับอุปกรณ์ภายนอก เพื่อที่จะถ่ายทอดข้อมูลเข้าออกภายในตัวควบคุม

รีจีสเตอร์คำสั่ง (Data Register : DR) เป็นรีจีสเตอร์ใช้รับข้อมูลจากอุปกรณ์ภายนอก เพื่อถ่ายทอดต่อไปยังหน่วยความจำที่ทำหน้าที่เก็บข้อมูลแสดงผล หรือนำข้อมูลไปสร้างตัวอักษรเพิ่มเติมในแรมเก็บตัวอักษร

แรมเก็บข้อมูลแสดงผล (Display Data RAM : DDRAM) เป็นหน่วยความจำแรมทำหน้าที่เก็บข้อมูลที่มาจากรีจีสเตอร์ DR ตัวควบคุมจะนำข้อมูลใน DDRAM นี้ไปเปิดตาราง (Look up-table) ของตัวอักษรที่เก็บไว้ในหน่วยความจำรอมและแรมเก็บตัวอักษร เพื่อนำไปแสดงที่ตัวแสดงผล

รอมเก็บตัวอักษร (Character Generator ROM : CGROM) เป็นหน่วยความจำรอมที่ใช้เก็บข้อมูลตัวอักษรหรือสัญลักษณ์ที่สามารถอ่านออกไปแสดงที่ตัวแสดงผลได้ มีขนาด 7,200 บิต โดยจะถูกอ่านด้วยค่าของข้อมูลใน DDRAM

แรมเก็บตัวอักษร (Character Generator RAM : CGRAM) เป็นหน่วยความจำแรมที่ใช้เก็บตัวอักษรที่มีการสร้างเพิ่มเติมขึ้นใหม่ ในกรณีที่ตัวอักษรใน CGROM ไม่เพียงพอ มีขนาด 512 บิต การเขียนและอ่านค่าไปใช้นั้นทำได้เช่นเดียวกับ CGROM คือเขียนข้อมูลลงใน DDRAM แล้วตัวควบคุมจะมาอ่านค่าจาก CGRAM เอง

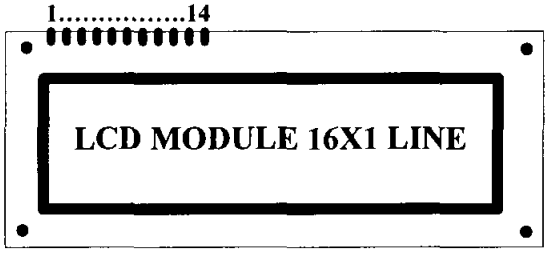
แฟลก (Flag) BUSY เป็นส่วนที่ทำหน้าที่แจ้งสถานะการทำงานของตัวควบคุมให้อุปกรณ์ภายนอกทราบว่าตัวควบคุมพร้อมที่จะรับข้อมูลหรือคำสั่งหรือไม่ ดังนั้นก่อนการส่งข้อมูลหรือคำสั่งมายังตัวควบคุมต้องตรวจสอบสถานะของแฟลก BUSY นี้เสียก่อน

2.8.2 โมดูลแอลซีดี ขนาด 16 ตัวอักษร 1 บรรทัด (LCD 16×1)

สำหรับโมดูลแอลซีดี ที่นำมาใช้ในการทำโครงงานนี้เป็นขนาด 16 ตัวอักษร 1 บรรทัด เนื่องจากราคาถูก หาง่าย และเป็นโมดูลแอลซีดีที่มีโครงสร้างเป็นมาตรฐาน มีผู้ผลิตหลายราย และมีการระบุเบอร์แตกต่างกันออกไปตามผู้ผลิต อาทิ LM020L ของฮิตาชิ (HITASHI), DMC-16117A ของออปเทร็กซ์ (Optrex) เป็นต้น แต่อย่างไรก็ตามคอนโทรลเลอร์ที่ใช้คือเบอร์เดียวกันนั่นคือ HD44750 ของฮิตาชิ

โดยที่แอลซีดีขนาด 16×1 มีขาต่อใช้งานทั้งสิ้น 14 ขา มีการจัดขาดังในรูปที่ 2-16 สำหรับรายละเอียดการทำงานของแต่ละขามีดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.23 รูปร่างและการจัดขาโมดูลแอลซีดีแบบอักขระ

Vss (ขา 1) : ต่อกราวด์

Vdd (ขา 2) : ต่อไฟเลี้ยง +5 โวลท์

Vo (ขา 3) : เป็นขาอินพุตรับแรงดันเพื่อปรับความเข้มของการแสดงผล

RS (ขา 4) : เป็นขาอินพุตใช้ในการแยกชนิดของข้อมูลที่ทำการประมวลผลในขณะนั้นว่าเป็นคำสั่งสำหรับรีจีสเตอร์ IR หรือเป็นข้อมูลสำหรับรีจีสเตอร์ DR โดยถ้าขานี้เป็น "0" ข้อมูลที่ส่งมาจะเป็นคำสั่ง แต่ถ้าขานี้เป็น "1" ข้อมูลที่ส่งมาจะเป็นข้อมูลสำหรับการแสดงผล

R/W (ขา 5) : เป็นขาที่ใช้ในการเลือกการอ่านหรือการเขียนข้อมูลกับโมดูลแอลซีดี ถ้าเป็น "0" เป็นการกำหนดให้เขียนข้อมูล แต่ถ้าเป็น "1" จะเป็นการอ่านข้อมูล

E (ขา 6) : เป็นขาสำหรับรับสัญญาณพัลส์เอ็นเอเบิล (Pulse Enable)โมดูลแอลซีดีให้ทำงาน

D0-D7 (ขา 7-14) : เป็นขาที่ใช้เป็นทางผ่านของข้อมูลระหว่างแอลซีดีกับอุปกรณ์ภายนอกขนาด 8 บิต

ตารางที่ 2.14 แสดงความสัมพันธ์ในการทำงานของ RS, RW และ E ของโมดูลแอลซีดีแบบอักขระ

RS	RW	E	การทำงาน
0	0	↓	เขียนคำสั่ง
0	1	⊓	อ่านสถานะของโมดูลแอลซีดี
1	0	↓	เขียนข้อมูล
1	1	⊓	อ่านข้อมูล

2.8.3 คำสั่งควบคุมโมดูลแอลซีดี

ในการเขียนคำสั่งลงในตัวควบคุม แน่นอนว่าต้องกำหนดให้ขา RS และ RW เป็น "0" แล้วเขียนคำสั่งตามไป คำสั่งควบคุมโมดูลแอลซีดีของชิปควบคุม *HD44780* ที่สำคัญมี 9 คำสั่ง

2.8.3.1 คำสั่งเคลียร์ตัวแสดงผล (clear display)

มีข้อมูลคำสั่งเป็น 01H เป็นคำสั่งที่ใช้เขียนข้อมูลช่องว่างหรือ space เข้าไปใน DDRAM ทั้งหมด เมื่อตัวควบคุมเอ็กซิคิวต์คำสั่งนี้ จะทำการกำหนดแอดเดรสของ DDRAM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็น "0" เคอร์เซอร์จะกลับไปอยู่ที่ตำแหน่งซ้ายมือสุดของจอแสดงผล และเซตบิต I/D (ซึ่งจะกล่าวถึงภายหลัง) ให้เป็น "1"

2.8.3.2 คำสั่ง *return home*

ต้องกำหนดให้บิต 1 ของข้อมูลเป็น "1" เป็นคำสั่งให้เคอร์เซอร์เคลื่อนที่กลับไปยังตำแหน่งซ้ายสุดของจอแสดงผล แต่ข้อมูลบนจอแสดงผลไม่เปลี่ยนแปลง นั่นคือ ข้อมูลคำสั่งของคำสั่งนี้จะเป็น 02H หรือ 03H ก็ได้

2.8.3.3 คำสั่งเลือกโหมดการป้อนข้อมูล (Entry mode Set)

มีรายละเอียดของรูปแบบข้อมูลคำสั่งดังนี้

ตารางที่ 2.15 ตารางแสดงรายละเอียดของรูปแบบข้อมูลคำสั่ง

บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0
0	0	0	0	0	0	I/D	S

บิต S เป็นบิตที่ใช้ในการกำหนดลักษณะของการแสดงผล เมื่อมีการป้อนข้อมูล ถ้าหากบิต S เป็น "1" เมื่อเกิดข้อมูลใหม่บนจอแสดงผลตัวเคอร์เซอร์จะอยู่กับที่ แต่ตัวอักษรข้อมูลเดิมจะถูกดันไปทางซ้าย แต่ถ้าหากบิตนี้เป็น "0" เมื่อเกิดข้อมูลใหม่ตัวเคอร์เซอร์จะเลื่อนไปทางขวามือ

บิต I/D เป็นบิตที่ใช้กำหนดว่า เมื่อเขียนหรืออ่านข้อมูลแล้ว แอดเดรสของ DDRAM เพิ่มขึ้นหรือลดลงหนึ่งแอดเดรส โดยถ้าบิตนี้เป็น "1" แอดเดรสของ DDRAM จะเพิ่มขึ้น แต่ถ้าเป็น "0" แอดเดรสจะลดลง ดังนั้นข้อมูลคำสั่งที่เกิดขึ้นสำหรับคำสั่งนี้ได้แก่ 04H-07H (4 ข้อมูลคำสั่ง) และที่ใช้บ่อยคือ 06H หมายถึงกำหนดให้เมื่อเกิดข้อมูลใหม่ เคอร์เซอร์จะเลื่อนไปทางขวามือ และแอดเดรสของ DDRAM เพิ่มขึ้น

2.8.3.4 คำสั่งควบคุมการแสดงผล

มีรายละเอียดของรูปแบบคำสั่งดังนี้

ตารางที่ 2.16 ตารางคำสั่งควบคุมการแสดงผลของแอลซีดี

บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0
0	0	0	0	0	D	C	B

บิต D ใช้ในการควบคุมการเปิดปิดจอแสดงผล ถ้าบิตนี้เป็น "1" จะเป็นการเปิดจอแสดงผล ถ้าเป็น "0" จะเป็นการปิดจอแสดงผล

บิต C ใช้ควบคุมการแสดงผลตัวเคอร์เซอร์บนจอแสดงผล ถ้าต้องการให้มีเคอร์เซอร์แสดงผลบนจอแสดงผล ต้องกำหนดให้บิตนี้เป็น "1" ถ้ากำหนดให้เป็น "0" จะเป็นการปิดเคอร์เซอร์ หรือไม่แสดงเคอร์เซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บิต B ใช้ควบคุมการกระพริบของเคอร์เซอร์ ถ้าบิตนี้เป็น "1" เคอร์เซอร์จะกระพริบ ดังนั้นจะมีข้อมูลคำสั่งตั้งแต่ 08H-0FH (8 รูปแบบคำสั่ง) ที่ใช้บ่อยคือ 0CH เป็นการสั่งให้เปิดจอแสดงผล แต่ไม่แสดงเคอร์เซอร์ และ 0FH เป็นการสั่งให้เปิดจอแสดงผลแสดงเคอร์เซอร์และสั่งให้เคอร์เซอร์กระพริบ

2.8.3.5 คำสั่งควบคุมการเลื่อนเคอร์เซอร์และข้อมูลตัวอักษร

มีรายละเอียดของรูปแบบข้อมูลคำสั่งดังนี้

ตารางที่ 2.17 แสดงคำสั่งควบคุมการเลื่อนเคอร์เซอร์และข้อมูลตัวอักษร

บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0
0	0	0	1	S/C	R/L	*	*

การควบคุมการเลื่อนเคอร์เซอร์และตัวอักษรบนจอแสดงผล ขึ้นอยู่กับการกำหนดบิต S/C และ R/L ซึ่งสามารถสรุปได้ดังนี้

ตารางที่ 2.18 แสดงลักษณะทิศทางการเลื่อนของเคอร์เซอร์

S/C	R/L	ลักษณะการเลื่อน	ข้อมูลคำสั่ง
0	0	เลื่อนเคอร์เซอร์ไปทางซ้าย	10H-13H
0	1	เลื่อนเคอร์เซอร์ไปทางขวา	14H-17H
1	0	เลื่อนตัวอักษรใหม่ไปทางซ้าย	18H-1BH
1	1	เลื่อนตัวอักษรใหม่ไปทางขวา	1CH-1FH

2.8.3.6 คำสั่งกำหนดฟังก์ชันการทำงาน

มีรายละเอียดของรูปแบบข้อมูลคำสั่งดังนี้

ตารางที่ 2.19 แสดงรูปแบบคำสั่งที่กำหนดฟังก์ชันการทำงานของแอลซีดี

บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0
0	0	1	DL	N	F	*	*

บิต DL ใช้กำหนดจำนวนบิตที่ใช้ติดต่อส่งผ่านข้อมูล ถ้าบิตนี้เป็น "0" จะเป็นการติดต่อแบบ 4 บิต แต่ถ้าเป็น "1" จะเป็นแบบ 8 บิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บิต N ใช้กำหนดจำนวนบรรทัดของการแสดงผล ถ้าเป็น "0" จะแสดงผล 1 บรรทัดถ้าเป็น "1" จะแสดง 2 บรรทัด ในกรณีที่จอแสดงผลสามารถแสดงได้มากกว่า 2 บรรทัด และต้องการให้แสดงผลมากกว่า 2 บรรทัดก็กำหนดบิต N ที่ให้เป็น "1" เนื่องจากแอดเดรสของ DDRAM แบ่งเป็น 2 ช่องคือ 00H และ 40H

บิต F ใช้เลือกความละเอียดของตัวอักษรให้การแสดงผล ถ้าบิตนี้เป็น "0" จะเป็นการแสดงผลแบบ 5×7 จุดและถ้าเป็น "1" จะแสดงผลเป็นแบบ 5×10 จุด ข้อมูลคำสั่งที่ใช้บ่อยคือ 38H เป็นการกำหนดให้โมดูลแอลซีดีทำงานในแบบ 8 บิต แสดงผล 2 บรรทัดและเลือกความละเอียดเป็น 5×7 จุด

2.8.3.7 คำสั่งเลือกแอดเดรสของ CGRAM

เมื่อต้องการกำหนดแอดเดรสของ CGRAM ต้องกำหนดให้บิต 7 เป็น "0" บิต 6 เป็น "1" ส่วนอีก 6 บิตที่เหลือจะแทนด้วยค่าแอดเดรสของ CGRAM จะต้องทำการกำหนดแอดเดรสด้วยคำสั่งนี้ ก่อนที่จะอ่านหรือเขียนข้อมูลให้ CGRAM โดยแอดเดรสของ CGRAM อยู่ระหว่าง 00H-3FH

2.8.3.8 คำสั่งเลือกแอดเดรสของ DDRAM

ใช้ในการเลือกแอดเดรสของ DDRAM ก่อนที่จะทำการอ่านหรือการเขียนข้อมูล โดยบิต 7 ต้องเป็น "1" และข้อมูลอีก 7 บิตที่เหลือจะเป็นค่าแอดเดรสของ DDRAM ซึ่งแอดเดรสของ DDRAM จะอยู่ระหว่าง 8CH-0FFH ทั้งนี้จำนวนแอดเดรสยังขึ้นกับการกำหนดสถานะที่บิต N ด้วย หากบิต N เป็นบิต "0" แอดเดรสของ DDRAM จะอยู่ระหว่าง 80H-0CFH และถ้าบิต N เป็น "1" แอดเดรสของ DDRAM จะมีสองช่วงคือ 8CH-87H และ 0C0H-0C7H

2.8.3.9 คำสั่งอ่านแฟลก BUSY และแอดเดรส

มีรายละเอียดของรูปแบบข้อมูลคำสั่งดังนี้

ตารางที่ 2.20 แสดงรูปแบบข้อมูลคำสั่งในการอ่านแฟลก Busy และแอดเดรส

บิต 7	บิต 6	บิต 5	บิต 4	บิต 3	บิต 2	บิต 1	บิต 0
BF	A	A	A	A	A	A	A

เป็นคำสั่งที่ใช้อ่านแฟลก BUSY (BF) โดยแฟลกนี้จะเป็นตัวบอกสถานะของตัวควบคุมแอลซีดีว่าพร้อมจะรับข้อมูลอยู่หรือไม่ ถ้าหากบิต BF เป็น "0" แสดงว่าตัวควบคุมแอลซีดีพร้อมรับข้อมูลหรือคำสั่ง แต่เป็น "1" แสดงว่า ขณะนี้ตัวควบคุมแอลซีดียังอยู่ในกระบวนการทำงานภายในหรือกำลังประมวลผลข้อมูลอยู่ ยังไม่พร้อมรับข้อมูลหรือคำสั่ง เมื่อต้องการอ่านแฟลกต้องกำหนดให้ขา R/W เป็น "1" ด้วย แต่สัญญาณที่ RS ยังต้องเป็น "0" อยู่ เพราะข้อมูลนี้เป็นข้อมูลคำสั่ง นอกจากนี้ยังใช้เป็นคำสั่งอ่านข้อมูลแอดเดรสของ CGRAM และ DDRAM ด้วย โดยบิต 0 – 6 เป็นค่าข้อมูลของแอดเดรสที่ต้องการอ่าน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.8.4 การเขียนคำสั่งและข้อมูลให้แก่โมดูลแอลซีดี

ในการเขียนข้อมูลเพื่อควบคุมให้โมดูลแอลซีดีแสดงผลตามที่ผู้ใช้งานต้องการ ต้องส่งคำสั่ง (instruction) แล้วกำหนดโหมดการทำงานให้แก่โมดูลแอลซีดีก่อน จากนั้นจึงค่อยส่งข้อมูล (data) ที่ต้องการแสดงผล เนื่องจากบัส (Bus) ข้อมูลแอลซีดีมี 8 เส้นคือ D0-D7 และใช้เป็นทางผ่านของทั้งคำสั่งและข้อมูล ดังนั้นในการส่งคำสั่งและข้อมูลจึงต้องอาศัยการกำหนดสัญญาณลอจิกที่ขา RS ถ้าหากที่ขา RS ได้ลอจิก "1" ข้อมูลที่ป้อนให้ขณะนั้นเป็นข้อมูลที่ใช้ในการแสดงผล

เมื่อต้องการเขียนหรืออ่านข้อมูลใน CGRAM และ DDRAM เริ่มต้นต้องกำหนดแอดเดรสที่ต้องการอ่านหรือเขียนก่อน โดยใช้คำสั่งเลือกแอดเดรส จากนั้นกำหนดให้ RS เป็น "1" เพื่อแจ้งให้ตัวควบคุมภายในโมดูลแอลซีดีทราบว่าข้อมูลที่ปรากฏต่อไปนี้เป็นข้อมูลปกติไม่ใช่คำสั่ง

ในกรณีที่ต้องการอ่านข้อมูลต้องกำหนดให้ขา R/W เป็น "1" ข้อมูลขนาด 8 บิต (หรือ 4 บิต) ก็จะปรากฏบนบัสข้อมูล โดยข้อมูลที่อ่านออกมาได้ จะเป็นข้อมูลจากแอดเดรสของ CGRAM หรือ DDRAM ตามที่ต้องการ

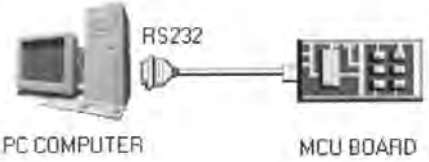
ในกรณีที่ต้องการเขียนข้อมูล เมื่อกำหนดแอดเดรสและป้อนลอจิก "1" ให้ขา RS แล้ว แล้วต้องกำหนดให้ขา R/W เป็น "0" ข้อมูลที่อยู่บนบัสข้อมูลจะถูกเขียนลงในรีจีสเตอร์ DR จากนั้นจึงถ่ายทอดลงใน DDRAM ต่อไป

2.8.5 จังหวะการทำงานของแอลซีดีโมดูล

ในการติดต่อกับโมดูลแอลซีดีจะต้องมีการหน่วงเวลา หลังจากที่ทำการส่งรหัสคำสั่งหรือข้อมูล เนื่องจากต้องรอให้คอนโทรลเลอร์ภายในแอลซีดีโมดูล แปลความหมายของรหัสคำสั่งทำงานตามคำสั่งให้เรียบร้อยก่อนจากนั้นจึงจะรับข้อมูลหรือดำเนินการต่อไป

ดังนั้นในการใช้งานโมดูลแอลซีดีผู้เขียนโปรแกรม ต้องมีโปรแกรมเพื่อหน่วงเวลารอให้โมดูลแอลซีดีพร้อมทำงานด้วย โดยเมื่อเริ่มจ่ายไฟให้แก่โมดูลแอลซีดีต้องรอประมาณ 10 มิลลิวินาที เพื่อให้โมดูลแอลซีดีทำการเตรียมความพร้อมหรืออินิเชียล (initial) หลังจากนั้นก็จะกำหนดลอจิกให้แก่ขา RS ของโมดูลแอลซีดีแล้วต้องหน่วงเวลาอีกประมาณ 2 มิลลิวินาทีเพื่อให้คอนโทรลเลอร์ในโมดูลแอลซีดีแปลความหมายของลอจิกที่ขา RS ว่า ข้อมูลต่อไปที่จะได้รับนั้นเป็นรหัสคำสั่งหรือเป็นข้อมูลที่ต้องการแสดงผล จากนั้นจะเป็นการส่งข้อมูลมารอที่บัสข้อมูล D0 - D7 (กรณีทำงานในโหมด 8 บิต) ขั้นตอนต่อไปจะเป็นการส่งสัญญาณพัลส์ไปที่ขา E เพื่อเอ็นเอเบิลโมดูลแอลซีดีให้รับข้อมูลจากบัสข้อมูลเข้าไป โดยพัลส์ที่ป้อนเข้าที่ ขา E ของโมดูลแอลซีดีต้องเป็นพัลส์ขอบขาขึ้น จากนั้นทำการหน่วงเวลา 2 มิลลิวินาที

2.9 การใช้งานพอร์ตอนุกรม RS232



รูปที่ 2.24 การเชื่อมต่อ COMPUTER กับ MCU BOARD โดย RS232

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การสื่อสารแบบอนุกรม นับว่ามีความสำคัญ ต่อการใช้งาน ไมโครคอนโทรลเลอร์มาก เพราะสามารถใช้เป็นพิมพ์ และจอภาพของ PC เป็น อินพุต และ เอาต์พุต ในการติดต่อ หรือ ควบคุม ไมโครคอนโทรลเลอร์ด้วยสัญญาณอย่างน้อยเพียง3เส้นเท่านั้นคือ

-สายส่งสัญญาณTX

-สายรับสัญญาณRX

-และสายGND

โดยปกติพอร์ตอนุกรม RS-232C จะสามารถต่อสายได้ยาว 50 ฟุตโดยประมาณ ขึ้นอยู่กับ ชนิดของ สายสัญญาณ, ระยะทาง, และ ปริมาณ สัญญาณ รบกวน

พอร์ตอนุกรมของ PC DB9 ตัวผู้ (Male) พอร์ตอนุกรมของอุปกรณ์ภายนอก DB9 ตัวเมีย (Female)

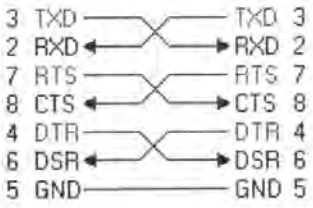
รูปที่ 2.25 พอร์ตอนุกรมของ DB9

พอร์ตอนุกรมของ PC จะเป็นคอนเน็คเตอร์แบบ DB9 ตัวผู้ (Male) ส่วนพอร์ตอนุกรมของอุปกรณ์ภายนอก จะเป็นคอนเน็คเตอร์แบบ DB9 ตัวเมีย(FeMale) ตารางที่ 2.21 แสดงการจัดขา ของ Connector อนุกรมแบบ DB9 และหน้าที่การใช้งานต่างๆ

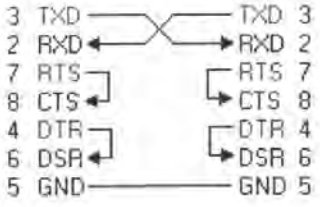
ตารางที่ 2.21 การแสดงขาของ DB9

Pin	Description	Type
1	Data Carrier Detect (DCD)	Input
2	Received Data (RXD)	Input
3	Transmitted Data (TXD)	Output
4	Data Terminal Ready (DTR)	Output
5	Signal Ground (GND)	Input
6	Data Set Ready (DSR)	Input
7	Request To Send (RTS)	Output
8	Clear to Send (CTS)	Input
9	Ring Indicator (RI)	Input

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



การเชื่อมต่ออุปกรณ์ภายนอกผ่าน DB9 แบบ Null modem



การต่ออุปกรณ์ภายนอกผ่าน DB9 แบบ 3 เส้น

รูปที่ 2.26 การเชื่อมต่ออุปกรณ์อุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์ด้วยสาย DB9

2.9.1 การทำงานของขาสัญญาณ DB9

TXD เป็นขาที่ใช้ส่งข้อมูล

RXD เป็นขาที่ใช้รับข้อมูล

DTR แสดงสภาวะพอร์ตว่าเปิดใช้งาน

DSR ตรวจสอบว่าพอร์ต ที่ติดต่อด้วย เปิดอยู่หรือไม่

- เมื่อเปิดพอร์ตอนุกรม ขา DTR จะ ON เพื่อให้อุปกรณ์ได้รับทราบว่าต้องการติดต่อด้วย
- ในขณะเดียวกันก็จะตรวจสอบขา DSR ว่าอุปกรณ์พร้อมหรือไม่

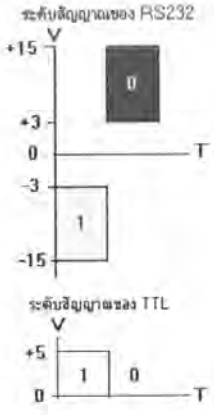
RTS แสดงสภาวะพอร์ตว่าต้องการส่งข้อมูล

CTS ตรวจสอบว่าพอร์ตที่ติดต่ออยู่ ต้องการส่งข้อมูลหรือไม่

- เมื่อต้องการส่งข้อมูลขา RTS จะ ON และจะส่งข้อมูลออกที่ขา TXD เมื่อส่งเสร็จก็จะ OFF
- ในขณะเดียวกันก็จะตรวจสอบขา CTS ว่าอุปกรณ์ต้องการที่จะส่งข้อมูลหรือไม่

GND ขา ground

2.9.2 ระดับสัญญาณของ RS232



รูปที่ 2.27 ระดับสัญญาณของ RS232C และระดับสัญญาณของ TTL

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญญาณรบกวนที่เกิดขึ้น ในสายนำสัญญาณ มักจะมีแรงดันเป็นบวก เมื่อเทียบกับกราวน์

- เพื่อป้องกันสัญญาณรบกวนนี้ จึงออกแบบแรงดัน ของโลจิก "1" เป็นลบ คืออยู่ในช่วง -3V ถึง-15Vส่วนแรงดันของโลจิก"0"อยู่ในช่วง+3Vถึง+15V

- และเหตุที่ ระดับสัญญาณ ของ RS232 อยู่ในช่วง +15V ถึง -15V ก็เพื่อให้ต่อสายสัญญาณไปได้ไกลขึ้น

ดังนั้นจึงจำเป็นจะต้องมีวงจรเปลี่ยนระดับแรงดันของ RS232 มาเป็นระดับแรงดันของ TTL เพื่อที่จะป้องกันสัญญาณจากสัญญาณรบกวน

อัตราการส่งข้อมูล (Baud rate)

- คือความเร็วของการรับ-ส่งข้อมูล เป็นจำนวนบิทต่อวินาทีเช่น 300, 1,200, 2,400, 4,800 , 9,600 ,14,400 ,19,200, 38,400 ,56,000 เป็นต้น

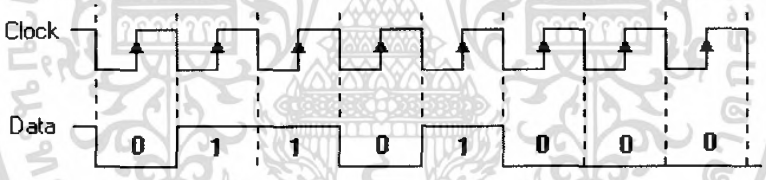
- การเลือกอัตราการส่งข้อมูลขึ้นอยู่กับ ชนิดของสายสัญญาณ, ระยะทาง,และปริมาณสัญญาณรบกวน

รูปแบบการสื่อสารแบบอนุกรม

มีด้วยกันอยู่ 2 แบบ คือแบบซิงโครนัส (Synchronous) และแบบอะซิงโครนัส(Asynchronous)

การสื่อสารแบบซิงโครนัส (Synchronous)

- การรับส่งข้อมูล จะมีสัญญาณนาฬิกา ซึ่งเป็นตัวกำหนด จังหวะเวลา การส่งข้อมูล ร่วมอยู่ด้วยอีกเส้นหนึ่ง ใช้คู่กับสัญญาณข้อมูล ตัวอย่างเช่น การส่งสัญญาณจากคีย์บอร์ด



รูปที่ 2.28 Diagram ของ Clock และ Data ของการสื่อสารแบบซิงโครนัส

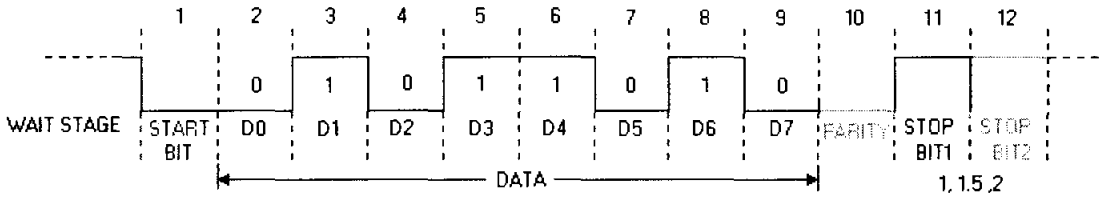
การสื่อสารแบบอะซิงโครนัส (Asynchronous)

- การรับส่งข้อมูล โดยที่ไม่จำเป็นต้อง มีสัญญาณนาฬิกา ร่วมด้วย แต่จะใช้ให้ตัวส่ง และตัวรับ มีอัตราส่งข้อมูล ที่เท่ากัน

รูปแบบข้อมูลแบบอะซิงโครนัส ประกอบด้วย 4 ส่วนคือ

1 บิตเริ่มต้น (Start bit) มีขนาด1 บิต
2 บิตข้อมูล (Data) มีขนาด 5,6,7 หรือ 8 บิต
3 บิตตรวจสอบพาริตี้ (Parity bit) มีขนาด 1 บิตหรือไม่มี
4 บิตหยุด (Stop bit) มีขนาด 1, 1.5, 2 บิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.29 รูปแบบข้อมูลแบบอะซิงโครนัส

- เมื่อไม่มีการส่งข้อมูล ขา data จะมีสถานะเป็นโลจิก "1" หรือ สถานะหยุดรอ (Waiting stage)
- เมื่อเริ่มต้นส่งข้อมูลจะให้ขา data เป็นโลจิก "0" เป็นจำนวน 1 บิต เรียกว่าบิตเริ่มต้น (Start bit)
- จากนั้นก็จะเริ่มต้นส่งข้อมูล โดยส่งบิตต่ำไปก่อน (LSB)
- แล้วตามด้วยพาริตี้บิต (จะมีหรือไม่ก็ได้ ขึ้นอยู่กับการติดตั้งค่า ของทั้งสองฝ่าย)
- สุดท้ายตามด้วยโลจิก "1" อย่างน้อย 1 บิต (มีขนาด 1, 1.5, หรือ 2 บิต) เพื่อแสดงว่าสิ้นสุดข้อมูล

การรับและส่งข้อมูลแบบอนุกรมยังแบ่งออกเป็นลักษณะการใช้งานได้ 3 แบบคือ

1. แบบซิมเพลกซ์ (Simplex) เป็นการส่ง หรือรับข้อมูล แบบทิศทางเดียว เท่านั้น
2 แบบฮาล์ฟดูเพลกซ์ (Half Duplex) เป็นการส่งและรับข้อมูลแบบสลับกัน คือเมื่อด้านหนึ่งส่ง อีกด้านหนึ่งเป็นฝ่ายรับ สลับกัน ไม่สามารถรับ-ส่งในเวลาเดียวกันได้
3. แบบฟูลดูเพลกซ์ (Full Duplex) สามารถรับ-ส่งข้อมูลในเวลาเดียวกันได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.10 ออฟโต้ไทรแอค

ออฟโต้ไทรแอคใช้ในการควบคุมปั๊มพ์อัดอากาศที่จะช่วยในการเพิ่มค่าออกซิเจนในน้ำ โดยออฟโต้ไทรแอคจะมีคุณสมบัติดังนี้

ค่าอัตตราขนาดสูงสุด

ตารางที่ 2.22 อัตราขนาดสูงสุดของออฟโต้ไทรแอค

พารามิเตอร์		สัญลักษณ์	อัตรา	หน่วย
อินพุท	กระแส forward	I_F	50	mA
	กระแส forward สูงสุด	I_{FM}	1	A
	แรงดันต้านกลับ	V_R	6	V
	กำลังไฟฟ้า	P	70	mW
เอาต์พุท	แรงดัน collector-emitter	V_{CEO}	35	V
	แรงดัน emitter-collector	V_{ECO}	6	V
	กระแสขา collector	I_C	50	mA
	กำลังไฟฟ้าที่ขา collector	P_C	150	mW
	กำลังไฟฟ้ารวมทั้งหมด	P_{TOT}	200	mW
	แรงดัน isolation	V_{ISO}	5	KVrms
	อุณหภูมิขณะปฏิบัติการ	T_{OPR}	-30 ถึง +100	C°
	อุณหภูมิเก็บสะสม	T_{STG}	-55 ถึง +125	C°
	อุณหภูมิในการหลอมละลาย	T_{SOL}	260	C°

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณลักษณะทางไฟฟ้าและทางออฟติคัล

ตารางที่ 2.23 แสดงคุณลักษณะต่างๆของตัวออฟโต้ไทรแอค

พารามิเตอร์		สัญลักษณ์	สภาพ	ต่ำสุด	TYP	สูงสุด	หน่วย
อินพุท	แรงดัน forward	V_F	I_f =20mA	-	1.2	1.4	V
	แรงดัน forward สูงสุด	V_{FM}	I_{fM} =0.5A	-	-	3.0	V
	กระแสย้อนกลับ	I_R	V_R = 4V	-	-	10	µs
	ค่าตัวเก็บประจุที่เทอร์มินัล	Ct	V=0,f=1KHz	-	30	250	pF
	กระแสขา collector-emitter	I_{CEO}	V_{CE} =20V,I=0	-	-	100	nA
เอาต์พุท	แรงดันเบรกดาวน์ collector-emitter	BV_{CEO}	I_c=0.1mA, I_f =0	35	-	-	V
	แรงดันเบรกดาวน์ emitter- collector	BV_{ECO}	I_E=10 µA, I_f =0	6	-	-	V
	กระแสขา collector	Ic	I_f=5mA,V_{CE}=5V	2.5	-	30	mA
คุณลักษณะการส่งผ่าน	แรงดันแซททูเรชั่น collector-emitter	$V_{CEO(SAT)}$	I_f=20mA, I_C= 1mA	-	0.1	0.2	V
	ความต้านทาน ไอโซเลชั่น	R_{ISO}	DC500V 40-60% RH	$5*10^{10}$	10^{11}	-	Ω
	ตัวเก็บประจุโฟลทติ้ง	Cf	V=0,f= 1MHz	-	0.6	1.0	Pf
	ความถี่คัทออฟ	Fc	V_{CE} =5V, I_c=2mA,RL=100Ω, -3dB	-	80	-	KHz
	เวลา Rise time	Ts	V_{CE} =2V, I_c=2mA,RL=100Ω	-	4	18	µs
	เวลา Fall time	Tr		-	3	18	µs

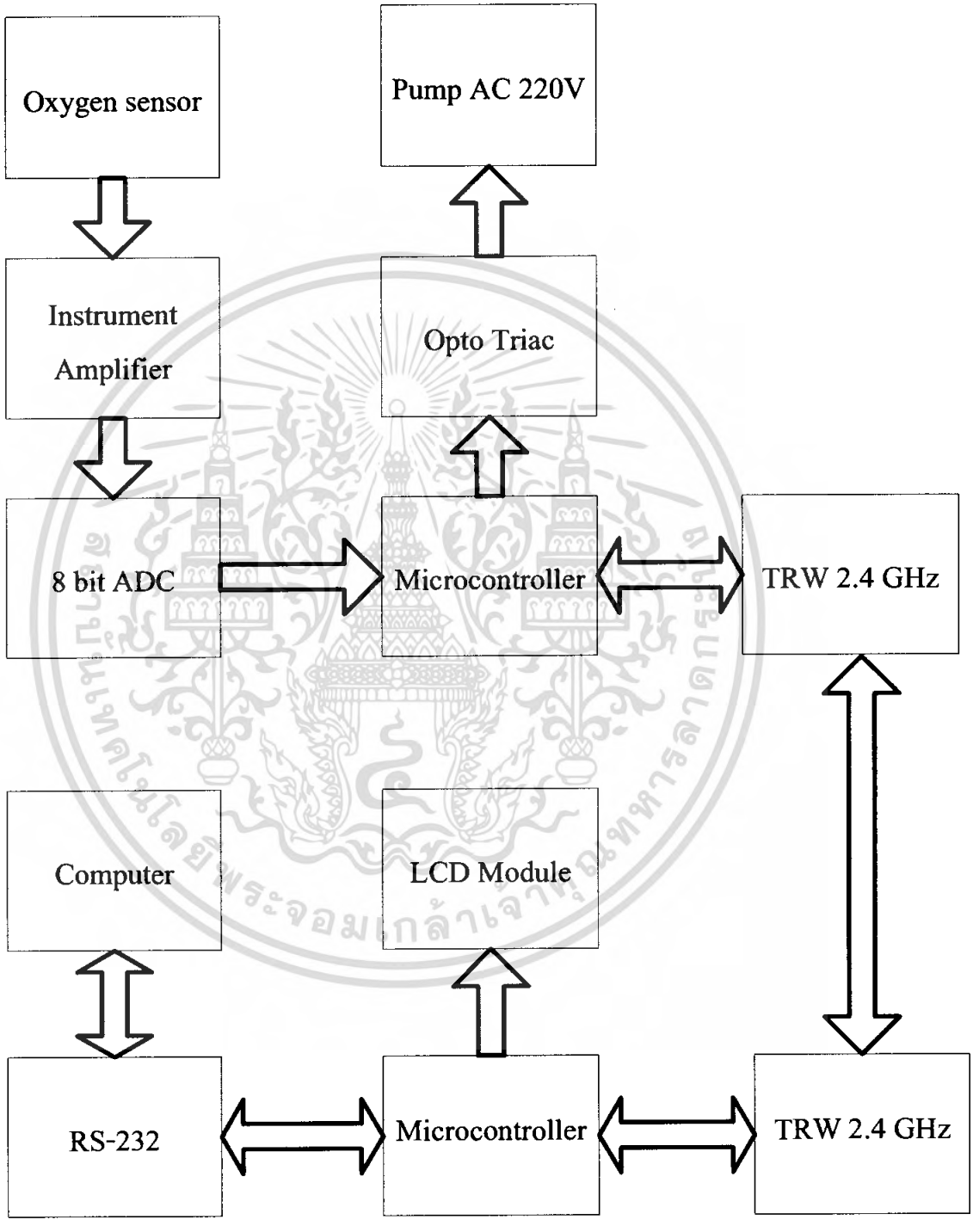
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การออกแบบวงจร

บทนี้จะกล่าวถึงรายละเอียดต่างๆในชิ้นงานโดยจะแบ่งชิ้นงานออกเป็นส่วนต่างๆดังนี้

3.1การออกแบบโครงงาน



รูปที่ 3.1 Block Diagram ของโครงงานทั้งหมด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โครงงานทั้งหมดแสดงในรูปที่ 3.1 หลักการทำงานของระบบน้ำเสียนี้คือ เมื่อตัวตรวจวัดค่าออกซิเจน วัดค่าปริมาณออกซิเจนในน้ำจะส่งค่าแรงดันไฟฟ้าซึ่งมีหน่วยเป็น mV (small signal) มายังที่ตัว Amplifier ซึ่งจะทำหน้าที่ขยายแรงดัน จากนั้นสัญญาณเข้ามายังตัวไอซีเบอร์ ADC0832 ที่จะแปลงสัญญาณอนาล็อกให้เป็นสัญญาณดิจิตอล แล้วทำการป้อนสัญญาณไปยังตัวไมโครคอนโทรลเลอร์เบอร์ P89LV51RD2BN เพื่อที่จะทำการส่งสัญญาณผ่านตัว TRW-2.4GHz และเมื่อสัญญาณออกจากตัว TRW-2.4 GHz ตัว TRW-2.4 GHz อีกตัวหนึ่งของอีกฝั่งหนึ่งจะทำการรับสัญญาณและส่งสัญญาณต่อไปยังตัวไมโครคอนโทรลเลอร์ เพื่อที่จะส่งสัญญาณไปยังตัว LCD Module เพื่อแสดงผลค่าออกซิเจนในน้ำและค่า Set Point ออกมา และสัญญาณจากตัวคอนโทรลเลอร์นี้ก็จะถูกส่งไปยังวงจร RS-232เพื่อจะส่งต่อไปยังตัวคอมพิวเตอร์เพื่อที่จะแสดงผลและตั้งค่า Set Point และจะส่งสัญญาณกลับไปเพื่อใช้ในการควบคุมปั๊มอีกที

จากรูปที่ 3.1 จะเห็นได้ว่า ไมโครคอนโทรลเลอร์ทั้งสองฝั่ง ตัว TRW-2.4 GHzทั้งสองฝั่ง รวมถึงวงจร RS-232และคอมพิวเตอร์นั้นสื่อสารกันแบบสองทิศทางคือสามารถทั้งรับ-ทั้งส่งได้ ซึ่งจะมีหลักการทำงานเหมือน Master-Slave

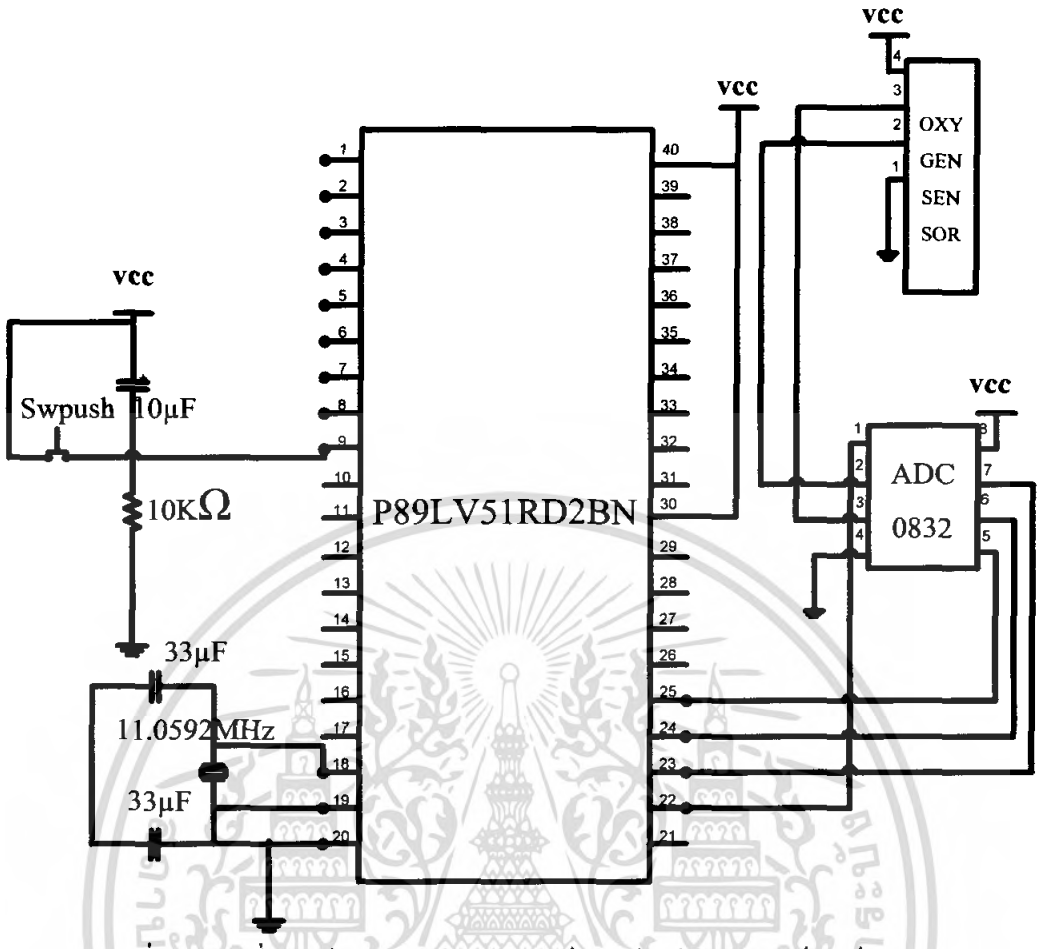
โดยเราจะให้ฝั่งที่ต่ออยู่กับคอมพิวเตอร์เป็นฝั่ง “Master” เพราะจะเป็นตัวสั่งการ ส่วนอีกฝั่งก็จะเป็นฝั่ง “Slave” เพราะจะเป็นตัวรับคำสั่ง

3.2การออกแบบวงจรทางด้าน Slave

3.2.1 การเชื่อมต่อไมโครคอนโทรลเลอร์กับตัวไอซีแปลงอนาล็อกเป็นดิจิตอล

การเชื่อมต่อระหว่างไมโครคอนโทรลเลอร์ของทางด้าน Slave กับไอซีแปลงสัญญาณอนาล็อกเป็นดิจิตอล ขาของตัวไอซีแปลงสัญญาณอนาล็อกเป็นดิจิตอลคือ CS, CLK , DO และ DI จะต่อเข้ากับพอร์ต 2.1-2.4 ของไมโครคอนโทรลเลอร์ ตามลำดับ โดยส่วนขา Ch1 และ Ch2 จะรับเอาท์พุทของสัญญาณจากตัวตรวจวัดปริมาณออกซิเจน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

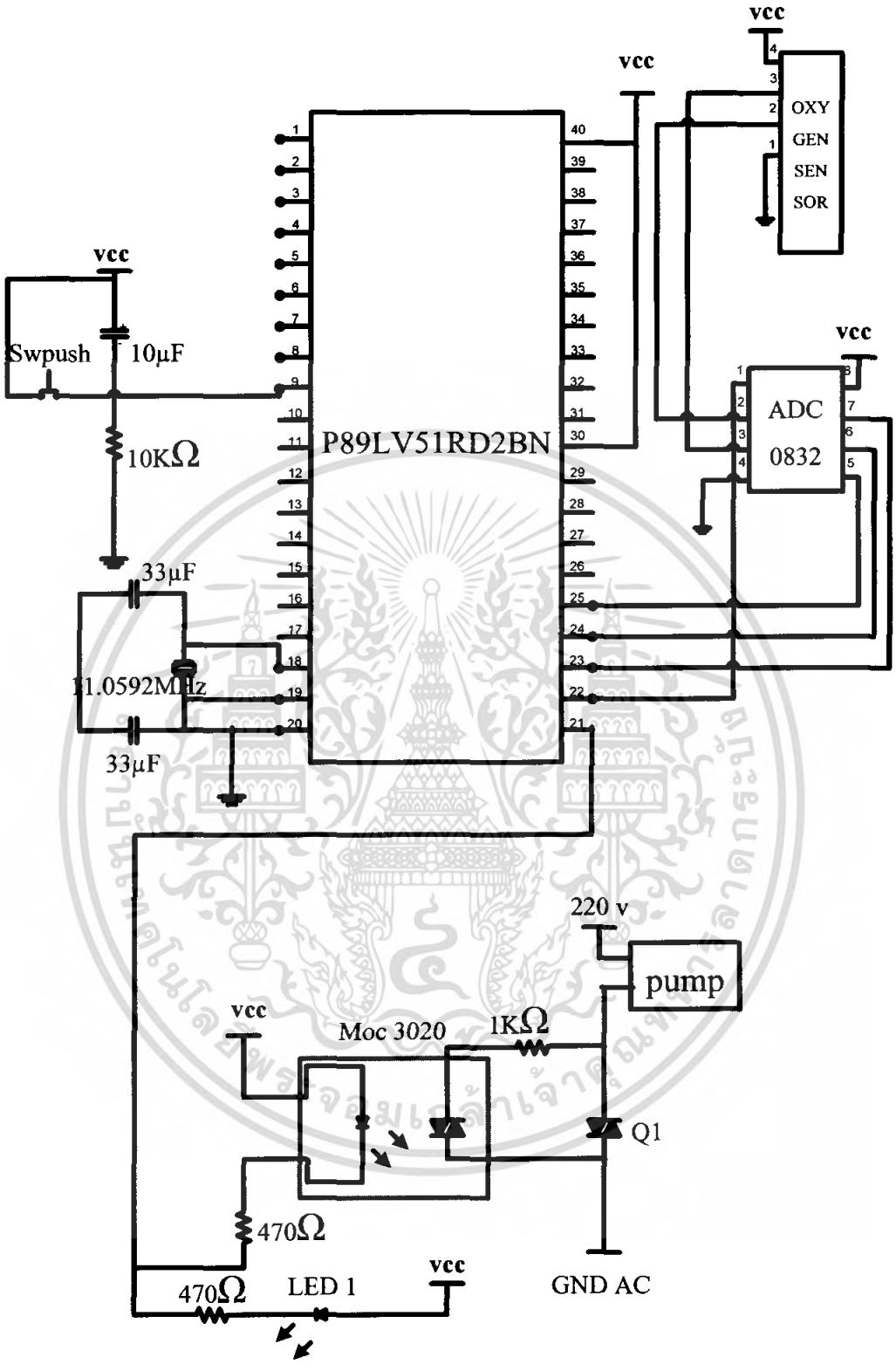


รูปที่ 3.2 การเชื่อมต่อไมโครคอนโทรลเลอร์กับตัวไอซีแปลงอนาล็อกเป็นดิจิตอล

3.2.2 การเชื่อมต่อไมโครคอนโทรลเลอร์กับ วงจรควบคุมปั้ม

ออฟโต้ไทรแอคเป็นอุปกรณ์ที่ใช้ควบคุมปั้ม และเนื่องจากเป็นอุปกรณ์ที่กินกระแสสูง จึงต้องใช้ในไมโครคอนโทรลเลอร์ที่ทำงานที่ Active Low จึงจะสามารถทำงานได้

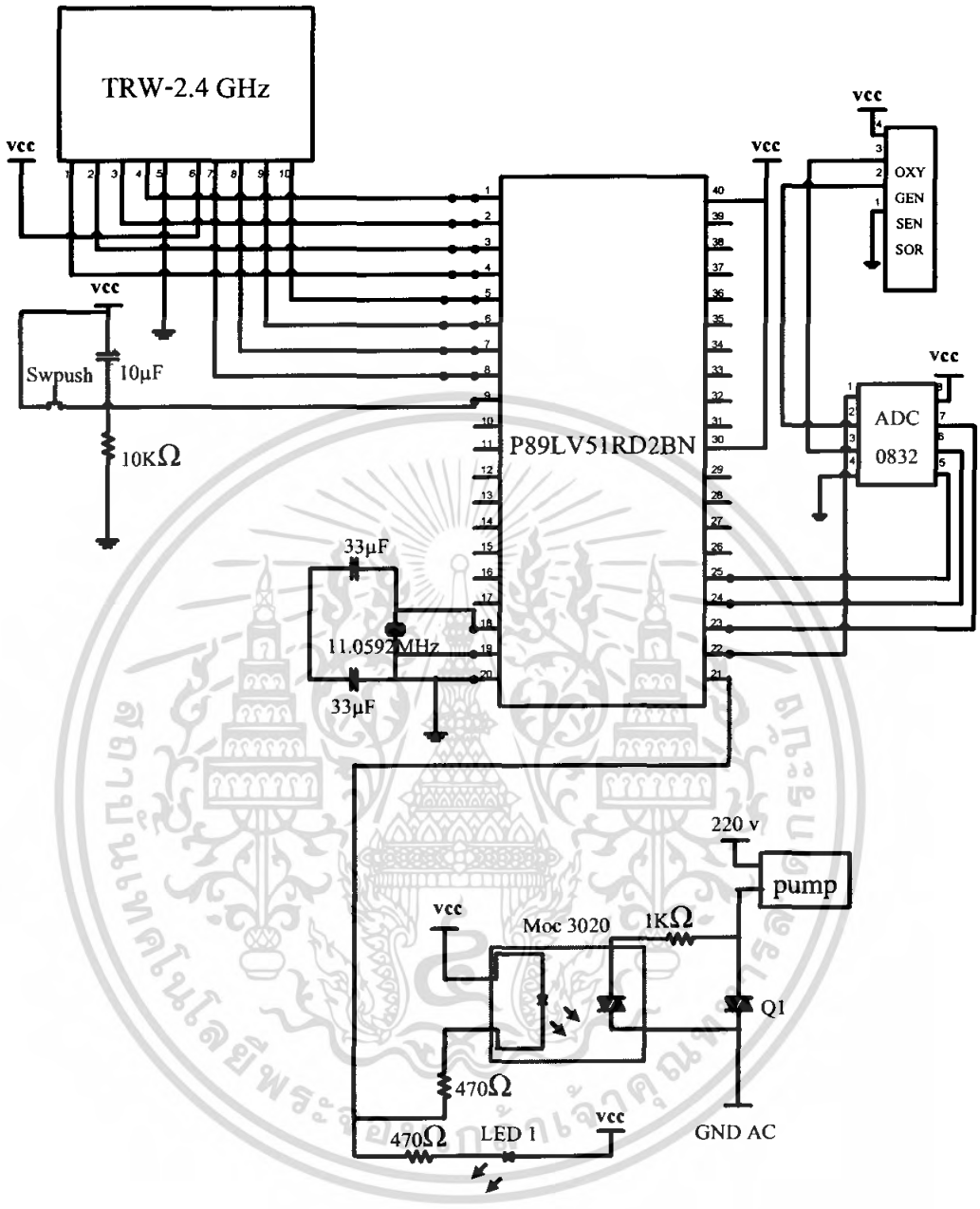
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.3 การเชื่อมต่อของไมโครคอนโทรลเลอร์กับ วงจรควบคุมปั๊ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.3 การเชื่อมต่อของไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz ทางด้าน Slave



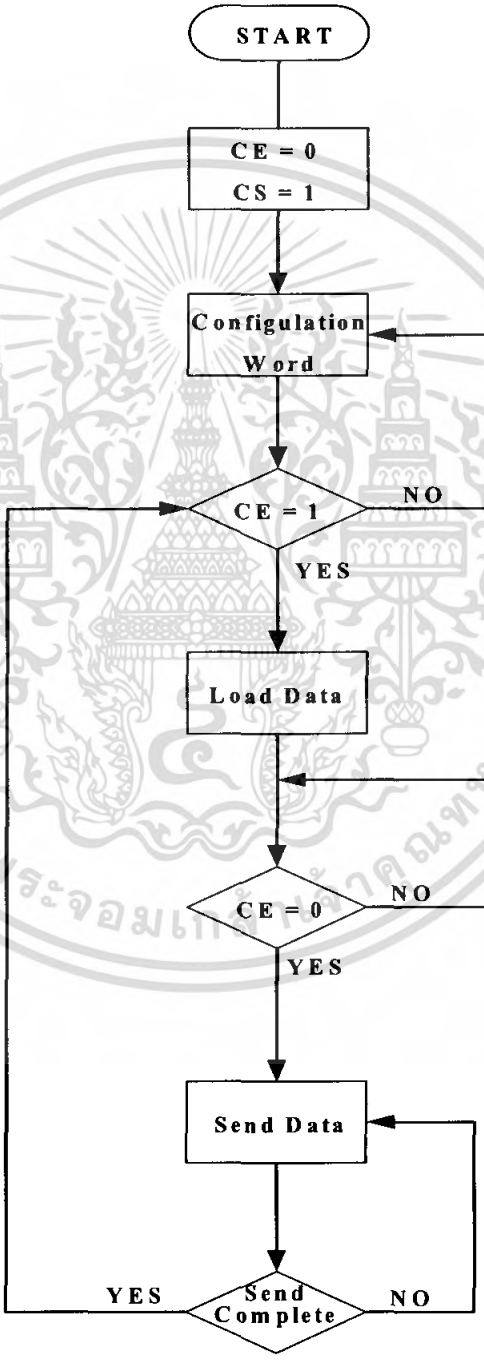
รูปที่ 3.4 การเชื่อมต่อของไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz ทางด้าน Slave

จากรูปที่ 3.4 เป็นการเชื่อมต่อโมดูลความถี่วิทยุกับ ไมโครคอนโทรลเลอร์ ผ่านทางพอร์ต P1.0-P1.7 ซึ่งโมดูลความถี่วิทยุเป็นตัวส่งข้อมูล โดยใช้การมอดูเลต (Modulate) ข้อมูลแบบ GFSK (Gaussian Frequency Shift Keying) ซึ่งสามารถกำหนดอัตราการส่งผ่านข้อมูลได้ โดยต้องใช้การโปรแกรมให้กับตัวไมโครคอนโทรลเลอร์ โดยลักษณะของโปรแกรมที่ใช้เขียนควบคุมตัวโมดูลความถี่วิทยุด้าน Slave ออกแบบตามผังงานรูปที่ 3.5

จากรูปที่ 3.5 เป็นผังงานในส่วนควบคุมโมดูลความถี่วิทยุทางด้าน Slave ซึ่งมีลักษณะการทำงานคือต้องทำการกำหนดค่าให้กับโมดูลความถี่วิทยุโดยทำการกำหนดค่าลงในเฟรมข้อมูล เพื่อให้ตัวโมดูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความถี่วิทยุทราบก่อนว่าต้องการให้โมดูลความถี่วิทยุทำงานเป็นตัวส่งก่อน ซึ่งมีจำนวนทั้งหมด 144 บิต ในการที่จะทำการส่งเฟรมข้อมูลต้องทำการกำหนดให้ขา CE มีสถานะ " low " และ CS มีสถานะ " high " ก่อน เมื่อทำการกำหนดเรียบร้อยแล้ว เมื่อต้องการที่จะส่งข้อมูลต้องทำการป้อนข้อมูลให้กับโมดูลความถี่วิทยุก่อน โดยต้องตั้งค่าให้ CE มีสถานะ " high " เพื่อโหลดข้อมูลไปเก็บไว้ภายในตัวโมดูลความถี่วิทยุหลังจากนั้นเมื่อส่งข้อมูลให้กับโมดูลความถี่วิทยุเรียบร้อยแล้วให้ตั้งค่า CE ให้อยู่ในสถานะ " low " เพื่อกระตุ้นให้โมดูลความถี่วิทยุทำการส่งข้อมูลออกไป

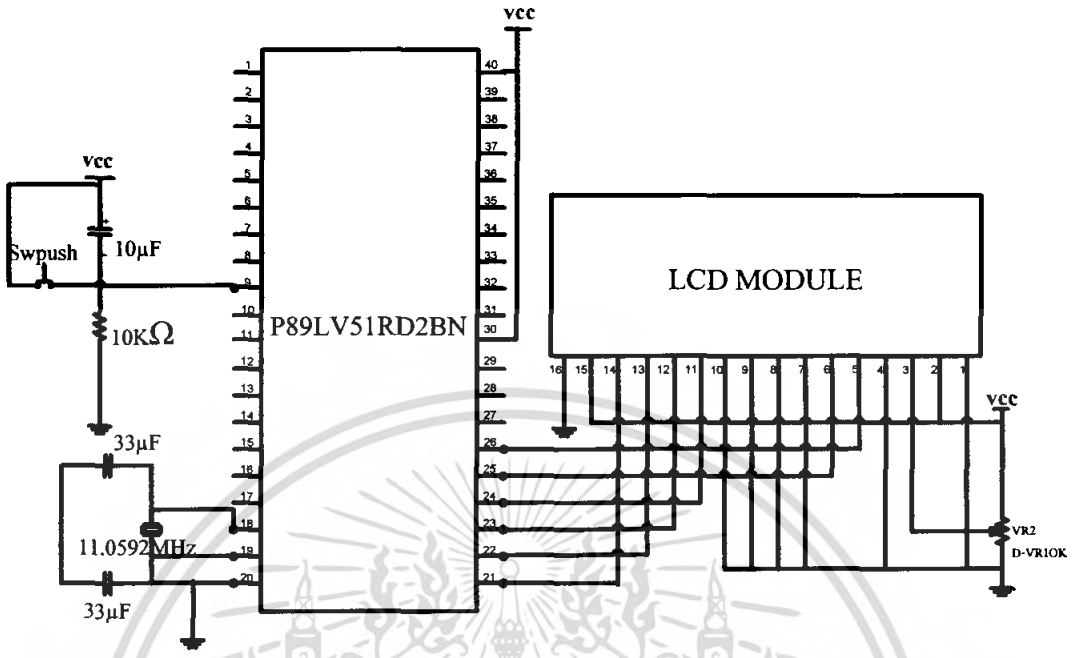


รูปที่ 3.5 รูปแบบผังงานแสดงการเขียนโปรแกรมควบคุมโมดูลความถี่ TRW-2.4 GHz ทางด้าน Slave

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3การออกแบบวงจรทางด้าน Master

3.3.1 การเชื่อมต่อไมโครคอนโทรลเลอร์กับตัวแอลซีดีโมดูล



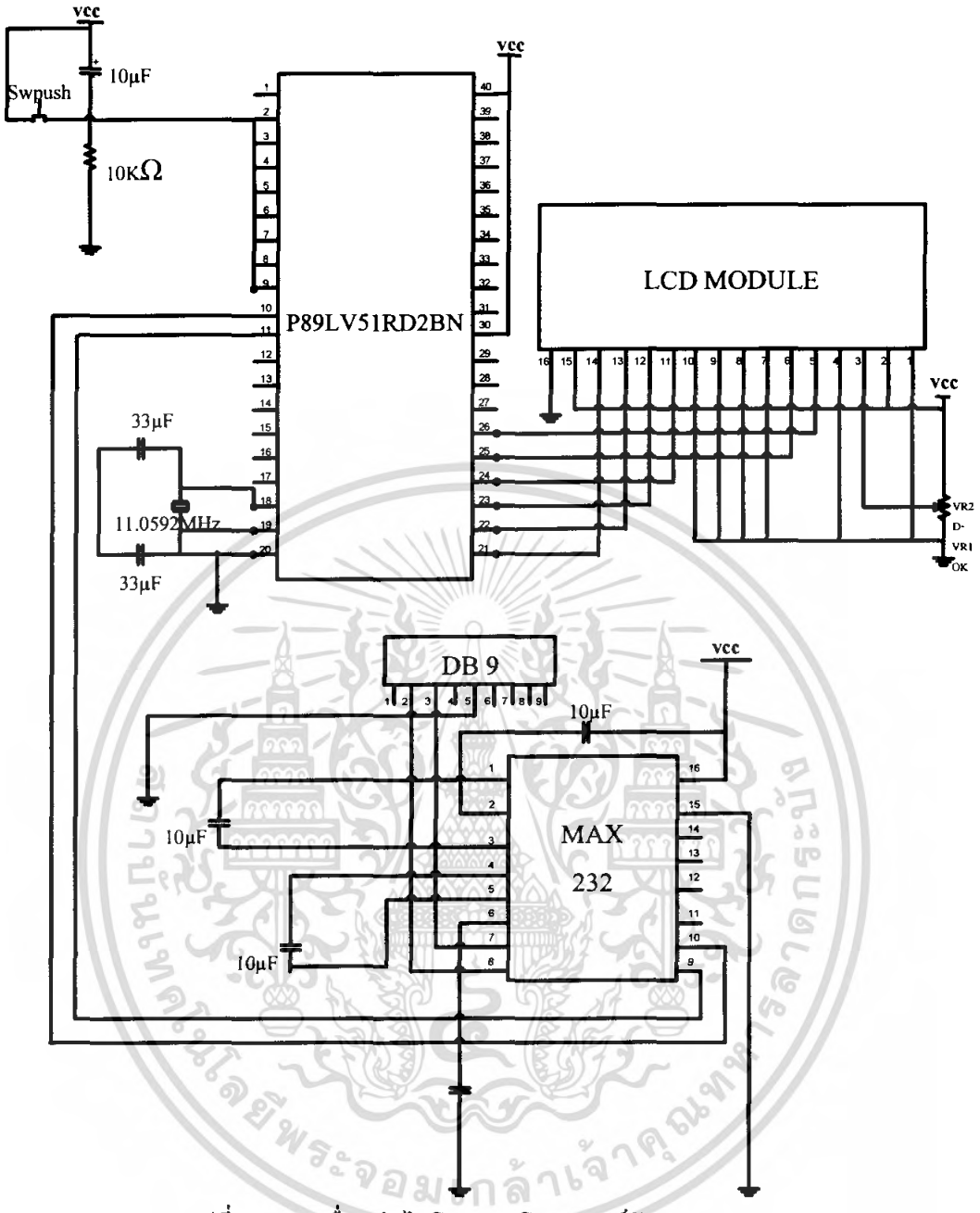
รูปที่ 3.6 การเชื่อมต่อไมโครคอนโทรลเลอร์กับตัวแอลซีดีโมดูล

LCD ที่เชื่อมต่อกับพอร์ต 2 ของไมโครคอนโทรลเลอร์ และสำหรับสัญญาณข้อมูลที่ใช้คือ D4 - D7 สามารถส่งข้อมูลสลับกันได้ระหว่างบิท high และบิท low สลับกันโดยจะเข้าที่ P2.0-P2.3 ไมโครคอนโทรลเลอร์ ในส่วนสัญญาณ , R และ E จะเชื่อมต่อกับ P2.4 , P2.5 ตามลำดับ

3.3.2 การเชื่อมต่อไมโครคอนโทรลเลอร์กับวงจร RS 232

เป็นการเชื่อมต่อเพื่อที่จะนำไปแสดงผลยังคอมพิวเตอร์โดยจะใช้ พอร์ตที่ 3.0 และพอร์ตที่ 3.1 โดยต่อเข้ากับขาเก้าและขาสิบของ ไอซี MAX 232 นอกจากนั้น ไอซี MAX 232 ยังต่อเข้ากับขา สองและขาสามของตัว DB9 เพื่อที่จะส่งสัญญาณไปแสดงผลยังหน้าจอคอมพิวเตอร์

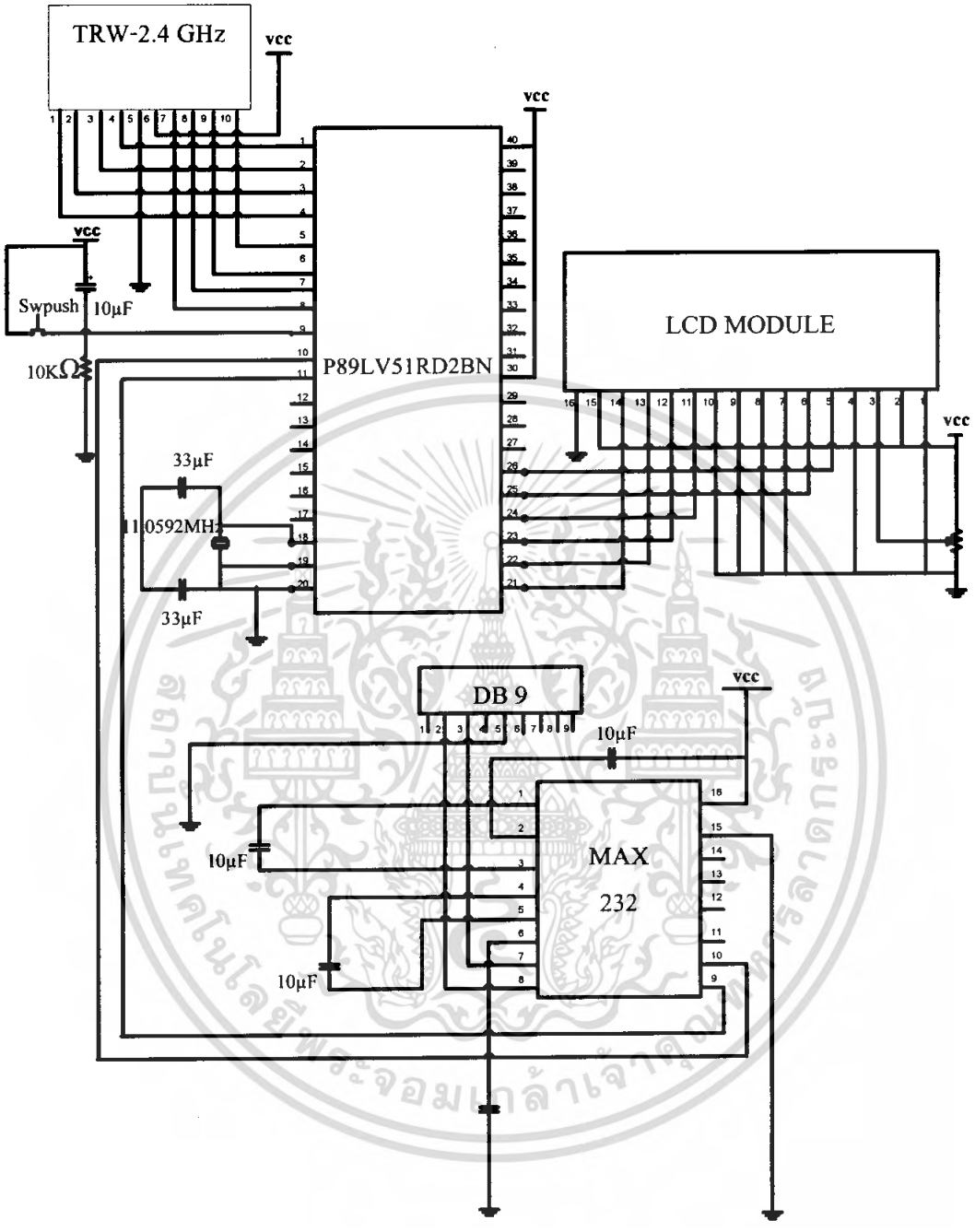
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.7 การเชื่อมต่อไมโครคอนโทรลเลอร์กับวงจร RS-232

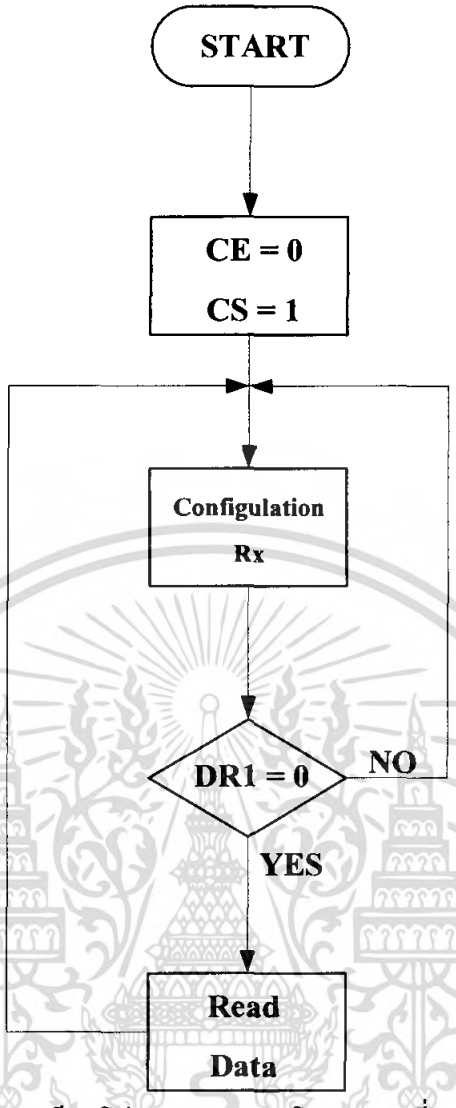
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3.3 การเชื่อมต่อไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz



รูปที่ 3.8 การเชื่อมต่อไมโครคอนโทรลเลอร์กับ TRW-2.4 GHz ทางด้าน Master

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.9 รูปแบบผังงานแสดงการเขียนโปรแกรมควบคุมโมดูลความถี่ TRW-2.4 GHz ทางด้าน Master

จากรูปที่ 3.9 เป็นผังงานในส่วนควบคุมโมดูลความถี่วิทยุทางด้าน Master ซึ่งมีลักษณะการทำงานคือต้องทำการกำหนดค่าให้กับโมดูลความถี่วิทยุโดยทำการกำหนดค่าลงในเฟรมข้อมูล เพื่อให้ตัวโมดูลความถี่วิทยุทราบก่อนว่าต้องการให้โมดูลความถี่วิทยุทำงานเป็นตัวรับก่อน ซึ่งมีจำนวนทั้งหมด 144 บิต ในการที่จะทำการส่งเฟรมข้อมูลต้องทำการกำหนดให้ขา CE มีสถานะ " low " และ CS มีสถานะ " high " ก่อน เมื่อทำการกำหนดเรียบร้อยแล้ว ทางด้านฝ่าย Master จะทำการรอรับข้อมูลโดยการตรวจเช็ค

ขา DR1 ซึ่งขา DR1 เป็นค่าที่แสดงความพร้อมที่จะรับข้อมูลเมื่ออยู่ในสถานะ " low " ดังนั้นเมื่อขา DR1 ของโมดูลความถี่วิทยุอยู่ในสถานะ " low " เมื่อมีข้อมูลเข้ามาจึงสามารถรับข้อมูลได้ทันที ในขณะทำการรับข้อมูล DR1 จะอยู่ในสถานะ " high " เมื่อรับข้อมูลเรียบร้อยแล้ว DR1 จะกลับมาอยู่ในสถานะ " low " อีกครั้งเพื่อเตรียมพร้อมที่จะรับข้อมูลที่จะมีเข้ามาใหม่ โดยข้อมูลที่รับมาจะทำการป้อนให้กับไมโครคอนโทรลเลอร์เพื่อให้ไมโครคอนโทรลเลอร์ทำการประมวลผลและทำการแสดงผลต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

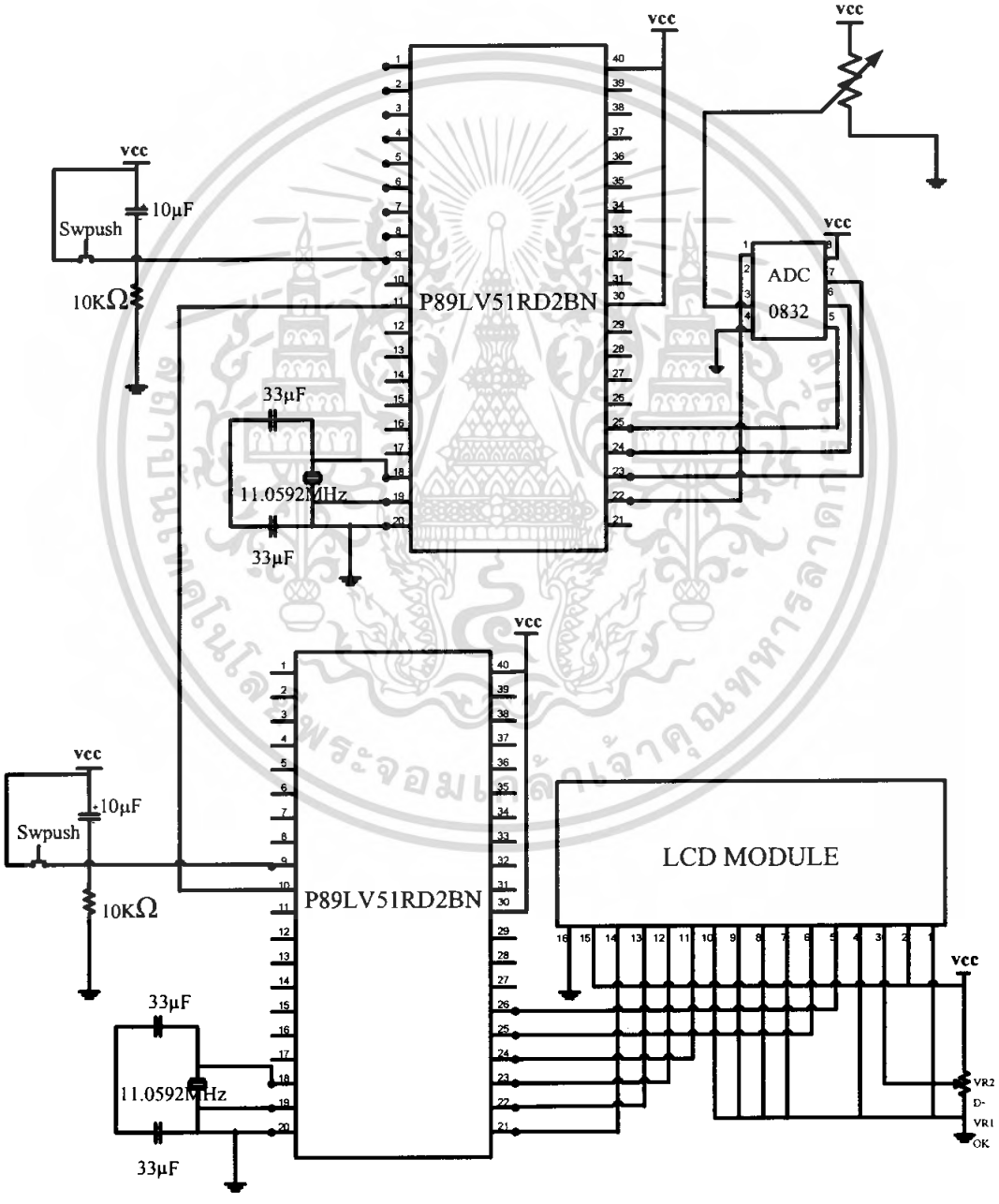
ผลการทดลอง

4.1 การทดลองวัดค่าแรงดันไฟฟ้าผ่านตัวไอซีแปลงสัญญาณอนาล็อกเป็นดิจิตอล

4.1.1 วัตถุประสงค์

เพื่อจำลองว่าถ้าแรงดันไฟฟ้าที่เป็นอนาล็อกออกจากตัวตรวจจับปริมาณออกซิเจนและผ่านตัวขยายสัญญาณ (Instrument Amplifier) มาเข้าตัวไอซีแปลงสัญญาณอนาล็อกเป็นดิจิตอล สัญญาณนั้นจะเป็นดิจิตอลและสามารถส่งไปให้ไมโครคอนโทรลเลอร์ประมวลผลได้ต่อไป

4.1.2 วิธีการทดลอง

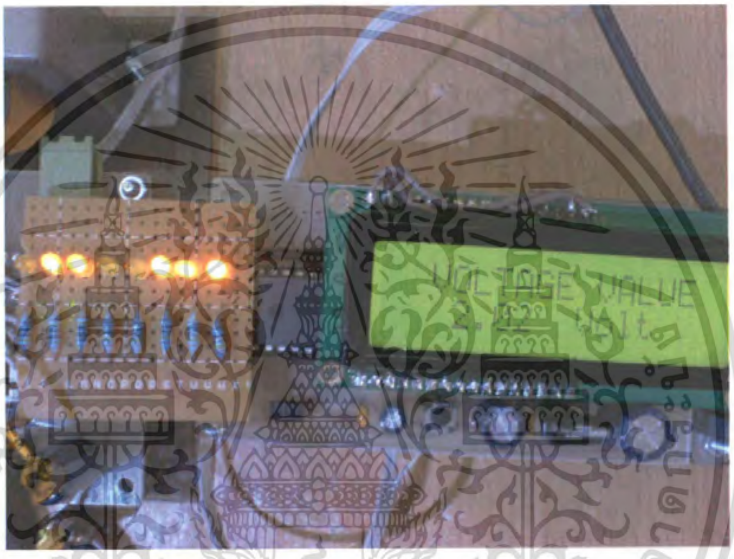


รูปที่ 4.1 การทดลองวัดค่าแรงดันไฟฟ้าผ่านตัวไอซีแปลงสัญญาณอนาล็อกเป็นดิจิตอล

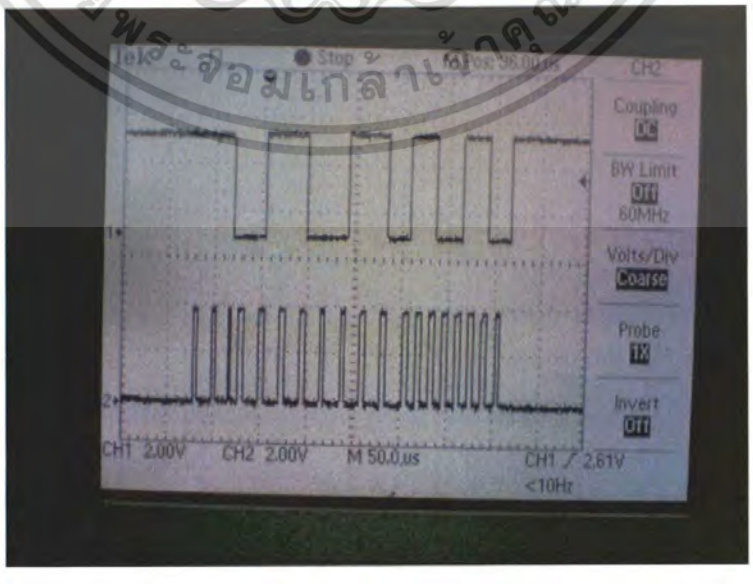
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อวงจรตามรูปที่ 4.1 โดยที่ใช้ไมโครคอนโทรลเลอร์ของภาค Slave ส่งสัญญาณ ซึ่งจะรับสัญญาณอินพุทมาจากไอซีตัวแปลงสัญญาณอนาล็อกเป็นดิจิตอลแล้วส่งต่อไปยังไมโครคอนโทรลเลอร์ของทางด้าน Master ไมโครคอนโทรลเลอร์ ก็จะส่งค่าไปแสดงผลยังตัว LCD โมดูล

โดยที่เราจะตั้งค่าให้ LCD โมดูลแสดงค่าแรงดันไฟฟ้าไว้ที่ 0-5 โวลท์ ขึ้นอยู่ที่เราจะปรับค่าแรงดันไว้ที่เท่าไหร่ สำหรับไอซีแปลงสัญญาณอนาล็อกเป็นสัญญาณดิจิตอลเบอร์ ADC0832 นี้นั้นค่าตัวเลขจะถูกกำหนดด้วยจำนวนบิตในการเข้ารหัส 8 บิตต่อ 1 ข้อมูลตัวเลข ซึ่งบิตจะมีการเปลี่ยนแปลงค่าได้ 2 ค่าคือ 0 กับ 1 ดังนั้นจึงแบ่งระดับการควอนไทซ์ได้ทั้งหมดเท่ากับ $2^8 = 256$ ระดับคือค่า 0-255 เพราะฉะนั้นถ้าเราต้องการให้เป็น 0-5 โวลท์ จะนำตัวเลข 51 มาหาร เพื่อทำให้ค่า 255 มีค่าสูงสุดอยู่ที่ 5 โวลท์



รูปที่ 4.2 ค่าผลการทดลองที่ 4.1



รูปที่ 4.3 ขาสัญญาณ CLK และ Data Out

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การวัดเปรียบเทียบค่า Dissolved Oxygen Sensor KDS25B กับเครื่อง Dissolved Oxygen meter

4.2.1 วัตถุประสงค์ในการทดลอง

- ทดสอบการทำงานของ Sensor
- เพื่อให้ทราบค่าแรงดันของ Sensor ที่ส่งออกมา
- เพื่อเปรียบเทียบค่าของ Sensor กับ Oxygen meter โดยตรง

4.2.2 ขั้นตอนการทดลอง

- ติดตั้งอุปกรณ์ เพื่อเปรียบเทียบกับ Oxygen meter
- เริ่มวัดค่าน้ำธรรมดา โดยเทียบ Sensor กับ meter
- เติมสาร Sodium Sulphite
- เขียนตารางเปรียบเทียบค่าระหว่าง Sensor กับ Oxygen meter
- ได้ค่าเปรียบเทียบค่าระหว่าง Sensor กับ Oxygen meter

4.2.3 ผลการทดลอง

ตารางที่ 4.1 ผลการทดลองการวัดเปรียบเทียบค่า Dissolved Oxygen Sensor KDS25B กับเครื่อง Dissolved Oxygen meter

DO SENSER (mv)	DO METER (mg/l)
10.4	9.85
8.3	7.9
7.1	6.9
6.0	5.8
5.1	5.0
4.0	3.9
3.2	3.0
2.1	2.1
1.2	0.9
0.1	0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4.4 เครื่อง Dissolved Oxygen meter

รูปที่ 4.5 สาร Sodium Sulphite

รูปที่ 4.6 วัดเปรียบเทียบค่า Dissolved Oxygen Sensor KDS25B กับเครื่อง Dissolved Oxygen meter

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3 การทดลองแสดงค่าปริมาณออกซิเจนผ่านทางโปรแกรม Visual basic

4.3.1 วัตถุประสงค์การทดลอง

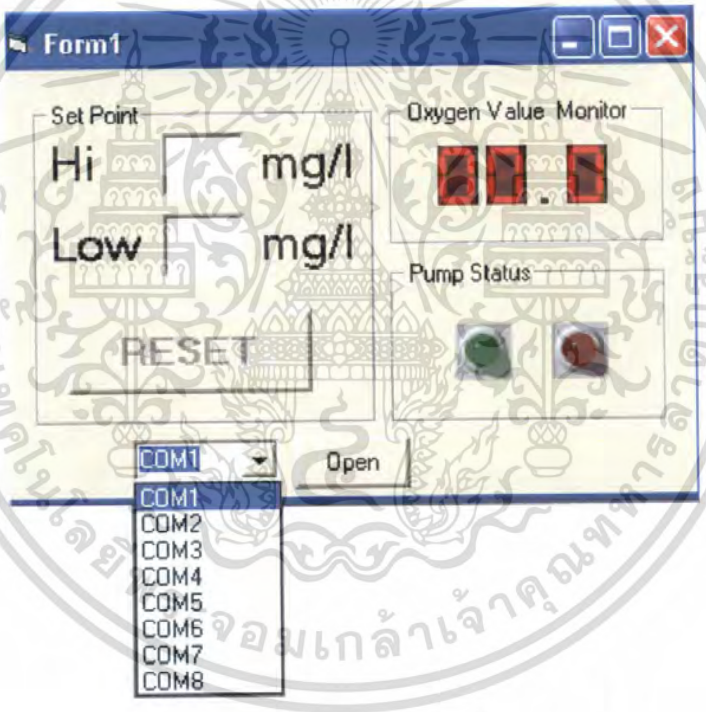
การทดลองนี้ถือเป็นจุดประสงค์หลักของโครงงานที่จะแสดงผลค่าปริมาณออกซิเจนภายในน้ำออกมาทางโปรแกรม Visual basic โดยสามารถที่จะควบคุมปั้มให้ทำงานได้ตามต้องการ

4.3.2 ขั้นตอนการทดลอง

ต่อวงจรของทางด้าน Slave และ Master ตามรูปที่ 3.4 และรูปที่ 3.8 สัญญาณจะส่งผ่านกันไปมาผ่านตัว TRW-2.4GHz ของทั้งสองด้าน ทำให้มีการติดต่อสื่อสารข้อมูลกัน ผลของค่าออกซิเจนและค่า Set Point ต่างๆจะแสดงทางด้านโปรแกรม Visual Basic และทางด้าน LCD

การเปิด Port

ก่อนที่เราจะใช้โปรแกรม Visual Basic เราต้องทำการเปิด Port ก่อนซึ่งขึ้นอยู่กับ Computer โดยเราสามรถเปิดจากบนหน้าจอของ Visual Basic นี้



รูปที่ 4.7 การเปิด Port ที่แสดงบนหน้าจอ Visual Basic

การตั้งค่า Set point

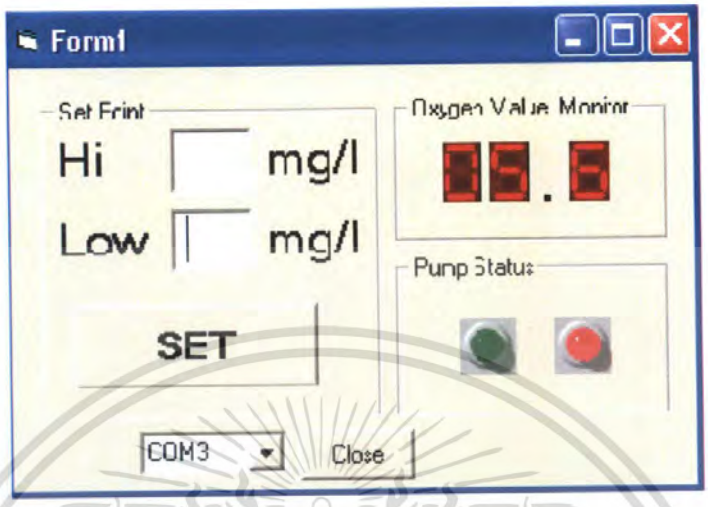
โดยเราสามารถตั้งค่าให้ปั้มออกซิเจนทำงานและหยุดทำงานได้ด้วยการตั้งค่า Set point 2 ค่า คือ ค่า Cut on และ Cut off

ค่าCut on หมายถึง ค่าที่ทำให้ปั้มเริ่มทำงาน คือถ้าปริมาณออกซิเจนภายในน้ำมีระดับต่ำกว่าค่าๆนี้ ปั้มก็จะเริ่มทำงาน ซึ่งในโปรแกรม Visual Basic ที่แสดงไว้จะหมายถึงค่า Low

ค่าCut off หมายถึง ค่าที่ทำให้ปั้มหยุดทำงาน คือเมื่อปั้มเพิ่มปริมาณออกซิเจนจนสูงกว่าค่าๆนี้ ปั้มออกซิเจนก็จะหยุดทำงาน โดยในโปรแกรม Visual Basic ที่แสดงไว้จะหมายถึงค่า Hi

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราสามารถตั้งค่า Set point ที่โปรแกรม Visual Basic โดยเวลาที่หน้าจอแสดงผลจะมีปุ่ม Reset และพอเราทำการคลิกเข้าไป ปุ่มจะขึ้นเป็น Set เราก็สามารถจะตั้งค่า Hi และ Low ได้

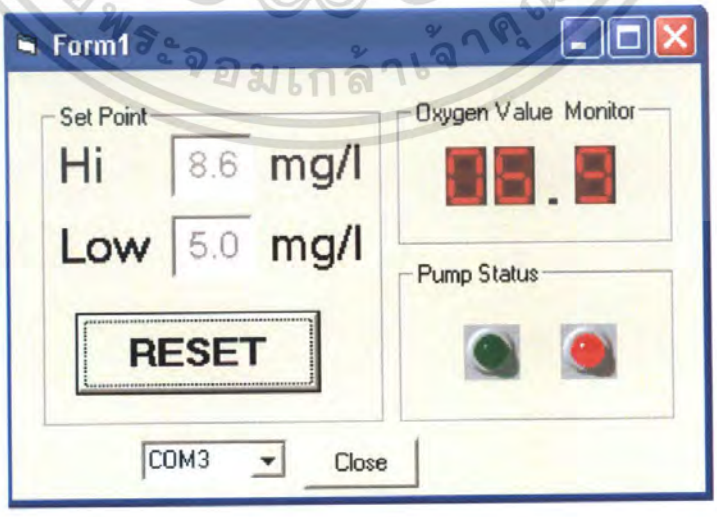


รูปที่ 4.8 การตั้งค่า Set point

4..3.3 ผลการทดลอง

1. แสดงผลในขณะที่ปั๊มไม่ทำงาน

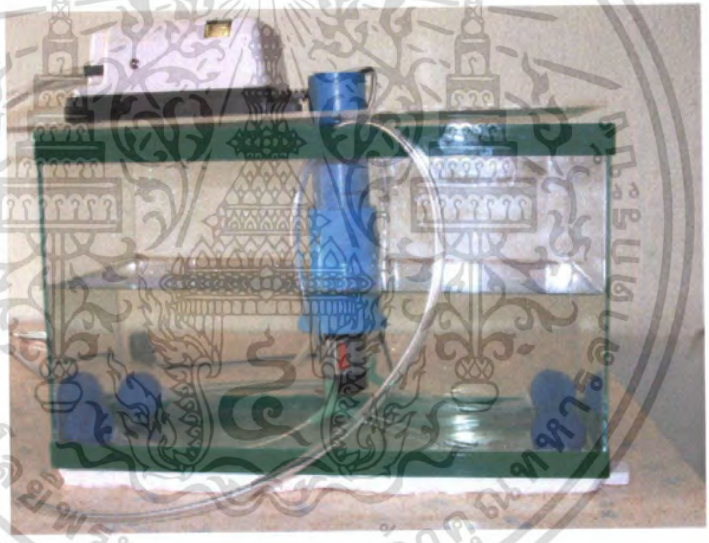
เมื่อทำการตั้ง ค่าการทำงานของปั๊มน้ำอยู่ที่ 5.0 mg/l และตั้งค่าการหยุดทำงานของปั๊มอยู่ที่ 8.6 mg/l เมื่อค่าออกซิเจนในบ่อเลี้ยงสัตว์น้ำอยู่ที่ 6.9 mg/l คุณสมบัติของปั๊มจะแสดงค่าเป็น 0 ปั๊มน้ำจะไม่ทำงาน จนค่าออกซิเจนจะลดลงไปถึงค่าของการทำงานซึ่งอยู่ที่ 5.0 mg/l ปั๊มจึงจะทำงาน ทำให้ค่าออกซิเจนในบ่อเลี้ยงสัตว์น้ำเพิ่มขึ้นมาจนถึงค่าเริ่มหยุดการทำงานอีกครั้งหนึ่ง สถานะ



รูปที่ 4.9 ปริมาณออกซิเจนในน้ำในขณะที่ปั๊มไม่ทำงานทาง Visual Basic

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 4.10 ปริมาณออกซิเจนในน้ำ ในขณะที่ปั้มไม่ทำงาน ทาง LCD

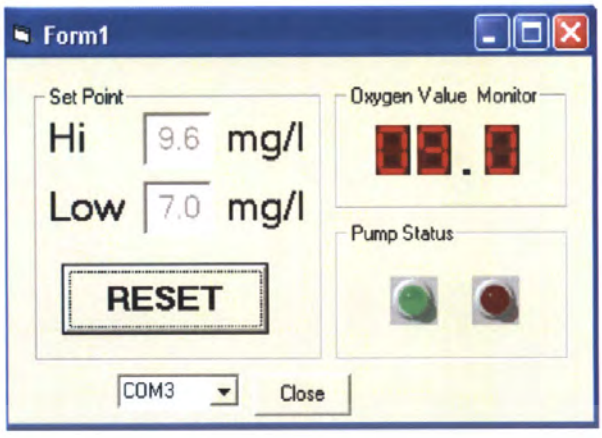


รูปที่ 4.11 ปั้มไม่ทำงานเมื่อออกซิเจนสูงค่ากว่าค่าการทำงาน

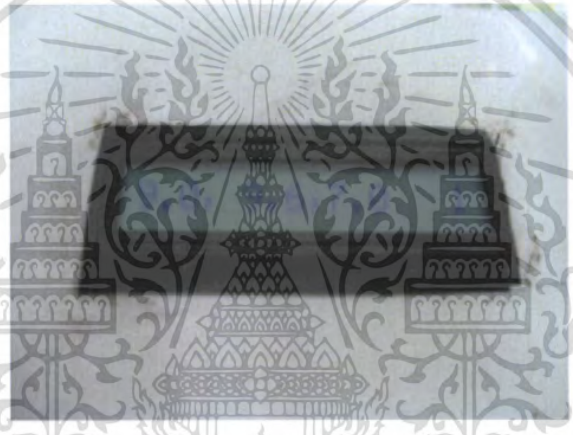
2. แสดงผลขณะที่ปั้มทำงาน

เมื่อทำการตั้ง ค่าการทำงานของปั้มน้ำอยู่ที่ 7.0 mg/l และตั้งค่าการหยุดทำงานของปั้มอยู่ที่ 9.6 mg/l เมื่อค่าออกซิเจนในบ่อเลี้ยงสัตว์น้ำอยู่ที่ 9.0 mg/l คุณสมบัติของปั้มจะแสดงค่าเป็น 1 ปั้มน้ำจะทำงานไปจนถึงค่าของการหยุดทำงานซึ่งอยู่ที่ 9.6 mg/l แล้วหยุดทำงานทำให้ค่าออกซิเจนในบ่อเลี้ยงสัตว์น้ำลดลงมาจนถึงค่าเริ่มทำงานอีกครั้งหนึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.12 ปริมาณออกซิเจนในน้ำ ในขณะที่ปั๊มทำงาน



รูปที่ 4.13 ปริมาณออกซิเจนในน้ำ ในขณะที่ปั๊มทำงาน ทาง LCD

รูปที่ 4.14 ปั๊มทำงานเมื่อออกซิเจนต่ำกว่าที่ตั้งไว้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
บทสรุปและวิจารณ์

5.1 สรุปผลการทดลอง

ในโครงงานระบบบำบัดน้ำเสียผ่านอุปกรณ์ไร้สายนี้ จะใช้การสั่งงานจากคอมพิวเตอร์ที่อยู่บนภาคพื้นดินเพื่อที่จะส่งสัญญาณจากภาคพื้นดินไปควบคุมปั๊มน้ำ เพื่อเพิ่มค่าออกซิเจนที่อยู่ในบ่อเลี้ยงสัตว์น้ำโดยการส่งและรับจะส่งข้อมูลค่าออกซิเจนที่อยู่ในน้ำ ผ่านโมดูลความถี่วิทยุเป็นหลักไปยังคอมพิวเตอร์ที่อยู่สถานีภาคพื้นดินเพื่อประมวลผลและตั้งค่าการทำงานของตัวปั๊มน้ำ เพื่อเพิ่มค่าออกซิเจนที่อยู่ในน้ำ โดยมีการตั้งค่าของช่วงการทำงานและช่วงหยุดการทำงานของปั๊มน้ำ เพื่อรักษาของช่วงค่าออกซิเจนให้อยู่ในช่วงที่ต้องการของสัตว์น้ำอยู่ตลอดเวลา ส่งกลับมาควบคุมตัวปั๊มน้ำที่อยู่ในบ่อน้ำ ทำให้ค่าออกซิเจนของน้ำที่อยู่ในระบบบำบัดน้ำเสียผ่านอุปกรณ์ไร้สายมีค่าจำกัดอยู่ในช่วงที่ผู้ใช้ต้องการให้อยู่เพื่อเพาะเลี้ยงสัตว์น้ำแต่ละชนิดได้ตามความต้องการอยู่ตลอดเวลา

5.1.1 สรุปผลการทดลองที่ 4.1

การทดลองที่ 4.1 เป็นการทดลองแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลซึ่งผลการทดลองที่ได้นั้นเราสามารถแปลงสัญญาณที่เป็นอนาลอกไปเป็นสัญญาณดิจิตอลได้ผ่านตัว ไอซี ADC 8032 โดยสามารถแสดงผลออกมาได้เป็นทั้งเลขบิท และแสดงค่าเป็นตัวเลขผ่านจอแอลซีดีทาง และจากการวัดขาสัญญาณ CLK และ Data Out ของไอซี ADC 8032 ก็จะมีสถานะ “0” กับ “1” ซึ่งจะตรงกับเลขบิทที่แสดงอยู่บนหลอด LED

เนื่องเป็นการควอนไทซ์แบบ 8 บิทและในการทดลอง เราได้ให้ค่าแรงดันสูงสุดไว้ที่ 5 V การควอนไทซ์แบบ 8 บิทมีค่าสูงสุดอยู่ที่ 256 ระดับคือ 0-255 เราจึงทำการแปลงให้ค่าบิทเป็นตัวเลขได้ โดยทำให้ค่า 5 V นั้นมีค่าสูงสุดโดยที่แสดงเป็นบิทนั้นทำให้เป็นตัวเลขได้โดยนำตัวเลข 51 มาหาร โดยจากรูปที่ 4.1 จะเห็นได้ว่าค่าที่แสดงบน LED คือ 1100111 แปลงเป็นเลขฐานสิบ 103 หารด้วย 51 จะได้เท่ากับ 2.02 ซึ่งตรงกับค่าที่แสดงบนแอลซีดี

5.1.2 สรุปผลการทดลองที่ 4.2

ในการทดลองนี้จะนำผลการทดลองที่ได้ไปเป็นผลอ้างอิงเพื่อเปรียบเทียบค่าออกซิเจนภายในน้ำกับแรงดันของตัว Sensor ที่ส่งออกมา โดยจะมี Dissolved Oxygen meter เป็นตัววัดค่าออกซิเจนในน้ำ ซึ่งผลที่ได้จะเห็นว่าค่าปริมาณออกซิเจนภายในน้ำจะมีอัตราส่วนที่ประมาณ 1:1 กับแรงดันไฟฟ้าที่ Oxygen Sensor ส่งออกมาโดยที่เราสามารถลดปริมาณออกซิเจนภายในน้ำได้โดยเติมสาร Sodium Sulphite

5.1.2 สรุปผลการทดลองที่ 4.3

ปกติแล้วภายในน้ำที่สัตว์น้ำสามารถเจริญเติบโตได้ดีจะมีปริมาณออกซิเจน 5-7 มิลลิกรัมต่อลิตร รวมทั้งควรจะมีค่า ph อยู่ที่ 7 ด้วย เพราะฉะนั้นเราจึงควรตั้งค่าไว้ให้ปริมาณออกซิเจนอยู่ในช่วง 5-7 มิลลิกรัมต่อลิตร เพราะถ้ามากหรือน้อยกว่านี้แล้วจะทำให้มันเป็นอันตรายต่อสัตว์น้ำได้

ข้อควรระวังอย่างมากอย่างหนึ่งก็คือ ส่วนหัวที่เป็นหน้าสัมผัสของตัว Oxygen Sensor นั้นห้ามแตะต้องโดยเด็ด เพราะมันจะทำให้ Oxygen Sensor นั้นเสียได้ เพราะฉะนั้นเวลาที่เราเคลื่อนย้าย ควรจะมีความระมัดระวัง หรือถ้าไม่จำเป็นก็ไม่ควรจะเคลื่อนย้ายเลย ส่วนที่ยากอีกอย่างหนึ่งของการทดลองนี้คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การปรับค่าแรงดันที่ออกมาจากตัว Oxygen Sensor ซึ่งต้องใช้วงจร Instrument Amplifier ในการปรับค่าเพื่อให้ได้แรงดันไฟฟ้าที่เหมาะสมไปเข้าวงจรของทางด้าน Slave และส่งมาแสดงผลในที่สุด

ส่วนการแสดงผลทางด้าน LCD นั้นค่าที่แสดงออกมาก็จะเหมือนกับค่าที่แสดงบนหน้าจอ Visual basic โดยที่ด้านซ้ายสุดนั้นจะเป็นค่าปริมาณออกซิเจนภายในน้ำต่อมาก็จะเป็นค่า Cut off และ Cut on ส่วนที่แสดงทางด้านขวาสุดคือสถานะของปั๊มนั่นเอง

สำหรับการตั้งค่า Set Point เราต้องตั้งให้ค่า Cut off นั้นมากกว่าค่า Cut on มิฉะนั้น การทำงานของปั๊มจะเกิดทำงานผิดพลาดจะทำงานแบบติดๆหยุดๆ

การส่งข้อมูลของตัว TRW 2.4 GHz ทั้งสองด้านจะมีการเกิดช่างเวลา Time out ขึ้นซึ่งจะแสดงขึ้นที่หน้าจอ LCD ซึ่งจะเป็นช่วงเวลาในการรอรับส่งข้อมูล

5.2 ปัญหาและแนวทางแก้ไข

1. เนื่องจากอุปกรณ์ในการรับ/ส่งสัญญาณข้อมูล ซึ่งก็คือโมดูลความถี่วิทยุยังไม่มีผู้นำมาใช้มากนัก ดังนั้นจึงต้องใช้เวลาในการศึกษาหาข้อมูล และศึกษาการเชื่อมต่อระหว่างโมดูลความถี่วิทยุกับไมโครคอนโทรเลอร์

2. การเปรียบเทียบค่าระหว่างออกซิเจนมิเตอร์กับออกซิเจนเซนเซอร์จำเป็นต้องใช้มิเตอร์ดิจิตอลซึ่งมีค่าเปลี่ยนแปลงที่ไวมาก อาจทำให้การเปรียบเทียบค่าคลาดเคลื่อนไปบ้างเล็กน้อย

3. เนื่องจากเป็นการจำลองจำเป็นต้องใช้เครื่องปั๊มขนาดเล็กทำให้กำลังขับในการเพิ่มค่าออกซิเจนมีการเปลี่ยนแปลงได้ช้า

5.3 แนวทางการพัฒนาโครงงาน

1. พัฒนาให้มีการควบคุมการทำงานหลายๆบ่อพร้อมกันผ่านหน้าจอคอมพิวเตอร์เดียวกัน

2. พัฒนาให้มีการส่งได้ระยะไกลเพื่อควบคุมพื้นที่ในการเลี้ยงสัตว์น้ำได้หลายๆบ่อ จากสถานีควบคุมภาคพื้นดินเดียวกัน

3. นำระบบไปใช้ได้จริงเลยเนื่องจากได้ปรับให้มีการขับโหลดที่ไฟฟ้า 220 V 1 เฟส 2สายอยู่แล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

[1] รองศาสตราจารย์ สมยศ จุณณปิยะ Microcontroller Application MCS-51.พิมพ์ครั้งที่ 5 พ.ศ 2546 ภาควิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. 2546

[2] อำนาจ มีมงคล, "ออกแบบและติดตั้งเครือข่าย Wireless LAN" นนทบุรี : ไอดีซี , 2547

[3] บัณฑิต จามรภูติ, "คู่มือการใช้งาน Protel 99" เชียงใหม่ : บัณฑิตเพรส 2544.

[4] พร้อมเลิศ หล่อวิจิตร, "คู่มือเรียน Visual Basic 2005"กรุงเทพ : โปรวิชั่น, 2549.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมของทางด้าน Slave

```
/***********************************************/
/* Program     : Test RF module 2.4G          */
/* Decription  : Use MCS-51 send data to      */
/*             : Module RF 2.4G with Whenshing */
/*             : Electronics                  */
/* CPU         : T89C51AC2 (Atmel corp USA)   */
/* Complier    : KeilV7.08                    */
/* Start Date  : 22/03/2004                   */
/* Solfware Eng.: P.Suthep                    */
/* email       : tape_4@hotmail.com           */
/***********************************************/
#include <REG51F.H>
#include <absacc.h>
#include <ctype.h>
#include <intrins.h>
#include <math.h>
#include <stdarg.h>
#include <stdio.h>
#include <stdlib.h>

/***********************************************/
/*  Define CODE command for TRW-2.4G at defual */
/***********************************************/

unsigned char Test_Package[] = {0x8E,0x08,0x1C}; // reserved for testing
unsigned char DATA2_W[]   = {0xC8};          // data2 for send to GFSK
unsigned char DATA1_W[]   = {0xC8};
unsigned char ADDR2[]     = {0xC8,0xC8,0xC8,0xAA,0x55};
unsigned char ADDR1[]     = {0xAA,0x55,0xAA,0x55,0xAA};
unsigned char ADDR_W_CRC[] = {0x0A3};
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
unsigned char RF_PGM[]    = {0x4F};
unsigned char RF_Send[]   = {0x44};
unsigned char RF_Recive_Data[] = {0x45};
unsigned char Data_Test[] = {0x01,0x02,0x04,0x10,0x20,0x40,0x80};
unsigned char BUFFER,Buffer1,i,k,j,Data_out,RxBuff;
unsigned char Low,High,ii;
unsigned char O2_Data,BuffData;
unsigned char Data_Buffer[40];
#define RUN 0
#define STOP 1

/************************************************/
/*  Define I/O Port                 */
/************************************************/
#define Port_Test P0
sbit Data  = P1^4;          // Port send data to module 2.4G chanel I
sbit CLK1  = P1^3;          // Port Clock befor send data chanel 1
sbit DR1   = P1^5;          // port select data to chanel1
 sbit CS    = P1^2;         // Port chip select
sbit CE    = P1^0;          // Port Enable Shift
sbit PUMP = P2^0;
sbit ADC_CS     = P2^1;
sbit ADC_CLK    = P2^2;
sbit ADC_DO     = P2^3;
sbit ADC_DI     = P2^4;

/************************************************/
/*  Subrutine for Delay time            */
/************************************************/
void dmsec (unsigned int count)
{    // mSec Delay 11.0592 Mhz
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    unsigned int iii;                       // Keil v7.08
    while (count) {
      iii = 225; while (iii>0) iii--;
      count--;
    }
}

/************************************************/
/* Subrutine delay time @ 18.432 MHZ          */
/************************************************/
void delay_5usec (void)
{
     _nop_();      // nop = 0.9 usec @ 1 machine cycle
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();      // nop = 0.9 usec @ 1 machine cycle
}
/************************************************/
/* Subrutine delay time @ 18.432 MHZ          */
/************************************************/
void delay_10usec (void)
{
     _nop_();      // nop = 0.9 usec @ 1 machine cycle
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();          // Total time approximate at 10 usec
    _nop_();      // nop = 0.9 usec @ 1 machine cycle
    _nop_();
    _nop_();
      // Total time approximate at 10 usec

}

/*********************************/
/* Delay 200 usec              */
/*********************************/
void delay_200usec (void)
{
   for (i=1;i<=20;i++)
     {delay_10usec();}
}

/************************************************/
/* Subrutine send data to port 1.4          */
/************************************************/
void TX_DATA(void)
{
          for (i=1;i<=8;i++)
  {
           delay_5usec();
   CLK1 = 0;
   Data= BUFFER & 0x80;        // Send data 1 bit
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            BUFFER = BUFFER << 1;       // Data shift 1 bit
            delay_5usec();
            CLK1 = 1;           // Clock send data
            delay_5usec();
          }
          delay_200usec();
}
void READ_DATA (void)
{
//    unsigned char j;
        for (j=1;j<=8;j++)
                    {
                                CLK1 =0;
                                delay_5usec();
                                CLK1 =1;
                          i <<=1;

                    if (Data ==1)
                          {  i |=0x01;}
                      else
                                {  i &=0xFE;}
                                 delay_5usec();

        }
}

/***************************************/
/* Subrutine  RF_CONFIG_SEND chanel 2   */
/***************************************/
void RF_CONFIG_SEND (void)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
CE = 0;
CS = 1;
delay_200usec();
BUFFER = Test_Package[0];
TX_DATA();
BUFFER = Test_Package[1];
TX_DATA();
BUFFER = Test_Package[2];
TX_DATA();
BUFFER = DATA2_W[0];
TX_DATA();
BUFFER = DATA1_W[0];
TX_DATA();
BUFFER = ADDR2[0];
TX_DATA();
BUFFER = ADDR2[1];
TX_DATA();
BUFFER = ADDR2[2];
TX_DATA();
BUFFER = ADDR2[3];
TX_DATA();
BUFFER = ADDR2[4];
TX_DATA();
BUFFER = ADDR1[0];
TX_DATA();
BUFFER = ADDR1[1];
TX_DATA();
BUFFER = ADDR1[2];
TX_DATA();
BUFFER = ADDR1[3];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    TX_DATA();
    BUFFER = ADDR1[4];
    TX_DATA();
    BUFFER = ADDR_W_CRC[0];
    TX_DATA();
    BUFFER = RF_PGM[0];
    TX_DATA();
    BUFFER = RF_Send[0];
    TX_DATA();
    delay_5usec();
    CLK1 = 0;
    CS   = 0;
                dmsec(1);
}

void RF_CONFIG_RECIVE(void)
{
    CE = 0;
    CS = 1;
    delay_200usec();
    BUFFER = Test_Package[0];
    TX_DATA();
    BUFFER = Test_Package[1];
    TX_DATA();
    BUFFER = Test_Package[2];
    TX_DATA();
    BUFFER = DATA2_W[0];
    TX_DATA();
    BUFFER = DATA1_W[0];
    TX_DATA();
    BUFFER = ADDR2[0];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    TX_DATA();
    BUFFER = ADDR2[1];
    TX_DATA();
    BUFFER = ADDR2[2];
    TX_DATA();
    BUFFER = ADDR2[3];
    TX_DATA();
    BUFFER = ADDR2[4];
    TX_DATA();
    BUFFER = ADDR1[0];
    TX_DATA();
    BUFFER = ADDR1[1];
    TX_DATA();
    BUFFER = ADDR1[2];
    TX_DATA();
    BUFFER = ADDR1[3];
    TX_DATA();
    BUFFER = ADDR1[4];
    TX_DATA();
    BUFFER = ADDR_W_CRC[0];
    TX_DATA();
    BUFFER = RF_PGM[0];
    TX_DATA();
    BUFFER = RF_Recive_Data[0];
    TX_DATA();
    delay_5usec();
    CLK1 = 0;
    CS  = 0;
          delay_10usec();
          //CE=1;// En Rx
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void Address_Send(void)
{
    CS = 0;
    CE = 1;
    delay_200usec();
    BUFFER = ADDR1[0];
    TX_DATA();
    BUFFER = ADDR1[1];
    TX_DATA();
    BUFFER = ADDR1[2];
    TX_DATA();
    BUFFER = ADDR1[3];
    TX_DATA();
    BUFFER = ADDR1[4];
    TX_DATA();
}
////////////////////////////////////////////////
//////////   ADC 0832  ////////////////////////////
//---------------------------------------
// read analog from ADC
// Single end mode(2 channel)
//---------------------------------------
unsigned char ReadADC(unsigned char channel)
{
  unsigned char i,k;
  unsigned char AdcResult;  // 8 bit

  ADC_CS=0;     // Active chip select
  k++;                        // Delay about 1 uS
  ADC_CLK=0;  // make clock low first
  k++;k++;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
channel = channel? 0xE0 : 0xC0;
k++;k++;                    // delay about 2 uS
//--- write command 3 bit ----------
for(i=0; i< 3;i++) {
  ADC_DI = (channel & 0x80) != 0;
  channel=channel<<1;
  ADC_CLK=1;
  k++;k++;                  // delay about 2 uS
  ADC_CLK=0;
}
//--- read ADC result 8 bit --------
AdcResult=0;
for(i=0;i<8;i++)
        {
  ADC_CLK=1;
  k++;k++;                  // delay about 2 uS
  k++;k++;                  // delay about 2 uS
  ADC_CLK=0;
  k++;k++;                  // delay about 2 uS
  k++;k++;                  // delay about 2 uS
  AdcResult=AdcResult<<1;
  AdcResult=AdcResult | (ADC_DO & 0x01);
}
for(i=0;i<8;i++)
        {
  ADC_CLK=1;
  k++;k++;                  // delay about 2 uS
  k++;k++;                  // delay about 2 uS
  ADC_CLK=0;
  k++;k++;                  // delay about 2 uS
  k++;k++;                  // delay about 2 uS
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//     AdcResult=AdcResult<<1;
//     AdcResult=AdcResult | (ADC_DO & 0x01);
   }
          dmsec(1);
   ADC_CS=1;
   return AdcResult;
}

/***********************************************/
/*  Main Program                   */
/***********************************************/

// Data Package
// ":OOOOTTTT CrLf
// ":SSSS CrLf
void main (void)
{
    CS  = 1;
          CE  = 1;
    TMOD = 0x21;
    TH1  = 0x0FD;
    TL1  = 0x0FD;
    EA   = 0;
    ES   = 1;
          TR1  = 1;
          SCON = 0x52;
   DR1 = 1;
    CLK1= 1;
          Port_Test=0;
          dmsec(200);
          Port_Test=128;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    dmsec(200);
    Port_Test=255;
  while(1)
        {
                    // Step #1 Config RF Module = Recieve Mode
              RF_CONFIG_RECIVE();
    delay_10usec();
                    CE=1;
                    DR1=1;
                    Port_Test=0;
                    // STEP #2 : Wait Data From Master
                    while(DR1==0)
                    {
                      //STEP #A Read O2 Data from A/D Converter
                      BuffData=(unsigned char)((float)ReadADC(1)*1.7);//Read O2
Data (RAW DATA)
                      // STEP #B =Filter O2 Data
                      if(BuffData>O2_Data)
                      {
                          O2_Data++;
                      }
                      else if(BuffData<O2_Data)
                      {
                          O2_Data--;
                      }

                      // STEP #C Display To LED
                      Port_Test=~O2_Data;
                      // STEP #D Control Pump
                      if(PUMP==STOP)
                      {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    if(O2_Data<=Low)PUMP=RUN;
                }
                else
                {
                    if(O2_Data>=High)PUMP=STOP;
                }
            }// Wait Recieve Data
            CE=0;// Low Pwr Mode
            Port_Test=254;

            // STEP #3 : Read Analog Data
            ii=0;
            while (DR1)// Read from Buffer
        {
                Port_Test=255;
    READ_DATA();
                Data_Buffer[ii]=i;
                Port_Test=i;
                ii++;
            }
                Port_Test=0;
            // STEP #4 :Decode Package
            //  Recieve "080040xxxxxxxxxxxxxxxxxx"
            High=Data_Buffer[1]-0x30;
            High=(High*10) + Data_Buffer[2]-0x30;
            High=(High*10) + Data_Buffer[3]-0x30;
            Port_Test=~High;
            dmsec(200);
            // Hign=80
            Low=Data_Buffer[4]-0x30;
            Low=(Low*10) + Data_Buffer[5]-0x30;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                        Low=(Low*10) + Data_Buffer[6]-0x30;
                                        Port_Test=~Low;
                                        dmsec(250);
                                        // STEP #5  Pack Data for Send to Master   4 Value : 1=PV,2=Cut
Off(High),3=Cut On(Low), 4=Pump Status(0=On,1=Off)
                                        // All Data will Store in 'Data_Buffer'
        sprintf(Data_Buffer,":%3d%3d%3d%1dxxxxxxxxxxxxxxxxx",(int)O2_Data,(int)High,(int)Low,(in
t)PUMP);
                                        for(j=0;j<25;j++)
                                        {
                                                if(Data_Buffer[j]==' ')Data_Buffer[j]='0';
                                        }

                                        // SYTEP #6 Set RF Moduke = Transmit Mode
        RF_CONFIG_SEND();
                Data_out=0x30;
                                        delay_200usec();

                                        // STEP #7 Send Data to Master
        Address_Send();
                                        delay_200usec();
        for (j=1;j<=25;j++)
        {
                                                BUFFER = Data_Buffer[j-1];
                                                Port_Test=~BUFFER;
                TX_DATA();
                                        }
        delay_5usec();
                                        CE=0;
    }
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมที่ใช้ด้าน Master

```
/*****************************************************/
/* Program   :Test RF module 2.4G reciver          */
/* Decription   : Use MCS-51 recive data to        */
/*              : Module RF 2.4G with Whenshing    */
/*              : Electronics                      */
/* CPU          : T89C51AC2 (Atmel corp USA)       */
/* Complier     : KeilV7.08                        */
/* Start Date   : 22/03/2004                       */
/* Solfware Eng.: P.Suthep                         */
/* email        : tape_4@hotmail.com               */
/*****************************************************/
#include <REG51F.H>
#include <absacc.h>
#include <ctype.h>
#include <intrins.h>
#include <math.h>
#include <stdarg.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define Port_Test P0
/*************************************************/
/*   Define I/O Port                             */
/*************************************************/
//#define Port_Test P0
sbit Data   = P1^4;             // Port send data to module 2.4G chanel 1
sbit CLK1   = P1^3;             // Port Clock befor send data chanel 1
sbit DR1    = P1^5;             // port select data to chanel1
            // Port seclect data to chanel 2
sbit CS     = P1^2;             // Port chip select
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
sbit CE    = P1^0;            // Port Enable Shift
#define ON                  0x02
#define OFF                 0x00
#define BLINK   0x01
// LCD bit define
sbit rs_LCD=P2^5;// select command (0) or data(1)
sbit en_LCD=P2^4;
sbit D3=P2^0;
sbit D2=P2^1;
sbit D1=P2^2;
sbit D0=P2^3;
/************************************************/
/*  Define CODE command for TRW-2.4G at defual */
/************************************************/
unsigned char Test_Package[] = {0x8E,0x08,0x1C}; // reserved for testing
unsigned char DATA2_W[]   = {0xC8};          // data2 for send to GFSK
unsigned char DATA1_W[]   = {0xC8};
unsigned char ADDR2[]     = {0xC8,0xC8,0xC8,0xAA,0x55};
unsigned char ADDR1[]     = {0xAA,0x55,0xAA,0x55,0xAA};
unsigned char ADDR_W_CRC[] = {0x0A3};
unsigned char RF_PGM[]    = {0x4F};
unsigned char RF_Send[]   = {0x44};
unsigned char RF_Recive_Data[]  = {0x45};
unsigned char Data_Test[]  = {0x01,0x02,0x04,0x10,0x20,0x40,0x80};
unsigned char BUFFER,Buffer1,i,k,j,Data_out;
unsigned char Data_Buffer[40];
unsigned char bdata TX_RX;
sbit Chk_TRx = TX_RX^0;
unsigned char High,Low,O2_Data,Index;
unsigned char RecieveBuffer[30];
bit PumpStatus;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
bit RxReady;
/************************************************/
/*  Subrutine for Delay time                    */
/************************************************/
void dmsec (unsigned int count) {     // mSec Delay 11.0592 Mhz
   unsigned int iii;                  // Keil v7.08
   while (count) {
      iii = 115; while (iii>0) iii--;
      count--;
   }
}
/************************************************/
/* Subrutine delay time @ 18.432 MHZ            */
/************************************************/
void delay_5usec (void)
{
     _nop_();      // nop = 0.9 usec @ 1 machine cycle
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
     _nop_();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    _nop_();
}
/***********************************************/
/* Subrutine delay time @ 18.432 MHZ          */
/***********************************************/
void delay_10usec (void)
{
    _nop_();     // nop = 0.9 usec @ 1 machine cycle
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();     //Total time approximate at 10 usec
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
    _nop_();
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/*********************************/
/* Delay 200 usec               */
/*********************************/
void delay_200usec (void)
{
//   unsigned char i;
    for (i=1;i<=20;i++)
      {delay_10usec();}
}
void TX_DATA(void)
{
//    unsigned char i;
          for (i=1;i<=8;i++)
   {
    CLK1 = 0;
    Data= BUFFER & 0x80;        // Send data 1 bit
          BUFFER = BUFFER << 1;       // Data shift 1 bit
          delay_5usec();
          CLK1 = 1;          // Clock send data
          delay_5usec();
         }
}
void READ_DATA (void)
{
//    unsigned char j;
        for (j=1;j<=8;j++)
                  {
                                    CLK1 =0;
                                    delay_5usec();
                                    CLK1 =1;
                              i <<=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    if (Data ==1)
                        { i |=0x01;}
                    else
                        { i &=0xFE;}
                        delay_5usec();
        }
}

/*****************************************/
/* Subrutine  RF_CONFIG_SEND chanel 2   */
/*****************************************/
void RF_CONFIG_SEND (void)
{
    CE = 0;
    CS = 1;
    delay_200usec();
    BUFFER = Test_Package[0];
    TX_DATA();
    BUFFER = Test_Package[1];
    TX_DATA();
    BUFFER = Test_Package[2];
    TX_DATA();
    BUFFER = DATA2_W[0];
    TX_DATA();
    BUFFER = DATA1_W[0];
    TX_DATA();
    BUFFER = ADDR2[0];
    TX_DATA();
    BUFFER = ADDR2[1];
    TX_DATA();
    BUFFER = ADDR2[2];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    TX_DATA();
    BUFFER = ADDR2[3];
    TX_DATA();
    BUFFER = ADDR2[4];
    TX_DATA();
    BUFFER = ADDR1[0];
    TX_DATA();
    BUFFER = ADDR1[1];
    TX_DATA();
    BUFFER = ADDR1[2];
    TX_DATA();
    BUFFER = ADDR1[3];
    TX_DATA();
    BUFFER = ADDR1[4];
    TX_DATA();
    BUFFER = ADDR_W_CRC[0];
    TX_DATA();
    BUFFER = RF_PGM[0];
    TX_DATA();
    BUFFER = RF_Send[0];
    TX_DATA();
    delay_5usec();
    CLK1 = 0;
    CS   = 0;
                dmsec(1);
}
void RF_CONFIG_RECIVE(void)
{
    CE = 0;
    CS = 1;
    delay_200usec();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
BUFFER = Test_Package[0];
TX_DATA();
BUFFER = Test_Package[1];
TX_DATA();
BUFFER = Test_Package[2];
TX_DATA();
BUFFER = DATA2_W[0];
TX_DATA();
BUFFER = DATA1_W[0];
TX_DATA();
BUFFER = ADDR2[0];
TX_DATA();
BUFFER = ADDR2[1];
TX_DATA();
BUFFER = ADDR2[2];
TX_DATA();
BUFFER = ADDR2[3];
TX_DATA();
BUFFER = ADDR2[4];
TX_DATA();
BUFFER = ADDR1[0];
TX_DATA();
BUFFER = ADDR1[1];
TX_DATA();
BUFFER = ADDR1[2];
TX_DATA();
BUFFER = ADDR1[3];
TX_DATA();
BUFFER = ADDR1[4];
TX_DATA();
BUFFER = ADDR_W_CRC[0];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    TX_DATA();
    BUFFER = RF_PGM[0];
    TX_DATA();
    BUFFER = RF_Recive_Data[0];
    TX_DATA();
    delay_5usec();
    CLK1 = 0;
    CS  = 0;
        delay_10usec();
        //CE=1;// En Rx
}
void Address_Send(void)
{
   CS = 0;
   CE = 1;
   delay_200usec();
   BUFFER = ADDR1[0];
   TX_DATA();
   BUFFER = ADDR1[1];
   TX_DATA();
   BUFFER = ADDR1[2];
   TX_DATA();
   BUFFER = ADDR1[3];
   TX_DATA();
   BUFFER = ADDR1[4];
   TX_DATA();
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
/***********************************************/
/*     Subrutine Hex to ASCII                  */
/***********************************************/
/*void Hex_To_ASCII (void)
{
   if (Buffer1 < 10)
      {
      Data_Send =   Buffer1 +(0x30);
      }
   else
      {
      Data_Send =   Buffer1 +(0x37);
      }
}
////////////////////////////////////////////////
//                                            //
//                  LCD                       //
//                                            //
////////////////////////////////////////////////*/
void dusec(unsigned char time)
{
 while(time--);
}
void LCD_Pulse(void)
{
   en_LCD=1;
   dusec(50);
   en_LCD=0;
   dusec(50);
}
void LCD_Command_write(unsigned char ch)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{

    rs_LCD=0;
    if((ch&128)==128)D3=1;          else D3=0;
    if((ch&64)==64)D2=1; else D2=0;
    if((ch&32)==32)D1=1; else D1=0;
    if((ch&16)==16)D0=1; else D0=0;
    LCD_Pulse();
    if((ch&8)==8)D3=1;    else D3=0;
    if((ch&4)==4)D2=1;    else D2=0;
    if((ch&2)==2)D1=1;    else D1=0;
    if((ch&1)==1)D0=1;    else D0=0;
    LCD_Pulse();
}
void LCD_Data_write(unsigned char ch)
{

    rs_LCD=1;
    if((ch&128)==128)D3=1;          else D3=0;
    if((ch&64)==64)D2=1; else D2=0;
    if((ch&32)==32)D1=1; else D1=0;
    if((ch&16)==16)D0=1; else D0=0;
    LCD_Pulse();
    if((ch&8)==8)D3=1;    else D3=0;
    if((ch&4)==4)D2=1;    else D2=0;
    if((ch&2)==2)D1=1;    else D1=0;
    if((ch&1)==1)D0=1;    else D0=0;
    LCD_Pulse();
}
void ClearLCD(void)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    LCD_Command_write(0x01);
    dmsec(50);
}
void CursorLCD(unsigned char attribute)
{
        if (attribute == BLINK)
                attribute |= ON;

    LCD_Command_write(0x0C | attribute);
}
void putc(char ch, unsigned char index)
{
        if (index > 0x01F)
                return;
        if (index > 0x07)
                index += 0x38;
        LCD_Command_write(0x80 | index);// set Cursor
        LCD_Data_write(ch);
}
void PrintLCD(char *str, unsigned char index)
{
    unsigned char iii;
        for (iii = 0; iii < strlen(str); iii++)
    {
                putc(str[iii], iii + index);
    }
}
void InitializeLCD(void)
{
    dmsec(500);
    LCD_Command_write(0x33);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        dmsec(90);
    LCD_Command_write(0x32);
        dmsec(50);
    LCD_Command_write(0x28);
        dmsec(50);
    LCD_Command_write(0x08);
        dmsec(50);
    LCD_Command_write(0x0C);
        dmsec(50);
        CursorLCD(BLINK);//
        dmsec(50);
        ClearLCD();
    LCD_Command_write(0x06);
        CursorLCD(OFF);//
}
// ":xxxxxxxxxxxxx" 0x0D 0x0A
void ConnectPC(void) interrupt 4  // serial interrupt vecter 4
{
        if(RI && RxReady==0)
        {
                if(SBUF==0x0D)
                {
                        RecieveBuffer[Index]=0;
                        if(Index>0)RxReady=1; else RxReady=0;
                        Index=0;
                }
                else if(SBUF==0x0A)
                {
                        Index=0;
                }
                else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                {
                        RecieveBuffer[Index]=SBUF;
                        Index++;
                }
                RI=0;
        }
}
// Send data (Byte)
void SendByte(unsigned char bData)  // RS232
{
        SBUF=bData;//  SBUF  -> TI=0;
        while(TI==0);// Wait Send Complete   ->TI=1;
        TI=0;
}
void initializeRS232(void)// initial at 9600 bps 8 bit none parity and 1 stop bit
{
                        RCAP2H = 0xFF;
                        RCAP2L = 0xB8;
  SCON = 0x40;
        T2CON = 0x30;
        TR2 = 1;
        REN=1;
}

/*************************************************/
/*          Mian Program                */
/*************************************************/
void main (void)
{
        int ii;
        unsigned char StrBuff[18];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        CE  = 1;
        CS  = 1;
InitializeLCD();
        initializeRS232();
DR1 = 1;
CLK1= 1;
        Data=1;
dmsec(200);
       IE=0x90;
while(1){
                                //STEP #1 Config RF Module to Transmit Data
                                dmsec(100);
            RF_CONFIG_SEND();
                                delay_200usec();
        Address_Send();
                                delay_200usec();
                                //  STEP #2 Packing Data

            sprintf(Data_Buffer,":%3d%3dxxxxxxxxxxxxxxxxxxx",(int)High,(int)Low);
                                for(j=0;j<25;j++)
                                {
                                        if(Data_Buffer[j]==' ')Data_Buffer[j]='0';
                                }
                                // STEP #3 Send Data
        for (j=1;j<=25;j++)
        {
                                        BUFFER = Data_Buffer[j-1];
                                        Port_Test=~BUFFER;
            TX_DATA();

                                }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
delay_5usec();
                    // STEP #4 Config RF Module to Recieve Mode
                    CE=0;
                RF_CONFIG_RECIVE();
                    CE=1;
                    DR1=1;
                    ii=500; // Set TimeOut
                    while(DR1==0 && ii>0)
                    {
                        ii--;dmsec(5);
                        if(RxReady)
                        {
                            if(RecieveBuffer[0]        !='R')
                            {
                                High=RecieveBuffer[0]-0x30;
                                High=(High*10) + RecieveBuffer[1]-0x30;
                                High=(High*10) + RecieveBuffer[2]-0x30;
                                // Hign=80
                                Low=RecieveBuffer[3]-0x30;
                                Low=(Low*10) + RecieveBuffer[4]-0x30;
                                Low=(Low*10) + RecieveBuffer[5]-0x30;
                            }
sprintf(Data_Buffer,"%3d%3d%3d%1d",(int)O2_Data,(int)High,(int)Low,(int)PumpStatus);
                                for(j=0;j<10;j++)
                                {
                                    SendByte(Data_Buffer[j]);
                                }
                                SendByte(13);
                                SendByte(10);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    RxReady=0;
                }
            }
            CE=0;// Low Pwr Mode
            ii=0;
            if(DR1==1)
            {
        while (DR1)// Read from Buffer
            {
          READ_DATA();
                delay_10usec();
                Data_Buffer[ii]=i;
                ii++;
            }
            if(ii>0)
            {
                StrBuff[ii]=0;
                // Read High ,Low From Slave
                O2_Data=Data_Buffer[1]-0x30;
                O2_Data=(O2_Data*10) + Data_Buffer[2]-
0x30;
                O2_Data=(O2_Data*10) + Data_Buffer[3]-
0x30;

                if(High==0 || Low==0)
                {
                    High=Data_Buffer[4]-0x30;
                    High=(High*10) + Data_Buffer[5]-0x30;
                    High=(High*10) + Data_Buffer[6]-0x30;
                    // Hign=80
                    Low=Data_Buffer[7]-0x30;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                                        Low=(Low*10) + Data_Buffer[8]-0x30;
                                        Low=(Low*10) + Data_Buffer[9]-0x30;
                                    }
                                    if(Data_Buffer[10]=='0')PumpStatus=0; else
PumpStatus=1;
                                    sprintf(StrBuff,"%2.1f,  ",(float)(O2_Data)/10.0);
                                    PrintLCD(StrBuff,0);
                            sprintf(StrBuff,"%2.1f,%2.1f",(float)(High)/10.0,(float)(Low)/10.0);
                                    PrintLCD(StrBuff,5);
                                    if(PumpStatus==0)putc('1',15); else putc('0',15);
                                }
        }
                            else
                            {
                                    PrintLCD("  Time Out     ",0);
                                    dmsec(300);
                            }
        }
}.
```

โปรแกรมที่ใช้ใน Visual Basic

```
Dim Set_Hi, Set_Low As String
Dim OS1, OS2, OS3, OH1, OH2, OH3, OL1, OL2, OL3, Pump As String
Private Sub Buffer_txt_Change()
   'Write HHH LLL  chr(13)
   'Read  OOO HHH LLL S chr(13)
   Dim f
   If Buffer_txt.Text = "" Then Exit Sub
   f = InStr(1, Buffer_txt.Text, Chr(13))
   If f = 0 Then Exit Sub
   On Error Resume Next
   Pump = Mid(Buffer_txt.Text, f - 1, 1)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    OL3 = Mid(Buffer_txt.Text, f - 2, 1)
    OL2 = Mid(Buffer_txt.Text, f - 3, 1)
    OL1 = Mid(Buffer_txt.Text, f - 4, 1)
    OH3 = Mid(Buffer_txt.Text, f - 5, 1)
    OH2 = Mid(Buffer_txt.Text, f - 6, 1)
    OH1 = Mid(Buffer_txt.Text, f - 7, 1)
    OS3 = Mid(Buffer_txt.Text, f - 8, 1)
    OS2 = Mid(Buffer_txt.Text, f - 9, 1)
    OS1 = Mid(Buffer_txt.Text, f - 10, 1)

    LED7seg1.LEDnumber = Asc(OS1) - &H30
    LED7seg2.LEDnumber = Asc(OS2) - &H30
    LED7seg3.LEDnumber = Asc(OS3) - &H30

    Setpoint_Hi.Text = OH1 & OH2 & "." & OH3
    Setpoint_Low.Text = OL1 & OL2 & "." & OL3

    If Pump = "0" Then
      LED_ON.LEDstate = 1
      LED_OFF.LEDstate = 0
    Else
      LED_ON.LEDstate = 0
      LED_OFF.LEDstate = 1
    End If

    Buffer_txt.Text = ""

End Sub

Private Sub Command1_Click()
    If Setpoint_Hi.Text = "" Or Setpoint_Low.Text = "" Then
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    If Command1.Caption = "RESET" Then
      Setpoint_Hi.Enabled = True
      Setpoint_Low.Enabled = True
      Command1.Caption = "SET"
      Timer1.Enabled = False
    Else
      MsgBox ("Setpoint Error")
    End If
  Else
    If Command1.Caption = "SET" Then
      Command1.Caption = "RESET"
      Set_Hi = Setpoint_Hi.Text
      Set_Low = Setpoint_Low.Text
      Setpoint_Hi.Enabled = False
      Setpoint_Low.Enabled = False
      MSComm1.Output = Format(Val(Set_Hi) * 10, "000") & Format(Val(Set_Low) * 10, "000") &
Chr(13)
      Timer1.Enabled = True
      Timer1.Enabled = True
    Else
    Timer1.Enabled = False
      Command1.Caption = "SET"
      Setpoint_Hi.Enabled = True
      Setpoint_Low.Enabled = True
    End If
  End If
End Sub

Private Sub Command2_Click()
If Command2.Caption = "Open" Then
  MSComm1.CommPort = Combo1.ListIndex + 1
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    On Error GoTo Er_1
    MSComm1.PortOpen = True
    Command2.Caption = "Close"
    Command1.Enabled = True
    Timer1.Enabled = True
  Else
    Command2.Caption = "Open"
  End If
  Exit Sub

  Er_1:
  MsgBox "Port Error!"

  End Sub

  Private Sub MSComm1_OnComm()
    Buffer_txt.Text = Buffer_txt.Text & MSComm1.Input
  End Sub
  Private Sub Timer1_Timer()
    MSComm1.Output = "R" & Chr(13)
  End Sub
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้