



ภาควิชาวิศวกรรม  
คณะครุศาสตร์อุตสาหกรรม  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง  
ใบรับรองปริญญาโท

ชื่อหัวข้อ ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต  
Electrical Devices Controller via Internet

ชื่อนักศึกษา 1. นายณัฐพร ศรีโน รหัสประจำตัว 47035564  
2. นางสาวธนพร พรหมผุย รหัสประจำตัว 47035566  
3. นายนิรันดร์ มะ รหัสประจำตัว 47035571  
4. นายวิชิต วรรณชิต รหัสประจำตัว 47035684

หลักสูตร ครุศาสตร์อุตสาหกรรมบัณฑิต สาขาวิชา คอมพิวเตอร์  
อาจารย์ที่ปรึกษา อ.วรวิทย์ สมหา  
อาจารย์ที่ปรึกษาร่วม อ.สุชิน อัจหาญ

คณะกรรมการสอบปริญญาโท	ลายมือชื่อ
1. อ.ปิยะ ศุภาราสุวัฒน์	
2. ผศ.ธีระพล เทพหัสดิน ณ อยุธยา	
3. อ.สุชิน อัจหาญ	
4. อ.วรวิทย์ สมหา	
5. อ.ไพบุลย์ พวงวงศ์ตระกูล	

วัน/เดือน/ปีที่สอบ วันพุธที่ 26 เดือนเมษายน พ.ศ. 2549 เวลา 9.00 น.

สถานที่สอบ ห้อง ค.310 คณะครุศาสตร์อุตสาหกรรม สจล.

ภาควิชารับรองแล้ว

ลงนาม.....  
(ผศ.สุรสิทธิ์ รัตรี)

หัวหน้าภาควิชาครุศาสตร์วิศวกรรม  
วันที่ 1 เดือน พ.ศ. 2549



<BT481262>

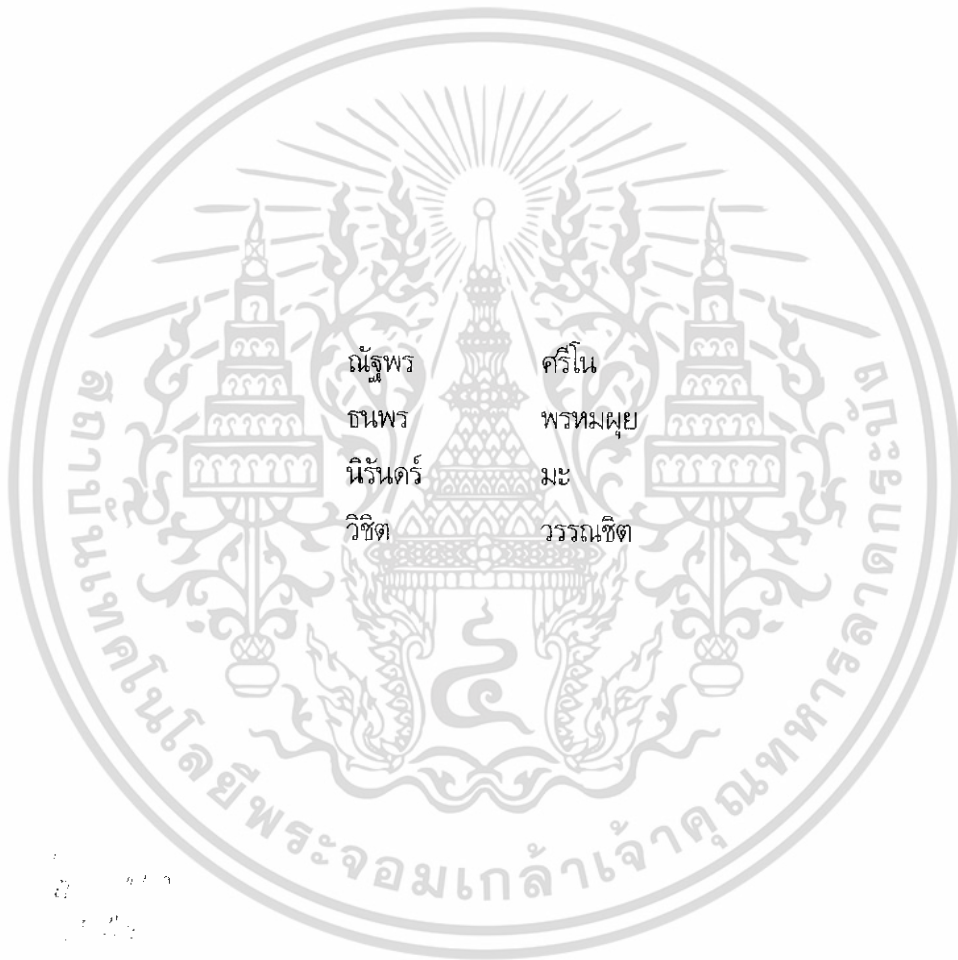
ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## ปริญญาโท

ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต

**ELECTRICAL DEVICES CONTROLLER VIA INTERNET**



เลขที่.....  
เลขประจำตัว..... **66676** .....

11660452

ปริญญาโทฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรครุศาสตรบัณฑิต  
สาขาวิชาคอมพิวเตอร์  
ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2548  
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## ปริญญาโท

เรื่อง ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต  
Electrical devices Controller via Internet

### วัตถุประสงค์

1. เพื่อศึกษาการทำงานของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตได้
2. เพื่อออกแบบวงจรควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตได้
3. เพื่อสร้างชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตได้
4. เพื่อทดสอบการทำงานของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตได้
5. เพื่อนำชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตไปใช้งานได้จริง

### ประโยชน์ที่คาดว่าจะได้รับ

1. ได้ความรู้เกี่ยวกับการทำงานของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต
2. ได้วงจรควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต
3. ได้ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต
4. ได้ผลการทดลองของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต
5. ได้นำชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตไปใช้งานได้จริง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชื่อหัวข้อ	ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต	
นักศึกษา	นายณัฐพร	ศรีโน
	นางสาวธนพร	พรหมผุย
	นายนิรันดร์	มะ
	นายวิชิต	วรรณชิต
อาจารย์ที่ปรึกษา	อาจารย์วรวิทย์	สมหา
อาจารย์ที่ปรึกษาร่วม	อาจารย์สุชิน	อาจหาญ
หลักสูตร	ครุศาสตร์อุตสาหกรรมบัณฑิต	
สาขาวิชา	คอมพิวเตอร์	
ปีการศึกษา	2548	

#### บทคัดย่อ

ปฏิญานิพนธ์ฉบับนี้เป็นการประยุกต์ใช้งานพอร์ตอานุกรมเพื่อเข้ามาใช้ควบคุมอุปกรณ์ไฟฟ้า ผู้ใช้สามารถใช้คอมพิวเตอร์จากที่ต่างๆ ผ่าน Hyper Terminal มาควบคุมอุปกรณ์ไฟฟ้า (เปิด/ปิด) พร้อมทั้งตรวจสอบสถานะของอุปกรณ์ไฟฟ้าภายในอาคารผ่านตัวไมโครคอนโทรลเลอร์ โดยการควบคุมจะกระทำได้ทั้งผ่านคอมพิวเตอร์และการกดสวิตช์จากตัวชุดควบคุม เราสามารถเปิด-ปิดอุปกรณ์ไฟฟ้าและตรวจสอบสถานะของอุปกรณ์ไฟฟ้าภายในอาคารจากที่ใดก็ได้ ถ้าจุดนั้นมีการเชื่อมต่อกับชุดควบคุม

Thesis Title	Electrical devices Controller via Internet
Students	Mr. Nuttaporn Srino Ms. Thanaporn Phomphui Mr. Niran Mah Mr. Vichit Vannachit
Advisor	Mr. Worawit Somha
Co-Advisor	Mr. Suchin Ardhan
Education Level	Bachelor of Science in Industrial Education
Program in	Computer
Academic Year	2005

**ABTRACT**

This project proposes to apply Serial port for electrical devices control. The user can use any computer via the Hyper Terminal to control the electrical devices (ON/OFF) and inspect status via microcontroller. It's can control via Computer and control switch. We can turn on/off any electrical devices and check status from anywhere in the organization if we have setup this module.

## กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สามารถสำเร็จลุล่วงได้ด้วยดีนั้น เนื่องมาจากความร่วมมือร่วมใจของสมาชิกภายในกลุ่มทุกท่าน คณะผู้จัดทำขอขอบพระคุณท่านอาจารย์วรวิทย์ สมหาและอาจารย์สุชิน อาจหาญ รวมไปถึงอาจารย์ประจำภาควิชาครุศาสตร์ศึกษาศาสตร์วิศวกรรมทุกท่านเป็นอย่างมากที่ได้กรุณาให้คำปรึกษาและคำแนะนำในการแก้ไขปัญหาต่างๆ ตลอดจนจนถึงข้อมูลและอุปกรณ์ที่เป็นประโยชน์ต่อการทดลองโครงการงาน และในการจัดทำปริญญานิพนธ์ฉบับนี้ ขอขอบคุณห้องสมุดคณะครุศาสตร์อุตสาหกรรม ห้องสมุดคณะวิศวกรรมศาสตร์ และสำนักหอสมุดกลางที่ช่วยอำนวยความสะดวกและเอื้อเฟื้อสถานที่ในการค้นคว้าหาข้อมูล ขอขอบคุณพ่อ แม่ที่ให้กำเนิดและให้การสนับสนุนในทุกเรื่องด้วยดีตลอดมา ตลอดจนญาติพี่น้องทุกคนที่ให้กำลังใจ ครูอาจารย์ทุกท่านที่ได้ให้สิ่งดีงามแก่คณะผู้จัดทำ และสุดท้ายขอขอบคุณเพื่อนร่วมงานทุกท่าน ที่ร่วมแรงร่วมใจเป็นหน้าหนึ่งใจเดียวจนงานสำเร็จลุล่วงไปด้วยดี



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## สารบัญ

เรื่อง	หน้า
บทคัดย่อภาษาไทย	I
บทคัดย่อภาษาอังกฤษ	II
กิตติกรรมประกาศ	III
สารบัญ	IV
สารบัญตาราง	VII
สารบัญรูป	VIII
บทที่ 1 บทนำ	1
1.1 ความเป็นมาและความสำคัญของปัญหา	1
1.2 จุดมุ่งหมายของโครงการ	1
1.3 สมมุติฐานของการจัดทำโครงการ	1
1.4 ขีดความสามารถของโครงการ	1
1.5 ขั้นตอนการทำโครงการ	2
1.6 เนื้อหาโดยสังเขป	2
บทที่ 2 ทฤษฎีและหลักการ	4
2.1 ระบบควบคุมระยะไกล	4
2.1.1 ระบบของสัญญาณควบคุม	5
2.1.1.1 ระบบใช้สาย	5
2.1.1.2 ระบบไร้สาย	5
2.1.1.3 การกำหนดช่องสัญญาณ	5
2.1.1.4 การจัดรหัสเป็นอนุกรมพัลส์	6
2.2 โปรโตคอล	7
2.2.1 รูปแบบของโปรโตคอล	7
2.2.1.1 ไบต์โอเรียนโปรโตคอล	8
2.2.1.2 บิตโอเรียนโปรโตคอล	8
2.2.2 โปรโตคอลของการสื่อสารแบบอนุกรม	8
2.3 พอร์ตอนุกรม	9
2.3.1 มาตรฐานพอร์ตอนุกรม	9
2.3.1.1 UART	11

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้ทำงานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## สารบัญ (ต่อ)

เรื่อง	หน้า
2.3.1.2 ชนิดของ UART	11
2.3.2 เปรียบเทียบมาตรฐานสัญญาณอนุกรมแบบต่างๆ	12
2.3.2.1 มาตรฐานสัญญาณอนุกรมแบบ RS-232C	12
2.3.2.2 มาตรฐานสัญญาณอนุกรมแบบ RS-422	12
2.3.2.3 มาตรฐานสัญญาณอนุกรมแบบ RS-485	13
2.4 ไมโครคอนโทรลเลอร์	14
2.4.1 ความรู้เบื้องต้นเกี่ยวกับไมโครคอนโทรลเลอร์	14
2.4.1.1 หน่วยประมวลผลกลางหรือซีพียู	15
2.4.1.2 หน่วยความจำ	16
2.4.1.3 รีจิสเตอร์	17
2.4.2 สถาปัตยกรรมและโครงสร้างทางฮาร์ดแวร์ของ PIC16F628	18
2.5 Java Servlet	26
2.5.1 จุดเด่นของ Java Servlet	26
2.5.2 สถาปัตยกรรมของ Servlet	27
2.5.3 การใช้ Java Servlet กับการพัฒนาระบบงานบนเครือข่ายอินเทอร์เน็ต	27
บทที่ 3 การออกแบบ การสร้างและการทำงาน	35
3.1 กล่าวนำ	35
3.2 สถาปัตยกรรมของระบบ	35
3.3 ฮาร์ดแวร์	37
3.3.1 แผงวงจรชุดแม่ข่าย	37
3.3.2 แผงวงจรไมโครคอนโทรลเลอร์	38
3.3.3 แผงวงจรจ่ายไฟ	39
3.4 โปรโตคอล	42
3.5 ซอฟต์แวร์	43
3.5.1 ซอฟต์แวร์ส่วนควบคุมและติดต่อกับฮาร์ดแวร์	44
3.5.2 ซอฟต์แวร์ส่วนแสดงผลผ่านอินเทอร์เน็ต	44
บทที่ 4 การทดลองและผลการทดลอง	45
4.1 กล่าวนำ	45

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## สารบัญ (ต่อ)

เรื่อง	หน้า
4.2 การทดลองการทำงานของแผงวงจรไมโครคอนโทรลเลอร์ PIC16F628	45
4.2.1 การทดลองติดต่อผ่านคอมพิวเตอร์	45
4.2.2 การทดลองติดต่อผ่านสวิตช์	49
4.3 สรุปผลการทดลอง	58
บทที่ 5 บทสรุป	59
5.1 บทสรุป	59
5.2 ปัญหาและวิธีการแก้ไข	59
5.3 แนวทางการพัฒนา	60
บรรณานุกรม	61
ภาคผนวก ก เครื่องต้นแบบ	62
ภาคผนวก ข วงจรและแผ่นวงจรพิมพ์	63
ภาคผนวก ค รายการอุปกรณ์	65
ภาคผนวก ง รายละเอียดและคุณสมบัติของอุปกรณ์	66
ภาคผนวก จ รหัสต้นฉบับโปรแกรม	84
ภาคผนวก ฉ คู่มือการใช้งาน	89
ประวัติผู้แต่ง	94

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## สารบัญตาราง

ตารางที่	หน้า
2.1 การเปรียบเทียบการสื่อสารแบบอนุกรมชนิดต่างๆ	14
2.2 รายละเอียดของขาต่อใช้งานของไมโครคอนโทรลเลอร์ PIC16F628	22
2.3 เปรียบเทียบคุณสมบัติของไมโครคอนโทรลเลอร์ PIC16F84 กับ PIC16F628	25
4.1 สถานะของอุปกรณ์ไฟฟ้าผ่านทาง Hyper Terminal module ตัวที่ 1	50
4.2 สถานะของอุปกรณ์ไฟฟ้าที่ดูจากชุดควบคุมอุปกรณ์ module ตัวที่ 1	51
4.3 สถานะของอุปกรณ์ไฟฟ้าผ่านทาง Hyper Terminal module ตัวที่ 2	52
4.4 สถานะของอุปกรณ์ไฟฟ้าที่ดูจากชุดควบคุมอุปกรณ์ module ตัวที่ 2	53
4.5 สถานะของอุปกรณ์ไฟฟ้าผ่านทาง Hyper Terminal module ตัวที่ 3	54
4.6 สถานะของอุปกรณ์ไฟฟ้าที่ดูจากชุดควบคุมอุปกรณ์ module ตัวที่ 3	55
4.7 สถานะของอุปกรณ์ไฟฟ้าผ่านทาง Hyper Terminal module ตัวที่ 4	56
4.8 สถานะของอุปกรณ์ไฟฟ้าที่ดูจากชุดควบคุมอุปกรณ์ module ตัวที่ 4	57

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## สารบัญรูป

รูปที่	หน้า
2.1 โครงสร้างและหลักการทำงานของระบบควบคุมระยะไกล	4
2.2 การทำงานของรีจิสเตอร์	6
2.3 การสร้างพัลส์อนุกรม	7
2.4 ลักษณะสมบัติทางไฟฟ้าของการอินเทอร์เฟสแบบ RS-232C	12
2.5 ลักษณะสมบัติทางไฟฟ้าของการอินเทอร์เฟสแบบ RS-422	13
2.6 ลักษณะสมบัติทางไฟฟ้าของการอินเทอร์เฟสแบบ RS-485	13
2.7 แผนผังการทำงานพื้นฐานของไมโครคอนโทรลเลอร์ที่ใช้สถาปัตยกรรมแบบฮาร์วาร์ด	18
2.8 แผนผังการทำงานของกระบวนการไปป์ไลน์ที่ใช้ในไมโครคอนโทรลเลอร์ PIC	19
2.9 แผนผังโครงสร้างการทำงานของไมโครคอนโทรลเลอร์ PIC16F628	20
2.10 การจัดขาของไมโครคอนโทรลเลอร์ PIC16F628	21
2.11 แผนผังการทำงานของ Servlet	26
3.1 สถาปัตยกรรมของระบบ	36
3.2 การเชื่อมต่อของวงจรชุดแม่ข่าย	37
3.3 การเชื่อมต่อของแผงวงจรไมโครคอนโทรลเลอร์	38
3.4 การเชื่อมต่อของแผงควบคุมอุปกรณ์ไฟฟ้า	39
3.5 วงจรแหล่งจ่ายไฟ	39
3.6 ผังงานการส่งค่าข้อมูลผ่านทางพอร์ต RS-232	40
3.7 ผังงานการส่งค่าข้อมูลโดยการกดสวิทช์	41
3.8 เฟรมข้อมูลที่คอมพิวเตอร์ส่งไปยังไมโครคอนโทรลเลอร์	42
3.9 เฟรมข้อมูลที่ไมโครคอนโทรลเลอร์ส่งไปยังคอมพิวเตอร์	42
3.10 โครงสร้างของระบบการรับส่งข้อมูลบนเครื่องเซิร์ฟเวอร์	43
4.1 การทดลองส่งค่าข้อมูลผ่านคอมพิวเตอร์ไปยังแผงวงจรควบคุมอุปกรณ์ไฟฟ้า	45
4.2 หน้าแรกของโปรแกรม Hyper Terminal	46
4.3 การกำหนดค่าการเชื่อมต่อของ Connect using	46
4.4 การกำหนดบอร์ดเวตในการเชื่อมต่อ	47
4.5 หน้าหลักโปรแกรม Hyper Terminal ที่ใช้ในการเชื่อมต่อ	47
4.6 โปรแกรมรับค่า 110 แล้วแสดงค่าสถานะออกผ่านทาง Hyper Terminal	48
4.7 โปรแกรมรับค่า 11C แล้วแสดงค่าสถานะออกผ่านทาง Hyper Terminal	48

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์สำหรับการใช้งานเพื่อการศึกษาเท่านั้น เมื่อผู้ยูเอตเห็นนำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## สารบัญรูป (ต่อ)

รูปที่	หน้า
4.8 โปรแกรมรับค่าการกดสวิตช์แล้วแสดงออกผ่านทาง Hyper Terminal	49



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

# บทที่ 1

## บทนำ

### 1.1 ความเป็นมาและความสำคัญของปัญหา

การควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าในระยะไกล ทำได้โดยใช้คำสั่งผ่านทางโทรศัพท์ โดยมีหลักการในการทำงานคือ เมื่อเราต้องการจะควบคุมเครื่องใช้ไฟฟ้า เราต้องโทรศัพท์กลับไปยังสถานที่นั้นๆ แล้วเครื่องตอบรับจะแจ้งสถานะ การทำงานของเครื่องใช้ไฟฟ้านั้นกลับมาและผู้ใช้งานจำเป็นต้องจำรหัสสำหรับการตรวจสอบเท่านั้น จึงทำให้ยุ่งยาก เสียเวลาและเสียค่าใช้จ่ายสูง หากผู้ใช้จำรหัสสำหรับการตรวจสอบผิดพลาด จะทำให้เกิดความเสียหายเป็นอย่างมาก

### 1.2 จุดมุ่งหมายของโครงการ

ระบบอินเทอร์เน็ตได้เข้ามามีบทบาทในชีวิตประจำวันเป็นอย่างมาก โดยเฉพาะการติดต่อสื่อสารทุกรูปแบบ ไม่ว่าเราจะอยู่ที่ใดก็ตามก็สามารถเข้าถึงอินเทอร์เน็ตได้ เนื่องจากอินเทอร์เน็ตนั้นครอบคลุมทั่วมุมโลก จากข้อดีของอินเทอร์เน็ตดังกล่าว คณะผู้จัดทำจึงเกิดแนวความคิดในการนำอินเทอร์เน็ตมาใช้ในการผสมผสานกับระบบอิเล็กทรอนิกส์ เพื่อประยุกต์ใช้ในการควบคุมระยะไกล ซึ่งประกอบไปด้วยส่วนของฮาร์ดแวร์และซอฟต์แวร์ เพียงเท่านี้เราก็จะสามารถนำประโยชน์ไปใช้ในงานด้านต่างๆ รวมไปถึงอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านได้อีกด้วย

### 1.3 สมมุติฐานของการจัดทำโครงการ

เมื่อนำชุดควบคุมอุปกรณ์ไฟฟ้ามาติดตั้งภายในตัวอาคารแล้ว ผู้ใช้สามารถสั่งการทำงานของอุปกรณ์ไฟฟ้าได้โดยการล็อกอินผ่านทางเว็บเบราว์เซอร์ รวมไปถึงยังสามารถตรวจสอบสถานะของอุปกรณ์ไฟฟ้าภายในตัวอาคารได้อีกด้วย

### 1.4 ขีดความสามารถของโครงการ

โครงการนี้มีขีดความสามารถดังนี้

1. สามารถโปรแกรมได้ 250 ชิ้น
2. สามารถควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตได้อย่างน้อย 4 ชิ้น
3. สามารถควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต
4. สามารถบอกสถานะการทำงานของอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. สามารถควบคุมอุปกรณ์ไฟฟ้าขนาดไม่เกิน 100 วัตต์
6. สามารถควบคุมอุปกรณ์ไฟฟ้าผ่านเว็บเบราว์เซอร์

## 1.5 ขั้นตอนการทำโครงการ

โครงการนี้ประกอบด้วยฮาร์ดแวร์และซอฟต์แวร์ ซึ่งการทำงานในระยะแรกจะเริ่มต้นจากการทำฮาร์ดแวร์ หลังจากนั้นเมื่อสร้างฮาร์ดแวร์ได้ระดับหนึ่งทีพอเพียงสำหรับการเขียนโปรแกรมควบคุมได้ก็จะเริ่มเขียนโปรแกรมทดสอบพร้อมกับการทำฮาร์ดแวร์ส่วนอื่นเพิ่มเติม และเมื่อทำโครงการเสร็จเรียบร้อยแล้วจะให้อาจารย์ที่ปรึกษาโครงการทำการตรวจสอบขีดความสามารถและประสิทธิภาพของชุดควบคุมอุปกรณ์ไฟฟ้าต่อไป

## 1.6 เนื้อหาโดยสังเขป

เนื้อหาภายในปฏิญานิพนธ์ฉบับนี้แบ่งออกเป็นบทต่างๆ เพื่อสะดวกต่อการศึกษาและทำความเข้าใจในแต่ละบทจะประกอบไปด้วยเนื้อหาดังต่อไปนี้

บทที่ 1 กล่าวถึงความเป็นมาและความสำคัญของปฏิญานิพนธ์ ขีดความสามารถของโครงการและเนื้อหาในบทต่างๆ โดยสังเขป

บทที่ 2 ประกอบด้วย ทฤษฎีต่างๆ เกี่ยวกับองค์ประกอบของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต รูปแบบการส่งข้อมูลของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต ลักษณะการเชื่อมต่อของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต หลักการทำงานของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต คำสั่งที่ใช้ในการเขียนโปรแกรมของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต

บทที่ 3 กล่าวถึงเนื้อหาเกี่ยวกับ สถาปัตยกรรมของระบบ แผนผังการทำงานของโครงการ ผังวงจรต่างๆ ที่ใช้ในโครงการ ตลอดจนการออกแบบและการสร้างส่วนประกอบต่างๆ เช่น แผงวงจรชุดแม่ข่าย แผงวงจรไมโครคอนโทรลเลอร์ การเชื่อมต่อของแผงควบคุมอุปกรณ์ไฟฟ้า วงจรจ่ายไฟ เป็นต้น พร้อมทั้งการทำงานของส่วนประกอบต่างๆ โดยละเอียด รวมไปถึงซอฟต์แวร์ที่ใช้ในการควบคุมและติดต่อกับฮาร์ดแวร์

บทที่ 4 ประกอบด้วย การทดลองและผลการทดลองของวงจรชุดควบคุมอุปกรณ์ไฟฟ้าผ่านสวิชต์ที่บอร์ดควบคุมและวงจรชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต

บทที่ 5 เป็นการสรุปผลการจัดทำโครงการ ปัญหาที่เกิดขึ้นและแนวทางในการแก้ไข รวมทั้งแนวทางในการพัฒนา

ภาคผนวก ก แสดงภาพเครื่องต้นแบบ การติดตั้ง การเชื่อมต่อกับอุปกรณ์อื่นๆ ขณะใช้งานจริง

ภาคผนวก ข แสดงประกอบด้วยผังรายละเอียดวงจรแผ่นวงจรพิมพ์

ภาคผนวก ค แสดงรายการอุปกรณ์ที่ใช้งานในแต่ละวงจร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ง แสดงรายละเอียดและคุณสมบัติของอุปกรณ์ควบคุมเพื่อประกอบการทำงานของ  
โครงการ

ภาคผนวก จ แสดงรหัสต้นฉบับโปรแกรม

ภาคผนวก ฉ แสดงคู่มือการใช้งาน



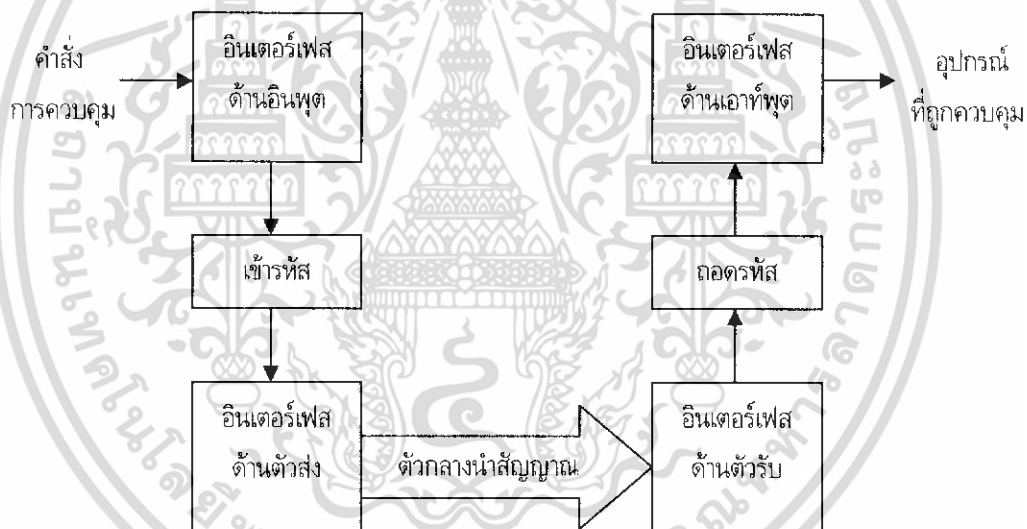
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## บทที่ 2

### ทฤษฎีและหลักการ

#### 2.1 ระบบควบคุมระยะไกล

โครงสร้างของระบบควบคุมระยะไกลแสดงดังรูปที่ 2.1 ในลักษณะทั่วไปคือ การควบคุมแบบทางเดียว เริ่มจากตัวกำหนดคำสั่งที่ใช้สำหรับการควบคุมเมื่อมีการกำหนดรูปของคำสั่งแล้ว รูปแบบของคำสั่งจะถูกส่งไปยังภาคส่งสัญญาณที่ทำหน้าที่แปลงสัญญาณหรือรวมสัญญาณควบคุมให้มีรูปแบบที่เหมาะสมกับวงจร



รูปที่ 2.1 โครงสร้างของระบบควบคุมระยะไกล

โดยอาจทำการเข้ารหัสสัญญาณให้แต่ละคำสั่งมีรหัสเฉพาะของมันเองให้เป็นสัญญาณทางไฟฟ้า ก่อนที่จะถูกส่งออกไปยังภาครับโดยตัวอินเตอร์เฟสด้านส่ง ที่เพื่อทำหน้าที่ส่งสัญญาณที่ภาครับต้องเข้าใจได้นั้นคือต้องเป็นระบบเดียวกัน สัญญาณที่ส่งออกมาอาจอยู่ในรูปของสัญญาณไฟฟ้า สัญญาณแสงหรือสัญญาณความถี่สูง สัญญาณนี้สามารถเดินทางผ่านตัวกลางที่เป็นสายนำสัญญาณหรือผ่านตัวกลางอากาศ ขึ้นอยู่กับระบบที่ถูกออกแบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญญาณที่เข้ามายังเครื่องรับหรือภาครับ จะถูกตัวอินเทอร์เฟสทำหน้าที่แปลงสัญญาณให้อยู่ในรูปของสัญญาณไฟฟ้าที่เข้ากับระบบของตัวรับ ก่อนถูกถอดรหัสเพื่อทราบวัตถุประสงค์ของคำสั่ง จากนั้นส่วนของวงจรรีโมตคอนโทรลด้านเอาต์พุตจะทำหน้าที่ควบคุมการทำงานของอุปกรณ์ที่ต้องการตามลักษณะคำสั่งที่ได้รับ

ระบบที่กล่าวมาเป็นการควบคุมระยะไกลแบบทางเดียวที่มีการสั่งงานที่จุดหนึ่งแล้วเกิดการทำงานที่จุดหนึ่ง หากจุดที่ถูกสั่งให้ทำงานสามารถสั่งการกลับมาไปยังจุดเริ่มต้นให้ทำงานได้ด้วย ถือว่าเป็นระบบควบคุมระยะไกลแบบสองทาง ซึ่งพบในระบบสื่อสารทั่วไป

### 2.1.1 ระบบของสัญญาณควบคุม

ประเภทของการควบคุมของรีโมตคอนโทรลอาจจำแนกได้เป็น 2 ประเภทตามลักษณะการส่งผ่านสัญญาณ

#### 2.1.1.1 ระบบใช้สาย

เป็นระบบควบคุมที่ต้องมีอุปกรณ์นำสัญญาณจากตัวส่งไปยังตัวรับซึ่งอุปกรณ์การนำสัญญาณนั้นขึ้นอยู่กับชนิดของสัญญาณพาห้ ซึ่งอาจได้แก่ สัญญาณไฟฟ้าและสัญญาณแสง

ในกรณีของสัญญาณไฟฟ้า อุปกรณ์นำสัญญาณอาจได้แก่ สายไฟ สายโคแอกเชียล สายโทรศัพท์ ฯลฯ แต่ถ้าในกรณีสัญญาณแสง อุปกรณ์นำสัญญาณจะเป็นเส้นใยแก้วนำแสง

#### 2.1.1.2 ระบบไร้สาย

เป็นระบบควบคุมที่ไม่ต้องมีอุปกรณ์ใดๆ เป็นตัวนำสัญญาณ โดยสัญญาณควบคุมจะเดินทางผ่านไปสู่อากาศ ชนิดของสัญญาณควบคุมอาจอยู่ในรูปของสัญญาณเสียง สัญญาณแสงและคลื่นวิทยุ ซึ่งปัจจุบันสัญญาณแสงที่นิยมใช้ได้แก่ ระบบอินฟราเรด

#### 2.1.1.3 การกำหนดของสัญญาณ

ในการกำหนดรหัสสัญญาณเฉพาะเพื่อใช้ควบคุมช่องสัญญาณ มักใช้รหัสที่เป็นสัญญาณดิจิทัลของเลขไบนารี โดยจำนวนช่องสัญญาณการทำงานขึ้นอยู่กับจำนวนบิตที่ใช้ โดยหลักการทำงานของวงจรรหัสช่องสัญญาณยังคงมีหลักการเหมือนกันทุกประการ

ในแต่ละบิตของสัญญาณควบคุมแสดงด้วยสัญญาณพัลส์สำหรับค่าของบิตที่เป็น 1 และไม่มีพัลส์หรือคงสถานะต่ำไว้เมื่อค่าของบิตเป็น 0 ข้อมูลทั้งหมดตามจำนวนบิตที่ใช้มักถูกจัดเรียงเป็นลำดับอนุกรมพัลส์และส่งไปยังเครื่องรับ เช่น หากเราออกแบบรีโมตคอนโทรลให้มีจำนวนบิตเป็น 4 บิต เราสามารถควบคุมการทำงานได้สูงสุด 16 ช่องสัญญาณ

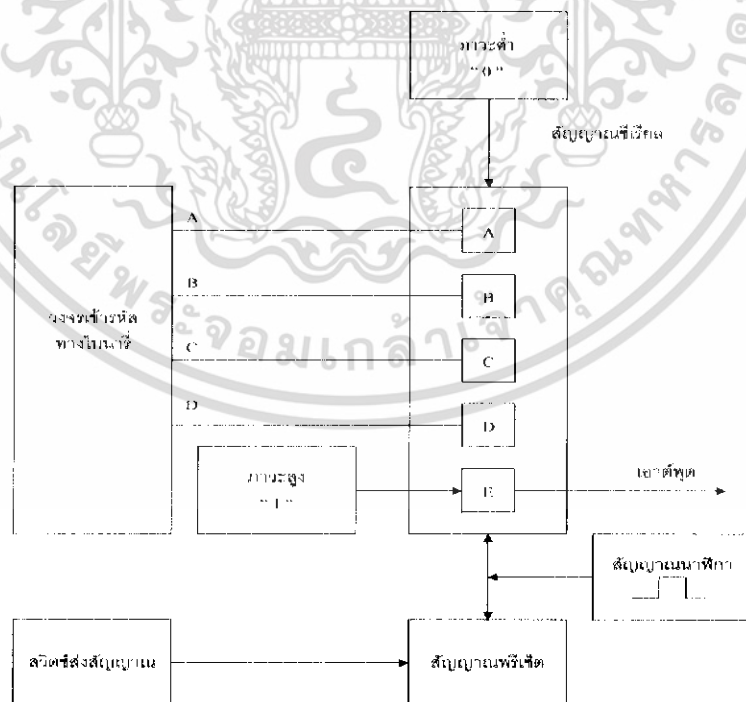
ลักษณะของรหัสสัญญาณที่กำหนด เมื่อถูกส่งออกจะจัดเรียงลำดับเป็นอนุกรมออกไป มีการเรียงลำดับความสำคัญจากมากที่สุดไปหาน้อยที่สุด เครื่องรับอาจมีการผิดพลาดในการตีความของสัญญาณได้ หากช่วงเวลาของการส่งรหัสควบคุมเกิดคลาดเคลื่อนไป จึงจำเป็นต้องมีการชิงโครนัสหรือสัญญาณที่จะบอกให้ทราบเวลาเริ่มต้นที่แน่นอนของรหัสควบคุมไปพร้อมกัน โดยทั่วไปมักใช้พัลส์ที่แสดงสถานะสูงเป็นตัวบอกจุดเริ่มต้นของเวลาในการส่งสัญญาณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

#### 2.1.1.4 การจัดรหัสเป็นอนุกรมพัลส์

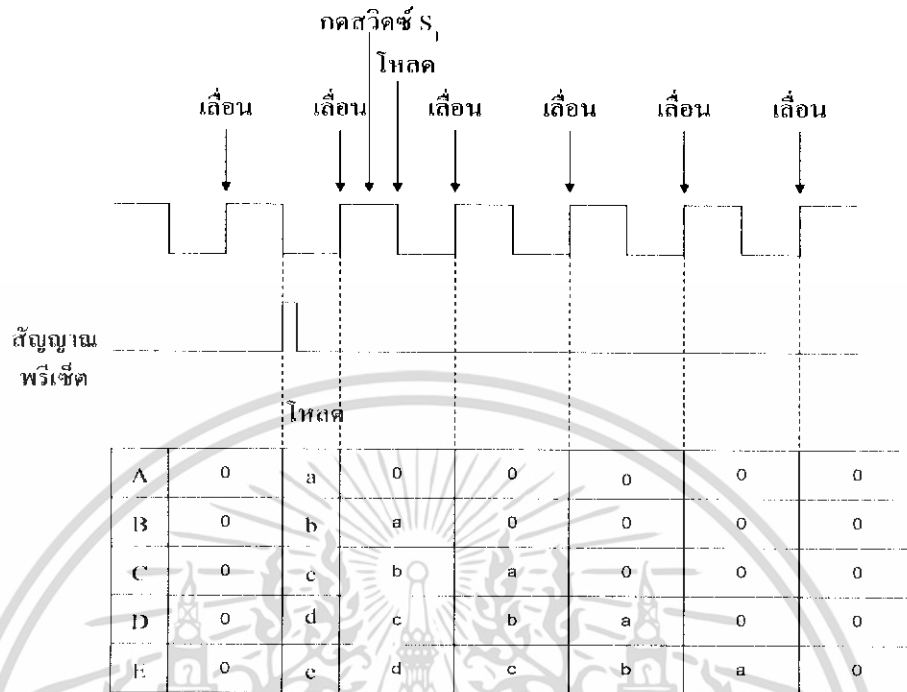
การสร้างรหัสสัญญาณดิจิทัลให้เป็นพัลส์อนุกรมสำหรับส่งออก สามารถใช้วงจรของชิฟต์รีจิสเตอร์ (Shift register) ดังรูปที่ 2.2 (ก) เมื่อต้องการการส่งสัญญาณควบคุมขนาด 4 บิต ที่มีพัลส์ซิงโครนัส ในสภาวะสูงนำหน้าการทำงานของวงจรถูกกำหนดความเร็วจากอัตราของสัญญาณนาฬิกาที่ใช้ในการกำหนดความกว้างของแต่ละบิตที่เป็นรหัสควบคุม

รหัสควบคุมที่ถูกสร้างขึ้นจากวงจรเข้ารหัสจะปรออยู่หน้ารีจิสเตอร์แต่ละตัวก่อน เมื่อมีการกดสวิตช์ส่งสัญญาณ จะเกิดการสร้างพัลส์ที่เป็น 1 บิตให้กับชิฟต์รีจิสเตอร์ จากนั้นรีจิสเตอร์แต่ละตัวจะทำการไหลข้อมูลจากวงจรเข้ามาเก็บไว้ ดังรูปที่ 2.2 (ข) เมื่อสัญญาณพัลส์เปลี่ยนกลับมาเป็น 0 รีจิสเตอร์ทำงานในลักษณะของการเลื่อนหรือชิฟต์ข้อมูลในแต่ละส่วนลงมา ในขณะที่สัญญาณนาฬิกาเกิดการเปลี่ยนแปลงจากสภาวะต่ำไปสูง โดยข้อมูลถูกเลื่อนมาเรื่อยๆ จาก A, B, C, D ไปยัง E ทำให้ได้เอาต์พุตที่ E เป็นสัญญาณพัลส์อนุกรมที่บรรจุข้อมูลเรียงลำดับจาก E, D, C, B และ A สำหรับใช้ส่งออกไป โดยที่สัญญาณสภาวะสูงของรีจิสเตอร์ E ตอนต้นใช้เป็นสัญญาณซิงโครนัส เมื่อข้อมูลที่ถูกลื่อนลงมาหมดแล้วรีจิสเตอร์จะมาสภาวะเป็น 0 เพราะไม่มีการไหลข้อมูลเข้าสู่รีจิสเตอร์ใหม่นอกจากต้องรอให้มีการสร้างสัญญาณพัลส์เป็น 1 จึงจะมีการไหลข้อมูลจากวงจรเข้ารหัสและเกิดการเลื่อนข้อมูลในลักษณะเดิมต่อไป



รูปที่ 2.2 การทำงานของรีจิสเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.3 การสร้างฟิลส์อนุกรม

## 2.2 โพรโตคอล

ในการสื่อสารระยะไกลนั้นสิ่งที่จะต้องทำเป็นก็คือ การกำหนดรูปแบบวิธีของการรับ-ส่งข้อมูล โดยฝ่ายรับและส่งต้องตกลงกันก่อนว่าจะใช้รูปแบบใดในการส่งข้อมูลร่วมกัน ดังนั้นโพรโตคอล หมายถึง วิธีการหรือกฎระเบียบที่ใช้ในการสื่อสารข้อมูล เพื่อให้ผู้รับและผู้ส่งสามารถเข้าใจกันหรือคุยกันรู้เรื่อง ซึ่งจะมีส่วนของซอฟต์แวร์ที่ทำหน้าที่ให้การดำเนินงานในการสื่อสารข้อมูลเป็นไปตามโปรแกรมที่กำหนด ตัวอย่างของโพรโตคอล ได้แก่ X25, BSC, SDLC, HDLC เป็นต้น ส่วนซอฟต์แวร์ เช่น IPX ในการต่อ LAN ของ Novell's Netware ในอดีตหรือ TCP/IP (Transmission Control Protocol/Internet Protocol) ที่ใช้ในการต่อในระบบ Internet ในปัจจุบัน

### 2.2.1 รูปแบบของโพรโตคอล

ในการสื่อสารข้อมูลจะต้องมีกฎหรือข้อกำหนดในการสื่อสารข้อมูลหรือที่นิยมเรียกว่าโพรโตคอล ซึ่งเป็นส่วนที่จะกำหนดมาตรฐานในการควบคุมและจัดระบบในการสื่อสารข้อมูลสำหรับรายละเอียดที่จะกล่าวในที่นี้จะเกี่ยวข้องเฉพาะในส่วนของการควบคุมการเชื่อมโยงข้อมูล (Data Link Control Protocol หรือ DLCP) ซึ่งจัดการในส่วนของการเชื่อมต่อและหลักการต่างๆ คือโครงสร้างและรายละเอียดของข้อมูล วิธีในการสื่อสารข้อมูล การตรวจสอบแก้ไขความผิดพลาดของข้อมูลและขบวนการในการควบคุมการสื่อสาร โดย DLCP แบ่งออกได้ตามโครงสร้างของข้อมูล 2 แบบ คือ Byte Oriented Protocol และ Bit Oriented

เอกสาร Protocol สารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 2.2.1.1 ไบต์โอเรียนโปรโตคอล (Byte Oriented Protocol)

โปรโตคอลแบบนี้เป็นโปรโตคอลที่การสื่อสารข้อมูลและการควบคุมการทำงานจะทำโดยใช้ลักษณะข้อมูลเป็นตัว (Character) หรืออาจเรียก Character Oriented Protocol ซึ่งแบ่งออกเป็น

#### 1. อะซิงโครนัสโปรโตคอล (Asynchronous Protocol)

โปรโตคอลในการสื่อสารข้อมูลนี้จะใช้สื่อสารข้อมูลแบบ half-Duplex ที่มีลักษณะการสื่อสารข้อมูลแบบอะซิงโครนัส ซึ่งเป็นในการสื่อสารข้อมูลแบบพื้นฐานที่ใช้งานมาเป็นเวลานานแล้ว จึงมีรายละเอียดและขั้นตอนในการสื่อสารข้อมูลที่ทำให้โอกาสเกิดความผิดพลาดได้น้อย ยังมีข้อดีที่การสื่อสารข้อมูลแบบนี้มีในโครงสร้างการทำงานที่ง่าย อุปกรณ์ที่ใช้ในการสื่อสารข้อมูลก็ไม่สลับซับซ้อนและมีราคาถูก โปรโตคอลแบบนี้จึงเหมาะสำหรับใช้ในระบขนาดเล็ก

#### 2. ไบนารีซิงโครนัสโปรโตคอล (Binary Synchronous Protocol)

โปรโตคอลแบบนี้จะมีลักษณะการทำงานที่ใช้งานข้อมูลเป็นลักษณะไบนารีและยังคงใช้การสื่อสารข้อมูลแบบซิงโครนัส มีรายละเอียดและขั้นตอนในการสื่อสารข้อมูลที่ทำให้ความน่าเชื่อถือมากกว่า อีกทั้งยังสามารถใช้อัตราเร็วในการสื่อสารข้อมูลที่สูงกว่า โดยตัวอย่างการสื่อสารข้อมูลแบบนี้ได้กำหนดเป็นมาตรฐานแล้ว คือการสื่อสารข้อมูลมาตรฐาน BSC (Binary Synchronous Communications) ซึ่งเป็นโปรโตคอลที่ลักษณะของข้อมูลแบบไบนารีที่นิยมนำไปใช้งาน

### 2.2.1.2 บิตโอเรียนโปรโตคอล (Bit Oriented Protocol)

โปรโตคอลแบบนี้เป็นโปรโตคอลที่การสื่อสารข้อมูลและการควบคุมการทำงานจะทำโดยใช้ลักษณะข้อมูลที่เป็นบิตโดยมีตัวอย่างของการสื่อสารข้อมูลในลักษณะนี้ที่มีการกำหนดขึ้นเป็นมาตรฐานแล้ว คือ HDLC (High-Level Data Link Control) โดยมีโครงสร้างของข้อมูลแบบซิงโครนัสเช่นเดียวกับ BSC แต่แตกต่างกันที่มีลักษณะของข้อมูลเป็นบิต ซึ่งโปรโตคอลแบบนี้ ข้อดีที่สามารถสื่อสารข้อมูลแบบ Full Duplex ได้ทำการสื่อสารข้อมูลได้รวดเร็วกว่า แต่โปรโตคอลแบบนี้ก็จะมีรายละเอียดและโครงสร้างในการสื่อสารข้อมูลที่สลับซับซ้อนมากทำให้การควบคุมการทำงานทำได้ยากและต้องใช้อุปกรณ์ที่มีราคาสูง จึงไม่เหมาะที่จะนำมาใช้กับงานที่มีขนาดเล็กๆ

## 2.2.2 โปรโตคอลของการสื่อสารแบบอนุกรม

เมื่อพิจารณาการส่งข้อมูลในแบบอนุกรมให้ตีจะพบว่า ปัญหาหนึ่งที่ควรจะต้องเกิดขึ้นอยู่เสมอ ก็คือการตัดสินใจว่าข้อมูลที่ได้รับนั้นมีจุดเริ่มต้นที่ใด ดังนั้นจึงมีการกำหนดข้อตกลงในการสื่อสารขึ้นเพื่อแก้ปัญหาในการสื่อสารข้อมูลแบบอนุกรมสามารถแบ่งออกเป็น 2 ประเภทใหญ่ๆ คือ โปรโตคอลสำหรับการสื่อสารข้อมูลแบบซิงโครนัส (Synchronous) และแบบอะซิงโครนัส (Asynchronous) การสื่อสารข้อมูลแบบซิงโครนัสนั้น ข้อมูลจะถูกส่งออกไปอย่างสม่ำเสมอช่วงเวลาระหว่างบิตและระหว่างเวิร์ดจะเท่ากันเสมอ ดังนั้นในการสื่อสารข้อมูลอนุกรมในแบบซิงโครนัสจึงต้องมีสายสัญญาณเพิ่มเติมกำกับกับการส่งว่าควรส่งเมื่อใดและควรหยุดเมื่อใด ระบบที่เป็นซิงโครนัสจะเป็นระบบที่มีความเร็วสูง แต่ก็ยังต่ำกว่าการสื่อสารแบบขนาน การสื่อสาร

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ไว้สำหรับการใช้ในเพื่อการศึกษาค้นคว้าเท่านั้น เมื่อผู้ดูแลเห็นประโยชน์ของเอกสารนี้  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ข้อแตกต่างของอุปกรณ์ DTE และอุปกรณ์ DCE อย่างหนึ่งที่เราเห็นได้ชัดก็คือ คอนเน็กเตอร์ของ DTE จะเป็นตัวผู้ ส่วนคอนเน็กเตอร์ของ DCE จะเป็นตัวเมีย ซึ่งพอร์ตอนุกรมของคอมพิวเตอร์ที่ใช้กันอยู่ทั่วไปจะเป็นแบบ DTE ส่วนคอนเน็กเตอร์ที่อยู่ที่ไม่เต็มจะเป็นแบบ DCE

สำหรับการใช้งานบนคอมพิวเตอร์ พอร์ตอนุกรม RS-232 มักถูกใช้เชื่อมต่อกับโมเด็มหรือเมาส์โดยสามารถรับส่งข้อมูลได้ด้วยความยาวของสายสัญญาณสูงสุดถึง 20 เมตร

มาตรฐานการเชื่อมต่อแบบ RS-232 จะใช้คอนเน็กเตอร์แบบ DB-25 จะมีขาต่อใช้งานเพียง 9 เส้น เช่นเดียวกับคอนเน็กเตอร์แบบ DB-9 เนื่องจากขาอื่นๆ ที่เคยใช้งานในอดีต ปัจจุบันมีการใช้งานไม่มากนัก จึงถูกยกเลิกไป

สำหรับการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอก ประกอบด้วยการเชื่อมต่อแบบ Null modem หรือการเชื่อมต่อโดยไม่ต้องผ่านโมเด็ม โดยมีการตรวจสอบหรือแฮนด์เช็กเต็มรูปแบบและการเชื่อมต่อแบบ Null modem ในลักษณะที่ใช้สายสัญญาณเพียง 3 เส้น โดยเส้นหนึ่งสำหรับส่งข้อมูล อีกเส้นสำหรับรับข้อมูล และเส้นสุดท้ายเป็นกราวด์ สำหรับรายละเอียดหน้าที่การทำงานในแต่ละขาของพอร์ตอนุกรม RS-232 มีดังนี้

1. Data Carrier Detect : DCD หรืออาจเรียกว่า Carrier Detect : CD ขานี้จะแอกตีฟเมื่อมีการส่งสัญญาณพาห์จากอุปกรณ์สื่อสารข้อมูล เช่น โมเด็ม สำหรับการใช้งานปกติ ขานี้จะไม่ได้ถูกใช้งานมากนัก

2. Receive Data : RD หรือ RxD ขานี้ใช้เพื่อรับสัญญาณอนุกรมเข้ามายังคอมพิวเตอร์โดยนำข้อมูลที่อ่านได้เก็บไว้ในรีจิสเตอร์ บัฟเฟอร์

3. Transmitted Data : TD หรือ TxD ขานี้ใช้เพื่อส่งข้อมูลออกจากคอมพิวเตอร์ โดยนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์สำหรับส่งข้อมูลออกไป

4. Data Terminal Ready : DTR เป็นขาสัญญาณที่ส่งออกจากคอมพิวเตอร์เพื่อให้อุปกรณ์ปลายทางรับรู้ว่าการติดต่อด้วย โดยขา DTR นี้จะต้องเชื่อมต่อกับขา DSR ของอุปกรณ์ปลายทางและขา DTR ของอุปกรณ์ปลายทางจะต้องเชื่อมต่อกับขา DSR ของคอมพิวเตอร์ ถ้าใช้การเชื่อมต่อเป็นแบบ Null Modem ซึ่งใช้สายในการเชื่อมต่อเพียง 3 เส้น จะต้องต่อขา DTR และ DSR ของตัวมันเองเข้าด้วยกันและต้องต่อกับขา DCD ด้วยกรณีที่ใช้โปรแกรมสื่อสารที่ใช้มีการตรวจสอบจับสัญญาณพาห์

5. Signal Ground : GND ขากราวด์ของระบบ

6. Data Set Ready : DSR ขานี้จะใช้งานคู่กับขา DTR เพื่อตรวจสอบการเชื่อมต่อกันระหว่างคอมพิวเตอร์กับอุปกรณ์ปลายทาง ซึ่งขา DSR นี้จะเป็นขาสำหรับรับข้อมูลจากภายนอกซึ่งถูกส่งมาจากขา DTR

7. Request To Send : RTS เป็นขาสำหรับส่งสัญญาณร้องขอให้ทางอุปกรณ์ปลายทางส่งข้อมูลกลับมายังคอมพิวเตอร์ โดยที่ขาที่รับสัญญาณ RTS ก็คือขา CTS ในกรณีที่ใช้การเชื่อมต่อแบบ Null Modem 3 สาย จะต้องเชื่อมต่อกับขา RTS และ CTS ของตัวมันเองเข้าด้วยกัน เพื่อจะให้การรับและส่งข้อมูลสามารถเกิดขึ้นได้ตลอดเวลา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8. Clear To Send : CTS ขานี้จะคอยรับสัญญาณจากขา RTS เมื่อรับสัญญาณได้ ข้อมูลที่ขา TxD จะถูกส่งออกไป ดังนั้นขานี้จึงถูกใช้เพื่อตรวจสอบอุปกรณ์ต่อพ่วงว่าพร้อมที่จะรับข้อมูลหรือไม่

9. Ring Indicator : RI ใช้แสดงสถานะสัญญาณเรียกจากสายโทรศัพท์ โดยปกติในการสื่อสารทั่วไปสายนี้จะไม่ถูกใช้งาน จะใช้งานก็ต่อเมื่อมีการเชื่อมต่อกับโมเด็มและโปรแกรมมีการตรวจสอบสัญญาณนี้เท่านั้น

### 2.3.1.1 UART

UART มาจากคำว่า Universal Asynchronous Receiver Transmitter ซึ่งหมายถึง อุปกรณ์ที่ทำหน้าที่รับและส่งข้อมูลแบบอะซิงโครนัสนั่นเอง สำหรับการสื่อสารอนุกรมบนคอมพิวเตอร์แล้ว UART ถือว่าเป็นหัวใจสำคัญของการสื่อสารอนุกรม

หน้าที่หลักของ UART คือ ทำหน้าที่แปลงข้อมูลที่อยู่ในรูปแบบขนานจากคอมพิวเตอร์ให้อยู่ในรูปแบบอนุกรมแบบอะซิงโครนัสแล้วส่งออกไปและทำหน้าที่แปลงสัญญาณอนุกรมแบบอะซิงโครนัสที่ป้อนเข้ามาให้ UART ให้เป็นแบบขนานก่อนที่จะส่งเข้าสู่คอมพิวเตอร์ ซึ่งนอกจาก UART จะส่งข้อมูลไปยังคอมพิวเตอร์แล้ว ยังแจ้งข้อมูลอื่นๆ ให้คอมพิวเตอร์ทราบด้วย เช่น อัตราเร็วในการรับส่งข้อมูล (buadrate) รูปแบบการส่งข้อมูล ความผิดพลาดที่เกิดขึ้นระหว่างการถ่ายทอดข้อมูล (ผิดพลาดจากพาริตีเฟรมข้อมูลและ โอเวอร์รัน) เป็นต้น

ภายใน UART จะมีส่วนของวงจรสร้าง buadrate แบบโปรแกรมได้ (Programmable buadrate generator) โดยการกำหนดค่าตัวหารให้กับสัญญาณนาฬิกาของ UART โดยตัวหารนี้มีขนาด 16 บิต ดังนั้นจึงสามารถกำหนดตัวหารอยู่ในช่วง 1-0.5,535 UART สามารถรับส่งข้อมูลได้ทั้งแบบฮาล์ฟดูเพล็กซ์ (half duplex) และฟูลดูเพล็กซ์ (full duplex) โดยการส่งแบบฮาล์ฟดูเพล็กซ์เป็นการส่งแบบทิศทางเดียว ส่วนการส่งแบบฟูลดูเพล็กซ์เป็นการส่งข้อมูลได้ในคราวเดียวกัน

### 2.3.1.2 ชนิดของ UART

ในเครื่องคอมพิวเตอร์ทั่วไป UART ที่ใช้งานกันอยู่ 2 เบอร์คือ 8250 ซึ่งเป็น UART มาตรฐานที่มีใช้กันมายาวนาน UART เบอร์นี้จะมียัพเพอร์สำหรับรับและส่งข้อมูลตำแหน่งเดียวกัน ทำให้การรับและส่งข้อมูลถูกจำกัดความเร็วอยู่ที่ 57.6 กิโลบิตต่อวินาทีเท่านั้น แต่ UART เบอร์นี้ก็ถือว่าเป็นต้นแบบของ UART ที่ใช้ในคอมพิวเตอร์ โดยคอมพิวเตอร์ทุกๆ รุ่นจะต้องสนับสนุนการทำงานตามรูปแบบของ UART เบอร์นี้

UART อีกเบอร์หนึ่งคือ 16450 มีความสามารถรับส่งข้อมูลได้ที่ความเร็ว 115,200 บิตต่อวินาที และเพิ่มรีจิสเตอร์สำหรับพักข้อมูลสำหรับ UART นอกจากนั้นยังเพิ่มส่วนของซีพียูรีจิสเตอร์แบบ FIFO (First In Frist Out) ขนาด 16 ไบต์เข้าไป ทำให้สามารถสนับสนุนความเร็วในการรับส่งข้อมูลที่ 256 กิโลบิตต่อวินาทีได้ โดยคอมพิวเตอร์ในปัจจุบันใช้ UART เบอร์นี้หรือใหม่กว่า เช่น เบอร์ TL16C750 ซึ่งมีรีจิสเตอร์แบบ FIFO ขนาด 64 ไบต์ ทำงานได้ที่ระดับแรงดัน +5V และ +3V มีโหมดประหยัดพลังงาน สามารถรับส่งข้อมูลได้ที่ความเร็ว 1 เมกะบิตต่อวินาที เมื่อใช้สัญญาณนาฬิกา 16 MHz

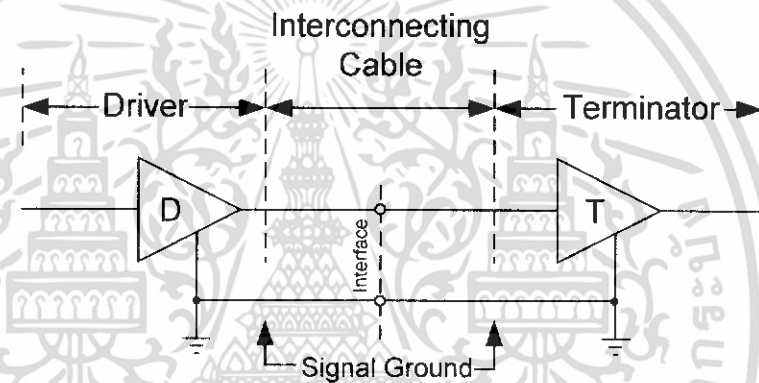
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อย่างไรก็ตาม ความเร็วในการส่งข้อมูลが多มายของ UART เวอร์ใหม่ๆ ก็ไม่ได้ช่วยให้การรับส่งข้อมูลของคอมพิวเตอร์เร็วขึ้น เนื่องจากคอมพิวเตอร์ยังใช้ความถี่ของสัญญาณนาฬิกาในการแปลงข้อมูลเพียง 1.8432 MHz เท่านั้น

## 2.3.2 เปรียบเทียบมาตรฐานสัญญาณอนุกรมแบบต่างๆ

### 2.3.2.1 มาตรฐานสัญญาณอนุกรมแบบ RS-232C

มาตรฐานการสื่อสารข้อมูลแบบอนุกรมที่กำหนดโดย EIA (Electronics Industries Association) มาตรฐาน RS-232 ได้ถูกตีพิมพ์ในปี ค.ศ. 1969 ตัวอักษร RS แทน "Recomm Standard" 232 แทนหมายเลขของมาตรฐาน ส่วนอักษร C แสดงให้รู้ว่ามาตรฐานได้รับการแก้ไขกี่ครั้ง



รูปที่ 2.4 ลักษณะสมบัติทางไฟฟ้าของการอินเตอร์เฟสแบบ RS-232C

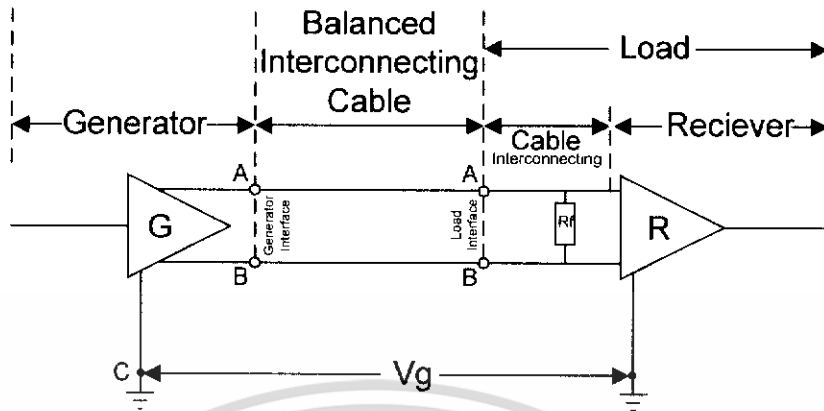
คุณลักษณะสมบัติทางไฟฟ้าของการอินเตอร์เฟสแบบ RS-232C

1. ถูกออกแบบให้ใช้กับอุปกรณ์พวกสัญญาณ Discrete
2. ใช้การอินเตอร์เฟสแบบ Unbalanced
3. ในแต่ละวงจรใช้ลวดนำในการนำสัญญาณ 1 เส้น และมีสายกราวด์รวม 1 เส้น
4. อัตราเร็วในการส่งข้อมูลมีค่า  $< 20$  กิโลบิตต่อวินาที
5. ระยะทางสูงสุดที่ใช้ในการส่งข้อมูลมีค่า  $< 15$  เมตร

### 2.3.2.2 มาตรฐานสัญญาณอนุกรมแบบ RS-422

มาตรฐาน RS-422 เป็นชุดขั้วแบบขยายความแตกต่างในรูปแบบของกระแส ทำให้อัตราเร็วในการส่งข้อมูลมีสูงขึ้นและระยะทางที่ใช้ส่งข้อมูลระหว่างชุดส่งและชุดรับมีระยะไกลขึ้น เพราะเป็นการส่งข้อมูลแบบฟูลดูเพล็กซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



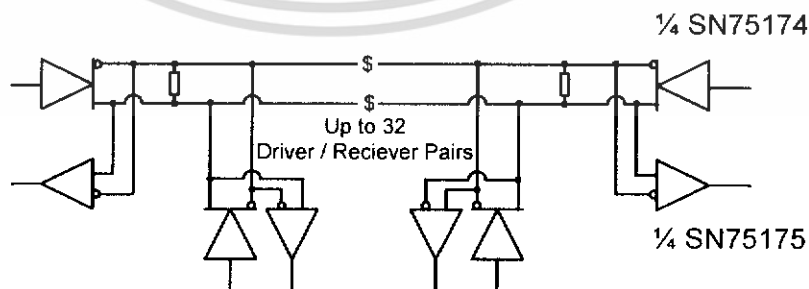
รูปที่ 2.5 ลักษณะสมบัติทางไฟฟ้าของการอินเตอร์เฟสแบบ RS-422

คุณลักษณะสมบัติทางไฟฟ้าของการอินเตอร์เฟสแบบ RS-422

1. ชุดคำสั่งสัญญาณแบบ Balanced
2. ชุดรับข้อมูลเป็นแบบขยายความแตกต่าง
3. ในแต่ละวงจรใช้ลวดตัวนำในการส่งสัญญาณจำนวน 2 เส้น
4. อัตราเร็วในการส่งข้อมูลสูงถึง 10 เมกะบิตต่อวินาที
5. ระยะทางที่ใช้ในการส่งข้อมูลได้ไกลถึง 400 ฟุต

**2.3.2.3 มาตรฐานสัญญาณอนุกรมแบบ RS-485**

มาตรฐาน RS-485 นี้ก็มีการพัฒนามาจาก RS-422 คือผู้ผลิตบางบริษัทได้ทำวงจรแบบขับสัญญาณ (ชุดส่ง) เป็นลักษณะ Tri-State ทำให้เราสามารถส่งข้อมูลได้สองทิศทางบนสายคู่เดียว (Single Pair) คุณสมบัตินี้จึงทำให้ระบบส่งข้อมูลมีโครงสร้างเป็นแบบ Multidrop ซึ่งอุปกรณ์หลายๆ ประเภทสามารถรับและส่งข้อมูลแบบฮาล์ฟดูเพล็กซ์บนสายเดี่ยวได้



รูปที่ 2.6 ลักษณะสมบัติทางไฟฟ้าของการอินเตอร์เฟสแบบ RS-485

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มาตรฐาน RS-485 นี้ ทางบริษัทผู้ผลิตได้ออกแบบชุดส่งและชุดรับให้สามารถต่อรวมอยู่บนบัสได้ถึงอย่างละ 32 ชุด แต่การส่งข้อมูลในขณะเวลาหนึ่งๆ นั้นจะส่งได้ที่ละชุด ฉะนั้นจะต้องมีการมวืติตรวจสอบบัสก่อนว่าว่างและพร้อมที่จะให้ข้อมูลผ่านไปได้ ทั้งนี้เพื่อป้องกันการสับสนของข้อมูลนั่นเอง การรับส่งข้อมูลสามารถที่จะรับส่งข้อมูลได้ถึง 32 บิต แต่ละชุดต้องมีการตรวจหัวข้อมูล (Header) หรือ Identify Code ก่อนเพื่อเป็นรหัสให้รับทราบว่าการสื่อสารกับชุดใด

ตารางที่ 2.1 การเปรียบเทียบการสื่อสารแบบอนุกรมชนิดต่างๆ

Specification	RS-232C	RS-422	RS-485
Mode Of Operated	Single Ended	Difference	Difference
No of Driver & Receivers	1 Driver	1 Driver	32 Drivers
Allowed On One Line	1 Receiver	32 Receivers	32 Reccivers
Max Cable Length	50 feet	4,000 feet	4,000 feet
Max Data Length	20 Kb/s	100 Kb/s	10 Mb/s
Driver Output Max Voltage	+/- 5 Vdc.	-0.25 to 6 Vdc.	-7 to 12 Vdc.
Driver Out (Load)	+/- 5 Vdc.	+/- 2 Vdc.	+/- 1.5 Vdc.
Signal Level (Unload)	+/- 15 Vdc.	+/- 5 Vdc.	+/- 5 Vdc.
Driver Load Impedance	3 to 7 K $\Omega$	100 $\Omega$	54 $\Omega$
Driver Output Current (ON)	-	-	+/- 100 $\mu$ A
High Impedance State OFF)	Vmax/300 $\Omega$	+/- 100 $\mu$ A	+/- 100 $\mu$ A
Receiver Input Volt Rang	+/- 15 Vdc.	+/- 7 Vdc.	-7 to 12 Vdc.
Receiver Input Sensitivity	+/- 3 Vdc.	+/- 200 m Vdc.	+/- 200 m Vdc.
Receiver Input Resistance	3 to 7 $\Omega$	4 K $\Omega$ (min)	12 K $\Omega$ (min)

## 2.4 ไมโครคอนโทรลเลอร์

### 2.4.1 ความรู้เบื้องต้นเกี่ยวกับไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) มาจากคำ 2 คำ คำหนึ่งคือ ไมโคร (Micro) หมายถึงขนาดเล็กและคำว่า คอนโทรลเลอร์ (Controller) หมายถึง ตัวควบคุมหรืออุปกรณ์ควบคุม ดังนั้นไมโครคอนโทรลเลอร์ จึงหมายถึง อุปกรณ์ควบคุมขนาดเล็ก แต่ในตัวอุปกรณ์ควบคุมขนาดเล็กนี้ได้บรรจุเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความสามารถที่คล้ายคลึงกับระบบคอมพิวเตอร์ กล่าวคือ ภายในไมโครคอนโทรลเลอร์ได้รวมเอาซีพียู หน่วยความจำและพอร์ต ซึ่งเป็นส่วนประกอบหลักสำคัญของระบบคอมพิวเตอร์เข้าไว้ด้วยกัน โดยบรรจุรวมกันอยู่ภายใต้ตัวถังเดียวกัน

ซีพียูจะติดต่อกับหน่วยความจำโปรแกรมเพื่ออ่านคำสั่งที่ระบุไว้ โดยต้องทำการอ้างตำแหน่งของหน่วยความจำผ่านสายสัญญาณที่เรียกว่า บัสแอดเดรส (address) แล้วทำการอ่านข้อมูลคำสั่งออกมาจากหน่วยความจำโปรแกรมในแอดเดรสนั้นๆ จากนั้นทำการประมวลผล โดยมีหน่วยความจำข้อมูลแรมเป็นที่พักของข้อมูลที่อยู่ในระหว่างการประมวลผลหรืออาจมองว่าหน่วยความจำข้อมูลแรมเป็นเสมือนกระดานหัดในการคำนวณก็ได้ ข้อมูลในการประมวลผลจะส่งผ่านสายสัญญาณที่เรียกว่าบัสข้อมูล (data bus) แล้วส่งต่อไปยังอุปกรณ์ภายนอกผ่านทางขาพอร์ตอินพุตเอาต์พุต

#### 2.4.1.1 หน่วยประมวลผลกลางหรือซีพียู (CPU : Central Processing Unit)

เป็นเสมือนมันสมองของไมโครคอนโทรลเลอร์ โดยซีพียูนี้จะทำหน้าที่ประมวลผลข้อมูลที่เข้ามาในระบบ แล้วทำการส่งต่อไปยังส่วนต่างๆ เพื่อควบคุมการทำงานต่อไป หัวใจหลักของซีพียูคือหน่วยคำนวณทางคณิตศาสตร์และลอจิก (ALU : Arithmetic and logic unit) ซึ่งได้รับการกำหนดจังหวะการทำงานจากส่วนควบคุมลำดับการทำงาน โดยจังหวะการทำงานนั้นจะสัมพันธ์กับสัญญาณนาฬิกา เมื่อซีพียูทำการติดต่อกับหน่วยความจำ สิ่งปรากฏขึ้นบนบัสข้อมูลภายในซีพียูคือ รหัสคำสั่ง (instruction code) ต้องผ่านการทำงานของส่วนถอดรหัสคำสั่ง (instruction decoder) เสียก่อน จะได้เป็นข้อมูลคำสั่งที่ซีพียูเข้าใจและสามารถดำเนินการต่อไปได้ หลังจากที่หน่วยคำนวณทางคณิตศาสตร์และลอจิกประมวลผลแล้วก็จะส่งข้อมูลมายังส่วนเชื่อมต่อรีจิสเตอร์ภายในซีพียู เพื่อติดต่อกับส่วนอื่นๆ ต่อไป

การทำงานของซีพียูมีด้วยกัน 2 จังหวะ คือ เฟตช์ (fetch) และเอ็กซีคิวต์ (executed) โดยการทำงานจะเริ่มจากการเฟตช์ ซึ่งก็คือการเรียกหรือการเข้าถึงคำสั่ง แล้วทำการถอดรหัสเป็นภาษาเครื่องเพื่อเตรียมประมวลผล จากนั้นจะเป็นจังหวะของการเอ็กซีคิวต์ ซึ่งก็คือการกระทำตามคำสั่งที่กำหนดให้จนเสร็จสิ้น

การที่จะระบุว่าไมโครคอนโทรลเลอร์มีขีดความสามารถในการประมวลผลเป็นอย่างไร จะพิจารณาที่ความสามารถในการประมวลผลข้อมูลของซีพียู หากซีพียูสามารถประมวลผลข้อมูลได้สูงสุด 8 บิต นั่นคือไมโครคอนโทรลเลอร์นี้เป็นแบบ 8 บิต (8-bit core) แต่ในซีพียูในไมโครคอนโทรลเลอร์สมัยใหม่บางตัวมีขนาด 8 บิต แต่สามารถประมวลผลกับข้อมูล 16 บิตได้ ทำให้ในบางครั้งผู้ผลิตจึงระบุออกมาว่าไมโครคอนโทรลเลอร์ตัวนี้ทำงานกับข้อมูล 16 บิต อาจกล่าวได้ว่า เป็นไมโครคอนโทรลเลอร์ 16 บิตเทียม เพราะถ้าหากเป็นแบบ 16 บิตแท้ ซีพียูต้องรองรับข้อมูลได้เต็ม 16 บิตหรือถ้าอ่านค่าในรายละเอียดของอุปกรณ์ของไมโครคอนโทรลเลอร์ตระกูลนั้นๆ จะต้องระบุว่าเป็น 16-bit core ดังนั้น จึงต้องพิจารณารายละเอียดตรงส่วนนี้ให้ดี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 2.4.1.2 หน่วยความจำ

ไมโครคอนโทรลเลอร์จะประกอบไปด้วยหน่วยความจำ 3 แบบ คือ หน่วยความจำโปรแกรม (program memory) หน่วยความจำข้อมูลแรม (RAM data memory) และหน่วยความจำข้อมูลอีอีพรอม (EEPROM data memory)

#### 1. หน่วยความจำโปรแกรม

หน่วยความจำโปรแกรมเป็นที่สำหรับเก็บข้อมูลคำสั่งของโปรแกรมควบคุมที่ผู้พัฒนาเขียนขึ้น หรือเรียกว่า โปรแกรมมอนิเตอร์ (monitor program) ซีพียูจะเข้ามาติดต่อกับเพื่ออ่านข้อมูลรหัสคำสั่งจาก หน่วยความจำในส่วนนี้แล้วนำไปประมวลผลเพื่อควบคุมการทำงานของระบบทั้งหมดต่อไปเรียกได้ว่ามีความสำคัญเท่าๆ กับซีพียูเลยทีเดียว หน่วยความจำโปรแกรมนี้นี้มีขนาดใหญ่และถ้ายังมีขนาดมากเท่าใด ก็จะสามารถบรรจุโปรแกรมที่มีความซับซ้อนหรือสามารถเก็บตารางข้อมูลที่ใช้ในการประมวลผลได้มากตาม โดยทั่วไปมีความจุไม่น้อยกว่า 512 ไบต์ แต่จะให้ดีควรมีความจุ 1 กิโลไบต์ขึ้นไป จึงจะช่วยให้การเขียน โปรแกรมควบคุมมีอิสระเพิ่มมากขึ้น ขนาดของหน่วยความจำโปรแกรมจะแปรตามความก้าวหน้าทาง เทคโนโลยี มีการพัฒนาให้ไมโครคอนโทรลเลอร์มีความจุของหน่วยความจำโปรแกรมสูงขึ้นเรื่อยๆ เป็น 4, 8, 16, 32 และ 64 กิโลไบต์และยังไม่สิ้นสุดเท่านี้ เชื่อแน่ว่าต้องมีการพัฒนาไมโครคอนโทรลเลอร์ให้มีความจุของ หน่วยความจำโปรแกรมสูงเป็นหลักร้อยกิโลไบต์หรือหลักหมื่นไบต์ในที่สุด

ชนิดของหน่วยความจำโปรแกรมที่ใช้ในไมโครคอนโทรลเลอร์ (นับถึงปี 2002) มีอยู่ 3 แบบ ที่นิยมกันคือ แบบอีพรอม (EPROM : Erasable Programmable Read-Only Memory) แบบอีอีพรอม (Electrically Erasable Programmable Read-Only Memory) และแบบแฟลช (flash) ความแตกต่างอยู่ที่จำนวนครั้งในการลบและเขียนข้อมูลทับลงไปใหม่ โดยสามารถสรุปได้ดังนี้

แบบอีพรอม ยังแบ่งเป็น 2 แบบ คือ แบบโปรแกรมได้หลายครั้งและแบบโปรแกรมได้ครั้งเดียว ถ้าหากเป็นแบบโปรแกรมได้หลายครั้งบนตัวถังของไมโครคอนโทรลเลอร์จะมีหน้าต่างกระจกติดอยู่สามารถมองเห็นชิปภายในได้ เวลาลบต้องลบด้วยแสงอัลตราไวโอเล็ต จำนวนรอบในการโปรแกรมใหม่อยู่ระหว่าง 10-100 ครั้ง แต่ถ้าเป็นแบบโปรแกรมได้ครั้งเดียวหรือ OTP (One-time programmable) จะไม่สามารถลบได้ ตัวถังของมันจะปิดมิดชิดเหมือนกับไอซีธรรมดา

แบบอีอีพรอม หน่วยความจำแบบนี้จะลบและเขียนใหม่ได้ด้วยสัญญาณไฟฟ้า ในอดีตเป็นที่นิยมมากเนื่องจากสามารถลบและเขียนใหม่ได้เป็นหลักร้อยรอบขึ้นไป ในบางตระกูลถึง 1 ล้านครั้ง แต่ในปัจจุบันแบบนี้ไม่เป็นที่นิยมใช้ในไมโครคอนโทรลเลอร์ เนื่องจากต้นทุนสูง

แบบแฟลช หน่วยความจำโปรแกรมชนิดนี้ สามารถลบและเขียนได้ด้วยสัญญาณไฟฟ้า แตกต่างกับแบบอีอีพรอมในเชิงการใช้งานตรงที่กระบวนการลบข้อมูล หน่วยความจำโปรแกรมแบบแฟลชจะไม่สามารถเลือกลบเฉพาะเจาะจงบางแอดเดรสบางตำแหน่งได้ เมื่อทำการลบข้อมูลจะต้องลบทั้งหมด หน่วยความจำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมแบบนี้ได้รับความนิยมมาก เนื่องจากราคาไม่สูงและสามารถโปรแกรมได้เป็นร้อยครั้งขึ้นไป แต่โดยปกติมักเริ่มที่ 1,000 ครั้งในบางรุ่นสูงเป็นหมื่นครั้งและเป็นแสนครั้งก็มี ขึ้นอยู่กับแรงดันที่ใช้ในการโปรแกรม

## 2. หน่วยความจำข้อมูลแรม

เป็นหน่วยความจำที่ต้องมีในไมโครคอนโทรลเลอร์ทุกตัว เพราะใช้เป็นพื้นที่สำหรับเก็บข้อมูลทั้งในระหว่างและหลังการประมวลผล ยิ่งมีมากยิ่งช่วยให้การทำงานสะดวก เพราะหน่วยความจำแรมมีอัตราเร็วในการอ่านเขียนสูงมากและไม่จำกัดจำนวนรอบในการอ่านเขียน ในพื้นที่ของหน่วยความจำข้อมูลแรมจะแบ่งออกเป็น 2 ส่วนคือ ส่วนของข้อมูลทั่วไปสำหรับเก็บค่าตัวแปรและส่วนของรีจิสเตอร์

โดยปกติแล้วหน่วยความจำข้อมูลแรมจะมีความจุไม่มากเมื่อเทียบกับหน่วยความจำโปรแกรมในบางตัวอยู่ในหลักสิบไบต์ แต่ถ้าไมโครคอนโทรลเลอร์มีความสามารถสูงขึ้นไป ความจุของหน่วยความจำข้อมูลแรมจะเพิ่มมากขึ้น ทั้งนี้เพราะต้องเพิ่มในส่วนของรีจิสเตอร์ตามความสามารถที่สูงขึ้นของไมโครคอนโทรลเลอร์

ทางด้านขนาดของหน่วยความจำข้อมูลแรม โดยส่วนใหญ่จะมีขนาด 8 บิต แต่สามารถต่อรวมกันเป็น 16 บิตได้ ส่วนการจัดสรรตำแหน่งแอดเดรสของหน่วยความจำข้อมูลจะขึ้นอยู่กับสถาปัตยกรรมของไมโครคอนโทรลเลอร์ หากเป็นแบบฮาร์วาร์ด (Harvard) จะได้รับการจัดสรรให้อยู่แยกจากหน่วยความจำโปรแกรม จึงทำให้มีค่าแอดเดรสเหมือนกันได้ นั่นคือผู้ใช้งานจะพบแอดเดรส 0000 ทั้งในหน่วยความจำโปรแกรมและหน่วยความจำข้อมูล แต่จริงๆ แล้วอยู่ต่างที่กัน จะพบในไมโครคอนโทรลเลอร์ MCS-51, PIC, AVR เป็นต้น แต่ถ้าเป็นแบบพริન્ซ์ตัน (Princeton) จะจัดสรรให้อยู่ในบริเวณเดียวกัน ดังนั้นค่าแอดเดรสจะไม่มีทางตรงกัน จะพบในไมโครคอนโทรลเลอร์ 68HCxxx ของ Motorola

## 3. หน่วยความจำข้อมูลอีอีพรอม

เป็นหน่วยความจำข้อมูลพิเศษที่ในไมโครคอนโทรลเลอร์บางเบอร์ บางรุ่น บางตระกูลไม่มีใช้สำหรับเก็บข้อมูลที่ต้องการรักษาไว้เมื่อไม่มีการจ่ายไฟเลี้ยงให้แก่ไมโครคอนโทรลเลอร์ การติดต่อเพื่ออ่านเขียนจะมีลักษณะเป็นพิเศษขึ้นอยู่กับไมโครคอนโทรลเลอร์แต่ละเบอร์ ขนาดของหน่วยความจำแบบนี้มักเท่ากับ 8 บิต ส่วนความจุก็จะแตกต่างกันไป มีตั้งแต่ไม่กี่สิบไบต์จนถึงเป็นกิโลไบต์

การอ่านเขียนหน่วยความจำแบบนี้จะใช้สัญญาณไฟฟ้าทั้งหมดและสามารถรักษาข้อมูลล่าสุดไว้แม้ว่าจะไม่มีการจ่ายไฟเลี้ยงให้แก่ไมโครคอนโทรลเลอร์แล้วก็ตาม สำหรับจำนวนรอบในการเขียนโดยปกติอยู่ในหลักล้านขึ้นไป

### 2.4.1.3 รีจิสเตอร์ (register)

เป็นหน่วยความจำพิเศษที่มีบทบาทสูงมากในการทำงานของไมโครคอนโทรลเลอร์ สามารถที่จะอ่านและเขียนข้อมูลได้ตลอดเวลา จนกว่าจะหยุดจ่ายไฟเลี้ยงให้แก่ไมโครคอนโทรลเลอร์ หน้าที่หลักคือ ใช้เก็บข้อมูลในการทำงานของไมโครคอนโทรลเลอร์ โดยข้อมูลที่เก็บนี้ทั้งข้อมูลแสดงสถานะการทำงาน ข้อมูลสำหรับควบคุมการทำงานโมดูลย่อยต่างๆ ภายในไมโครคอนโทรลเลอร์ ข้อมูลที่รับเข้ามาจากพอร์ติอินพุต

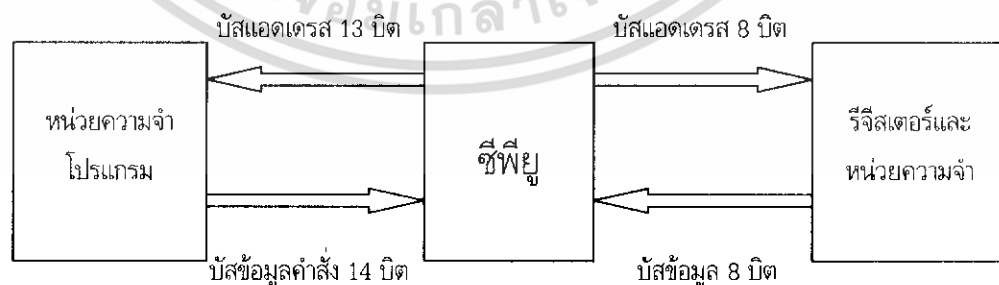
ข้อมูลที่ต้องการส่งออกไปยังอุปกรณ์ภายนอกผ่านพอร์ตเอาต์พุต โดยข้อมูลแต่ละประเภทก็จะถูกเก็บลงในรีจิสเตอร์ที่แตกต่างกันไปตามหน้าที่การทำงาน

#### 2.4.2 สถาปัตยกรรมและโครงสร้างทางฮาร์ดแวร์ของ PIC16F628

ไมโครคอนโทรลเลอร์ตระกูล PIC เป็นไมโครคอนโทรลเลอร์ที่มีสถาปัตยกรรมแบบฮาร์วาร์ด (Harvard architecture) กล่าวคือ มีการจัดแยกหน่วยความจำโปรแกรมและหน่วยความจำข้อมูลออกจากกัน มีบัสสำหรับติดต่อแยกกัน ดังแสดงในรูปที่ 2.6 จะเห็นได้ว่าซีพียูภายในไมโครคอนโทรลเลอร์จะติดต่อกับหน่วยความจำโปรแกรมด้วยบัสของแอดเดรส 13 บิต และบัสของข้อมูลหน่วยความจำโปรแกรม 14 บิต ในขณะที่บัสติดต่อหน่วยความจำข้อมูลและรีจิสเตอร์ภายในเป็นแบบ 8 บิต

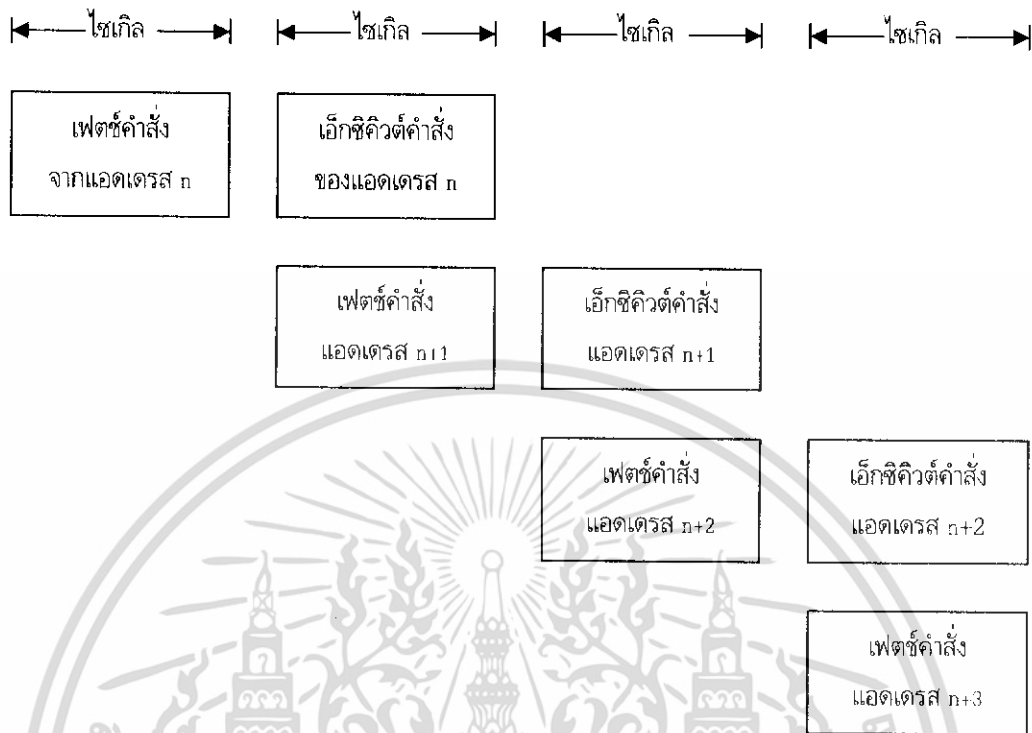
นอกจากการจัดสถาปัตยกรรมแบบนี้แล้ว การกระทำคำสั่งทำงานของไมโครคอนโทรลเลอร์ PIC ยังใช้กระบวนการที่เรียกว่า ไปป์ไลน์ (pipeline) ทำให้สามารถเฟตช์คำสั่งถัดไป ในขณะที่กำลังเอ็กซีคิวต์คำสั่งในปัจจุบัน ส่งผลให้ความเร็วในการทำงานของไมโครคอนโทรลเลอร์เพิ่มมากขึ้น นั่นจึงเป็นที่มาของความสามารถในการกระทำคำสั่ง 1 คำสั่งภายในสัญญาณนาฬิกา 1 ลูก (กระบวนการเฟตช์ (fetch) เป็นกระบวนการเรียกคำสั่งออกจากหน่วยความจำโปรแกรมแล้วทำการแปลคำสั่งนั้นให้เป็นรหัสเลขฐานสิบหกเพื่อให้ซีพียูเข้าใจ ส่วนกระบวนการเอ็กซีคิวต์ (execute) เป็นกระบวนการกระทำคำสั่งให้เกิดผลลัพธ์ตามที่คำสั่งนั้นๆ กำหนด) สำหรับกระบวนการไปป์ไลน์แสดงดังรูปที่ 2.7

เมื่อเริ่มต้นกระทำคำสั่งที่ 1 ซีพียูจะเฟตช์คำสั่งจากหน่วยความจำโปรแกรมที่แอดเดรส  $n$  จากนั้นทำการเอ็กซีคิวต์ในไซเคิลต่อมา และที่ไซเคิลของการเอ็กซีคิวต์คำสั่งที่แอดเดรส  $n$  นั้น ซีพียูก็จะเริ่มต้นเฟตช์คำสั่งจากแอดเดรส  $n+1$  ทันที เมื่อเอ็กซีคิวต์คำสั่งที่แอดเดรส  $n$  เรียบร้อย ซีพียูก็จะสามารถเอ็กซีคิวต์คำสั่งที่แอดเดรส  $n+1$  ต่อเนื่องกันไปได้ในทันที และในทำนองเดียวกัน ขณะที่กำลังเอ็กซีคิวต์คำสั่งแอดเดรส  $n+1$  ซีพียูก็จะดำเนินการเฟตช์คำสั่งที่แอดเดรส  $n+2$  ต่อไป



รูปที่ 2.7 แผนผังการทำงานพื้นฐานของไมโครคอนโทรลเลอร์ที่ใช้สถาปัตยกรรมแบบฮาร์วาร์ด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



**รูปที่ 2.8** แผนผังการทำงานของกระบวนการไปป์ไลน์ที่ใช้ในไมโครคอนโทรลเลอร์ PIC

**โครงสร้างของไมโครคอนโทรลเลอร์ PIC16F628**

แสดงดังในรูปที่ 2.8 ส่วนประกอบหลักก็จะเหมือนกับไมโครคอนโทรลเลอร์ PIC16F84 แต่จะมีส่วนเพิ่มเติมเข้ามาอีกพอควร ได้แก่ วงจรบราวเอาต์รีเซต (brown-out reset) สำหรับสร้างสัญญาณรีเซตที่พียู เมื่อไฟเลี้ยงลดต่ำลงเกินกว่าที่กำหนด วงจรโปรแกรมข้อมูลด้วยแรงดันต่ำ (low-voltage programming) ไทมเมอร์ที่มีมากถึง 3 ตัว โมดูลแรงดันเปรียบเทียบ (reference voltage module) โมดูลเปรียบเทียบแรงดันแอนะล็อก 2 ชุด (analog comparator) วงจรสื่อสารอนุกรม (USART : Universal Synchronous Asynchronous Receiver Transmitter) และโมดูลตรวจจับสัญญาณ เปรียบเทียบข้อมูลและวงจรสร้างสัญญาณมอดูเลชันทางความกว้างของพัลส์หรือ PWM (CCP : Capture Compare Pulse-width modulation module)

นอกจากนั้นขนาดของหน่วยความจำทั้งส่วนโปรแกรม ข้อมูล รีจิสเตอร์และหน่วยความจำอีพีรอม ในไมโครคอนโทรลเลอร์ PIC16F628 ก็มีเพิ่มมากขึ้น การจัดขาของ PIC16F628 แสดงในรูปที่ 2.9 ในตารางที่ 2.2 เป็นรายละเอียดของการทำงานในแต่ละขาสัญญาณของไมโครคอนโทรลเลอร์ PIC16F62x ส่วนในตารางที่ 2.3 แสดงการเปรียบเทียบคุณสมบัติของไมโครคอนโทรลเลอร์ในอนุกรม PIC16F62x ทั้ง 2 เบอร์ PIC16F627 กับ PIC16F84 ทั้งนี้เพื่อใช้เป็นข้อมูลในการเลือกใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



RA2/AN2/V <sub>REF</sub>	1	18	RA1/AN1
RA3/AN3/CMP1	2	17	RA0/AN0
RA4/T0CK1/CMP2	3	16	RA7/OSC1/CLKIN
RA5/MCLR/V <sub>pp</sub>	4	15	RA6/OSC2/CLKOUT
VSS	5	14	V <sub>dd</sub>
RB0/INT	6	13	RB7/T10SI
RB1/RXD/DT	7	12	RB6/T10SO/T1CK1
RB2/TXD/CK	8	11	RB5
RB3/CCP1	9	10	RB4/LVP

PDIP - 18

### รูปที่ 2.10 การจัดขาของไมโครคอนโทรลเลอร์ PIC16F628

คุณสมบัติทางเทคนิคของ PIC16F62x

1. ซีพียูเป็นแบบ RISC (Reduce Instruction-Set Computer) มีคำสั่งใช้งานเพียง 35 คำสั่ง
2. ความถี่สัญญาณนาฬิกา ตั้งแต่ไฟตรงถึง 20 MHz (สูงสุด)
3. ขนาดของหน่วยความจำโปรแกรม 1 กิโลเวิร์ด สำหรับ PIC16F627 และ 2 กิโลเวิร์ด สำหรับ

PIC16F628

4. หน่วยความจำแรมข้อมูล 224 ไบต์
5. หน่วยความจำข้อมูลอี้อัปรอม 128 ไบต์
6. ตอบสนองแหล่งกำเนิดอินเทอร์รัปต์ได้ 10 แหล่ง
7. มีสแต็ก 8 ระดับ
8. มีวงจรเพาเวอร์อนรีเซต (POR) เพาเวอร์อัปไทมเมอร์ (PWRT) และออสซิลเลเตอร์สตาร์ทอัปไทมเมอร์ (OST)
9. มีวอตช์ด็อกไทมเมอร์ (WDT) ที่มีวงจรออสซิลเลเตอร์ในตัว ทำให้มีความน่าเชื่อถือในการทำงานสูง
10. เลือกป้องกันข้อมูลทั้งในหน่วยความจำโปรแกรมและหน่วยความจำข้อมูลและสามารถเลือกระดับการป้องกันได้
11. เลือกใช้วงจรกำเนิดสัญญาณนาฬิกาได้ 6 โหมดหลัก
  - 11.1 โหมด EC ใช้สัญญาณนาฬิกาจากภายนอก
  - 11.2 โหมด ER ใช้ตัวต้านทานภายนอก
  - 11.3 โหมด INTRC ใช้วงจร RC ภายในไมโครคอนโทรลเลอร์ มี 2 ความถี่ให้เลือก
  - 11.4 โหมด LP ใช้คริสตอลพลังงานต่ำ ความถี่สูงสุดไม่เกิน 200 kHz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

11.5 โหมด XT ใช้คริสตอล ความถี่ตั้งแต่ 100 kHz สูงสุดไม่เกิน 4 MHz

11.6 โหมด HS ใช้คริสตอลความถี่สูง สูงสุดไม่เกิน 20 MHz (ต้องใช้กับรุ่นที่รองรับความถี่ 20 MHz ด้วย)

12. สามารถโปรแกรมโดยใช้แรงดัน +5V ได้

13. สามารถโปรแกรมในวงจรได้

14. ไฟเลี้ยง +3V ถึง +5.5V

15. กระแสซิงก์และซอร์สของพอร์ต 25 mA

16. ขาพอร์ตปกติ 15 บิต สูงสุด 16 บิต เมื่อทำงานในโหมด INTRC และกำหนดให้ MCLR เป็นพอร์ตอินพุต

17. ไทมเมอร์ 3 ตัว (ไทมเมอร์ 0, ไทมเมอร์ 1 และไทมเมอร์ 2)

18. มีโมดูล CCP (Capture/Compare/PWM) 1 ชุด

19. มีโมดูลเปรียบเทียบแรงดันแอนะล็อก 2 ชุด

20. มีโมดูลสร้างแรงดันอ้างอิง

21. มีโมดูลสื่อสารข้อมูลอนุกรม USART

22. มีวงจรตรวจจักระดับแรงดันไฟเลี้ยงหรือบราวเอาต์ดีเทกชัน (Brown-out detection) เพื่อสร้างสัญญาณรีเซ็ตซีพียูหรือเรียกว่า บราวเอาต์รีเซ็ต (Brown-out reset : BOR)

23. การใช้พลังงานไฟฟ้าในกรณีไม่ขับโหลด

**ตารางที่ 2.2** รายละเอียดของขาต่อใช้งานของไมโครคอนโทรลเลอร์ PIC16F628 โดยแบ่งออกเป็น 2 ตารางย่อยแยกได้ 3 กลุ่มคือ ขาต่อไฟเลี้ยง ขาพอร์ต A ที่มีทั้งหมด 8 ขา และขาพอร์ต B ที่มีทั้งหมด 8 ขา

ชื่อขา	ตำแหน่งขา	ชนิดของขา	ชนิดของวงจรบัฟเฟอร์	รายละเอียดการทำงาน
VDD	14	อินพุต	-	- ขาต่อไฟเลี้ยง ตั้งแต่ +3V ถึง +5.5V
VSS	5	อินพุต	-	- ขาต่อกราวด์
ขาพอร์ต A เป็นขาพอร์ต 2 ทิศทาง				
RA0/AN0	17	อินพุต/เอาต์พุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ต RA0 - อินพุตวงจรเปรียบเทียบแรงดันแอนะล็อก ช่อง 0
RA1/AN1	18	อินพุต/เอาต์พุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ต RA1 - อินพุตวงจรเปรียบเทียบแรงดันแอนะล็อก ช่อง 1
RA2/AN2/ V <sub>ref</sub>	1	อินพุต/เอาต์พุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ต RA2 - อินพุตวงจรเปรียบเทียบแรงดันแอนะล็อก ช่อง 2 - เอาต์พุตแรงดันอ้างอิง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษานั้น ไม่อนุญาตให้เผยแพร่โดยไม่ขออนุญาต

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

**ตารางที่ 2.2 (ต่อ)** รายละเอียดของขาต่อใช้งานของไมโครคอนโทรลเลอร์ PIC16F628 โดยแบ่งออกเป็น 2 ตารางย่อยแยกได้ 3 กลุ่มคือ ขาต่อไฟเลี้ยง ขาพอร์ต A ที่มีทั้งหมด 8 ขา และขาพอร์ต B ที่มีทั้งหมด 8 ขา

ชื่อขา	ตำแหน่งขา	ชนิดของขา	ชนิดของวงจรมัลติไพลีเออร์	รายละเอียดการทำงาน
RA3/AN3/ CMP1	2	อินพุต/เอาต์พุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ต RA3 - อินพุตวงจรเปรียบเทียบแรงดันแอนะล็อก ช่อง 3 - เอาต์พุตวงจรเปรียบเทียบแอนะล็อกชุดที่ 1
RA4/TOCKI/ CMP2	3	อินพุต/เอาต์พุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ต RA4 กรณีใช้งานเป็นพอร์ตเอาต์พุตจะมีโครงสร้างเป็นแบบเดรนเปิด - อินพุตสัญญาณนาฬิกาของไทมเมอร์ 0 - เอาต์พุตวงจรเปรียบเทียบแอนะล็อกชุดที่ 2
RA5/MCLR/ THV	4	อินพุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ตอินพุต RA5 - ขาร์ีเฟิร์สหลัก - อินพุตรับแรงดันสูงสำหรับโปรแกรม
RA6/OSC2/CLKOUT	15	อินพุต/เอาต์พุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ต RA6 เมื่อทำงานในโหมด INTRC - เอาต์พุตสัญญาณนาฬิกาหลัก เมื่อทำงานในโหมด ER มีความถี่เท่ากับ ¼ ของความถี่ที่ขา OSC1 - ขาต่อคริสตัล เมื่อทำงานในโหมด LP, XT และ HS
RA7/OSC1/ CLKIN	16	อินพุต/เอาต์พุต	ซิมิตต์ทริกเกอร์	- ขาพอร์ต RA7 เมื่อทำงานในโหมด INTRC - อินพุตสัญญาณนาฬิกาหลักเมื่อทำงานในโหมด EC - ต่อตัวต้านทานเพื่อกำหนดความถี่ในโหมด ER - ขาต่อคริสตัล เมื่อทำงานในโหมด LP, XT และ HS
ขาพอร์ต B เริ่มขาพอร์ต 2 ที่คทาง สามารถกำหนดให้ต่อตัวต้านทานเพูล์อัปภายในเมื่อทำงานเป็นอินพุตได้ทางซอฟต์แวร์				
RB0/INT <sup>1</sup>	6	อินพุต/เอาต์พุต	พืทีแอล/ ซิมิตต์ทริกเกอร์ <sup>1)</sup>	- ขาพอร์ต RB0 - อินพุตรับสัญญาณอินเตอร์รัปต์จากภายนอก
RB1/RxD/DT <sup>2</sup>	7	อินพุต/เอาต์พุต	พืทีแอล/ ซิมิตต์ทริกเกอร์ <sup>2)</sup>	- ขาพอร์ต RB1 - ขารับข้อมูลของวงจรถ่ายสารอนุกรม USART - ขาติดต่อสัญญาณข้อมูลซิงโครนัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.2 (ต่อ) รายละเอียดของขาต่อใช้งานของไมโครคอนโทรลเลอร์ PIC16F628 โดยแบ่งออกเป็น 2 ตารางย่อยแยกได้ 3 กลุ่มคือ ขาต่อไฟเลี้ยง ขาพอร์ต A ที่มีทั้งหมด 8 ขา และขาพอร์ต B ที่มีทั้งหมด 8 ขา

ชื่อขา	ตำแหน่งขา	ชนิดของขา	ชนิดของวงจรมัลติพอร์	รายละเอียดการทำงาน
RB2/TxD/CK	8	อินพุต/เอาต์พุต	ทีทีแอล/ ชมิตต์ทริกเกอร์ <sup>(3)</sup>	- ขาพอร์ต RB2 - ขาส่งข้อมูลของวงจรถ่ายข้อความ USART - ขาดัดต่อสัญญาณข้อมูลเชิงไดนามิก
RB3/CCP1	9	อินพุต/เอาต์พุต	ทีทีแอล/ ชมิตต์ทริกเกอร์ <sup>(3)</sup>	- ขาพอร์ต RB3 - ขาอินพุต/เอาต์พุตของโมดูล CCP
RB4/PGM	10	อินพุต/เอาต์พุต	ทีทีแอล/ ชมิตต์ทริกเกอร์ <sup>(3)</sup>	- ขาพอร์ต RB4 - สามารถกำเนิดสัญญาณอินเทอร์รัปต์อันเนื่องจากการเปลี่ยนแปลงลอจิกขึ้นที่ขาในกรณี enable ได้
RB5	11	อินพุต/เอาต์พุต	ทีทีแอล	- ขาพอร์ต RB5 - สามารถกำเนิดสัญญาณอินเทอร์รัปต์อันเนื่องจากการเปลี่ยนแปลงลอจิกขึ้นที่ขาในกรณี enable ได้ - ป้อนสัญญาณกระตุ้นให้ไมโครคอนโทรลเลอร์ทำงาน กรณีอยู่ในโหมดประหยัดพลังงาน (sleep)
AB6/T1OSO/TICK1/PGC	12	อินพุต/เอาต์พุต	ทีทีแอล/ ชมิตต์ทริกเกอร์ <sup>(2)</sup>	ขาพอร์ต RB6 - ขาเอาต์พุตสัญญาณนาฬิกาของไมโคร 1 - อินพุตรับสัญญาณนาฬิกาสำหรับไมโคร 1 - ขาสัญญาณนาฬิกาของการโปรแกรม - ป้อนสัญญาณกระตุ้นให้ไมโครคอนโทรลเลอร์ทำงาน กรณีอยู่ในโหมดประหยัดพลังงาน (sleep)
RB7/T1OSI/PGD	13	อินพุต/เอาต์พุต	ทีทีแอล/ ชมิตต์ทริกเกอร์ <sup>(2)</sup>	- ขาพอร์ต RB7 - สามารถกำเนิดสัญญาณอินเทอร์รัปต์อันเนื่องจากการเปลี่ยนแปลงลอจิกขึ้นที่ขาในกรณี enable ได้ - ขาอินพุตสัญญาณนาฬิกาของไมโคร 1 - ขาสัญญาณข้อมูลของการโปรแกรม - ป้อนสัญญาณกระตุ้นให้ไมโครคอนโทรลเลอร์ทำงาน กรณีอยู่ในโหมดประหยัดพลังงาน (sleep)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หมายเหตุ

1. อินพุตของวงจรรีเซ็ตเฟิร์จะเป็นแบบซิมิตต์ทริกเกอร์ เมื่อใช้งานเป็นขาอินพุตรับสัญญาณอินเทอร์รัปต์จากภายนอก
2. อินพุตของวงจรรีเซ็ตเฟิร์จะเป็นแบบซิมิตต์ทริกเกอร์ เมื่อทำงานในโหมดโปรแกรมข้อมูลอนุกรม (Serial programming mode)
3. อินพุตเอาต์พุตของวงจรรีเซ็ตเฟิร์จะเป็นแบบซิมิตต์ทริกเกอร์ เมื่อกำหนดให้ทำงานในโหมดสื่อสารข้อมูลอนุกรม
4. อินพุตเอาต์พุตของวงจรรีเซ็ตเฟิร์จะเป็นแบบซิมิตต์ทริกเกอร์ เมื่อกำหนดให้ทำงานในโหมด CCP
5. อินพุตของวงจรรีเซ็ตเฟิร์จะเป็นแบบซิมิตต์ทริกเกอร์ เมื่อทำงานในโหมดโปรแกรมด้วยแรงดันต่ำ (low voltage programming)

ตารางที่ 2.3 เปรียบเทียบคุณสมบัติของไมโครคอนโทรลเลอร์ PIC16F84 กับ PIC16F628

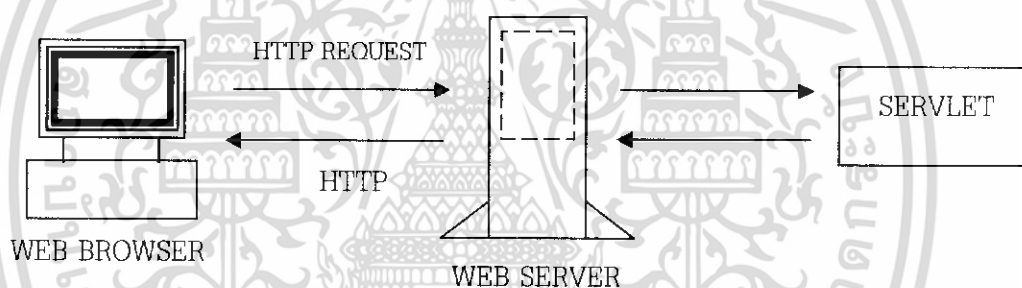
คุณสมบัติสำคัญ	PIC16F84A	PIC16F627	PIC16F628
ความถี่สัญญาณนาฬิกา	ไฟตรง -20MHz	ไฟตรง -20MHz	ไฟตรง -20MHz
จำนวนขา	18	18	18
จำนวนพอร์ตอินพุต/เอาต์พุต	พอร์ต A,B 13 บิต	พอร์ต A,B 16 บิต	พอร์ต A,B 16 บิต
ส่วนรีเซ็ต (และหน่วงเวลา)	POR,BOR	POR,BOR	POR,BOR
หน่วยความจำโปรแกรม	1K × 14 บิต	1K × 14 บิต	2K × 14 บิต
หน่วยความจำข้อมูลแรม	68 ไบต์	224 ไบต์	224 ไบต์
หน่วยความจำข้อมูลอีพีรอม	64 ไบต์	128 ไบต์	128 ไบต์
จำนวนแหล่งกำเนิดอินเทอร์รัปต์	4	10	10
จำนวนไทเมอร์/เคาน์เตอร์	1	3	3
โมดูลแคปเจอร์/เปรียบเทียบ/PWM	-	1	1
ส่วนสื่อสารข้อมูลอนุกรม	-	USART	USART
โมดูลเปรียบเทียบแรงดันอนาล็อก	-	2 ชุด	2 ชุด
โมดูลสร้างแรงดันอ้างอิง	-	1 ช่อง	1 ช่อง
โหมดสัญญาณนาฬิกา	RC,LP,XT,HS	EC,ER,INTRC,LP,XT,HS	EC,ER,INTRC,LP,XT,HS
จำนวนคำสั่ง	35	35	35

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## 2.5 Java Servlet

Java Servlet คือโปรแกรมภาษาจาวาที่ถูกสร้างมาให้ทำงานได้บน HTTP Server โดยการทำงานของ Servlet มีลักษณะเป็น Server-side ซึ่งต่างจาก Java Applet โดยทั่วไปที่ทำงานในลักษณะ Client-side โดย Servlet เป็น modules ที่ทำงานอยู่ภายในกระบวนการ request/response oriented server และ Servlet ยังสามารถนำมาประยุกต์ใช้ในการพัฒนาฐานข้อมูล (database) ต่างๆ ได้อีกด้วย

ประโยชน์ของ Servlet นั้นช่วยให้เราเขียน Program ที่สั่งให้ HTTP Server ทำงานหรือโต้ตอบกับ Program ของ Client เหมือนที่ CGI (Common Gateway Interface) ทำได้ ซึ่ง Servlet ถูกนำมาใช้แทน CGI script ได้อย่างมีประสิทธิภาพและให้ความสะดวกได้อย่างมากคือสามารถเขียนได้ง่ายกว่าและทำการ RUN ได้เร็วกว่า CGI และ Servlet ยังได้มีการจัดการในส่วนของโปรแกรมทางด้าน server-side ไว้ โดยเฉพาะ Servlet ได้ถูกพัฒนาโดยใช้ Java Servlet API ซึ่งเป็นส่วนเพิ่มเติมมาตรฐานของ Java



รูปที่ 2.11 แผนผังการทำงานของ Servlet

### 2.5.1 จุดเด่นของ Servlet

1. Servlet สามารถที่จะฝังตัวทำงานอยู่บน Server เมื่อทำการให้บริการ Client เสร็จสิ้นแล้วจะคอยรับ Request ใหม่ที่อาจจะมาจาก Client อีกโดยไม่ต้องถูกสร้างขึ้นใหม่จึงทำให้ Servlet ใช้ทรัพยากรของ Server น้อยมาก

2. Servlet สามารถจัดการกับงานประเภท Multiple Connection ได้ดี โดยอาศัยความสามารถของ Thread ที่มีอยู่ในตัว Java

3. Servlet สามารถติดต่อกับ Applet บน Client ได้อย่างต่อเนื่อง ซึ่งเป็นผลให้การรับส่งข้อมูลระหว่างกันดำเนินไปอย่างต่อเนื่อง

4. Servlet สามารถถูก Upload จาก Client เพื่อสั่งให้ไปทำงานบน server ใดๆ บน Network ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประมวลผลที่เครื่องต้นทาง สามารถทำได้ที่ตัว Host นั้นๆ เลย เสร็จแล้วค่อยส่งผลรับกลับคืน จึงสามารถลดเวลาและ traffic บนระบบ Network ลงอย่างมาก

### 2.5.2 สถาปัตยกรรมของ Servlet

ใจความสำคัญของ Servlet API คือ การสนับสนุนการ interface ของ Servlet ทั้งแบบทางตรงหรือทางอ้อม โดยประกอบไปด้วยคลาสที่สนับสนุนการ interface เช่น Http Servlet ซึ่ง Servlet interface ประกอบด้วย method ที่ใช้เป็นตัวจัดการ Servlet และติดต่อสื่อสารกับ Client ผู้เขียน Servlet จะกำหนด method นี้ทั้งหมดหรือบางส่วนเมื่อต้องการพัฒนา Servlet เมื่อ Servlet ได้รับการเรียกร้องจาก Client จะมีการรับ object อยู่ 2 object ซึ่งอันแรกคือ Servlet Request และอีกอันคือ Servlet Response โดยคลาส Servlet Request นั้นจะเป็นตัวติดต่อสื่อสารจาก Client ไปยัง Server ในขณะที่คลาส Servlet Response เป็นตัวติดต่อสื่อสารจาก Servlet กลับไปยัง Client

Servlet Request interface อนุญาตให้ Servlet สามารถเข้าถึงข้อมูลได้ เช่น ชื่อของพารามิเตอร์ที่ผ่านเข้ามาโดย Client โพรโตคอลและชื่อ host ของ Client ที่เป็นตัวส่ง request และของ Server ที่เป็นตัวรับ request ด้วยรวมไปถึงการกำหนดให้ Servlet เข้าถึง input stream ได้ด้วยโดย Servlet Input Stream โดย Servlet ทำหน้าที่รับข้อมูลจาก Client โดยผ่านโปรโตคอล เช่น HTTP, POST และ PUT method โดยคลาสย่อยของคลาส Servlet Request จะอนุญาตให้ Servlet สามารถรับข้อมูลที่เป็นลักษณะเฉพาะของโปรโตคอลกลับมาได้

Servlet Response interface ประกอบไปด้วย method สำหรับการตอบกลับไปยัง Client โดยอนุญาตให้ Servlet กำหนดขนาดและ mime type ในการตอบกลับและเป็นตัวจัดเตรียม output stream ด้วย โดยผู้เขียน Servlet สามารถใช้ Servlet Output Stream ในการส่งข้อมูลกลับได้ sub-class ของ Servlet Response ทำให้ Servlet มีโปรโตคอลที่แตกต่างกันได้หลายชนิด ตัวอย่างเช่น Http Servlet Response ประกอบด้วย method ที่อนุญาตให้ Servlet สามารถถ่ายทอดข้อความจาก HTTP-specific header ได้

### 2.5.3 การใช้ Java Servlets กับการพัฒนาระบบงานบนเครือข่ายอินเทอร์เน็ต

Servlet API สามารถใช้ได้บน Server พื้นฐานทั่วไปที่มีการสนับสนุน Servlet และถูกกำหนดโดยคลาสที่อยู่ในแพ็คเกจดังนี้ java.servlet, java.servlet.http และ java.servlet.html

Package : java.servlet

Java.servlet เป็นแพ็คเกจที่กำหนดการทำงานของ Servlet จากสัดส่วนของ protocol independent ในระดับนี้โปรโตคอลที่ถูกใช้ในการเข้าถึง Servlet ได้แก่ HTTP, โปรโตคอลของระบบไฟล์เครือข่าย หรือโปรโตคอลที่ใช้ในการขนส่งอื่นๆ Servlet ถูกออกแบบให้ทำงานบน server-side โดยใช้คลาสในแพ็คเกจนี้ซึ่งประกอบไปด้วย interface class 4 คลาส และ implementation class 3 คลาส ดังนี้

1. Servlet Interface Class ประกอบไปด้วย method พื้นฐาน 4 method ดังนี้คือ

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์สำหรับการใช้งานเพื่อการศึกษาเท่านั้น เมื่อผู้ผู้จัดทำเห็นประโยชน์ในการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.1 `init(ServletStub)` ถูกเรียกโดยอัตโนมัติเมื่อระบบต้องการหาค่าเริ่มต้นของ Servlet โดยกำหนดตัวแปรเริ่มต้นของ Servlet ในไฟล์ `servlet.properties` ได้โดยการเรียก `getIniParameter()` หรือ `getInitParameters()`

1.2 `destroy()` ถูกเรียกโดยระบบเพื่อคืนทรัพยากรให้ระบบเมื่อ Servlet ถูกยกเลิกการใช้งาน

1.3 `service(Servlet Request, Servlet Response)` ถูกเรียกเพื่อนำมาใช้ในกระบวนการรับ request ที่เข้ามาและสร้าง response ตอบกลับ

1.4 `getServletInfo()` ถูกเรียกเพื่อรับข้อมูล (information) ที่เป็น String ซึ่งบอกลักษณะของ Servlet

2. Servlet Context Interface Class เป็นการกำหนด method เพื่อให้ Servlet สามารถทำงานได้แบบสลับจาก (interact) ได้โดย method ต่างๆ มีดังนี้

2.1 `getMimeType(String filename)` ส่งคืนค่า MIME type ของแต่ละไฟล์โดยใส่ชื่อของไฟล์ (filename) ลงใน `mime.properties` file

2.2 `getRealPath (String path)` การประยุกต์ใช้ alias rules ใน `alias.properties` file เพื่อเป็นการกำหนด virtual path และตอบสนองด้วยการส่ง real path หากการส่งข้อมูลเกิดการผิดพลาดค่าที่ส่งกลับคืน null

2.3 `getServletInfo()` ส่งคืนค่าที่เป็นแบบ String ที่บ่งบอกชื่อ, version และฐานของ server software

2.4 `getServlet(String)` ส่งคืน Servlet ที่ตรงกับชื่อที่เรียกมา

2.5 `getServlets()` ส่งคืน Servlets ทั้งหมดที่ถูกโหลดใน server โดยรวมไปถึง Servlet ที่ถูกเรียกด้วย

2.6 `log(Servlet, String)` เขียนข้อความลงใน Servlet log ของ Servlet ที่ถูกกำหนด

3. Servlet Request Interface Class ประกอบไปด้วย method ที่ใช้สำหรับการค้นหาข้อมูลที่เกี่ยวข้องกับ Servlet request ซึ่งเป็นอิสระจากโปรโตคอลที่ใช้อยู่จริงในการส่งผ่านข้อมูลระหว่าง Client และ server โดย request จะถูกนำพาโดย IP พื้นฐานของโปรโตคอลเช่น HTTP ซึ่งการส่งผ่านข้อมูลของ ServletRequest โดยอาศัยโปรโตคอล เช่น HTTP นั้นจะต้องมี method ที่มีความสัมพันธ์กับโปรโตคอลดังนี้

3.1 `getRemoteAddr()` ส่งคืนค่า IP address ของ Client ที่ส่ง request

3.2 `getRemoteHost()` ส่งคืนค่า Host name ของตัวแทนที่ส่ง request

3.3 `getServerName()` ส่งคืนค่า Host name ของ server ที่รับ request

3.4 `getServerPort()` ส่งคืนค่า port number ที่รับ request

3.5 `getProtocol()` ส่งคืนค่าที่เป็น String ซึ่งแสดง protocol และ version

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

request ประกอบไปด้วยพารามิเตอร์และข้อมูลต่างๆ โดย Servlet สามารถหาขนาดความยาวของ content ได้โดย `getContentLength()` และหาชนิดของ content ได้จาก `getContentType()` เมื่อหา content พบ Servlet สามารถอ่านได้โดยใช้ `HttpInputStream` ซึ่งได้จาก `getInputStream()` โดย method ที่มีความสัมพันธ์กับ content มีดังนี้

3.6 `getParameter(String)` ส่งคืนพารามิเตอร์ที่กำหนดมาสำหรับ request

3.7 `getParameters()` ส่งคืนพารามิเตอร์ทั้งหมดของ request

3.8 `getContentType()` ส่งคืน MIME type ของข้อมูลที่รวมมากับ request หรือค่า null ถ้าไม่มีการกำหนด MIME type มากับ request

3.9 `getInputStream()` ส่งคืน input stream เพื่ออ่านข้อมูลที่มากับ request

4. Servlet Response Interface class เป็นตัวกำหนดการสร้าง response สำหรับการตอบสนอง request โดยทั่วไปแล้ว response จะประกอบไปด้วยส่วนที่บอกลักษณะของ MIME และขนาดของ content โดยการเขียน content ใน response นั้น Servlet จะเรียก `getOutputStream()` เพื่อทำหน้าที่รับ `ServletOutputStream` จากนั้นเรียก `print()` เพื่อส่ง response โดย method ที่ใช้ในการสร้าง response มีดังนี้

4.1 `setContentType(String)` เซ็ต MIME type ของ response

4.2 `setContentLength(int)` เซ็ตขนาดความยาวของ response

4.3 `getOutputStream()` ส่งคืน `ServletOutputStream` เพื่อทำการเขียนข้อมูล response ให้กลับไปยัง client

5. `GenericServlet` Class เป็นคลาสพื้นฐานที่สนับสนุนการส่งผ่านข้อมูลของ Servlet และใช้สำหรับการปลุก Servlet จาก server-side include โดยประกอบด้วย method ดังนี้

5.1 `init()` ถูกเรียกหลังจาก Servlet ได้รับการโหลดให้เป็นขั้นเริ่มต้น โดยจะข้าม method นี้ไปได้เมื่อมีการรับพารามิเตอร์เริ่มต้นต่างๆ หรือมีการทำงานที่ฟังก์ชันอื่นๆ ซึ่งในแต่ละครั้งเมื่อ Servlet ถูกโหลด Servlet จะมีค่าอยู่ที่ค่าเริ่มต้นหนึ่งครั้งเท่านั้น

5.2 `getInitParameter(String name)` เป็นการรับค่าพารามิเตอร์เริ่มต้นของตัว Servlet ทั้งแบบ `servlet.properties` file และอยู่ในรูปของ `<Servlet>` tag สำหรับ server-side includes servlets

5.3 `getInitParameters()` ส่งคืนชื่อและค่าพารามิเตอร์เริ่มต้นของ FOCUS01>

5.4 `getServletContext()` ส่งคืน Servlet context.

5.5 `Log(String)` log ข้อความลงใน servlet log file

6. `ServletInputStream` เป็น subclass ของ `InputStream` ที่เป็นตัวกำหนด `readLine(byte[], int offset, int count)` ซึ่งทำการอ่าน byte ลงใน byte array จนกระทั่ง array เต็มหรือมีการอ่านบรรทัดใหม่เข้ามา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7. ServletOutputStream Abstract Class เป็น subclass ของ OutputStream ซึ่งคล้ายกับ PrintStream แต่เป็นเพียงการกำหนด method ที่ใช้ print ข้อมูล โดยทั่วไปชนิดของข้อมูลจะใช้ ints, longs และ strings การทำงานของ ServletOutputStream จะทำการเปลี่ยน code ที่ใช้ในโปรแกรม (internal) ให้เป็น US-ASCII ซึ่งเป็น code ที่ใช้กันทั่วไปในระบบ Internet protocol โดยประกอบด้วย method ดังนี้

7.1 print(int), println(int)

7.2 print(long), println(long)

7.3 print(String), println(String)

7.4 println()

โดยสามารถแปลงข้อมูลชนิดอื่นก่อนการส่ง print ออกทาง ServletOutputStream ได้โดยใช้ toString()

Package : java.servlet.http

Java.servlet.http เป็นแพ็คเกจที่สนับสนุน Servlet ในการใช้ HTTP โดยเฉพาะ แพ็คเกจนี้ ประกอบไปด้วย interface class 2 คลาส implementation class 2 คลาส และคลาสสนับสนุนที่เป็น subclass ดังนี้

1. HttpServlet Abstract Class เป็นเครื่องมือที่ใช้ในการค้นหาและแปลงข้อมูลระหว่างตัว ServletRequest และ ServletResponse โดยประกอบด้วย method ดังนี้

1.1 service (ServletRequest, ServletResponse) ทำหน้าที่ค้นหาและแปลงข้อมูลทั่วไป ระหว่าง HTTP-specific request กับ response โดยทั่วไปแล้วจะไม่มีการเขียนทับ method นี้

1.2 service (HttpServletRequest, HttpServletResponse) เขียนทับ abstract method เพื่อรับ HTTP service request

2. HttpServletRequest Interface Class เป็น subclass ของคำสั่ง ServletRequest ซึ่งเป็นตัวจัดการข้อมูลของ extra request และแสดงลักษณะของ content โดยแบ่งประเภทของข้อมูลที่ได้จาก HttpServletRequest เป็น 4 ลำดับ ดังนี้

2.1 General Request Information ประกอบด้วย method ที่ให้ข้อมูลทั่วไปเกี่ยวกับ request รวมไปถึงชื่อของ method ที่ใช้ในการเข้าถึงข้อมูลและคำถามต่างๆ มีดังนี้คือ

2.1.1 getMethod() ส่งคืนค่า string ที่บรรจุ method ที่ใช้ในการ request เช่น "GET", "HEAD" หรือ "POST"

2.1.2 getQueryString() ส่งคืน query string ซึ่งเป็นส่วนหนึ่งของ URL หรือ ถ้าไม่มีจะส่งคืนค่า null โดยที่ query string จะตามหลังเครื่องหมาย ? ใน URL

2.2 URL Information ประกอบด้วย method ที่ให้ข้อมูลเกี่ยวกับการร้องขอ URL ดังนี้คือ เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.1 `getRequestURL()` ส่งคืน request URL ส่วนที่อ้างอิงถึงการปลุก Servlet

2.2.2 `getServletPath()` ส่งคืน `ServletPath` ส่วนที่อ้างอิงถึงการปลุก Servlet

2.2.3 `getPathInfo()` ส่งคืนข้อมูลของ extra path ที่ตามหลัง Servlet path ซึ่งนำหน้า query string

2.2.4 `getPathTranslated()` ส่งคืนข้อมูลของ extra path ที่จะส่งผ่านไปยังตัว real path โดยใช้ `alias.properties` rule

2.2.5 `getRequestPath()` ส่งคืน request URL ส่วนที่ตอบสนอง Servlet path ที่เพิ่มเติมด้วยข้อมูลของ extra path

2.3 Authentication-Related Information การเข้าถึงข้อมูลที่เกี่ยวข้องกับชื่อและที่อยู่ของ client นั้น Servlet สามารถทำได้โดยใช้ `getHostName()` และ `getHostAddress()` method แต่สำหรับชนิดของ HTTP level authentication สามารถแสดงได้โดยใช้ `getAuthType()` และการติดต่อในแบบ basic mode จะมีการส่งคืนชื่อผู้ใช้โดย `getRemoteUser()` method

2.3.1 `getAuthType()` ส่งคืนรูปแบบของการรับรอง (Authentication) ของ request ในกรณีที่หาไม่พบค่าที่ส่งคืนคือ null

2.3.2 `getRemoteUser()` ส่งคืนชื่อของผู้ใช้ที่สร้าง request หรือถ้าไม่มีชื่อมากับ request นั้นจะให้ค่าเป็น null

2.4 Access to HTTP Request Header Fields สำหรับ `HttpServletRequest` มีการเปิดช่องทางให้มีการเข้าถึง HTTP request header fields โดยตรงเพื่อรับข้อมูลที่ได้รับการสนับสนุนเพียงเล็กน้อยจาก `HttpServletRequest` method ดังอย่างเช่น `include` `user-agent`, `referer`, `preferred languages` และ `cookie` (สถานะของ client) ซึ่งสิ่งเหล่านี้มี header fields ที่ไม่แน่นอน

ต่อไปจะกล่าวถึง method 2 method ที่ใช้ในการเข้าถึง header เริ่มต้นเพื่อที่จะใส่ค่าดัชนีให้กับ header นั้นโดยค่าดัชนีที่เป็น 0 จะส่งคืน header field อันดับแรก และถ้าค่าดัชนีมีค่ามากกว่าจำนวนของ field ที่เป็นไปได้ที่ส่งคืนคือ null

2.4.1 `getHeader(int)` ส่งคืนจำนวนของ header field (n) ถ้ามีค่าน้อยกว่า header field (n) ค่าที่ส่งคืนคือ null

2.4.2 `getHeaderName(int)` ส่งคืนชื่อของจำนวน (n) ถ้ามีค่าน้อยกว่า (n) ค่าที่ส่งกลับคือ null

ในกรณีต่อไปเป็นการค้นหา request header ที่ตรงกันกับชื่อของ header field และส่งคืนค่าของ header field นั้นกลับไป โดยเลือกใช้ method ที่กำหนดให้ต่อไปนี้ซึ่งขึ้นอยู่กับรูปแบบประเภทของข้อมูลที่ต้องการจะได้กลับคืนมา

2.4.3 `getHeader(String fieldname)` ส่งคืนค่าของ header field ถ้าไม่มีการกำหนด method นี้ ค่าที่ส่งคืนคือ null

2.4.4 `getHeader(String fieldname, int default)` ส่งคืนค่าที่เป็น integer ซึ่งเป็นค่าของ header field หลังจากมีการเปลี่ยนแปลงข้อมูลที่เป็น string ให้เป็น int ถ้าไม่พบ field นี้ค่าที่ส่งคืนจะไม่ได้ถูกกำหนดไว้

2.4.5 `getDateHeader(String fieldname, long default)` ส่งคืนค่าวันเดือนปีของ header field หลังจากแปลงข้อมูลที่เป็น string ให้เป็น date ถ้าไม่พบ field นี้จะไม่มีการกำหนดค่าที่ส่งกลับ

3. `HttpServletResponse` Interface Class เป็นแบบ subclass ของ `ServletResponses` และเป็นเครื่องมือในการสร้าง HTTP response header โดยแบ่งตามการนำไปใช้ประโยชน์ได้ 4 แบบ

3.1 `Setting the Response Status` เป็นการเซตสถานะของ response โดยค่าที่ส่งคืนคือ 200 หรือ OK. ถ้ามี error ส่งคืนหรือ OK. อยู่ในตำแหน่งที่ไม่ถูกต้อง Servlet สามารถระบุแยกรายละเอียดของทั้ง error code และข้อความที่บอกถึงลักษณะของ response ได้เอง ถ้ามีเพียง error เกิดขึ้นเท่านั้น Servlet จะแปลง code ให้อยู่ในรูปข้อความที่เป็น string แทนซึ่ง method ต่างๆ มีดังนี้

3.1.1 `setStatus (status Code)` เซ็ต status code ในการส่งคืน HTTP header โดย status code ถูกแปลงให้เป็นข้อความที่เป็น string

3.1.2 `setStatus (statusCode, messageString)` เซ็ต status code ในการส่งคืน HTTP header

3.2 `Specifying Non-Standard Header Fields` เป็นการรวม field ต่างๆ กับ response header นั้น Servlet ได้กำหนดให้ใช้ `setHeader()` method แต่สำหรับ non-standard header field ซึ่งไม่ได้รับการสนับสนุนจาก `HttpServletResponse` ซึ่งกำหนดให้ชื่อของ header field (ไม่มีเครื่องหมาย ;) อยู่ที่ argument แรก และข้อมูลอยู่ที่ argument ที่ 2 `setHeader` ที่ต่างกันนั้นแยกด้วยชนิดของพารามิเตอร์ที่เป็น String, int และ date โดยประกอบด้วย method ดังนี้

3.2.1 `setHeader(String fieldname, String)` เซ็ตค่าของชื่อ header field เพื่อแสดงค่าออกมา

3.2.2 `setIntHeader(String fieldname, int)` เซ็ตค่าของชื่อ header field ที่เป็นแบบ integer หลังจากมีการแปลงค่าให้เป็น string

3.2.3 `setDateHeader(String fieldname, long)` เซ็ตค่าของชื่อ date field หลังจากแปลง long เป็น Internet-standard date-time string

3.2.4 `unsetHeader(String fieldname)` ย้ายชื่อของ header field ออกจาก response

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 Returning Error Response การสร้างและส่ง standard error response เพื่อส่งคืน error นั้นมี method ที่ทำหน้าที่นี้อยู่ภายใต้ HttpServletResponse คือ sendError() method ซึ่งประกอบด้วย status code และข้อความที่อธิบายรายละเอียดของ error ซึ่งการส่งค่าคืนนี้จะส่งไปบน page โดยประกอบด้วย header ซึ่งแสดงค่า status code และข้อความมาตรฐานของ status code ในขณะที่ข้อความที่อธิบายรายละเอียดของ error นั้นจะอยู่บน body ของ page ที่ส่งคืน ตัวอย่างเช่น เมื่อมีการเรียก sendError (404, "Couldn't find it!") จะมีการสร้าง page ซึ่งประกอบด้วย header "404 Not Found" และข้อความคือ "Couldn't find it!" โดย method ที่กล่าวถึงมีดังนี้

3.3.1 sendError(int) ส่ง error response ไปที่ client โดยระบุแต่ status code และไม่มีข้อความเพิ่มเติม

3.3.2 sendError(int, String) ส่ง error response ไปที่ client โดยต้องระบุทั้ง status code และข้อความที่อธิบายรายละเอียดของ error

3.4 Returning Redirect Response ในการสร้างและส่ง standard redirect response นั้น HttpServletResponse ได้กำหนดให้ใช้ sendRedirect() method โดยจะมีการสร้างและส่งค่า "302" เป็น response ค่าที่ส่งไปจะเป็น string ซึ่งเป็นตัวแสดงตำแหน่งใหม่ของ page โดย string ที่กล่าวถึงนี้ก็คือ URL นั้นเอง เนื่องจาก URL คือที่อยู่ที่ต้องการส่ง page ไปเหมือนกับการ link นั้นเอง อย่างไรก็ตาม sendRedirect() ไม่สามารถตรวจสอบได้ว่า string ที่เข้ามาั้นตรงกับ URL ที่ถูกต้องหรือไม่ ถ้าผู้เขียนมี URL ที่ถูกต้องแล้วให้แปลง URL นั้นเป็น string โดยใช้ toString() ก่อนที่จะเรียก sendRedirect() โดยการเรียก method นี้จะอยู่ในรูป sendRedirect(String) ซึ่งจะส่ง redirect response ไปยัง client โดยการระบุที่อยู่ของ URL ที่ต้องการมาด้วย

4. FormServlet Class เป็นคลาสของ Servlet ที่ถูกระบุให้เป็นส่วนหนึ่งของ Servlet API และด้วยเหตุนี้มันจึงสามารถอยู่บน web server ทุกชนิดที่สนับสนุน Servlet โดยทำหน้าที่สร้างแบบฟอร์มการทำงานของ Servlet โดย subclass ของ FormServlet และเขียนทับบน sendResponse() method ซึ่ง method นี้ทำหน้าที่ควบคุมแบบฟอร์มของข้อความและสร้าง response ซึ่งเป็น subclass ของ HttpServletResponse โดย method ต่อไปนี้ถูกกำหนดไว้ใน FormServlet Class

4.1 getServletInfo() เขียนทับตัว method นี้เพื่อส่งคืน string ที่บรรจุอยู่ในข้อความของ Servlet เช่น ชื่อผู้เขียนหรือเวอร์ชัน

4.2 service (HttpServletRequest, HttpServletResponse) ผู้เขียนสามารถเขียนทับตัว method นี้ เมื่อมีความต้องการ special request processing sendResponse (HttpServletResponse, Hashtable) เขียนทับ method นี้เมื่อต้องการแปลง code เพื่อจัดรูปแบบและสร้าง response 5 HttpUtils Class เป็นคลาสที่ทำหน้าที่กำหนด static utility routine ในการสร้าง Servlet

Package : java.servlet.html

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แพ็คเกจลำดับที่ 3 ซึ่งกำหนดใน Servlet API จะกล่าวถึงต่อไปก็คือ java.servlet.html โดยแพ็คเกจนี้ประกอบด้วยคลาสที่สนับสนุน HTML 13 คลาส ซึ่งสนับสนุนการปฏิบัติการของ HTML เช่น ตาราง การแทรกรูปภาพและฟอร์ม ผู้เขียนจึงสามารถสร้าง HTML page โดยใช้ภาษา HTML จากนั้นจึงส่ง HTML page นั้นเขียนลง ServletOutputStream โดยคลาสต่างๆ มีดังนี้

1. HtmlElement สนับสนุนการกำหนด basic interface ประกอบด้วย 2 method ดังนี้
  - 1.1 wrap() สำหรับรวม HTML tag pairs (เช่น <h1> และ </h1>) เข้ากับ element โดย tag ถูกระบุให้เป็น argument ใน wrap()
  - 1.2 write() เขียน HTML ไปที่ OutputStream
2. HtmlText สนับสนุนข้อความ (text) ที่อยู่ภายใน tag ประกอบด้วย 2 method ดังนี้
  - 2.1 add(String), add(String tag) เพิ่มข้อความ (text) ซึ่งรวมไปถึงข้อความที่อยู่ใน tag
  - 2.2 addTag(String), addTag(string,String) เพิ่ม tag เดียวที่ไม่เป็นคู่ (non-paired tag) เช่น <p>
3. HtmlContainer Class สนับสนุนในการทำงานของ list คือ HtmlElement ที่อยู่ภายใน tags ซึ่ง HtmlContainer ประกอบด้วย beginning tags, list ของ HtmlElement และ ending tags Htmlcontainer สนับสนุน wrap(), write() และชนิดต่างๆ ของ add() และ addtag() method โดย HtmlContainer ประกอบด้วย method ต่างๆ ดังนี้
  - 3.1 addImg(String) เพิ่ม <IMG> tag โดยที่ String คือ URL ที่บรรจุรูปภาพที่ต้องการ
  - 3.2 addLink(String text, String URL) เพิ่ม link เมื่อ "text" อยู่ระหว่าง tags และ URL โดยเป็น URL ที่ต้องการจะ Link ไป
4. Htmlapplet Class สนับสนุน applet HTML tag กับรายละเอียดของความกว้างและความสูง
5. HtmlChoicd Class สนับสนุน HTML tag ที่ถูกเลือก
6. HtmlDefiinitionList Class สนับสนุน dl, dt และ dd HTML tags
7. HtmlForm Class สนับสนุนการสร้าง HTML form ซึ่งประกอบด้วย input fields, check boxes, radio buttons, text area, selections และ submit buttons
8. HtmlFramset Class สนับสนุน client-side image map HTML tags
9. HtmlImageMap Class สนับสนุน client-side image map HTML tags
10. HtmlList Class สนับสนุน ul และ ol HTML list tags
11. HtmlPages Class สนับสนุนการสร้าง web page ที่ประกอบด้วย header และ body areas
12. HtmlTable Class สนับสนุนการสร้างตารางจาก row element
13. HtmlRow Class สนับสนุน HTML เพื่อกำหนดแถวของตาราง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีกรนำไปใช้

## บทที่ 3

### การออกแบบ การสร้าง และการทำงาน

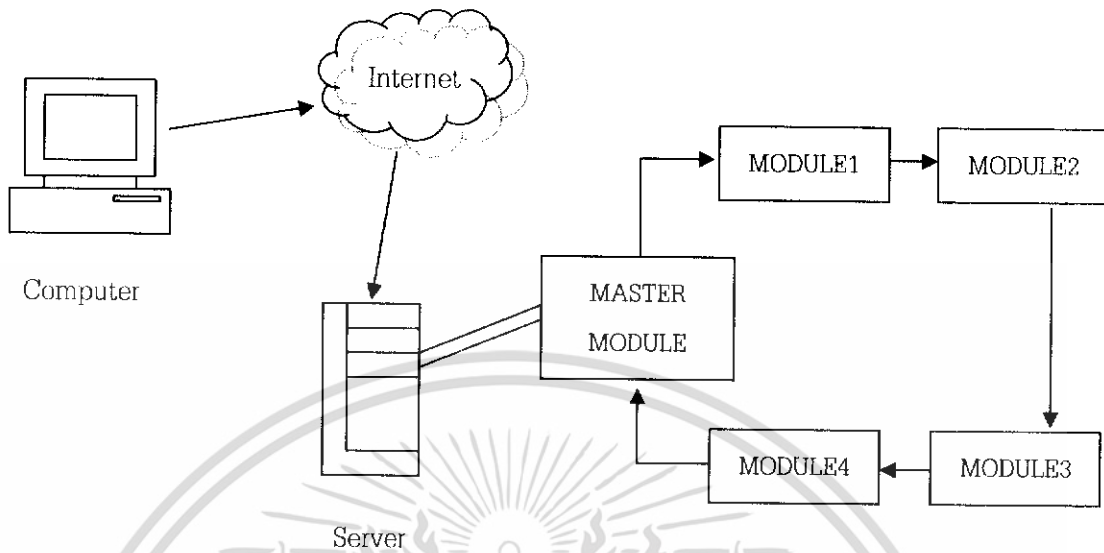
#### 3.1 กล่าวนำ

การออกแบบและการสร้างชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายคอมพิวเตอร์นั้น ได้แบ่งออกเป็น 2 ส่วนคือ ส่วนควบคุมอุปกรณ์ และซอฟต์แวร์ ซึ่งในส่วนของชุดควบคุมอุปกรณ์นั้น จะประกอบไปด้วย 5 ส่วนใหญ่ๆ คือ ส่วนของแผงวงจรควบคุมหลัก ซึ่งเป็นแผงวงจรไมโครคอนโทรลเลอร์ PIC 16F628 และส่วนของแผงวงจรเชื่อมต่อต่างๆ เช่น แผงวงจรชุดแม่ข่าย แผงวงจรสวิตซ์ แผงวงจรควบคุมอุปกรณ์ไฟฟ้า และแผงวงจรจ่ายไฟ โดยการทำงานของแผงวงจรชุดแม่ข่ายจะเป็นเชื่อมต่อเพื่อรับและส่งข้อมูลผ่านแผงวงจรควบคุมหลักเท่านั้น ในส่วนของแผงวงจรควบคุมหลักหรือแผงวงจรไมโครคอนโทรลเลอร์ จะมีหน้าที่สำคัญในการประมวลผลชุดคำสั่ง เพื่อสั่งให้แผงวงจรควบคุมอุปกรณ์ไฟฟ้าทำงานตามชุดคำสั่งนั้นๆ ส่วนของโปรแกรมก็จะใช้ในการทดสอบการทำงานของแผงวงจรต่างๆ ซึ่งจะเน้นในเรื่องของการส่งข้อมูลจากแผงวงจรควบคุมไฟฟ้าผ่านไปยังโปรแกรมที่ทำงานอยู่บนเครือข่ายบนอินเทอร์เน็ต รวมไปถึงการควบคุมการทำงานอุปกรณ์ไฟฟ้าโดยการกดสวิตซ์อีกช่องทางหนึ่งด้วย

#### 3.2 สถาปัตยกรรมของระบบ

โครงการนี้ได้ออกแบบโดยการนำระบบคอมพิวเตอร์มาใช้รับและส่งข้อมูลผ่านระบบอินเทอร์เน็ต ซึ่งข้อดีจากการรับและส่งข้อมูลผ่านอินเทอร์เน็ตนั้น คือผู้ใช้งานสามารถสั่งการทำงานของชุดควบคุมอุปกรณ์ไฟฟ้าได้ ไม่ว่าจะอยู่ที่ใดๆ ก็ตาม เพียงแค่ผู้ใช้งานสามารถเข้าใช้ระบบอินเทอร์เน็ตได้ ก็จะสามารถสั่งการทำงานของอุปกรณ์ไฟฟ้าที่อยู่ห่างไกลได้ การทำงานของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตนี้จะมีลักษณะการทำงานเป็นแบบไคลเอนท์-เซิร์ฟเวอร์ ซึ่งผู้ใช้งานจะต้องส่งข้อมูลมายังเครื่องเซิร์ฟเวอร์ก่อนแล้วเครื่องที่เป็นเซิร์ฟเวอร์ก็จะทำการตรวจสอบสิทธิ์การเข้าใช้งานของชุดควบคุมอุปกรณ์ไฟฟ้า ว่าผู้ใช้งานคนนี้มีสิทธิ์เข้ามาใช้งานชุดควบคุมอุปกรณ์ไฟฟ้าหรือไม่ หากตรวจสอบแล้วพบว่าไม่มีสิทธิ์ เครื่องเซิร์ฟเวอร์ก็จะรอรับชุดคำสั่งของผู้ใช้งาน โดยการส่งข้อมูลนั้นจะส่งมาในรูปแบบเฟรมข้อมูลตามชุดรูปแบบที่ได้กำหนดไว้ในโปรโตคอลที่ใช้สื่อสาร หากเครื่องเซิร์ฟเวอร์ตรวจสอบไม่พบสิทธิ์การเข้าใช้งานของผู้ใช้ที่จะเข้ามาควบคุมอุปกรณ์ไฟฟ้า ก็จะทำการปฏิเสธการร้องขอเข้าใช้งานชุดควบคุมอุปกรณ์ไฟฟ้าของผู้ใช้ และรอการป้อนชุดคำสั่งที่ถูกต้องต่อไป โครงสร้างสถาปัตยกรรมโดยรวมมีลักษณะ ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.1 สถาปัตยกรรมของระบบ

จากรูป เครื่องคอมพิวเตอร์ที่เชื่อมต่ออินเทอร์เน็ตจะทำการส่งชุดข้อมูลมาควบคุมหรือรับค่าจากอุปกรณ์ไฟฟ้าผ่านเครื่องคอมพิวเตอร์ที่เป็นเซิร์ฟเวอร์ (ในโครงการนี้จะใช้พีซีคอมพิวเตอร์ แทนเครื่องเซิร์ฟเวอร์) โดยเครื่องคอมพิวเตอร์ที่เป็นเซิร์ฟเวอร์จะตรวจสอบสิทธิ์การใช้งานของผู้ใช้งานก่อน หากตรวจสอบแล้วพบว่าไม่มีสิทธิ์ใช้งาน ก็จะส่งชุดข้อมูลที่ได้รับมานั้นไปควบคุมการทำงานของแผงวงจรไมโครคอนโทรลเลอร์โดยจะผ่านแผงวงจรชุดแม่ข่ายก่อน เพื่อทำการแปลงแรงดันที่ออกมาจากเครื่องคอมพิวเตอร์ซึ่งปกติจะประมาณ 12-15 V ให้กลายเป็น TTL ค่าประมาณ 0-5 V เมื่อชุดข้อมูลที่ได้ผ่านแผงวงจรชุดแม่ข่ายแล้ว ก็จะถูกส่งไปยังแผงวงจรไมโครคอนโทรลเลอร์เพื่อให้แผงวงจรไมโครคอนโทรลเลอร์ทำการตรวจสอบชุดข้อมูลว่าถูกต้องตามที่ได้กำหนดเอาไว้หรือไม่ เมื่อตรวจสอบชุดข้อมูลว่าถูกต้องแล้วจึงทำการประมวลผลชุดคำสั่ง เพื่อส่งค่าชุดข้อมูลไปยังแผงวงจรควบคุมอุปกรณ์ไฟฟ้าที่ต่อพ่วงอยู่ เพื่อควบคุมอุปกรณ์ไฟฟ้าแต่ละชุดหรือทั้งระบบ เมื่อแผงวงจรควบคุมอุปกรณ์ไฟฟ้าทำงานตามชุดคำสั่งที่ได้รับ ก็จะแสดงสถานะการทำงานของอุปกรณ์ไฟฟ้านั้นๆ กลับมาให้ผู้ใช้ทราบผ่านทางเว็บเพจ โดยการต่อพ่วงของแผงวงจรควบคุมอุปกรณ์ไฟฟ้านั้นจะต่อพ่วงกันแบบวงแหวน โดยทำการตรวจสอบชุดคำสั่งแบบวนรอบทั้งระบบ แล้วส่งค่ากลับไปยังแผงวงจรไมโครคอนโทรลเลอร์เพื่อแสดงผลสถานะการทำงานของอุปกรณ์ไฟฟ้าแต่ละชุดออกไป

ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตจะแบ่งการทำงานออกเป็น 2 ส่วนหลักๆ ดังนี้

1. ส่วนฮาร์ดแวร์ (Hardware)
2. ส่วนซอฟต์แวร์ (Software)

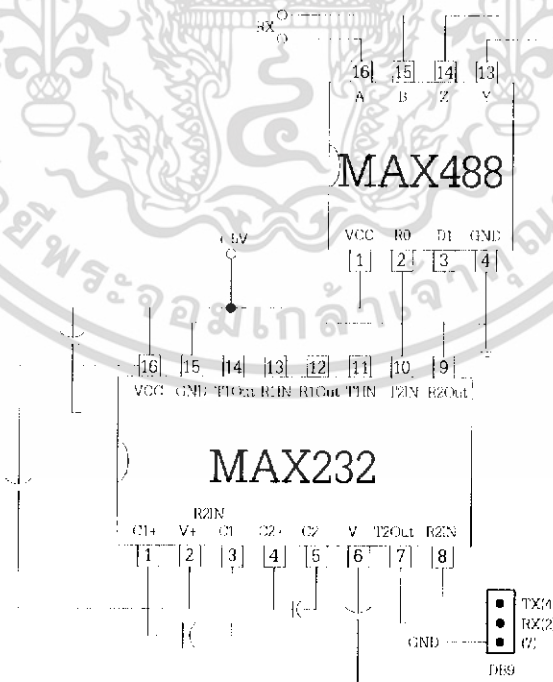
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 3.3 ฮาร์ดแวร์ (Hardware)

โครงการนี้จะเลือกใช้ไมโครคอนโทรลเลอร์ตระกูล PIC 16F628 เนื่องจากไมโครคอนโทรลเลอร์ตระกูล PIC16F62x มีคุณสมบัติสามารถรับส่งข้อมูลแบบอนุกรมภายในตัว มีการตรวจจับสัญญาณอินเตอร์รัพต์ อีกทั้งตัวไมโครคอนโทรลเลอร์ตระกูล PIC16F62x มีความเหมาะสมในการทำงานเนื่องจากมีราคาประหยัดและมีหน่วยความจำโปรแกรม 2 กิโลเวิร์ด กับหน่วยความจำข้อมูล (RAM) ถึง 224 ไบต์ที่พอเพียงต่อการใช้ในโครงการนี้

#### 3.3.1 แผงวงจรชุดแม่ข่าย

แผงวงจรชุดแม่ข่ายจะทำหน้าที่แปลงระดับแรงดันไฟฟ้า โดยเครื่องคอมพิวเตอร์เซิร์ฟเวอร์จะส่งข้อมูลมาทางพอร์ตอนุกรม (RS-232) แบบ 9 ขา (DB-9) ซึ่งระดับแรงดันของคอมพิวเตอร์ที่ส่งมาจะมีค่าสูงกว่าระดับแรงดันของไมโครคอนโทรลเลอร์ที่เป็นลอจิก TTL คือ 0-5 โวลต์ ดังนั้น จึงต้องมีการแปลงระดับสัญญาณให้อยู่ในช่วงของลอจิก TTL โดยใช้ไอซี MAX232 ที่ทำหน้าที่แปลงระดับแรงดันเครื่องคอมพิวเตอร์ +/-15 โวลต์ เป็นระดับแรงดัน +5 โวลต์ กับ 0 โวลต์ ของไมโครคอนโทรลเลอร์ จากนั้นเมื่อข้อมูลที่ได้ผ่าน MAX232 มาแล้ว ชุดข้อมูลจะเข้าไปที่ MAX488 เพราะต้องการแปลงชุดข้อมูลที่เป็น RS-232 แบบ TTL ให้กลายเป็นไลน์ไดรฟ์เวอร์ (Line Driver) RS-485 เพื่อที่ชุดข้อมูลจะสามารถส่งไปได้ไกลขึ้นและเข้าไปยังแผงวงจรไมโครคอนโทรลเลอร์ต่อไป



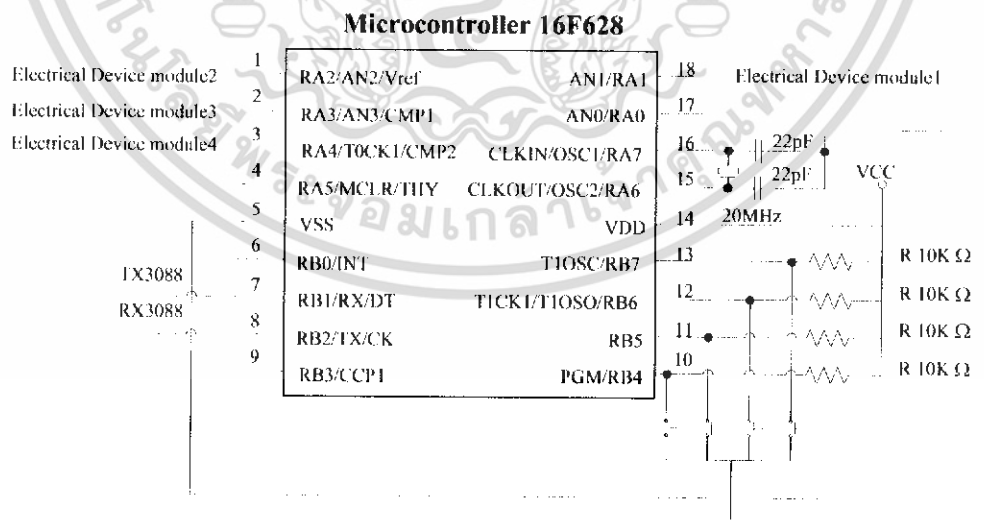
รูปที่ 3.2 การเชื่อมต่อของวงจรชุดแม่ข่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 3.3.2 แผงวงจรไมโครคอนโทรลเลอร์

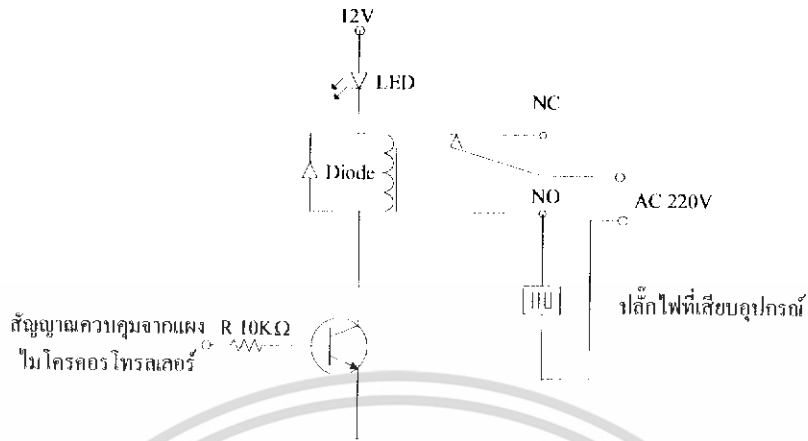
แผงวงจรไมโครคอนโทรลเลอร์จะทำหน้าที่ควบคุมการทำงาน การรับส่งข้อมูลจากแผงวงจรชุดแม่ข่ายก่อนที่จะส่งไปยังแผงวงจรควบคุมอุปกรณ์ไฟฟ้าที่ต่อพ่วงอยู่เพื่อสั่งให้อุปกรณ์ไฟฟ้าทำงาน พร้อมทั้งตรวจสอบความถูกต้องของข้อมูลและรายงานผลต่างๆ ที่เกิดขึ้นจากโหนดภายในแผงวงจรไมโครคอนโทรลเลอร์ไปยังเครื่องคอมพิวเตอร์ที่เป็นเซิร์ฟเวอร์

ที่แผงวงจรไมโครคอนโทรลเลอร์นั้นจะมีไอซี MAX3088 มาแปลงไลน์ไดรฟ์เวอร์ (Line Driver) RS-485 ที่ถูกส่งออกมาจากแผงวงจรชุดแม่ข่าย ให้กลายเป็นชุดข้อมูล RS-232 แบบ TTL เพื่อที่จะสามารถรับส่งค่าผ่านไมโครคอนโทรลเลอร์ได้ เมื่อแผงวงจรไมโครคอนโทรลเลอร์ได้รับค่ามาแล้ว ก็จะทำการส่งค่าชุดข้อมูลที่ตรวจสอบแล้วพบว่าถูกต้อง ไปยังแผงวงจรควบคุมอุปกรณ์ไฟฟ้าที่ต่อพ่วงอยู่ในกับแผงวงจรไมโครคอนโทรลเลอร์ เมื่อแผงวงจรควบคุมอุปกรณ์ไฟฟ้าทำงานตามคำสั่งแล้ว ก็จะส่งค่าของข้อมูลที่ได้รับการควบคุมนั้นออกไปให้แผงวงจรไมโครคอนโทรลเลอร์ เมื่อแผงวงจรไมโครคอนโทรลเลอร์ได้รับชุดข้อมูลของอุปกรณ์ไฟฟ้ามาแล้ว ก็จะส่งสถานะของอุปกรณ์ไฟฟ้านั้นออกมา โดยจะผ่าน MAX3088 ทำการแปลงมาตรฐานการส่งให้เป็นแบบ RS-485 อีกครั้ง เพื่อให้สามารถส่งข้อมูลไปยังอีกแผงวงจรได้ไกลขึ้น และชุดข้อมูลที่ถูส่งออกมาก็จะเข้าไปยังแผงวงจรไมโครคอนโทรลเลอร์อีกชุดต่อไป เมื่อข้อมูลถูกส่งจนครบระบบแล้ว ก็จะส่งค่าสถานะทั้งหมดของอุปกรณ์ไฟฟ้าออกมาโดยผ่านวงจรแม่ข่ายเพื่อแปลงระดับแรงดันที่คอมพิวเตอร์สามารถรับค่าได้ออกมาให้ผู้ใช้ทราบ



**รูปที่ 3.3** การเชื่อมต่อของแผงวงจรไมโครคอนโทรลเลอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

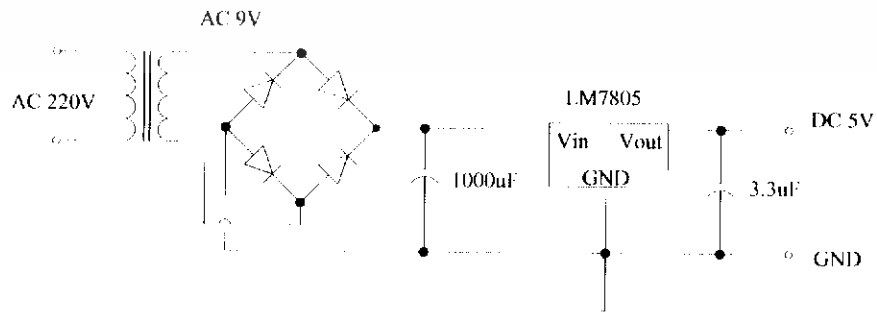


รูปที่ 3.4 การเชื่อมต่อของแผงควบคุมอุปกรณ์ไฟฟ้า

ที่แผงวงจรควบคุมอุปกรณ์ไฟฟ้า เมื่อมีสัญญาณควบคุมมาจากวงจรไมโครคอนโทรลเลอร์ที่ขา B ของทรานซิสเตอร์จะได้รับการไบอัส จากนั้นกระแสจากขา C จะไหลไปยังขา E ทำให้ขดลวดของรีเลย์มีกระแสไหลผ่าน ทำให้มีสนามแม่เหล็กไปเหนี่ยวนำ จึงทำให้หน้าสัมผัสเปลี่ยนจาก Normal Close ไปยัง Normal Open จากนั้นไฟ 220 V ก็จะถูกจ่ายไปที่กับปลั๊ก พร้อมทั้งจะจ่ายให้กับอุปกรณ์ที่นำมาเสียบเข้ากับปลั๊กนั้นๆ

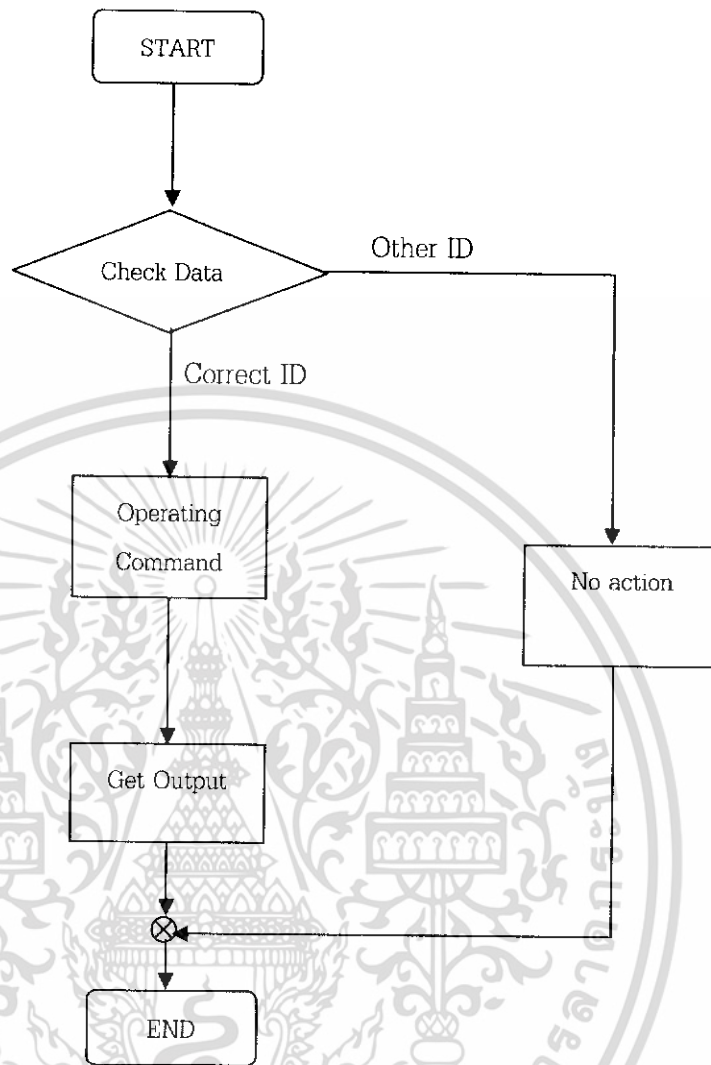
### 3.3.3 แผงวงจรจ่ายไฟ

จากหม้อแปลงกระแสสลับ 220 V/ 9 V เราใช้ไดโอดบริดจ์ ไอซีเบอร์ LM7805 และตัวเก็บประจุต่อร่วมกัน โดยตัวเก็บประจุตัวที่มีค่า 1000 uF นั้นจะทำหน้าที่กรองกระแสให้เรียบ เพื่อให้แรงดันไม่มี ripple ที่สูงมากนัก ตัวเก็บประจุตัวที่มีค่า 2.2 mF จะทำหน้าที่เป็นตัวกรองสัญญาณความถี่สูงที่ได้รับมาจากตัวเก็บประจุตัวแรก เพื่อไม่ให้มีสัญญาณรบกวนกับวงจรมากเกินไป ตัวเก็บประจุทั้ง 2 ตัวนี้เป็นวงจรควบคุมแรงดันเพื่อป้องกันไฟกระชากตรง 9 V เป็นไฟเลี้ยงให้กับวงจรทั้งหมด



รูปที่ 3.5 วงจรแหล่งจ่ายไฟ

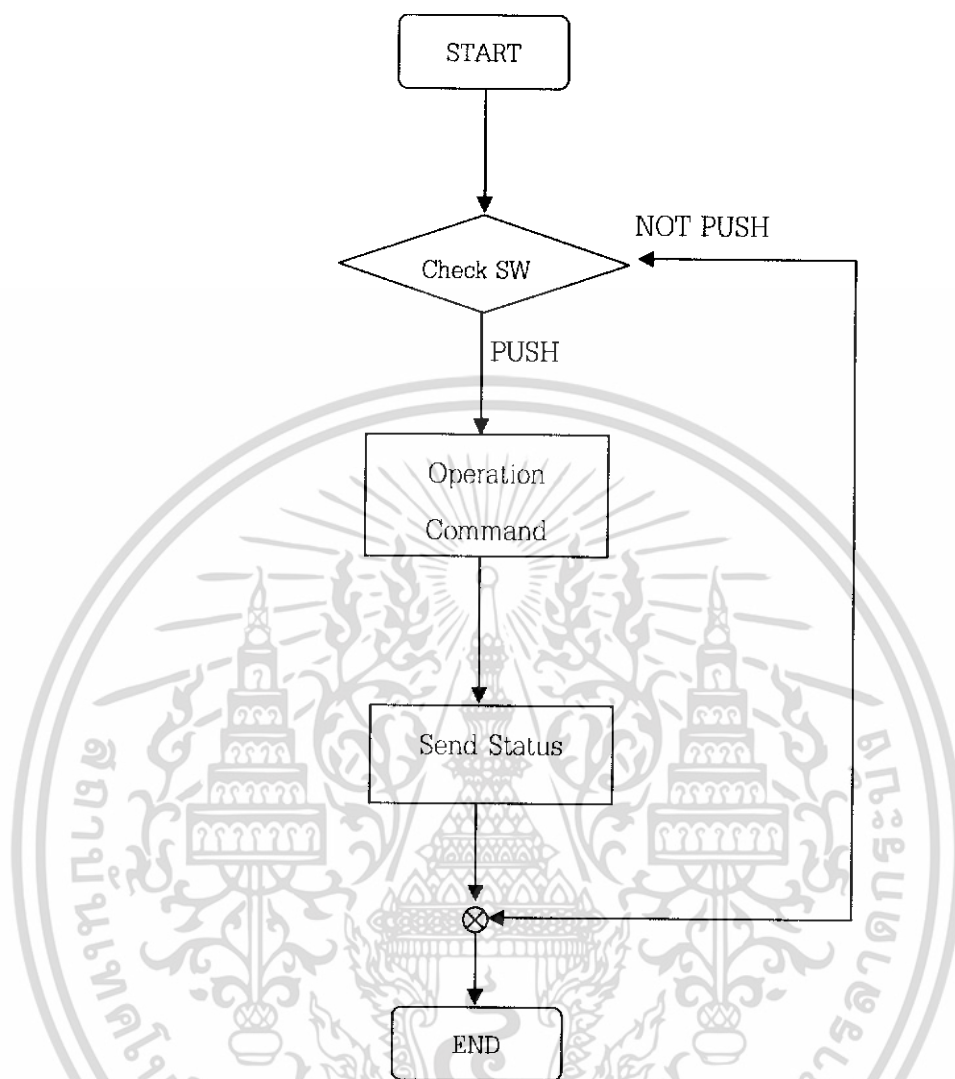
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น เมื่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.6 ผังงานการส่งค่าข้อมูลผ่านทางพอร์ต RS-232

เมื่อชุดควบคุมได้รับข้อมูลเข้ามาก็จะตรวจสอบข้อมูลที่เข้ามานั้นว่าตรงกับรหัสของตัวเองหรือไม่ ถ้าไม่ใช่ก็จะส่งข้อมูลไปยังชุดต่อไป แต่ถ้าเป็นรหัสของตนเอง ก็จะทำการประมวลผลคำสั่งที่เข้ามาว่าเป็นคำสั่งให้ทำอะไร จากนั้นก็สั่งให้อุปกรณ์ไฟฟ้าทำตามคำสั่งนั้นๆ คำสั่งที่ใช้ในการควบคุมอุปกรณ์ไฟฟ้ามีอยู่ 3 คำสั่งคือ เปิด ปิด และแจ้งสถานะ โดยการสั่งงานอุปกรณ์ไฟฟ้านั้นสามารถสั่งการทำงานที่ละเอียด หรือสั่งให้อุปกรณ์ไฟฟ้าทำงานทั้งหมดได้ เมื่ออุปกรณ์ไฟฟ้าประมวลผลคำสั่งและทำงานตามคำสั่งแล้ว ก็จะแสดงสถานะกลับไปยังเครื่องคอมพิวเตอร์นั้นๆ เพื่อแสดงผลของการสั่งงานกลับไปให้ผู้ใช้งานทราบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



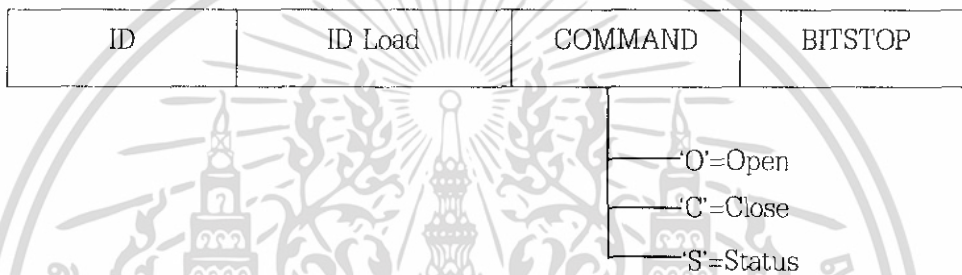
รูปที่ 3.7 ผังงานการส่งค่าข้อมูลโดยการกดสวิตช์

การทำงานของอุปกรณ์ไฟฟ้าผ่านการกดสวิตช์นั้น ถูกออกแบบโดยให้มีการตรวจสอบสถานะของการกดสวิตช์ ถ้าหากยังไม่มีมีการกดสวิตช์ อุปกรณ์ไฟฟ้าก็จะยังไม่ทำอะไรหรือแสดงสถานะของอุปกรณ์ไฟฟ้าให้ผู้ใช้ทราบ แต่ถ้ามีการกดสวิตช์เกิดขึ้น อุปกรณ์ไฟฟ้าก็จะทำการประมวลผลของชุดคำสั่งนั้นๆ เมื่อเป็นชุดข้อมูลของตนเองก็จะสั่งให้อุปกรณ์ไฟฟ้าทำงานตามคำสั่ง แล้วแสดงสถานะออกไปยังเครื่องคอมพิวเตอร์ เพื่อให้ผู้ใช้ทราบ หากมีการกดสวิตช์อีกครั้งที่อุปกรณ์ไฟฟ้าตัวเดิม ก็จะทำให้การเปลี่ยนสถานะปัจจุบันของอุปกรณ์ไฟฟ้าตัวนั้นให้เป็นสถานะตรงกันข้าม

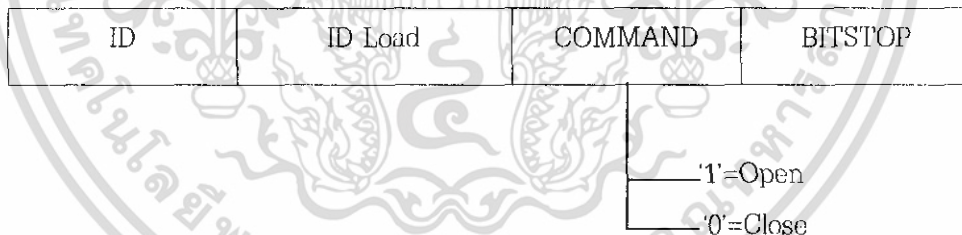
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 3.4 โปรโตคอล

ในการออกแบบนั้น การส่งข้อมูลเป็นสิ่งสำคัญอย่างยิ่ง เพราะจะต้องกำหนดรูปแบบการส่งข้อมูลให้ตรงกันถึงจะสามารถสื่อสารกันได้ และข้อมูลที่ส่งไป ถ้าเกิดผิดพลาดจะทำให้การทำงานนั้นขาดความน่าเชื่อถือและไม่มีประสิทธิภาพในการทำงาน ดังนั้นจึงได้มีการออกแบบรูปแบบการส่งข้อมูลที่เรียกว่า โปรโตคอล เพื่อใช้เป็นรูปแบบในการรับ/ส่งข้อมูลของฮาร์ดแวร์ในโครงการนี้ ข้อมูลที่ส่งในหนึ่งครั้งจะเรียกว่า เฟรมข้อมูล ซึ่งในหนึ่งเฟรมจะประกอบด้วยไบนารีข้อมูลที่แต่ละบิตมีความหมายในตัว เพื่อให้ฮาร์ดแวร์สื่อสารกันได้อย่างถูกต้องตรงกัน ลักษณะของเฟรมจะแสดงดังรูป



รูปที่ 3.8 เฟรมข้อมูลที่คอมพิวเตอร์ส่งไปยังไมโครคอนโทรลเลอร์



รูปที่ 3.9 เฟรมข้อมูลที่ไมโครคอนโทรลเลอร์ส่งไปยังคอมพิวเตอร์

ID	หมายถึง รหัสของแผงวงจรไมโครคอนโทรลเลอร์ที่จะได้รับชุดข้อมูล
ID Load	หมายถึง รหัสของแผงวงจรควบคุมอุปกรณ์ไฟฟ้า
COMMAND	หมายถึง ชุดคำสั่ง
BITSTOP	หมายถึง บิตสุดท้าย
"O"	หมายถึง ชุดคำสั่งให้เปิดการทำงานของอุปกรณ์ไฟฟ้า
"C"	หมายถึง ชุดคำสั่งให้ปิดการทำงานของอุปกรณ์ไฟฟ้า
"S"	หมายถึง ชุดคำสั่งให้แสดงสถานะของอุปกรณ์ไฟฟ้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น เมื่อผู้ผู้ใดเห็นนำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

"1"	หมายถึง ชุดคำสั่งแสดงสถานะการเปิดของอุปกรณ์ไฟฟ้า
"0"	หมายถึง ชุดคำสั่งแสดงสถานะการปิดของอุปกรณ์ไฟฟ้า

ในการติดต่อกับคอมพิวเตอร์ทุกครั้งเฟรมข้อมูลที่ส่งจะทำการตรวจสอบความถูกต้องของข้อมูลโดย

1. ตรวจสอบเฟรมที่ส่งเข้ามาว่าเท่ากับเฟรมที่เราได้กำหนดไว้หรือไม่ (ต้องเท่ากับ 4 ไบต์)
2. ตรวจสอบชุดคำสั่งที่เข้ามาว่ามีค่าเท่ากับค่าที่เรากำหนดไว้หรือไม่

(ในโครงงานนี้เราใช้ 01-อุปกรณ์ไฟฟ้าตัวที่ 1, 02-อุปกรณ์ไฟฟ้าตัวที่ 2, 03-อุปกรณ์ไฟฟ้าตัวที่ 3 และ 04-อุปกรณ์ไฟฟ้าตัวที่ 4, และ 09-อุปกรณ์ไฟฟ้าทุกตัว)

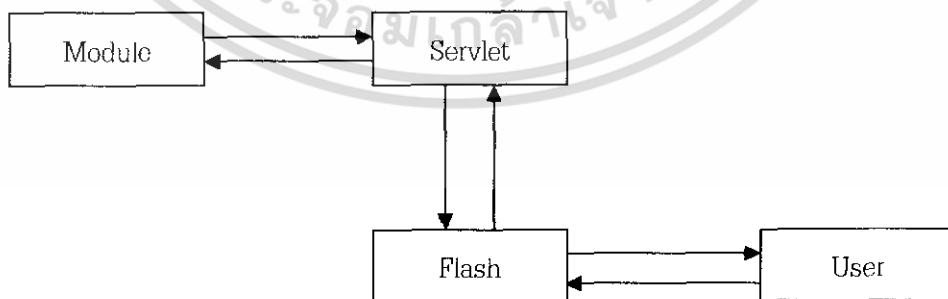
3. หากตรวจสอบแล้วพบว่าชุดคำสั่งที่ป้อนเข้ามานั้นไม่เท่ากับค่าที่เราได้ตั้งเอาไว้ ชุดคำสั่งนั้นก็จะถูกยกเลิกออกไป โดยแสดงผลที่เครื่องคอมพิวเตอร์ออกมาว่า Error

### 3.5 ซอฟต์แวร์ (Software)

ภาษาจาวา (JAVA) เป็นภาษาโปรแกรมซึ่งไม่ยึดติดกับระบบปฏิบัติการใดๆ จาวาจึงไม่สามารถติดต่อกับฮาร์ดแวร์ได้โดยตรง แต่มีทีมงานที่พัฒนาภาษาจาวาให้สามารถติดต่อกับพอร์ตอณุกรมขึ้นมาเฉพาะและผู้จัดทำได้นำ class ซึ่งถูกพัฒนามาดัดแปลงแก้ไข เพื่อใช้ติดต่อกับชุดควบคุมอุปกรณ์ไฟฟ้า โครงงานนี้เลือกใช้ภาษาซีในการอัดโปรแกรมลงไปยังไมโครคอนโทรลเลอร์ เพื่อให้ไมโครคอนโทรลเลอร์ทำงานตามชุดคำสั่ง

ซอฟต์แวร์ที่ใช้แบ่งออกเป็น 2 ส่วน ได้แก่

1. ซอฟต์แวร์ส่วนควบคุมและติดต่อกับฮาร์ดแวร์
2. ซอฟต์แวร์ส่วนแสดงผลผ่านอินเทอร์เน็ต



รูปที่ 3.10 โครงสร้างของระบบการรับส่งข้อมูลบนเครื่องเซิร์ฟเวอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 3.5.1 ซอฟต์แวร์ส่วนควบคุมและติดต่อกับฮาร์ดแวร์

ซอฟต์แวร์ที่ใช้ในการควบคุมแผงวงจรไมโครคอนโทรลเลอร์ เลือกใช้ Java Servlet ในการควบคุมการทำงานและรับส่งค่าข้อมูลไปยังซอฟต์แวร์ส่วนแสดงผล เนื่องจาก Servlet นั้นช่วยให้เราเขียน Program ที่สั่งให้ HTTP Server ทำงานหรือโต้ตอบกับ Program ของ Client เหมือนที่ CGI (Common Gateway Interface) ทำได้ ซึ่ง Servlet ถูกนำมาใช้แทน CGI script ได้อย่างมีประสิทธิภาพและให้ความสะดวกได้ อย่างมากคือสามารถเขียนได้ง่ายกว่าและทำการ RUN ได้เร็วกว่า CGI และ Servlet ยังได้มีการจัดการใน ส่วนของการโปรแกรมทางด้าน server-side ไว้โดยเฉพาะ Servlet ได้ถูกพัฒนาโดยใช้ Java Servlet API ซึ่งเป็นส่วนเพิ่มเติมมาตรฐานของ Java

การทำงานจะเริ่มจากการรัน Web Server ขึ้นมา หลังจากนั้นก็จะทำการเรียก Servlet ที่ถูกเขียนอยู่บนเครื่องเซิร์ฟเวอร์ เพื่อให้ Servlet ทำการดึงค่าข้อมูลต่างๆ ของแผงวงจรไมโครคอนโทรลเลอร์ หรือรอรับค่า ข้อมูลจากผู้ใช้งานผ่านซอฟต์แวร์ส่วนแสดงผล ส่งไปยังแผงวงจรไมโครคอนโทรลเลอร์ แล้วแสดงสถานะของ อุปกรณ์ไฟฟ้าที่แผงวงจรไมโครคอนโทรลเลอร์ควบคุมอยู่ ผ่านซอฟต์แวร์ส่วนแสดงผลออกมาให้ผู้ใช้งาน ทราบ

### 3.5.2 ซอฟต์แวร์ส่วนแสดงผลผ่านอินเทอร์เน็ต

ซอฟต์แวร์ที่ใช้ในการแสดงผลบนอินเทอร์เน็ตนั้น เลือกใช้โปรแกรม Flash เนื่องจากโปรแกรม Flash มีความยืดหยุ่นในการใช้งาน สามารถสร้างข้อความ ภาพ เสียงที่เคลื่อนไหวให้สามารถโต้ตอบกับ ผู้ใช้งานได้ รวมไปถึงเมื่อเราสร้างงานด้วยโปรแกรม Flash แล้ว ไฟล์ที่ได้จะมีขนาดไม่ใหญ่มากนัก ทำให้ใช้ เวลาในการ Download ไม่นานเกินไป

การทำงานจะเริ่มจากบราวเซอร์เรียกโปรแกรม Flash ที่รันอยู่บนอินเทอร์เน็ตขึ้นมา แล้วรอรับค่า จากผู้ใช้งานเพื่อส่งค่าข้อมูลไปยัง Servlet ให้ทำการประมวลผลและสั่งการทำงานไปยังแผงวงจร ไมโครคอนโทรลเลอร์ หลังจากนั้นก็รอรับค่าจาก Servlet ที่ส่งสถานะของอุปกรณ์ไฟฟ้าออกมา นำมาแสดงให้ ผู้ใช้งานได้ทราบต่อไป

## บทที่ 4

### การทดลองและผลการทดลอง

#### 4.1 กล่าวนำ

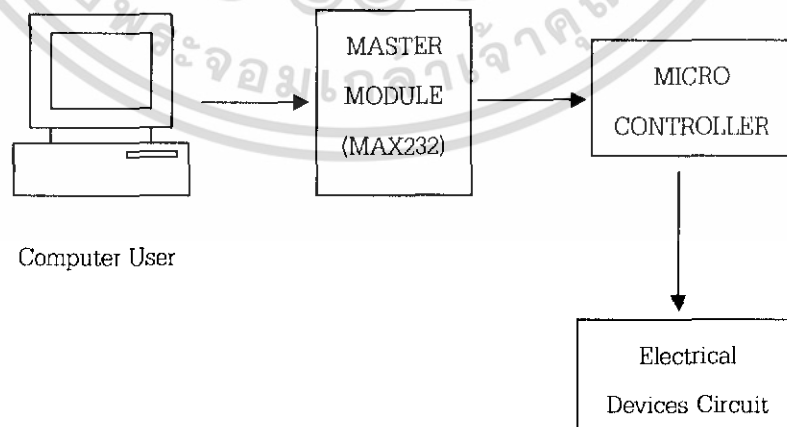
ในส่วนของบทนี้จะกล่าวถึงการทดลองและผลการทดลองของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต หลังจากที่ไดรวมวงจรชุดแม่ข่าย วงจรไมโครคอนโทรลเลอร์ วงจรสวิตช์ วงจรควบคุมอุปกรณ์ไฟฟ้าและวงจรจ่ายไฟเข้าด้วยกันแล้ว ก็จะใช้โปรแกรมที่ได้เขียนขึ้นมา นำทดลองการทำงานของชุดควบคุมอุปกรณ์ไฟฟ้างานนี้

#### 4.2 การทดลองการทำงานของแผงวงจรไมโครคอนโทรลเลอร์ 16F628

หน้าที่หลักของแผงวงจรไมโครคอนโทรลเลอร์ 16F628 คือ การรับข้อมูลจากคอมพิวเตอร์หรือสวิตช์ และส่งข้อมูลออกทางแผงวงจรเชื่อมต่อ โดยจะทำการติดต่อผ่านทางพอร์ตอนุกรม (DB-9) ดังนั้นในการทดลองแผงวงจรชุดนี้จึงสามารถแบ่งการทดลองได้ดังนี้

##### 4.2.1 การทดลองติดต่อผ่านคอมพิวเตอร์

หลังจากการประกอบวงจรชุดแม่ข่าย วงจรไมโครคอนโทรลเลอร์ วงจรสวิตช์ วงจรควบคุมอุปกรณ์ไฟฟ้าและวงจรจ่ายไฟ เข้าด้วยกันเรียบร้อยแล้ว ให้ต่อคอนเนคเตอร์เข้ากับพอร์ตอนุกรม COM1 เปิดโปรแกรม Hyper Terminal ขึ้นมาเพื่อทำการรับชุดคำสั่งที่ผู้ใช้ส่งมา

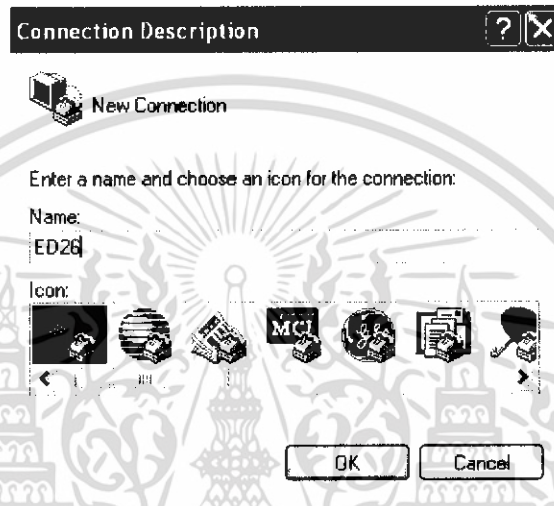


รูปที่ 4.1 การทดลองส่งค่าข้อมูลผ่านคอมพิวเตอร์ไปยังแผงวงจรควบคุมอุปกรณ์ไฟฟ้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากการทดลอง เริ่มจากการนำเครื่องคอมพิวเตอร์ที่ต่อกับชุดควบคุมอุปกรณ์ไฟฟ้าโดยผ่านพอร์ตอนุกรม RS-232 แล้วทำการเปิดโปรแกรม Hyper Terminal โดยไปที่ Start → All program → Accessories → Communications → hyper Terminal

เมื่อเปิดโปรแกรมขึ้นมาแล้ว ก็ทำการใส่ชื่อของการ Connection โดยการทดลองครั้งนี้ เราจะใส่ชื่อ ED26 หลังจากนั้น กด OK ดังรูปที่ 4.2



รูปที่ 4.2 หน้าแรกของโปรแกรม Hyper Terminal

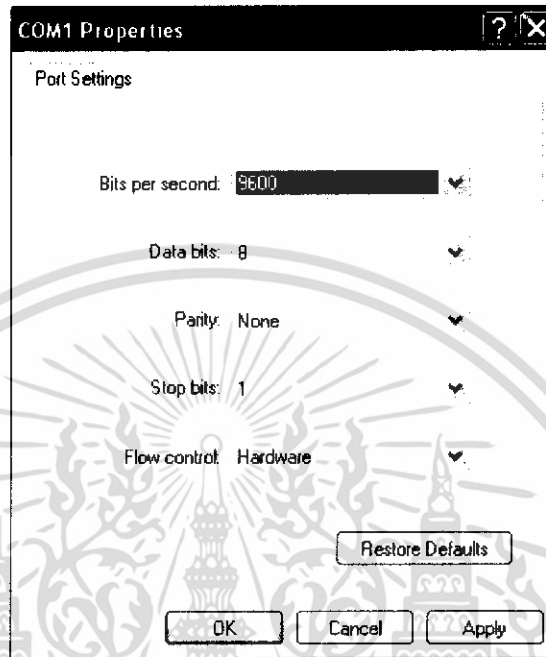
เมื่อกำหนดชื่อการ Connection ขึ้นมาแล้วจะขึ้นหน้าต่างพอร์ตในการเชื่อมต่อ กำหนด Connect using ไปที่ Com1 ดังรูปที่ 4.3



รูปที่ 4.3 การกำหนดค่าการเชื่อมต่อของ Connect using

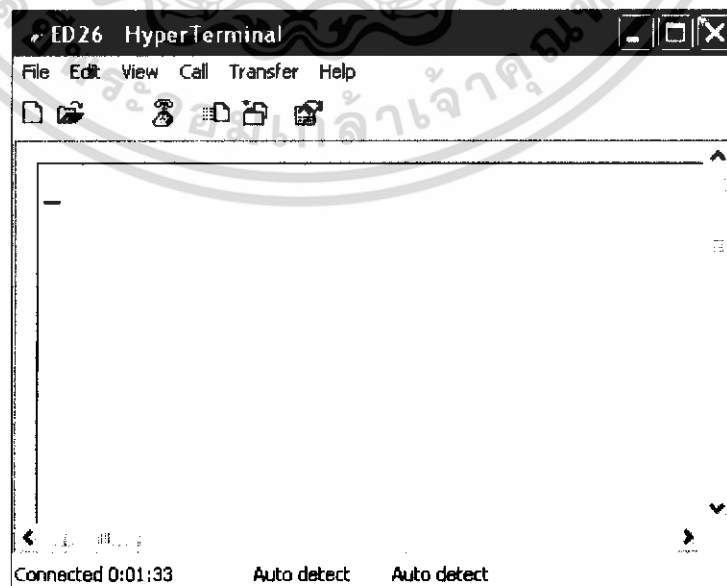
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อกำหนดพอร์ตในการเชื่อมต่อแล้ว ต่อไปจะเป็นการกำหนดบอร์ดเรตในการเชื่อมต่อ ในการทดลองครั้งนี้เรากำหนดให้อยู่ที่ 9600 ดังรูปที่ 4.4



รูปที่ 4.4 การกำหนดบอร์ดเรตในการเชื่อมต่อ

เมื่อกำหนดค่าต่างๆ ในการเชื่อมต่อเสร็จแล้ว จะขึ้นหน้าโปรแกรม Hyper Terminal ขึ้นมา เพื่อให้เราพิมพ์ชุดคำสั่ง ดังรูปที่ 4.5



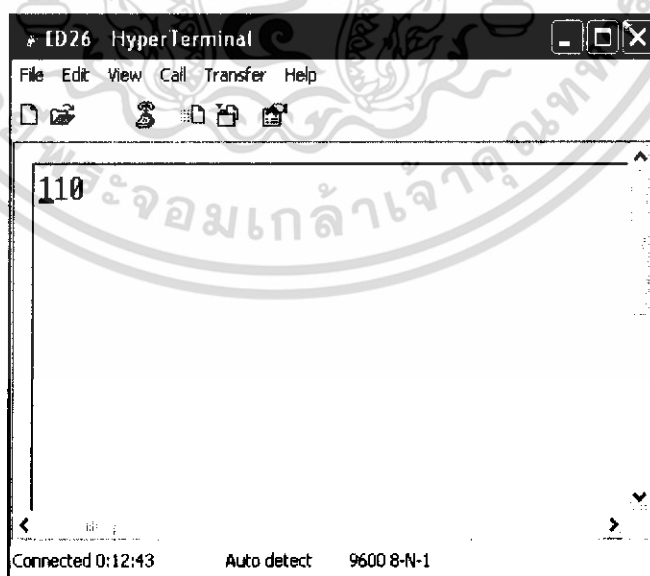
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
รูปที่ 4.5 หน้าหลักโปรแกรม Hyper Terminal ที่ใช้ในการเชื่อมต่อ  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมีเหตุดเบี่ยงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการส่งค่าข้อมูลแต่ละครั้ง จะส่งผ่านทางโปรแกรม Hyper Terminal โดยการพิมพ์ชุดคำสั่งตามที่ เราได้กำหนดเอาไว้ในโปรโตคอล เช่น หากเราต้องการจะสั่งให้โมดูลที่ 1 และอุปกรณ์ตัวที่ 1 ทำการเปิด เรา ต้องใช้คำสั่ง 110 โปรแกรมจะรับค่าสถานะของอุปกรณ์เข้ามาโดยจะแสดงค่าเป็น 111 ดังรูปที่ 4.6



รูปที่ 4.6 โปรแกรมรับค่า 110 แล้วแสดงค่าสถานะออกผ่านทาง Hyper Terminal

เช่นเดียวกัน หากเราต้องการจะสั่งให้โมดูลที่ 1 และอุปกรณ์ตัวที่ 1 ทำการปิด เราต้องใช้คำสั่ง 11C โปรแกรมจะรับค่าสถานะของอุปกรณ์เข้ามาโดยจะแสดงค่าเป็น 110 ดังรูปที่ 4.7

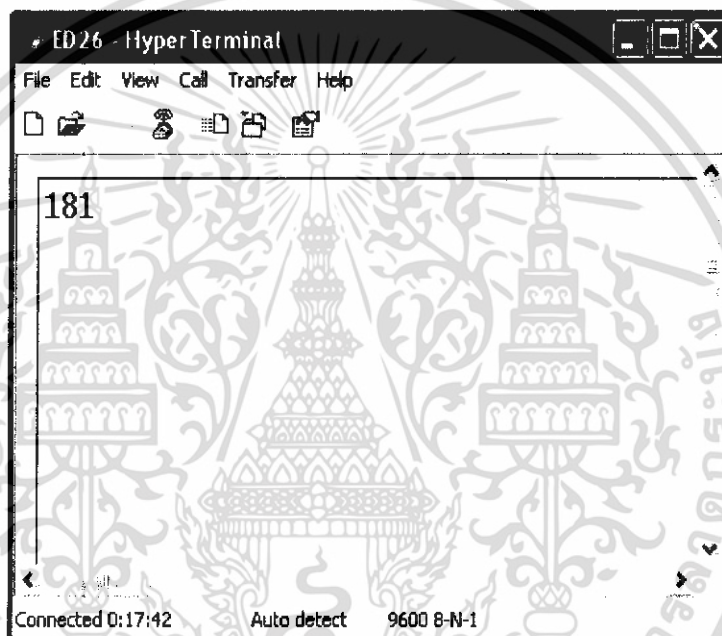


รูปที่ 4.7 โปรแกรมรับค่า 11C แล้วแสดงค่าสถานะออกผ่านทาง Hyper Terminal

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

#### 4.2.2 การทดลองติดต่อผ่านสวิทช์

หลังจากที่ทดลองการรับและส่งข้อมูลผ่านทางโปรแกรม Hyper Terminal ไปแล้ว ให้นำวงจรเดิมของการทดลองการติดต่อผ่านทางคอมพิวเตอร์นั้นมาทดลองผ่านการกดสวิทช์ดู จะพบว่า หากเรากดสวิทช์ที่อยู่บนบอร์ดตัวที่ 1 ของแผงวงจรควบคุมอุปกรณ์ไฟฟ้าตัวที่ 1 ผลที่ได้ก็คือ อุปกรณ์ไฟฟ้าตัวที่ 1 ของแผงควบคุมอุปกรณ์ไฟฟ้าชุดที่ 1 ไฟ LED สว่างขึ้น แล้วทดลองทำเช่นนี้กับแต่ละชุดของแผงควบคุมอุปกรณ์ไฟฟ้า พบว่าอุปกรณ์ไฟฟ้าทุกตัวทำงานตามการกดสวิทช์ แสดงว่า วงจรไมโครคอนโทรลเลอร์นั้นทำงานได้อย่างถูกต้อง



รูปที่ 4.8 โปรแกรมรับค่าการกดสวิทช์แล้วแสดงออกผ่านทาง Hyper Terminal

รูปที่ 4.8 เป็นการส่งค่าชุดข้อมูลของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านทางสวิทช์ โดยค่าข้อมูลจะแสดงออกมายังโปรแกรม Terminal โดยที่ 1 คือ ชุดโมดูลตัวที่ 1, 8 คือ โทลด์ตัวที่ 4 และ 1 คือ สถานะเปิดของอุปกรณ์ตัวนั้นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.1 สถานะของอุปกรณ์ไฟฟ้าผ่านทาง Hyper Terminal module ตัวที่ 1

	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
อุปกรณ์ที่																
กดเปิดอุปกรณ์ตัวที่ 1	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 2	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 3	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 4	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 1	ดับ	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 2	ดับ	ดับ	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.2 สถานะของอุปกรณ์ไฟฟ้าที่อุทกชุดควบคุมการณ module ตัวที่ 1

	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
อุปกรณ์ที่																
กรณีอุปกรณ์ตัวที่ 1	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 1	ดับ	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ดับ	ดับ	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.3 สถานะของอุปกรณ์ไฟฟ้าทาง Hyper Terminal module ตัวที่ 2

อุปกรณ์ที่	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
กรณีอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.4 สถานะของอุปกรณ์ไฟฟ้าที่ดูจากชุดควบคุมอุปกรณ์ module ตัวที่ 2

	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
อุปกรณ์ที่																
กดเปิดอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ติด	ติด	ติด	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ติด	ติด	ติด	ติด	ติด	ติด	ติด	ติด	ติด	ติด	ติด	ติด
กดปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.5 สถานะของอุปกรณ์ไฟฟ้าบนทาง Hyper Terminal module ตัวที่ 3

อุปกรณ์ที่	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
กรณีอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.6 สถานะของอุปกรณ์ไฟฟ้าที่ดูจากชุดควบคุมอุปกรณ์ module ตัวที่ 3

อุปกรณ์ที่	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
อุปกรณ์ที่	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
กรณีอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กรณีอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.7 สถานะของอุปกรณ์ไฟฟ้าผ่านทาง Hyper Terminal module ตัวที่ 4

	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
อุปกรณ์ที่																
กดเปิดอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดเปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กดปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 4.8 สถานะของอุปกรณ์ไฟฟ้าที่ดูจากชุดความดูแลอุปกรณ์ module ตัวที่ 4

	Module1				Module2				Module3				Module4			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
อุปกรณ์ที่																
กวดเปิดอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ตัวที่ 2	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ตัวที่ 3	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ตัวที่ 4	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ
กวดเปิดอุปกรณ์ทั้งหมดของโมดูลตัวที่ 1	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ	ดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

### 4.3 สรุปผลการทดลอง

จากการทดลองการทำงานของวงจรไมโครคอนโทรลเลอร์ โดยการส่งข้อมูลผ่านทางพอร์ตอนุกรม สามารถทำการส่งได้โดยใช้โปรแกรม Hyper Terminal โดยเริ่มจากการต่อชุดควบคุมอุปกรณ์ไฟฟ้าเข้ากับคอมพิวเตอร์ผ่านทางพอร์ตอนุกรม แล้วทำการเปิดโปรแกรม Hyper Terminal ขึ้นมา จากนั้นสามารถส่งข้อมูลจากคอมพิวเตอร์ไปยังชุดควบคุมอุปกรณ์ไฟฟ้า โดยการพิมพ์ชุดคำสั่ง 3 คำ (คำแรกคือ การกำหนดชุดไมโคร ถ้าใส่ 1 แสดงว่าต้องการให้ส่งค่าไปยังไมโครตัวที่ 1, คำที่สอง คือ การกำหนดโหนดในการทำงาน ในไมโครที่ใช้ในการทดลองจะมี 4 คำ คือ 1, 2, 4 และ 8) คำสุดท้ายคือ การกำหนดคำสั่งในการทำงานในอุปกรณ์ที่ไมโครนั้นๆ ทำงาน โดยที่ C คือ ปิด และ O คือ เปิด

เมื่อทำการสั่งค่าให้อุปกรณ์ที่ไมโครทำงาน พบว่า วงจรไมโครคอนโทรลเลอร์สามารถประมวลผลและส่งชุดคำสั่งที่ผู้ใช้ป้อนเข้าไปให้กับวงจรควบคุมอุปกรณ์ไฟฟ้าได้ดี และเมื่อนำอุปกรณ์ไฟฟ้ามาต่อใช้งานที่แผงวงจรควบคุม อุปกรณ์ไฟฟ้าตัวนั้นสามารถทำงานได้ตามชุดคำสั่งที่ป้อนเข้ามาโดยผ่านทางโปรแกรม Hyper Terminal พร้อมทั้งแสดงสถานะการทำงานของอุปกรณ์นั้นๆ ออกมาด้วย

รวมไปถึงการทดลองการกดสวิตซ์ที่แผงวงจรไมโครคอนโทรลเลอร์ พบว่าอุปกรณ์ไฟฟ้าที่ต่อร่วมอยู่กับวงจรควบคุมอุปกรณ์ไฟฟ้าที่ถูกควบคุมโดยวงจรไมโครคอนโทรลเลอร์ สามารถเปิดและปิดตามการกดสวิตซ์ของตำแหน่งอุปกรณ์ตัวนั้นๆ ได้ดี

## บทที่ 5

### บทสรุป

#### 5.1 บทสรุป

จากการออกแบบและการทดลองของโครงงานชิ้นนี้ เราได้พบว่าพอร์ตอนุกรมสามารถรับและส่งชุดข้อมูลไปควบคุมอุปกรณ์ไฟฟ้าในระยะไกลได้ แต่การควบคุมผ่านเว็บไม่สามารถควบคุมได้เนื่องจากการเขียนโปรแกรมของภาษาจาวามีปัญหา เนื่องจากไม่สามารถแก้ไขคลาสที่ติดต่อฮาร์ดแวร์ได้ แต่สามารถส่งข้อมูลผ่านเทอร์มินัลได้ ดังนั้นในการทดลองครั้งนี้ จึงเลือกใช้โปรแกรม Hyper Terminal ในการเชื่อมต่อชุดควบคุมเข้ากับคอมพิวเตอร์ เริ่มจากการออกแบบวงจรชุดแม่ข่าย โดยใช้ไอซีเบอร์ MAX232 ในการแปลงระดับแรงดันไฟฟ้าจากพอร์ตอนุกรม (DB-9) ให้เป็นระดับแรงดันแบบ TTL 0-5 V เพื่อให้สามารถส่งเข้าใช้ในวงจรไมโครคอนโทรลเลอร์ได้ โดยก่อนจะส่งไปยังวงจรมิโครคอนโทรลเลอร์นั้นจะต้องทำการแปลงให้เป็นมาตรฐาน RS-485 โดยใช้ไอซีเบอร์ MAX488 เพื่อทำให้ให้ชุดข้อมูลสามารถส่งไปในระยะไกลขึ้น ที่วงจรมิโครคอนโทรลเลอร์นั้นจะมีไอซี MAX3088 ในการรับชุดข้อมูลที่เป็นมาตรฐานแบบ RS-485 เพื่อทำการแปลงให้เป็นมาตรฐาน RS-232 ให้สามารถส่งเข้ามายังแผงวงจรมิโครคอนโทรลเลอร์ได้ เมื่อวงจรมิโครคอนโทรลเลอร์ได้ทำการประมวลผลแล้วก็จะส่งชุดคำสั่งที่ได้ไปให้วงจรควบคุมอุปกรณ์ไฟฟ้าที่ต่อเชื่อมกันกับวงจรมิโครคอนโทรลเลอร์ตัวนั้นๆ ทำงานตามชุดคำสั่ง พร้อมทั้งเก็บค่าสถานะของอุปกรณ์ไฟฟ้าตัวที่ถูกส่งงาน หลังจากนั้นก็จะส่งชุดคำสั่งออกไปยังแผงวงจรมิโครคอนโทรลเลอร์ชุดต่อๆ ไป จนครบทั้งระบบแล้วระบบจึงนำสถานะของอุปกรณ์ไฟฟ้าทุกตัวที่ถูกสั่งให้ทำงานนำไปแสดงผลอยู่บนเครื่องคอมพิวเตอร์

นอกจากนี้เรายังสามารถควบคุมอุปกรณ์ไฟฟ้าได้อีก 1 ช่องทางคือ การกดสวิตช์ ในกรณีที่เราไม่ต้องการให้มีการควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต เราก็สามารถจะทำได้โดยการกดสวิตช์ยกเลิกการสั่งงานผ่านทางเว็บเซิร์ฟเวอร์ โดยเว็บเซิร์ฟเวอร์จะทำหน้าที่แสดงสถานะของอุปกรณ์ภายในบ้านเท่านั้น แต่จะไม่สามารถควบคุมการทำงานของอุปกรณ์ไฟฟ้าได้

#### 5.2 ปัญหาและวิธีการแก้ไข

จากการดำเนินการสร้างและทดสอบโครงงานปรากฏว่ามีปัญหาเกิดขึ้นหลายประการ ซึ่งสามารถสรุปได้ดังนี้

1. ภาษาจาวาไม่สามารถทำงานบนพอร์ตอนุกรมได้

**วิธีการแก้ไข** เปลี่ยนมาใช้โปรแกรม Hyper Terminal ในการเชื่อมต่อ

2. ในกรณีที่มีคนอาศัยอยู่ในบ้านและกำลังใช้อุปกรณ์ไฟฟ้าอยู่ แต่คนที่อยู่ภายนอกไม่ทราบและ

ต้องการสั่งเปิด/ปิด อุปกรณ์ไฟฟ้า อาจจะทำให้เกิดอันตรายกับคนที่กำลังใช้ไฟฟ้าในบ้านได้ เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น เมื่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

**วิธีการแก้ไข** เพิ่มแผงวงจรสวิตช์เข้าไปยังชุดควบคุมอุปกรณ์ไฟฟ้า เพื่อป้องกันไม่ให้บุคคลภายนอกสามารถสั่งเปิด/ปิด อุปกรณ์ไฟฟ้าผ่านเว็บเซิร์ฟเวอร์ได้ แต่สามารถรับรู้สถานะของอุปกรณ์ไฟฟ้าได้

3. การเปิด/ปิดอุปกรณ์ไฟฟ้าทั้ง 16 ดวง ที่มีความผิดพลาด 2% เกิดจากอุปกรณ์รีเลย์ที่มีคุณสมบัติต่ำ เพราะซื้อในราคาที่ถูก

**วิธีการแก้ไข** เปลี่ยนมาใช้อุปกรณ์รีเลย์ที่มีคุณสมบัติที่ดีกว่า แต่มีราคาแพง

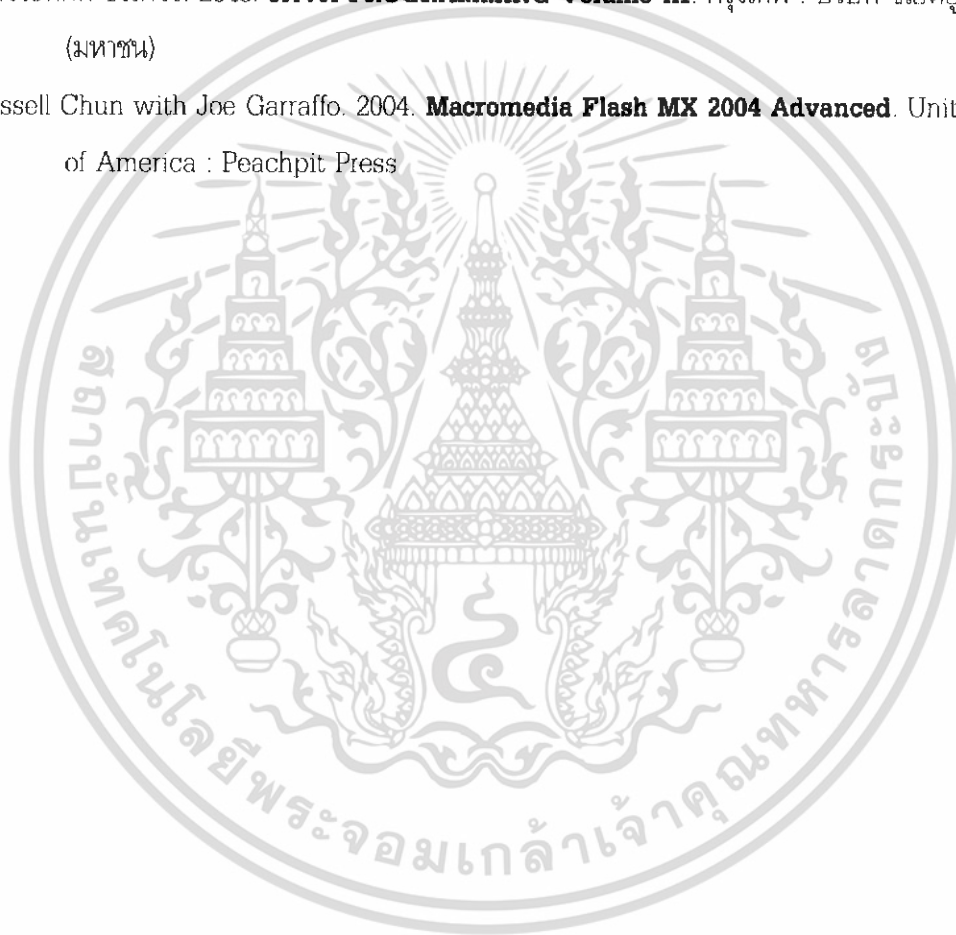
### 5.3 แนวทางการพัฒนา

1. โครงการนี้เป็นการควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต ที่ใช้คอมพิวเตอร์เป็นตัวสั่งการควบคุม แต่เนื่องจากปัจจุบันเทคโนโลยีทางด้านมือถือ ได้พัฒนาก้าวหน้าไปเป็นอันมาก สามารถรับส่งข้อมูลต่างๆ ผ่านทางมือถือ เช่น Bluetooth ได้แล้ว ดังนั้นโครงการนี้จึงเป็นแนวทางเพื่อให้มีการพัฒนาการควบคุมผ่านทางระบบมือถือต่อไป

2. นอกจากนี้ยังสามารถเขียนโปรแกรมตั้งเวลาการเปิด/ปิด อุปกรณ์ไฟฟ้า ซึ่งสามารถจะทำให้ตั้งเวลาในการเปิด/ปิด อุปกรณ์ไฟฟ้าได้อีกทางหนึ่ง

## บรรณานุกรม

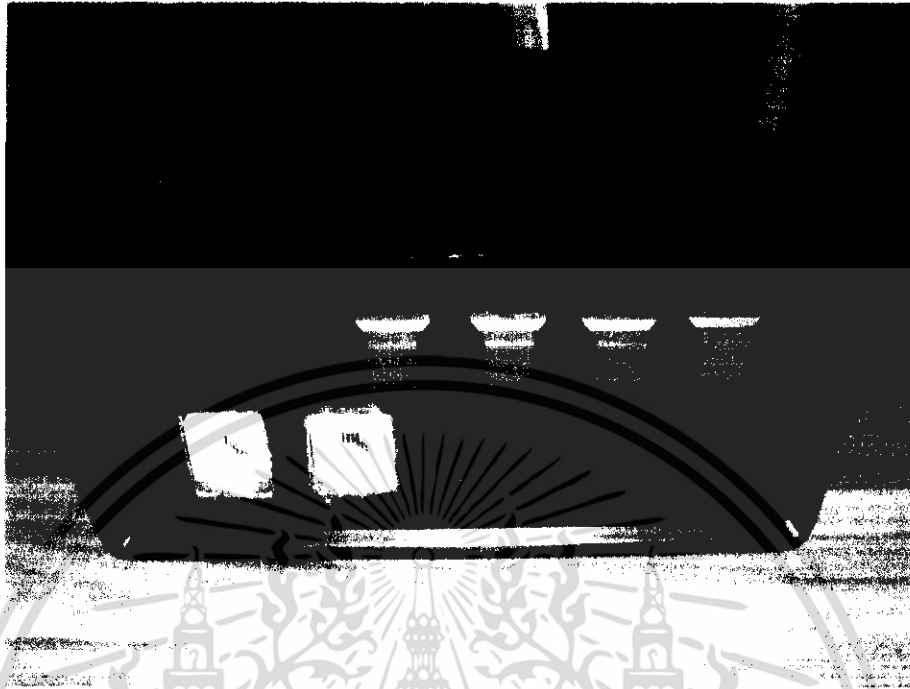
- ณัฐรุพล วงศ์สุนทรชัย, ชัยวัฒน์ ลิ้มพรจิตรวิไล. 2535. **เรียนรู้และปฏิบัติไมโครคอนโทรลเลอร์ PIC16F628 Microcontroller theory & experiment**. กรุงเทพฯ : บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน)
- ดวงพร ขอเจริญพร. 2521. **เขียนโปรแกรม Java บน Web ด้วย Servlets และ JSP**. กรุงเทพฯ : บริษัท เคทีพี คอมพ์ แอนด์ คอนซัลท์ จำกัด
- ดร.วีระศักดิ์ ชิงถาวร. 2548. **JAVA PROGRAMMING Volume III**. กรุงเทพฯ : บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน)
- Russell Chun with Joe Garraffo. 2004. **Macromedia Flash MX 2004 Advanced**. United States of America : Peachpit Press



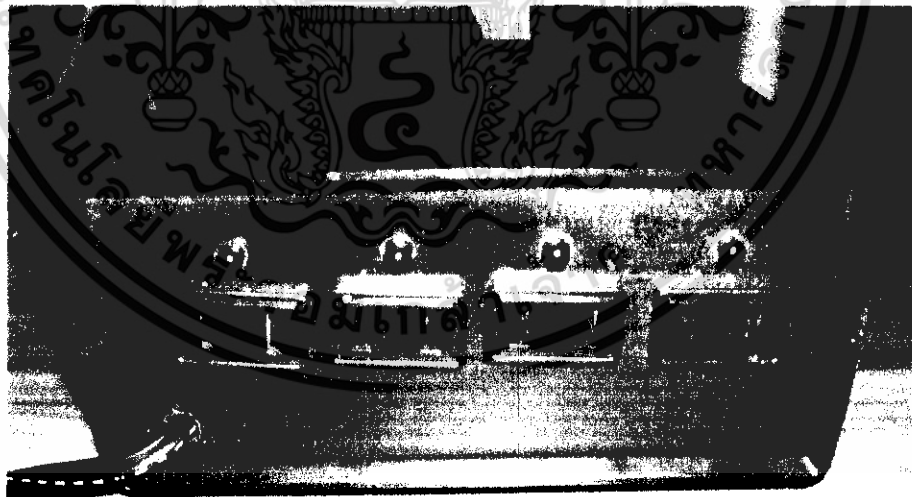
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ ก.1 ส่วนประกอบและปุ่มควบคุมของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต (ด้านหน้า)

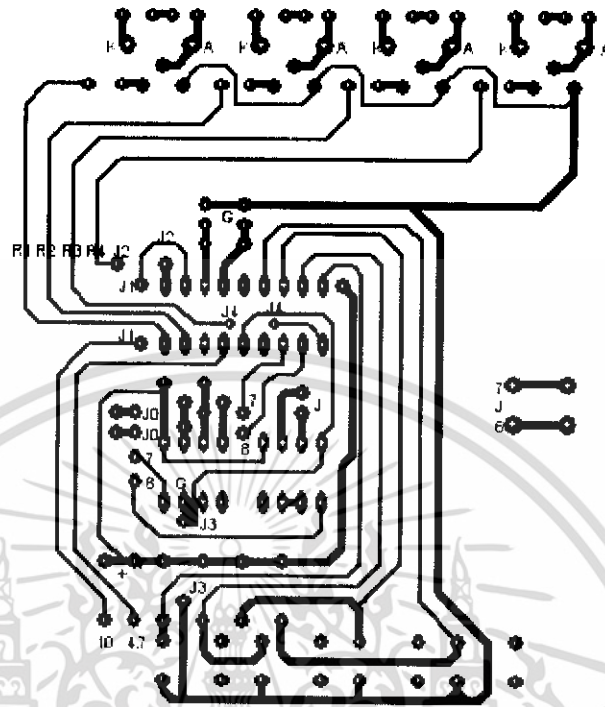


รูปที่ ก.2 ส่วนประกอบและปุ่มควบคุมของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต (ด้านหลัง)

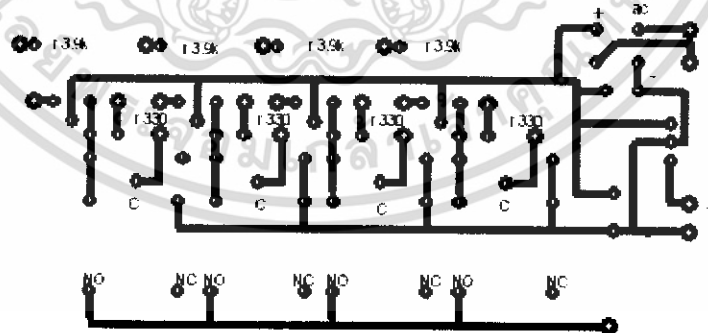
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

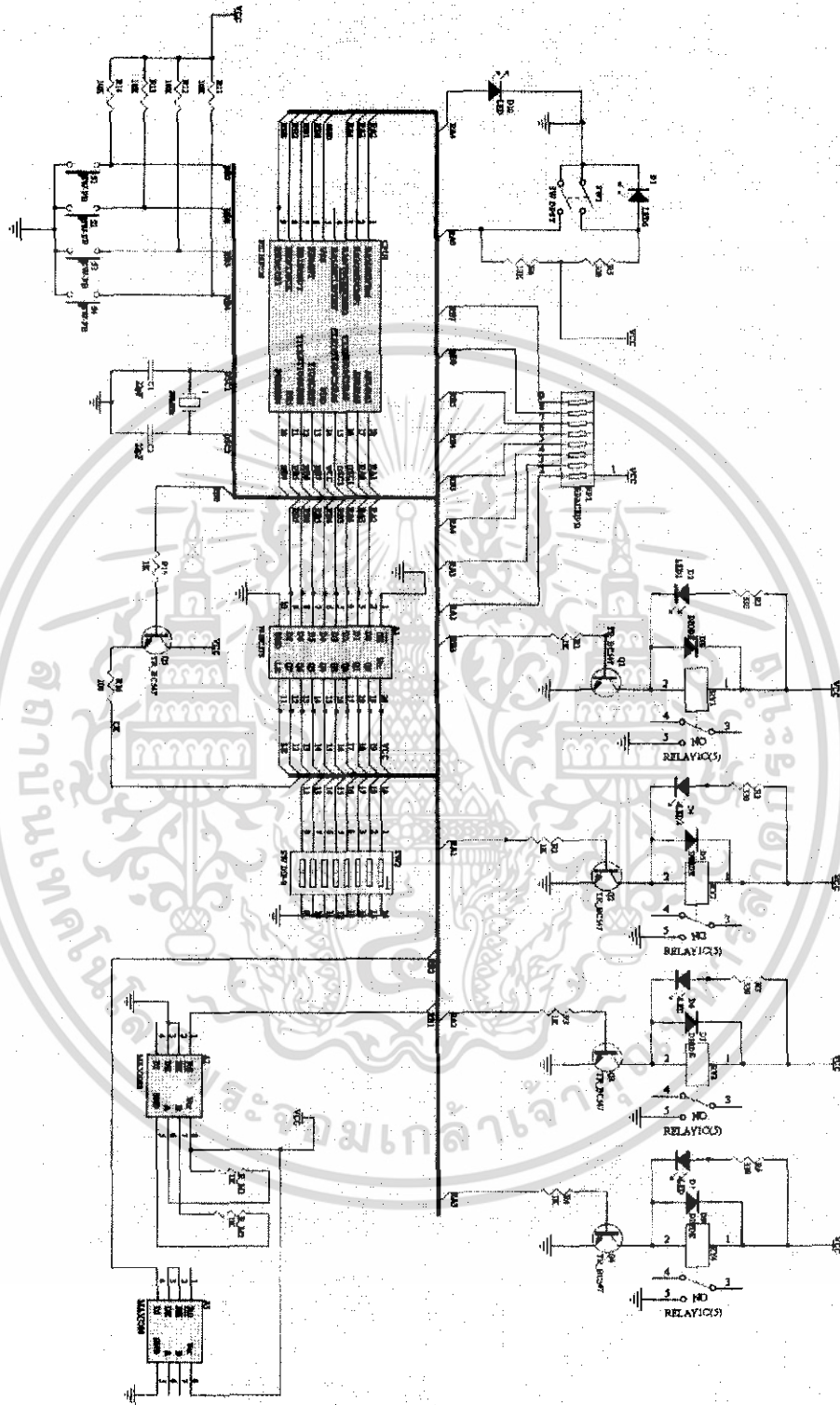


รูปที่ ข.1 แผงวงจรพิมพ์ของวงจรไมโครคอนโทรลเลอร์



รูปที่ ข.2 แผงวงจรพิมพ์ของวงจรรีเลย์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



**รูปที่ ข.3** การเชื่อมต่อและรายละเอียดอุปกรณ์ของชุดควบคุมอุปกรณ์ไฟฟ้าทั้งระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ ค.1 รายการอุปกรณ์ของแผงวงจรชุดแม่ข่าย

ชื่ออุปกรณ์	รายละเอียด	จำนวน
วงจรวม		
IC1	MAX232	1 ตัว
IC2	MAX488	1 ตัว
ตัวเก็บประจุ		
C1 - C5	0.1 $\mu$ F / 16v	5 ตัว

ตารางที่ ค.2 รายการอุปกรณ์ของแผงวงจรไมโครคอนโทรลเลอร์

ชื่ออุปกรณ์	รายละเอียด	จำนวน
วงจรวม		
IC1	MAX232	1 ตัว
ตัวเก็บประจุ		
C1 - C2	22 pF เซรามิก	2 ตัว
ตัวต้านทาน		
R1 - R4	10k $\Omega$ $\frac{1}{4}$ w 5%	4 ตัว
อุปกรณ์อื่น ๆ		
sw	แบบกดติด ปลดยับ 4 ขา	4 ตัว
คริสตอล	20 MHz	1 ตัว

ตารางที่ ค.4 รายการอุปกรณ์ของแผงวงจรจ่ายไฟ

ชื่ออุปกรณ์	รายละเอียด	จำนวน
อุปกรณ์สารกึ่งตัวนำ		
Q1	C458	1 ตัว
LED	ขนาด 0.5 mm	1 ตัว
ตัวต้านทาน		
R1	10k $\Omega$ $\frac{1}{4}$ w 5%	1 ตัว
อุปกรณ์อื่น ๆ		
Relay	ขนาดแรงดัน 12 v 1 คอนแทก	1 ตัว
คริสตอล	20 MHz	1 ตัว
ที่เสียบปลั๊ก	แบบ 2 แชนแนล	1 ตัว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



# PIC16F627A/628A/648A

## 18-pin Flash-Based 8-Bit CMOS Microcontrollers with nanoWatt Technology

- Operating speeds from DC - 20 MHz
- Interrupt capability
- 8-level deep hardware stack
- Direct, Indirect and Relative Addressing modes
- 35 single word instructions
  - All instructions single cycle except branches

### Special Microcontroller Features:

- Internal and external oscillator options
  - Precision Internal 4 MHz oscillator factory calibrated to  $\pm 1\%$
  - Low Power Internal 37 kHz oscillator
  - External Oscillator support for crystals and resonators
- Power saving Sleep mode
- Programmable weak pull-ups on PORTB
- Multiplexed Master Clear/Input-pin
- Watchdog Timer with independent oscillator for reliable operation
- Low voltage programming
- In-Circuit Serial Programming™ (via two pins)
- Programmable code protection
- Brown-out Reset
- Power-on Reset
- Power-up Timer and Oscillator Start-up Timer
- Wide operating voltage range (2.0 - 5.5V)
- Industrial and extended temperature range
- High Endurance Flash/EEPROM Cell
  - 100,000 write Flash endurance
  - 1,000,000 write EEPROM endurance
  - 100 year data retention

- Standby Current:
  - 100 nA @ 2.0V, typical
- Operating Current:
  - 12  $\mu$ A @ 32 kHz, 2.0V, typical
  - 120  $\mu$ A @ 1 MHz, 2.0V, typical
- Watchdog Timer Current:
  - 1  $\mu$ A @ 2.0V, typical
- Timer1 oscillator current:
  - 1.2  $\mu$ A @ 32 kHz, 2.0V, typical
- Dual Speed Internal Oscillator:
  - Run-time selectable between 4 MHz and 37 kHz
  - 4  $\mu$ s wake-up from Sleep, 3.0V, typical

### Peripheral Features:

- 16 I/O pins with individual direction control
- High current sink/source for direct LED drive
- Analog comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (VREF) module
  - Selectable internal or external reference
  - Comparator outputs are externally accessible
- Timer0: 8-bit timer/counter with 8-bit programmable prescaler
- Timer1: 16-bit timer/counter with external crystal/clock capability
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Capture, Compare, PWM module
  - 16-bit Capture/Compare
  - 10-bit PWM
- Addressable Universal Synchronous/Asynchronous Receiver/Transmitter USART/SCI

Device	Program Memory	Data Memory		I/O	CCP (PWM)	USART	Comparators	Timers 8/16-bit
	Flash (words)	SRAM (bytes)	EEPROM (bytes)					
PIC16F627A	1024	224	128	16	1	Y	2	2/1
PIC16F628A	2048	224	128	16	1	Y	2	2/1
PIC16F648A	4096	256	256	16	1	Y	2	2/1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

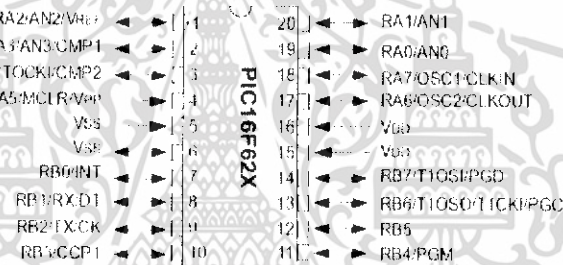
# PIC16F62X

## Pin Diagrams

### PDIP, SOIC



### SSOP



## Device Differences

Device	Voltage Range	Oscillator	Process Technology (Microns)
PIC16F627	3.0 - 5.5	(Note 1)	0.7
PIC16F628	3.0 - 5.5	(Note 1)	0.7
PIC16LF627	2.0 - 5.5	(Note 1)	0.7
PIC16LF628	2.0 - 5.5	(Note 1)	0.7

**Note 1:** If you change from this device to another device, please verify oscillator characteristics in your application.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## PIC16F62X

TABLE 2-1: PIC16F62X PINOUT DESCRIPTION (CONTINUED)

Name	Function	Input Type	Output Type	Description
RB4/PGM	RB4	TTL	CMOS	Bi-directional I/O port. Can be software programmed for internal weak pull-up.
	PGM	ST		Low voltage programming input pin. Interrupt-on-pin change. When low voltage programming is enabled, the interrupt-on-pin change and weak pull-up resistor are disabled.
RB5	RB5	TTL	CMOS	Bi-directional I/O port. Interrupt-on-pin change. Can be software programmed for internal weak pull-up.
RB6/T1OSO/T1CKI/PGC	RB6	TTL	CMOS	Bi-directional I/O port. Interrupt-on-pin change. Can be software programmed for internal weak pull-up.
	T1OSO		XTAL	Timer1 oscillator output.
	T1CKI	ST	—	Timer1 clock input.
	PGC	ST	—	ICSP™ Programming Clock.
RB7/T1OSI/PGD	RB7	TTL	CMOS	Bi-directional I/O port. Interrupt-on-pin change. Can be software programmed for internal weak pull-up.
	T1OSI	XTAL	—	Timer1 oscillator input. Wake-up from SLEEP on pin change. Can be software programmed for internal weak pull-up.
	PGD	ST	CMOS	ICSP Data I/O
Vss	Vss	Power	—	Ground reference for logic and I/O pins
VDD	VDD	Power	—	Positive supply for logic and I/O pins

Legend: O = Output  
 — = Not used  
 TTL = TTL Input

CMOS = CMOS Output  
 I = Input  
 OD = Open Drain Output

P = Power  
 ST = Schmitt Trigger Input  
 AN = Analog

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

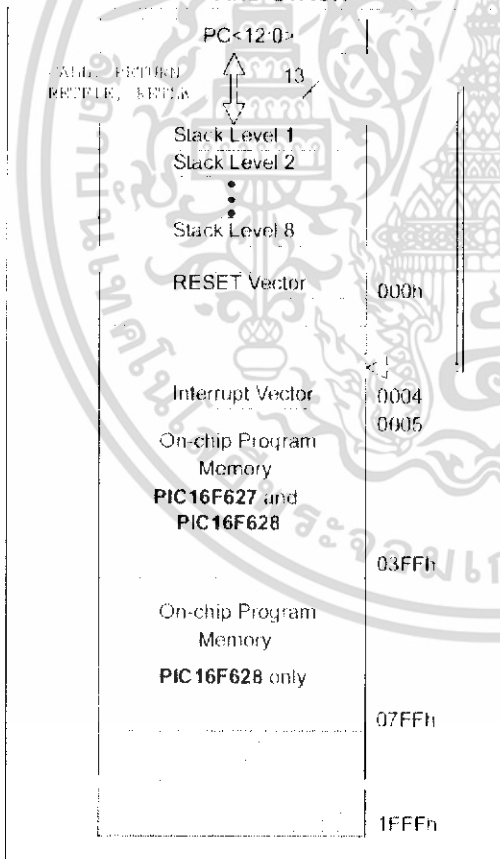
# PIC16F62X

## 3.0 MEMORY ORGANIZATION

### 3.1 Program Memory Organization

The PIC16F62X has a 13-bit program counter capable of addressing an 8K x 14 program memory space. Only the first 1K x 14 (0000h - 03FFh) for the PIC16F627 and 2K x 14 (0000h - 07FFh) for the PIC16F628 are physically implemented. Accessing a location above these boundaries will cause a wrap-around within the first 1K x 14 space (PIC16F627) or 2K x 14 space (PIC16F628). The RESET vector is at 0000h and the interrupt vector is at 0004h (Figure 3-1).

**FIGURE 3-1: PROGRAM MEMORY MAP AND STACK**



### 3.2 Data Memory Organization

The data memory (Figure 3-2) is partitioned into four banks, which contain the general purpose registers and the Special Function Registers (SFR). The SFRs are located in the first 32 locations of each Bank. Register locations 20-7Fh, A0h-FFh, 120h-14Fh, 170h-17Fh and 1F0h-1FFh are general purpose registers implemented as static RAM.

The Table below lists how to access the four banks of registers:

	RP1	RP0
Bank0	0	0
Bank1	0	1
Bank2	1	0
Bank3	1	1

Addresses F0h-FFh, 170h-17Fh and 1F0h-1FFh are implemented as common RAM and mapped back to addresses 70h-7Fh.

#### 3.2.1 GENERAL PURPOSE REGISTER FILE

The register file is organized as 224 x 8 in the PIC16F62X. Each is accessed either directly or indirectly through the File Select Register FSR (See Section 3.4).

# PIC16F62X

## 3.2.2.1 STATUS Register

The STATUS register, shown in Register 3-1, contains the arithmetic status of the ALU, the RESET status and the bank select bits for data memory (SRAM).

The STATUS register can be the destination for any instruction, like any other register. If the STATUS register is the destination for an instruction that affects the Z, DC or C bits, then the write to these three bits is disabled. These bits are set or cleared according to the device logic. Furthermore, the TO and PD bits are not writable. Therefore, the result of an instruction with the STATUS register as destination may be different than intended.

For example, `CLRF STATUS` will clear the upper-three bits and set the Z bit. This leaves the STATUS register as `00001111` (where n = unchanged).

It is recommended, therefore, that only `BCF`, `BSF`, `SWAPF` and `MOVWF` instructions are used to alter the STATUS register because these instructions do not affect any STATUS bit. For other instructions, not affecting any STATUS bits, see the "Instruction Set Summary".

**Note 1:** The C and DC bits operate as a Borrow and Digit Borrow out bit, respectively, in subtraction. See the `SUBLW` and `SUBWF` instructions for examples.

**REGISTER 3-1: STATUS REGISTER (ADDRESS: 03h, 83h, 103h, 183h)**

	R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
	IRP	RP1	RP0	TO	PD	Z	DC	C
bit 7								
bit 7	<b>IRP:</b> Register Bank Select bit (used for indirect addressing) 1 = Bank 2, 3 (100h - 1FFh) 0 = Bank 0, 1 (00h - FFh)							
bit 6-5	<b>RP1:RP0:</b> Register Bank Select bits (used for direct addressing) 00 = Bank 0 (00h - 7Fh) 01 = Bank 1 (80h - FFh) 10 = Bank 2 (100h - 17Fh) 11 = Bank 3 (180h - 1FFh)							
bit 4	<b>TO:</b> Timeout bit 1 = After power-up, <code>CLRWDT</code> instruction, or <code>SLEEP</code> instruction 0 = A WDT timeout occurred							
bit 3	<b>PD:</b> Power-down bit 1 = After power-up or by the <code>CLRWDT</code> instruction 0 = By execution of the <code>SLEEP</code> instruction							
bit 2	<b>Z:</b> Zero bit 1 = The result of an arithmetic or logic operation is zero 0 = The result of an arithmetic or logic operation is not zero							
bit 1	<b>DC:</b> Digit carry/borrow bit ( <code>ADDWF</code> , <code>ADDLW</code> , <code>SUBLW</code> , <code>SUBWF</code> instructions) (for borrow the polarity is reversed) 1 = A carry-out from the 4th low order bit of the result occurred 0 = No carry-out from the 4th low order bit of the result							
bit 0	<b>C:</b> Carry/borrow bit ( <code>ADDWF</code> , <code>ADDLW</code> , <code>SUBLW</code> , <code>SUBWF</code> instructions) 1 = A carry-out from the Most Significant bit of the result occurred 0 = No carry-out from the Most Significant bit of the result occurred							

**Note 1:** For borrow the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (`RRF`, `RLF`) instructions, this bit is loaded with either the high or low order bit of the source register.

Legend:  
 R = Readable bit      W = Writable bit      U = Unimplemented bit, read as '0'  
 -n = Value at POR      '1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## PIC16F62X

### 3.2.2.2 OPTION Register

The OPTION register is a readable and writable register which contains various control bits to configure the TMR0/WDT prescaler, the external RB0/INT interrupt, TMR0, and the weak pull-ups on PORTB.

**Note:** To achieve a 1:1 prescaler assignment for TMR0, assign the prescaler to the WDT (PSA = 1). See Section 6.3.1.

#### REGISTER 3-2: OPTION REGISTER (ADDRESS: 81h, 181h)

	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
	RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7								bit 0
bit 7	<b>RBPU:</b> PORTB Pull-up Enable bit 1 = PORTB pull-ups are disabled 0 = PORTB pull-ups are enabled by individual port latch values							
bit 6	<b>INTEDG:</b> Interrupt Edge Select bit 1 = Interrupt on rising edge of RB0/INT pin 0 = Interrupt on falling edge of RB0/INT pin							
bit 5	<b>T0CS:</b> TMR0 Clock Source Select bit 1 = Transition on RA4/T0CKI pin 0 = Internal instruction cycle clock (CLKOUT)							
bit 4	<b>T0SE:</b> TMR0 Source Edge Select bit 1 = Increment on high-to-low transition on RA4/T0CKI pin 0 = Increment on low-to-high transition on RA4/T0CKI pin							
bit 3	<b>PSA:</b> Prescaler Assignment bit 1 = Prescaler is assigned to the WDT 0 = Prescaler is assigned to the Timer0 module							
bit 2-0	<b>PS2:PS0:</b> Prescaler Rate Select bits							
	Bit Value	TMR0 Rate	WDT Rate					
	000	1:2	1:1					
	001	1:4	1:2					
	010	1:8	1:4					
	011	1:16	1:8					
	100	1:32	1:16					
	101	1:64	1:32					
	110	1:128	1:64					
	111	1:256	1:128					

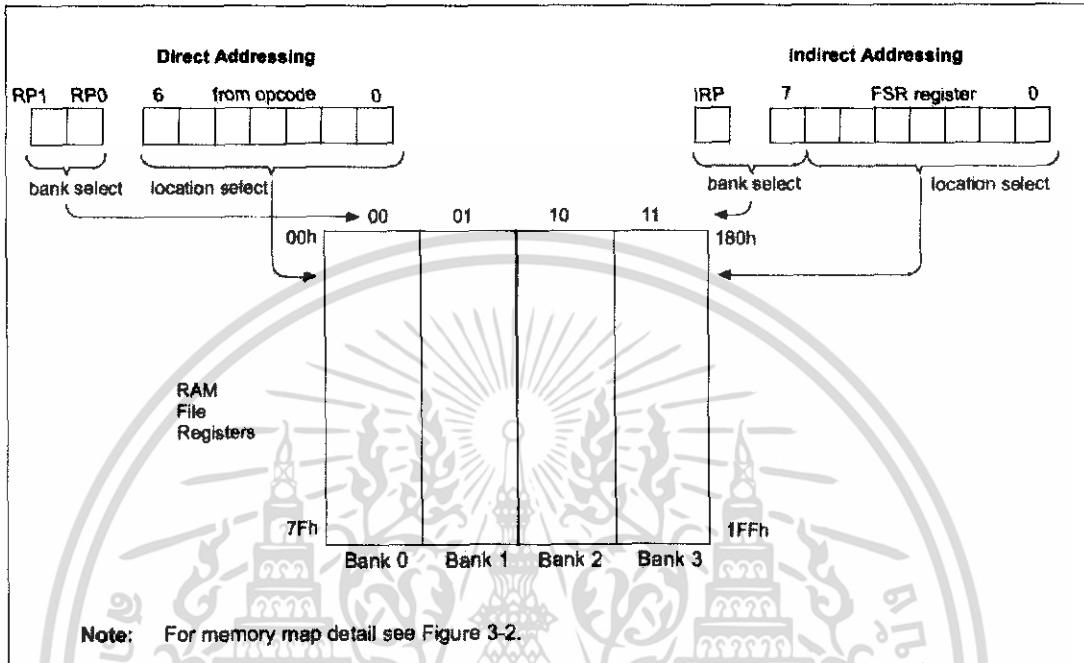
#### Legend:

R = Readable bit      W = Writable bit      U = Unimplemented bit, read as '0'  
-n = Value at POR      '1' = Bit is set      '0' = Bit is cleared      x = Bit is unknown

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

# PIC16F62X

FIGURE 3-4: DIRECT/INDIRECT ADDRESSING PIC16F62X



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## PIC16F62X

### 3.2.2.5 PIR1 Register

This register contains interrupt flag bits.

**Note:** Interrupt flag bits get set when an interrupt condition occurs regardless of the state of its corresponding enable bit or the global enable bit, GIE (INTCON<7>). User software should ensure the appropriate interrupt flag bits are clear prior to enabling an interrupt.

#### REGISTER 3-5: PIR1 REGISTER (ADDRESS: 0Ch)

	R/W-0	R/W-0	R-0	R-0	U-0	R/W-0	R/W-0	R/W-0
	EEIF	CMIF	RCIF	TXIF	—	CCP1IF	TMR2IF	TMR1IF
	bit 7							bit 0
bit 7	<b>EEIF:</b> EEPROM Write Operation Interrupt Flag bit 1 = The write operation completed (must be cleared in software) 0 = The write operation has not completed or has not been started							
bit 6	<b>CMIF:</b> Comparator Interrupt Flag bit 1 = Comparator output has changed 0 = Comparator output has not changed							
bit 5	<b>RCIF:</b> USART Receive Interrupt Flag bit 1 = The USART receive buffer is full 0 = The USART receive buffer is empty							
bit 4	<b>TXIF:</b> USART Transmit Interrupt Flag bit 1 = The USART transmit buffer is empty 0 = The USART transmit buffer is full							
bit 3	<b>Unimplemented:</b> Read as 0							
bit 2	<b>CCP1IF:</b> CCP1 Interrupt Flag bit <u>Capture Mode</u> 1 = A TMR1 register capture occurred (must be cleared in software) 0 = No TMR1 register capture occurred <u>Compare Mode</u> 1 = A TMR1 register compare match occurred (must be cleared in software) 0 = No TMR1 register compare match occurred <u>PWM Mode</u> Unused in this mode							
bit 1	<b>TMR2IF:</b> TMR2 to PR2 Match Interrupt Flag bit 1 = TMR2 to PR2 match occurred (must be cleared in software) 0 = No TMR2 to PR2 match occurred							
bit 0	<b>TMR1IF:</b> TMR1 Overflow Interrupt Flag bit 1 = TMR1 register overflowed (must be cleared in software) 0 = TMR1 register did not overflow							

#### Legend

R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
-n = Value at POR	1 = Bit is set	0 = Bit is cleared    x = Bit is unknown

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## PIC16F62X

### 3.2.2.6 PCON Register

The PCON register contains flag bits to differentiate between a Power-on Reset, an external MCLR Reset, WDT Reset or a Brown-out Detect.

**Note:** BOD is unknown on Power-on Reset. It must then be set by the user and checked on subsequent RESETS to see if BOD is cleared, indicating a brown-out has occurred. The BOD STATUS bit is a "don't care" and is not necessarily predictable if the brown-out circuit is disabled (by clearing the BODEN bit in the Configuration word).

#### REGISTER 3-6: PCON REGISTER (ADDRESS: 0Ch)

	U-0	U-0	U-0	U-0	R/W-1	U-0	R/W-q	R/W-q
	—	—	—	—	OSCF	—	POR	BOD
bit 7								bit 0
bit 7-4	<b>Unimplemented:</b> Read as '0'							
bit 3	<b>OSCF:</b> INTRC/ER oscillator frequency 1 = 4 MHz typical <sup>(1)</sup> 0 = 37 KHz typical							
bit 2	<b>Unimplemented:</b> Read as '0'							
bit 1	<b>POR:</b> Power-on Reset STATUS bit 1 = No Power-on Reset occurred 0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)							
bit 0	<b>BOD:</b> Brown-out Detect STATUS bit 1 = No Brown-out Reset occurred 0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)							
	<b>Note 1:</b> When in ER Oscillator mode, setting OSCF = 1 will cause the oscillator frequency to change to the frequency specified by the external resistor.							
	Legend							
	R = Readable bit	W = Writable bit	U = Unimplemented bit, read as 0					
	—n = Value at POR	1 = Bit is set	0 = Bit is cleared	x = Bit is unknown				

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

# PIC16F62X

## 4.0 GENERAL DESCRIPTION

The PIC16F62X are 18-Pin FLASH-based members of the versatile PIC16CXX family of low cost, high performance, CMOS, fully static, 8-bit microcontrollers.

All PICmicro<sup>®</sup> microcontrollers employ an advanced RISC architecture. The PIC16F62X have enhanced core features, eight-level deep stack, and multiple internal and external interrupt sources. The separate instruction and data buses of the Harvard architecture allow a 14-bit wide instruction word with the separate 8-bit wide data. The two-stage instruction pipeline allows all instructions to execute in a single cycle, except for program branches (which require two cycles). A total of 35 instructions (reduced instruction set) are available. Additionally, a large register set gives some of the architectural innovations used to achieve a very high performance.

PIC16F62X microcontrollers typically achieve a 2:1 code compression and a 4:1 speed improvement over other 8-bit microcontrollers in their class.

PIC16F62X devices have special features to reduce external components, thus reducing system cost, enhancing system reliability and reducing power consumption.

The PIC16F62X has eight oscillator configurations. The single pin ER oscillator provides a low cost solution. The LP oscillator minimizes power consumption. XT is a standard crystal, INTRC is a self-contained internal oscillator. The HS is for High Speed crystals. The EC mode is for an external clock source.

The SLEEP (Power-down) mode offers power savings. The user can wake-up the chip from SLEEP through several external interrupts, internal interrupts, and RESETS.

A highly reliable Watchdog Timer with its own on-chip RC oscillator provides protection against software lock-up.

Table 4-1 shows the features of the PIC16F62X mid-range microcontroller families.

A simplified block diagram of the PIC16F62X is shown in Figure 2.1.

The PIC16F62X series fits in applications ranging from battery chargers to low power remote sensors. The FLASH technology makes customization of application programs (detection levels, pulse generation, timers, etc.) extremely fast and convenient. The small footprint packages make this microcontroller series ideal for all applications with space limitations. Low cost, low power, high performance, ease of use and I/O flexibility make the PIC16F62X very versatile.

### 4.1 Development Support

The PIC16F62X family is supported by a full featured macro assembler, a software simulator, an in-circuit emulator, a low cost development programmer and a full-featured programmer. A Third Party "C" compiler support tool is also available.

TABLE 4-1: PIC16F62X FAMILY OF DEVICES

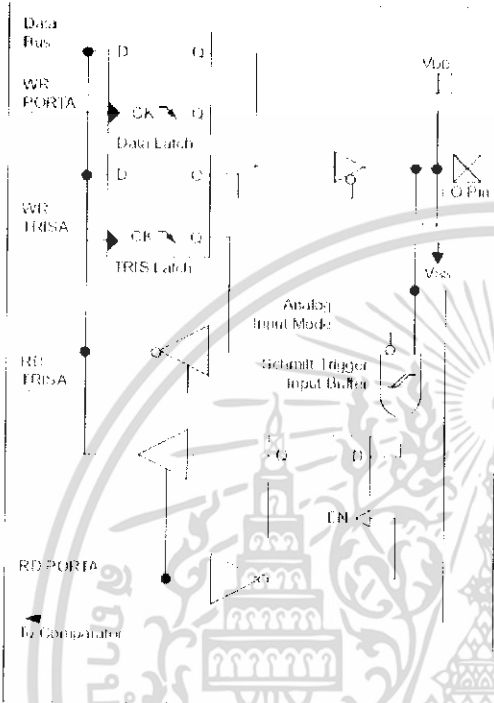
		PIC16F627	PIC16F628	PIC16LF627	PIC16LF628
Clock	Maximum Frequency of Operation (MHz)	20	20	4	4
	FLASH Program Memory (words)	1024	2048	1024	2048
Memory	RAM Data Memory (bytes)	224	224	224	224
	EEPROM Data Memory (bytes)	128	128	128	128
Peripherals	Timer Modules/Comparators	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2	TMR0, TMR1, TMR2
	Capture/Compare/PWM modules	1	2	1	2
	Serial Communications	USART	USART	USART	USART
	Internal Voltage Reference	Yes	Yes	Yes	Yes
Features	Interrupt Sources	10	10	10	10
	I/O Pins	16	16	16	16
	Voltage Range (Volts)	3.0-5.5	3.0-5.5	2.0-5.5	2.0-5.5
	Brown-out Detect	Yes	Yes	Yes	Yes
	Packages	18-pin DIP, SOIC, 20-pin SSOP	18-pin DIP, SOIC, 20-pin SSOP	16-pin DIP, SOIC, 20-pin SSOP	18-pin DIP, SOIC, 20-pin SSOP

All PICmicro<sup>®</sup> Family devices have Power-on Reset, selectable Watchdog Timer, selectable code protect, and high I/O current capability. All PIC16F62X Family devices use serial programming with clock pin RB6 and data pin RB7.

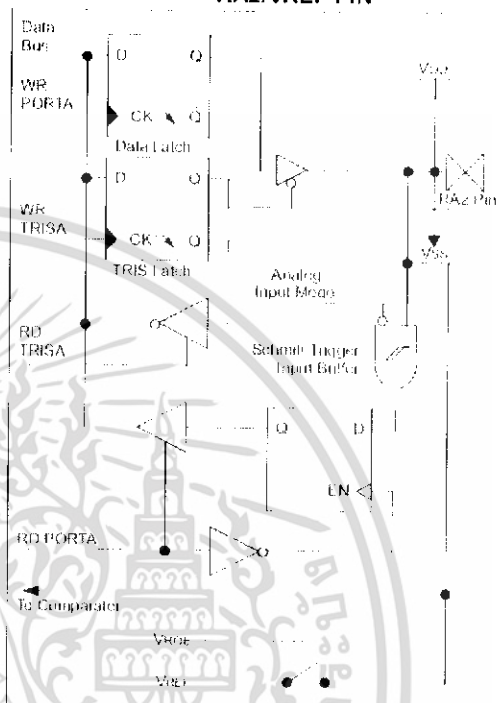
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

# PIC16F62X

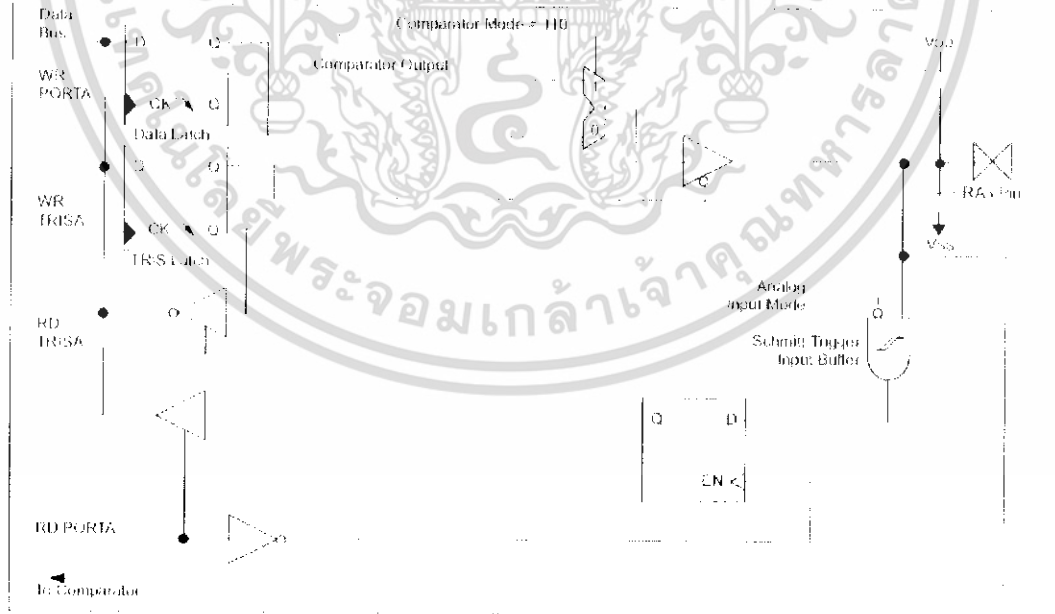
**FIGURE 5-1: BLOCK DIAGRAM OF RA0/AN0:RA1/AN1 PINS**



**FIGURE 5-2: BLOCK DIAGRAM OF RA2/VREF PIN**



**FIGURE 5-3: BLOCK DIAGRAM OF THE RA3/AN3 PIN**



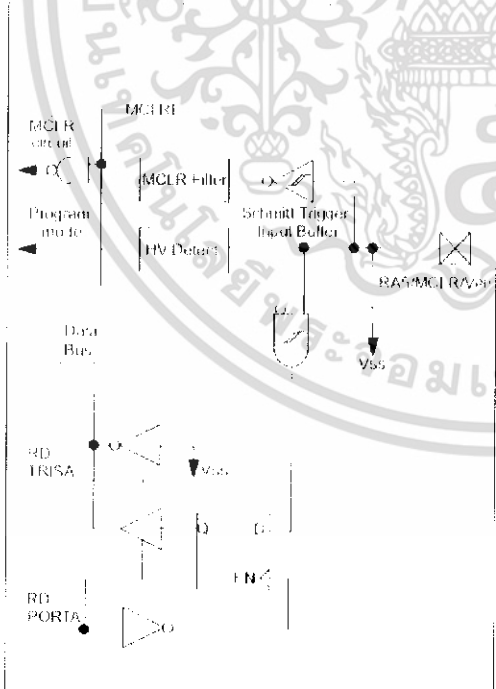
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

# PIC16F62X

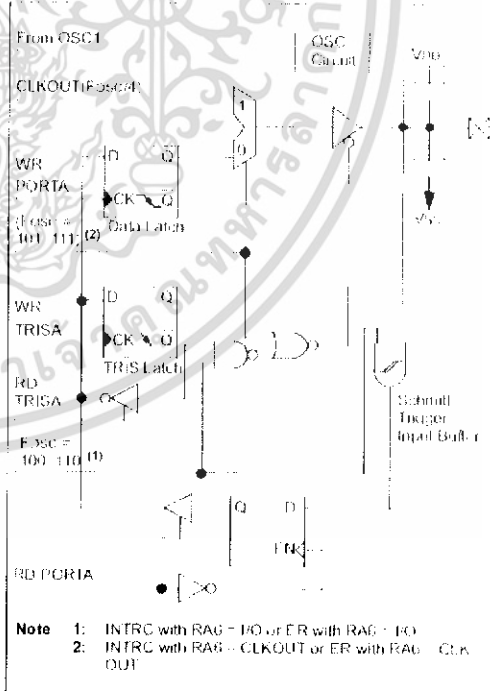
**FIGURE 5-4: BLOCK DIAGRAM OF RA4/T0CKI PIN**



**FIGURE 5-5: BLOCK DIAGRAM OF THE RA5/MCLR/VPP PIN**



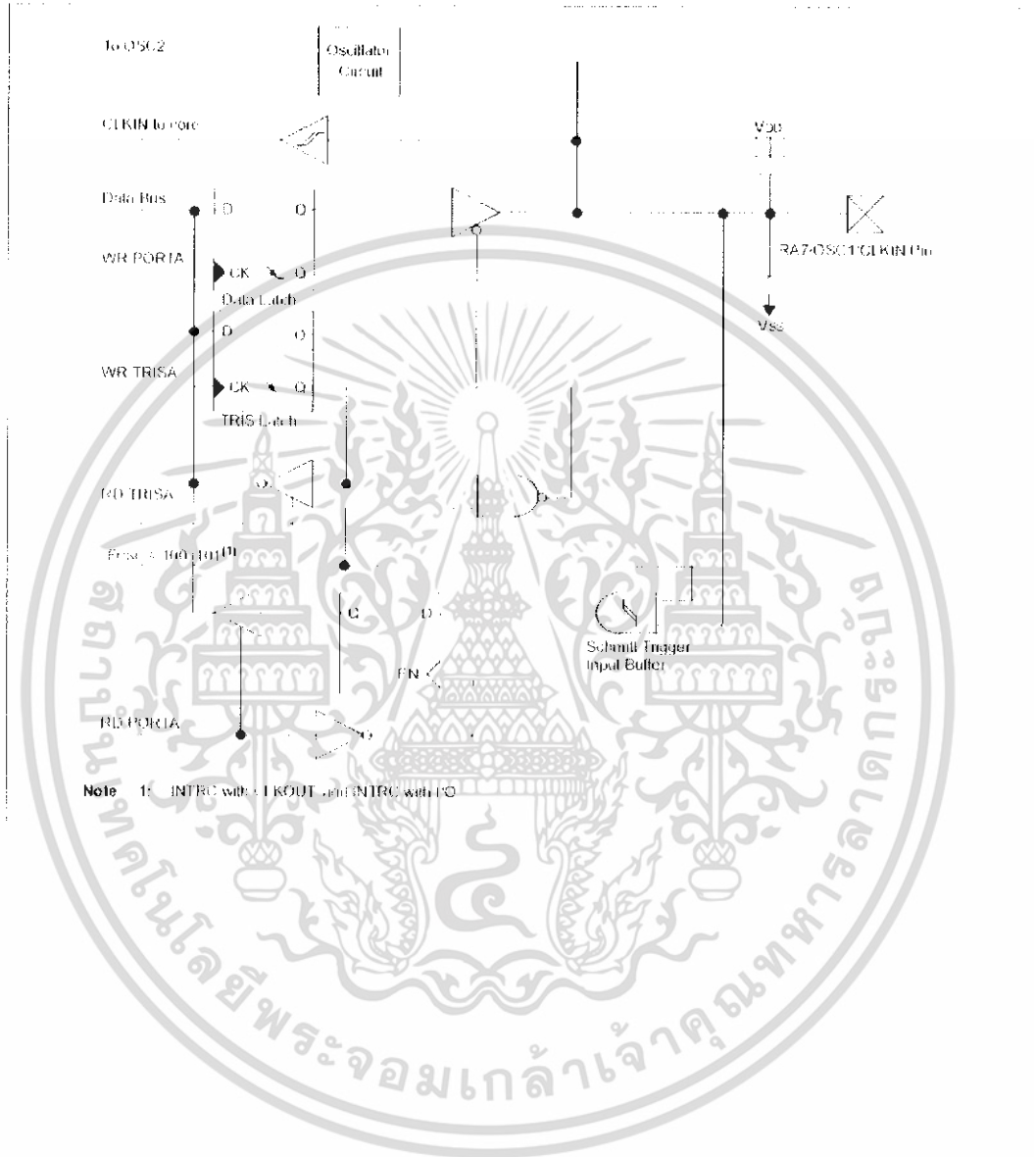
**FIGURE 5-6: BLOCK DIAGRAM OF RA6/OSC2/CLKOUT PIN**



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## PIC16F62X

FIGURE 5-7: BLOCK DIAGRAM OF RA7/OSC1/CLKIN PIN



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## PIC16F62X

TABLE 5-1: PORTA FUNCTIONS

Name	Function	Input Type	Output Type	Description
RA0/AN0	RA0	ST	CMOS	Bi-directional I/O port
	AN0	AN	—	Analog comparator input
RA1/AN1	RA1	ST	CMOS	Bi-directional I/O port
	AN1	AN	—	Analog comparator input
RA2/AN2/VREF	RA2	ST	CMOS	Bi-directional I/O port
	AN2	AN	—	Analog comparator input
	VREF	—	AN	VREF output
RA3/AN3/CMP1	RA3	ST	CMOS	Bi-directional I/O port
	AN3	AN	—	Analog comparator input
	CMP1	—	CMOS	Comparator 1 output
RA4/T0CKI/CMP2	RA4	ST	OD	Bi-directional I/O port
	T0CKI	ST	—	External clock input for TMR0 or comparator output. Output is open drain type
	CMP2	—	OD	Comparator 2 output
RA5/MCLR/VPP	RA5	ST	—	Input port
	MCLR	ST	—	Master clear
	VPP	HV	—	Programming voltage input. When configured as MCLR, this pin is an active low RESET to the device. Voltage on MCLR/VPP must not exceed VDD during normal device operation
RA6/OSC2/CLKOUT	RA6	ST	CMOS	Bi-directional I/O port.
	OSC2	XTAL	—	Oscillator crystal output. Connects to crystal resonator in Crystal Oscillator mode.
	CLKOUT	—	CMOS	In ER/INTRC mode, OSC2 pin can output CLKOUT, which has 1/4 the frequency of OSC1
RA7/OSC1/CLKIN	RA7	ST	CMOS	Bi-directional I/O port
	OSC1	XTAL	—	Oscillator crystal input
	CLKIN	ST	—	External clock source input. ER biasing pin.

Legend: ST = Schmitt Trigger input    HV = High Voltage    OD = Open Drain    AN = Analog

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## PIC16F62X

TABLE 5-2: SUMMARY OF REGISTERS ASSOCIATED WITH PORTA<sup>(1)</sup>

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR	Value on All Other RESETS
05h	PORTA	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0	xxxx 0000	xxxx 0000
85h	TRISA	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	1111 1111	1111 1111
1Fh	CMCON	C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0	0000 0000	0000 0000
9Fh	VRCON	VREN	VROE	VRR	—	VR3	VR2	VR1	VR0	000 0000	000 0000

Legend: — = Unimplemented locations, read as '0', u = unchanged, x = unknown

Note 1: Shaded bits are not used by PORTA

### 5.2 PORTB and TRISB Registers

PORTB is an 8-bit wide bi-directional port. The corresponding data direction register is TRISB. A '1' in the TRISB register puts the corresponding output driver in a Hi-impedance mode. A '0' in the TRISB register puts the contents of the output latch on the selected pin(s).

PORTB is multiplexed with the external interrupt, USART, CCP module and the TMR1 clock input/output. The standard port functions and the alternate port functions are shown in Table 5-3. Alternate port functions override TRIS setting when enabled.

Reading PORTB register reads the status of the pins, whereas writing to it will write to the port latch. All write operations are read-modify-write operations. So a write to a port implies that the port pins are first read, then this value is modified and written to the port data latch.

Each of the PORTB pins has a weak internal pull-up ( $\approx 200 \mu\text{A}$  typical). A single control bit can turn on all the pull-ups. This is done by clearing the  $\overline{\text{RBP}}\text{U}$  (OPTION<7>) bit. The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on Power-on Reset.

Four of PORTB's pins, RB<7:4>, have an interrupt-on-change feature. Only pins configured as inputs can cause this interrupt to occur (i.e., any RB<7:4> pin configured as an output is excluded from the interrupt-on-change comparison). The input pins (of RB7:RB4) are compared with the old value latched on the last read of PORTB. The "mismatch" outputs of RB7:RB4 are OR'ed together to generate the RBIF interrupt (flag latched in INTCON<0>).

This interrupt can wake the device from SLEEP. The user, in the interrupt service routine, can clear the interrupt in the following manner:

- Any read or write of PORTB. This will end the mismatch condition.
- Clear flag bit RBIF.

A mismatch condition will continue to set flag bit RBIF. Reading PORTB will end the mismatch condition and allow flag bit RBIF to be cleared.

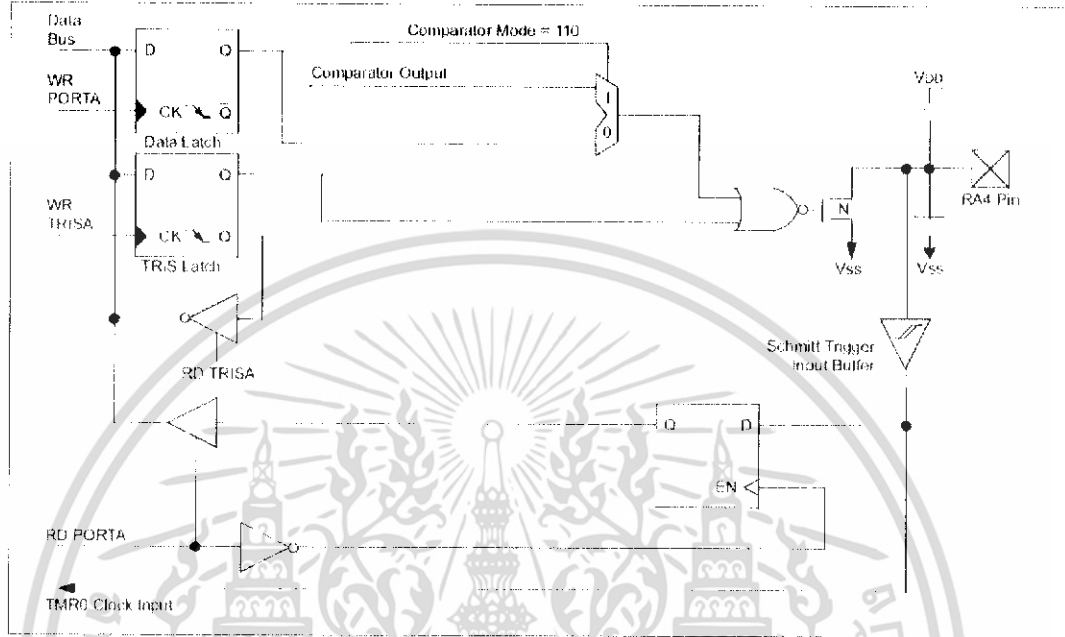
This interrupt on mismatch feature, together with software configurable pull-ups on these four pins allow easy interface to a key pad and make it possible for wake-up on key-depression. (See AN552)

**Note:** If a change on the I/O pin should occur when a read operation is being executed (start of the Q2 cycle), then the RBIF interrupt flag may not get set.

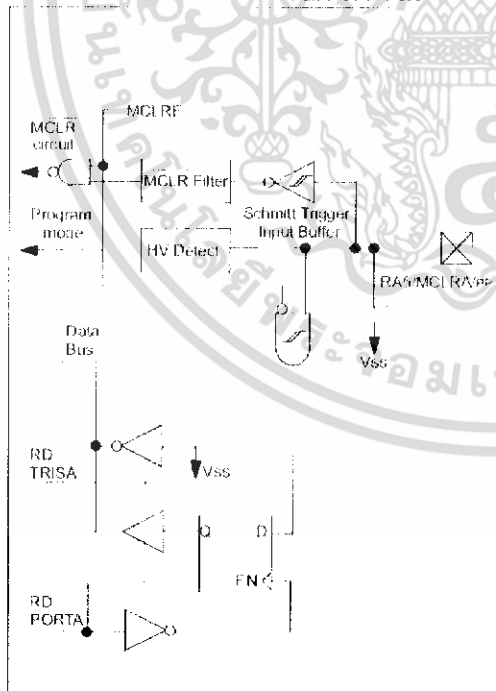
The interrupt-on-change feature is recommended for wake-up on key depression operation and operations where PORTB is only used for the interrupt-on-change feature. Polling of PORTB is not recommended while using the interrupt-on-change feature.

# PIC16F62X

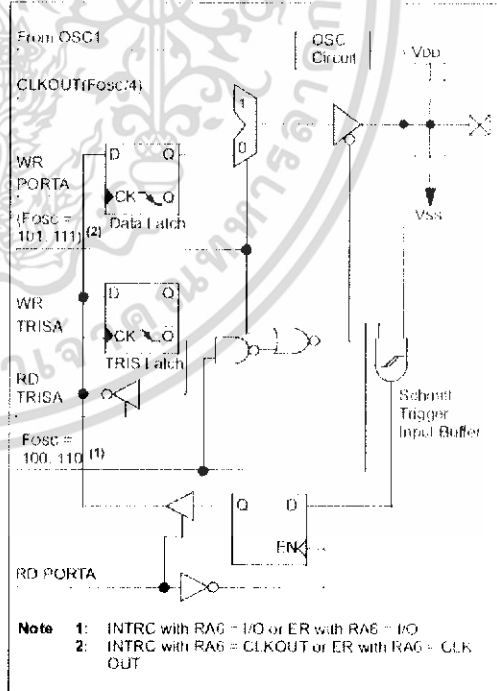
**FIGURE 5-4: BLOCK DIAGRAM OF RA4/T0CKI PIN**



**FIGURE 5-5: BLOCK DIAGRAM OF THE RA5/MCLR/VPP PIN**



**FIGURE 5-6: BLOCK DIAGRAM OF RA6/OSC2/CLKOUT PIN**



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

# PIC16F62X

FIGURE 5-10: BLOCK DIAGRAM OF RB2/TX/CK PIN

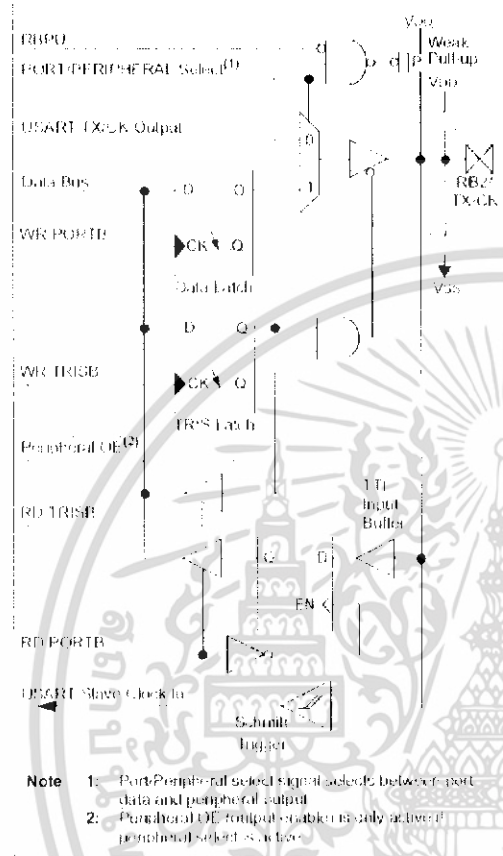
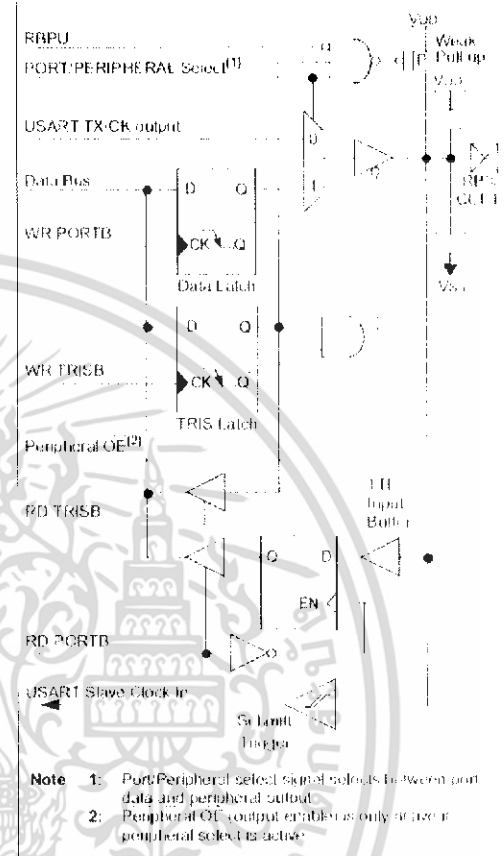


FIGURE 5-11: BLOCK DIAGRAM OF RB3/CCP1 PIN



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้





เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

#include <16F628.h>
#include <string.h>
#define HS,NOPROTECT,NOLVP,NOWDT, MCLR
#define use delay(clock=20000000)
#define use rs232(baud=9600, xmit=pin_b2,rcv=pin_b1)

#define use fast_io(A)
#define use fast_io(B)

#define bit Load1=0x6.3
#define bit Load2=0x5.1
#define bit Load3=0x5.2
#define bit Load4=0x5.3

char sbuf[4]; // buffer for serial port
int1 serialFlag=0;
char id,idx;

/* check key press */
int1 kbhit (void) {
    if ((input_b ()&0xf0)===0xf0) return (FALSE);
    else return (TRUE);
}

checkSwitch(void){

    char sw;
    sw =(input_b()&0xf0);
    while (!kbhit ());

    if (!(sw&0b00010000))

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

    { Load1^=1;
      printf("%c%d%d%c",id,1,Load1,13); }
    else if (!(sw&0b00100000))
    { Load2^=1;
      printf("%c%d%d%c",id,2,Load2,13); }
    else if (!(sw&0b01000000))
    { Load3^=1;
      printf("%c%d%d%c",id,4,Load3,13); }
    else if (!(sw&0b10000000))
    { Load4^=1;
      printf("%c%d%d%c",id,8,Load4,13); }

    delay_ms(1000);

    //while(sw!=0xf0);
}

#int_RDA
RDA_isr()
{

int c;

c=getc ();

if (serialFlag) { // serial not to decode
    idx=0;
    return;
}

sbuf[idx]=c; // keep data

if (c==13) { // return detected
    sbuf[idx]=13; // fill terminate

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

    serialFlag=1; // serial ready flag
    idx=0; // reset index
    return;
}
if (++idx>4) // increase index for next data
    idx=0;
}

```

```

void out_load(char LoadID,int1 b)
{
    if (LoadID=='1') Load1=b;
    else if (LoadID=='2') Load2=b;
    else if (LoadID=='4') Load3=b;
    else if (LoadID=='8') Load4=b;
    else if (LoadID=='9') Load1=Load2=Load3=Load4=b;

    printf("%c%c%d%C",id,LoadID,b,13);
}

```

```

void main()
{

```

```

    setup_timer_0(RTCC_INTERNALRTCC_DIV_1);
    setup_timer_1(T1_DISABLED);
    setup_timer_2(T2_DISABLED,0,1);
    setup_comparator(NC_NC_NC_NC);
    setup_vref(FALSE);
    enable_interrupts(INT_RDA);

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
enable_interrupts(GLOBAL);
```

```
set_tris_a(0x00);
```

```
set_tris_b(0xf2);
```

```
id='3';
```

```
Load1=0;
```

```
Load2=0;
```

```
Load3=0;
```

```
Load4=0;
```

```
while(true)
```

```
{
```

```
//printf(" Yao Love M. \n");
```

```
//load1=load2=load3=load4=0;
```

```
///delay_ms (1000);
```

```
//load1=load2=load3=load4=1;
```

```
//delay_ms (1000);
```

```
if (serialFlag)
```

```
{
```

```
// printf("%c%c%c%c",sbuf[0],sbuf[1],sbuf[2],sbuf[3]);
```

```
if( (sbuf[1]!='1') || (sbuf[1]!='2') || (sbuf[1]!='4') || (sbuf[1]!='8') || (sbuf[1]!='9'))
```

```
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

if ((sbuf[2]!='o') || (sbuf[2]!='c') || (sbuf[2]!='s') )
{
    if (sbuf[3]!='13')

        {
            if(sbuf[0]==id)
                {
                    if (sbuf[2]!='s') ;
                    else if (sbuf[2]!='c') out_load(sbuf[1],0);
                    else if (sbuf[2]!='o') out_load(sbuf[1],1);
                }
            else
                {
                    printf("%c%c%c%c",sbuf[0],sbuf[1],sbuf[2],sbuf[3]);
                }
        }
    serialFlag=0;
}
if (!kbdhit ()) continue; // no key press

checkSwitch();
}

/

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คู่มือการใช้งาน  
ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต



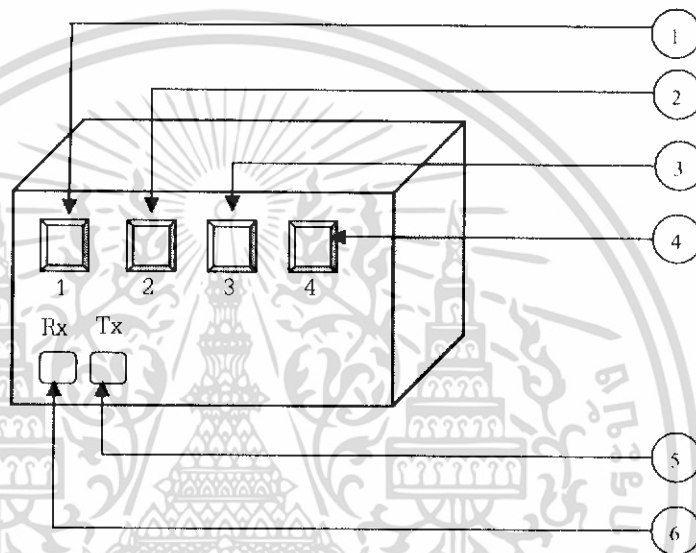
ภาควิชาครุศาสตร์วิศวกรรม  
คณะครุศาสตร์อุตสาหกรรม  
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง  
ปีการศึกษา 2548

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

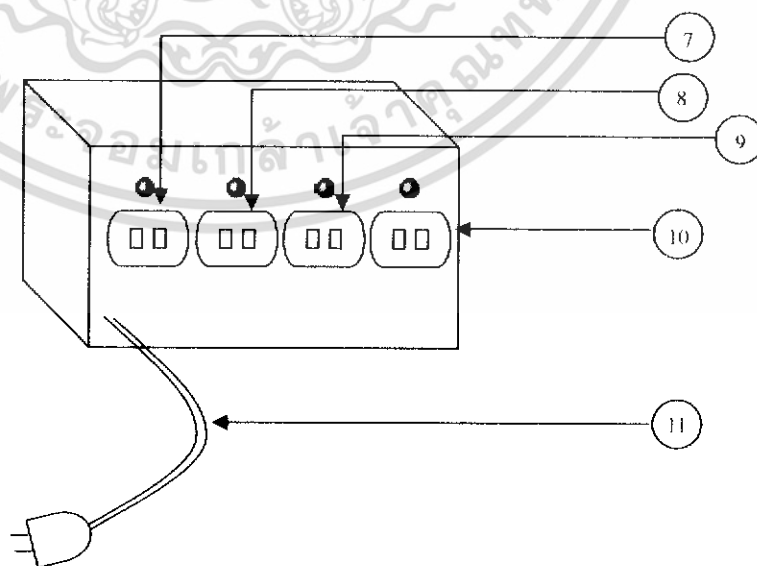
**1. คำแนะนำเบื้องต้น**

ก่อนที่จะลงมือใช้งานชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตนั้น ควรทำการศึกษาการใช้งานจากคู่มือให้เข้าใจ เพื่อการให้บริการที่ถูกต้องและเป็นการป้องกันการเสียหายที่อาจเกิดขึ้นกับชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต

**2. ส่วนประกอบและส่วนควบคุม**



รูปที่ ๑.1 ส่วนประกอบและปุ่มควบคุมของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต (ด้านหน้า)



รูปที่ ๑.2 ส่วนประกอบและปุ่มควบคุมของชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต (ด้านหลัง)  
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ ฉ.1 และ ฉ.2 มีรายละเอียดต่างๆ ดังนี้

- ① สวิตช์เปิด/ปิดอุปกรณ์ไฟฟ้าตัวที่ 1
- ② สวิตช์เปิด/ปิดอุปกรณ์ไฟฟ้าตัวที่ 2
- ③ สวิตช์เปิด/ปิดอุปกรณ์ไฟฟ้าตัวที่ 3
- ④ สวิตช์เปิด/ปิดอุปกรณ์ไฟฟ้าตัวที่ 4
- ⑤ ช่องเสียบสายโทรศัพท์ Tx
- ⑥ ช่องเสียบสายโทรศัพท์ Rx
- ⑦ ปลั๊กไฟของอุปกรณ์ไฟฟ้าตัวที่ 4
- ⑧ ปลั๊กไฟของอุปกรณ์ไฟฟ้าตัวที่ 3
- ⑨ ปลั๊กไฟของอุปกรณ์ไฟฟ้าตัวที่ 2
- ⑩ ปลั๊กไฟของอุปกรณ์ไฟฟ้าตัวที่ 1
- ⑪ สายไฟฟ้ากระแสสลับ 220 โวลต์

### 3. การติดตั้งและการใช้งาน

3.1 นำสายไฟฟ้าเสียบเข้ากับปลั๊กไฟกระแสสลับ 220 โวลต์ (หมายเลข 11)

3.2 เปิดโปรแกรม Hyper Terminal ขึ้นมา โดยกำหนดชื่อการเชื่อมต่อเป็น ED26 , Connection Using เป็น Com1 หลังจากนั้นกำหนดค่าบอรรถเรตไว้ที่ 9600 จากนั้นจะขึ้นโปรแกรมดังภาพ



รูปที่ ฉ.3 หน้าหลักโปรแกรม Terminal ที่ใช้ในการเชื่อมต่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 หลังจากที่เปิดโปรแกรมขึ้นมาแล้ว เราจะสามารถส่งข้อมูลจากคอมพิวเตอร์ผ่านไปยังชุดควบคุมอุปกรณ์โดย การใส่ชุดคำสั่ง 3 คำ

ตัวที่ 1 คือ รหัสโมดูล (มีอยู่ 4 คำ คือ 1, 2, 3, 4)

- 1 คือ โมดูลตัวที่ 1
- 2 คือ โมดูลตัวที่ 2
- 3 คือ โมดูลตัวที่ 3
- 4 คือ โมดูลตัวที่ 4

ตัวที่ 2 คือ รหัสของอุปกรณ์ไฟฟ้าผ่านในโมดูล (มีอยู่ 5 คำ คือ 1, 2, 4, 8, 9)

- 1 คือ อุปกรณ์ไฟฟ้าตัวที่ 1
- 2 คือ อุปกรณ์ไฟฟ้าตัวที่ 2
- 4 คือ อุปกรณ์ไฟฟ้าตัวที่ 3
- 8 คือ อุปกรณ์ไฟฟ้าตัวที่ 4
- 9 คือ อุปกรณ์ไฟฟ้าทุกตัวในโมดูล

ตัวที่ 3 คือ รหัสของคำสั่ง (มี 2 คำ คือ C และ O)

- C คือ การสั่งให้อุปกรณ์ไฟฟ้าเปิด
- O คือ การสั่งให้อุปกรณ์ไฟฟ้าปิด

3.4 การควบคุมโดยการกดสวิตช์ที่ชุดควบคุมอุปกรณ์ สามารถทำได้โดย กดสวิตช์ที่ระบุการต่อพ่วงกับอุปกรณ์ไฟฟ้าตัวนั้นๆ ก็จะสามารถเปิด/ปิด อุปกรณ์ไฟฟ้าที่เราต้องการได้เช่นเดียวกัน

#### หมายเหตุ

1. หากการพิมพ์คำสั่งไม่ได้เป็นไปตามเงื่อนไขที่กำหนด ชุดคำสั่งนั้นจะถูกโยนทิ้งไป และจะไม่มีเหตุการณ์ใดๆ เกิดขึ้นกับอุปกรณ์ไฟฟ้าตัวนั้นๆ และอุปกรณ์ไฟฟ้าก็จะรอชุดคำสั่งที่ถูกต้องต่อไป
2. สามารถควบคุมอุปกรณ์ไฟฟ้าได้ 4 ชนิดในแต่ละโมดูล

#### 4. การแก้ปัญหาเบื้องต้น

เมื่อท่านประสบปัญหาในการใช้งานชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ต สามารถตรวจสอบแนวทางในการแก้ไขปัญหาเบื้องต้นได้จากตารางข้างล่างนี้

### ตารางที่ ๑.1 การแก้ปัญหาเบื้องต้น

อาการ	สาเหตุและ/หรือวิธีแก้ไข
ชุดควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเทอร์เน็ตไม่ทำงาน	ตรวจสอบว่าได้เสียบสายไฟฟ้าเข้ากับปลั๊กไฟหรือไม่
อุปกรณ์ไฟฟ้าที่ต้องการควบคุมไม่ทำงาน	ตรวจสอบหัวต่อจากพอร์ต RS-232 ของคอมพิวเตอร์ว่าต่ออยู่กับพอร์ต Tx และ Rx ของโมดูลชุดแม่ข่ายหรือไม่

## 5. การดูแลรักษาและข้อควรระวัง

### 5.1 การดูแลรักษา

1. เช็ดทำความสะอาดตัวเครื่องด้วยผ้านุ่ม อย่าใช้สารใดๆ ที่เป็นตัวทำละลายเพราะอาจจะทำให้ตัวเครื่องเป็นรอยและเสียหายได้
2. ตรวจสอบขั้วต่อสายไฟของวงจรภายในเครื่องให้อยู่ในสภาพพร้อมใช้งานอยู่เสมอ
3. ควรมีการซ่อมบำรุงตัวเครื่องเป็นระยะเพื่อป้องกันและลดอัตราการเสื่อมสภาพของชุดควบคุมอุปกรณ์ไฟฟ้า เพื่อให้การใช้งานตัวเครื่องเป็นไปอย่างมีประสิทธิภาพ

### 5.2 ข้อควรระวัง

1. ควรศึกษาคู่มือการใช้งานของเครื่องก่อนการใช้งานตัวเครื่อง
2. เครื่องจะสร้างสนามแม่เหล็กที่มีความเข้มข้นสูง จึงไม่ควรนำโทรศัพท์วางบนเครื่องหรือนำบัตรที่มีแถบแม่เหล็กเข้าไปใกล้บริเวณที่ติดตั้งชุดควบคุมอุปกรณ์ไฟฟ้า
3. การเคลื่อนย้ายควรระมัดระวังอย่าให้มีการกระแทก เพื่อป้องกันความเสียหายของระบบกลไกต่างๆ ภายในชุดควบคุม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## ประวัติผู้แต่ง



ชื่อ-สกุล	นายณัฐพร ศรีโน
วัน เดือน ปีเกิด	วันที่ 9 มกราคม 2527
ภูมิลำเนา	42 หมู่ 9 ตำบลสันกลาง อำเภอสันป่าตอง จังหวัดเชียงใหม่ 50120
ประวัติการศึกษา	โรงเรียนบ้านห้วยส้ม จังหวัดเชียงใหม่ โรงเรียนน้ำบ่อหลวงวิทยาคม จังหวัดเชียงใหม่ วิทยาลัยเทคนิคเชียงใหม่ จังหวัดเชียงใหม่ วิทยาลัยเทคนิคเชียงใหม่ จังหวัดเชียงใหม่ สาขาวิชาคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สจล.
ผลงานที่ได้รับรางวัล	ได้ที่ 4 ระดับภาค ที่ 3 ระดับประเทศในการแข่งขันหุ่นยนต์ของ กรมอาชีวศึกษา "ตำนานสะพานแห่งความรัก"
ความสนใจพิเศษ	อินเทอร์เน็ต, ฟุตบอล, คอมพิวเตอร์
คติพจน์	ไม่มีคำว่าสิ้นหวัง ถ้ายังมีลมหายใจ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## ประวัติผู้แต่ง



ชื่อ-สกุล	นางสาวธนพร พรหมผุย
วัน เดือน ปีเกิด	วันศุกร์ที่ 30 ธันวาคม 2526
ภูมิลำเนา	16 หมู่ 20 ถ.เสี้ยวเมือง ต.ขามใหญ่ อ.เมือง จังหวัดอุบลราชธานี 34000
ประวัติการศึกษา	<p>ประถมศึกษา โรงเรียนอนุบาลวิทยาคม จังหวัดอุบลราชธานี</p> <p>มัธยมศึกษาตอนต้น โรงเรียนนารีนุกูล จังหวัดอุบลราชธานี</p> <p>มัธยมศึกษาตอนปลาย โรงเรียนนารีนุกูล จังหวัดอุบลราชธานี</p> <p>ประกาศนียบัตรวิชาชีพชั้นสูง วิทยาลัยเทคนิคอุบลราชธานี จังหวัดอุบลราชธานี</p> <p>ปริญญาตรี สาขาวิชาคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สจล.</p>
ทุนการศึกษา	ทุนนักเรียนดีเด่นประจำปีการศึกษา 2546 วิทยาลัยเทคนิคอุบลราชธานี
ความสนใจพิเศษ	อ่านหนังสือ เล่นอินเทอร์เน็ต
คติพจน์	ไม่มีคำว่าทำไม่ได้ในสิ่งที่ยังไม่ได้ทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## ประวัติผู้แต่ง



ชื่อ-สกุล	นายนิรันดร์ มะ
วัน เดือน ปีเกิด	วันที่ 4 มกราคม 2527
ภูมิลำเนา	36/22 ถ.แสงจันทร์ ต.บางนาค อ.เมือง จังหวัดนราธิวาส 96000
ประวัติการศึกษา	
ประถมศึกษา	โรงเรียนพินานวิทยายนราธิวาส จังหวัดนราธิวาส
มัธยมศึกษาตอนต้น	โรงเรียนนราธิวาส จังหวัดนราธิวาส
ประกาศนียบัตรวิชาชีพ	วิทยาลัยเทคนิคนราธิวาส จังหวัดนราธิวาส
ประกาศนียบัตรวิชาชีพชั้นสูง	วิทยาลัยเทคนิคนราธิวาส จังหวัดนราธิวาส
ปริญญาตรี	สาขาวิชาคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สจล.
ผลงานที่ได้รับรางวัล	รองชนะเลิศอันดับสามสิ่งประดิษฐ์ของคนรุ่นใหม่ ระดับภาคใต้
ความสนใจพิเศษ	ซ่อมคอมพิวเตอร์ ทีวี เล่นฟุตบอล เล่นดนตรี
คติพจน์	ความสำเร็จของคุณ คือความสุขของพ่อแม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

## ประวัติผู้แต่ง



ชื่อ-สกุล	นายวิจิต วรรณเขต
วัน เดือน ปีเกิด	วันที่ 13 กรกฎาคม 2526
ภูมิลำเนา	373/1 ม1. ต.ควนลิ่ง อ.หาดใหญ่ จังหวัดสงขลา 90110
ประวัติการศึกษา	
ประถมศึกษา	โรงเรียนพลวิทยา จังหวัดสงขลา
มัธยมศึกษาตอนต้น	โรงเรียนหาดใหญ่วิทยาลัยสมบูรณ์กุลกันยา จังหวัดสงขลา
ประกาศนียบัตรวิชาชีพ	วิทยาลัยเทคนิคหาดใหญ่ จังหวัดสงขลา
ประกาศนียบัตรวิชาชีพชั้นสูง	วิทยาลัยเทคนิคหาดใหญ่ จังหวัดสงขลา
ปริญญาตรี	สาขาวิชาคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สจล.
ผลงานที่ได้รับรางวัล	ชนะเลิศการแข่งขันการเขียนโปรแกรมไมโครคอนโทรลเลอร์ PIC ของกรมอาชีวศึกษา ระดับภาคใต้
ทุนการศึกษา	ทุนยกเว้นหน่วยกิต, ทุนผู้ปกครองรายได้น้อย
ความสนใจพิเศษ	เขียนโปรแกรมคอมพิวเตอร์
คติพจน์	ความพยายามอยู่ที่ไหนความสำเร็จอยู่ที่นั่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า  
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้