ระบบควบคุมไฟส่องสว่าง

LIGHT CONTROL SYSTEM

ปิยรัตน์ พรหมโชติ

โรสลีต้า หมาดสกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2556

ระบบควบคุมไฟส่องสว่าง

LIGHT CONTROL SYSTEM

ปิยรัตน์ พรหมโชติ

โรสลีต้า หมาดสกุล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมการวัดคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2556

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

LIGHT CONTROL SYSTEM

PIYARAT PROMCHOT

ROSELITA MADSAKUL

A THESIS SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENT FOR THE DEGREE OF

BACHELOR OF ENGINEERING IN INSTRUMENTATION ENGINEERRING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUE OF TECHNOLOGY LARDKRABANG

ACADEMIC YEAR 2013

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของผลงานทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2556

ภาควิชาวิศวกรรมการวัดคุม คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

.....

หัวข้อปริญญานิพนธ์	ระบบควบคุมไฟส่องสว่าง LIGHT CONTROL SYSTEM	
นักศึกษาผู้จัดทำ	นางสาวปิยรัตน์ พรหมโชติ	**รหัสนักศึกษา** 53011992
	นางสาวโรสลีต้า หมาดสกุล	**รหัสนักศึกษา** 53011373
ปริญญา	วิศวกรรมศาสตรบัณฑิต	
สาขาวิชา	วิศวกรรมการวัดคุม	
ปีการศึกษา	2556	

อาจารย์ผู้ควบคุมปริญญานิพนธ์	**ลายมือชื่อ**
อาจารย์สุธรรม สัทธรรมสกุล	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อปริญญานิพนธ์	ระบบควบคุมไฟส่องสว่าง LIGHT CONTROL SYSTEM	
นักศึกษาผู้จัดทำ	นางสาวปิยรัตน์ พรหมโชติ	รหัสนักศึกษา 53011992
	นางสาวโรสลีต้า หมาดสกุล	รหัสนักศึกษา 53011373
อาจารย์ที่ปรึกษา	อาจารย์สุธรรม สัทธรรมสกุล	
ปีการศึกษา	2556	

บทคัดย่อ

โครงงานนี้นำเสนอระบบควบคุมไฟส่องสว่างที่ใช้ไมโครคอนโทรลเลอร์ตระกูล PIC18F458 เป็นตัวควบคุมการเปิด-ปิดไฟส่องสว่างผ่านทางสวิทช์ ค่าสถานะของไฟส่องสว่างจะถูกส่งจากอุปกรณ์ผ่าน CAN BUS ไปยัง SNAP โดยการแสดงสถานะสามารถทำได้ผ่านทางเว็บบราวเซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Thesis Title	LIGHT CONTROL SYSTEM
Authors	Miss Piyarat Promchot
	Miss Roselita Madsakul
Thesis Advisor	Mr. Sutham Satthamsakul
Year	2012

ABSTRACT

This project was presented the light control system by using microcontroller (PIC18F458) with opening and closing of light controlled via switch. The status of light from the device transfer to CAN BUS and SNAP which the status of light can show on browser.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยดี ทางคณะผู้จัดทำขอขอบพระคุณอาจารย์ที่ปรึกษา อาจารย์สุธรรม สัทธรรมสกุลและอาจารย์ประจำภาควิชาวิศวกรรมการวัดคุมทุกท่านที่ได้ให้คำปรึกษา ให้ความเมตตาและคำแนะนำแก่ผู้จัดทำอีกทั้งยังเอื้อเฟื้ออุปกรณ์และเครื่องมือ ในการจัดทำปริญญานิพนธ์นี้ และขอกราบขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณอาจารย์ประจำภาควิชาวิศวกรรมการวัดคุมทุกท่าน ที่ให้คำปรึกษาให้ความเมตตาและคำแนะนำแก่ผู้จัดทำ และช่วยเหลืออันเป็นประโยชน์ต่อการทำปริญญานิพนธ์ฉบับนี้ที่สำคัญขอกราบขอบพระคุณคุณพ่อและคุณแม่อันเป็นที่เคารพอย่างยิ่ง ที่สนับสนุนในการทำปริญญานิพนธ์ฉบับนี้

ความรู้และประโยชน์ที่ได้จากปริญญานิพนธ์ฉบับนี้ ผู้จัดทำขอมอบแด่ผู้มีพระคุณทุกท่าน

คณะผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

หน้า

บทคัดย่อภาษาไทย.....I

บทคัดย่อภาษาอังกฤษ.....II

กิตติกรรมประกาศ.....III

สารบัญ.....IV

สารบัญตาราง.....VI

สารบัญภาพ.....VII

บทที่ 1 บทนำ.....1

1.1 ความสำคัญของปริญญานิพนธ์.....1

1.2 วัตถุประสงค์ของปริญญานิพนธ์.....1

1.3 ขอบเขตของปริญญานิพนธ์.....1

1.4 ขั้นตอนการศึกษา.....1

บทที่ 2 ทฤษฎี.....2

2.1 ไมโครคอนโทรลเลอร์ตระกูล PIC.....2

2.1.1 คุณสมบัติของไมโครคอนโทรลเลอร์ตระกูลPICในแต่ละกลุ่ม.....3

2.2 ET-CAN DRIVER.....4

2.2.1 คุณสมบัติของบอร์ด ET-CAN DRIVER.....4

2.2.2 โครงสร้างบอร์ด ET-CAN DRIVER.....5

2.2.3 การเลือก Node สำหรับ CAN BUS.....5

2.3 Control Area Network (CAN).....7

2.3.1 ประวัติของ Control Area Network (CAN).....7

2.3.2 โครงสร้างของ Control Area Network (CAN).....8

2.3.3 รูปแบบของ CAN (Control Area Network).....9

2.3.3.1 คำอธิบาย CAN2.0A.....9

2.3.3.2 คำอธิบาย CAN2.0B.....9

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

2.3.4 ข้อแตกต่างระหว่าง CAN 2.0A และ CAN 2.0B....10
2.3.5 หลักการทำงานของ CAN....10
2.3.5.1 การเข้ารหัสบิต (Bit Encoding)....11
2.3.5.2 ความมีเสถียรภาพในการทำงาน....11
2.3.5.3 ความยืดหยุ่นของระบบเครือข่าย(Network Flexibility and Expansion)....11
2.3.5.4 การจัดการลำดับข้อความ....11
2.3.5.5 การหยุดพักหรือการเริ่มต้นทำงานใหม่ (Sleep Mode / Wake-up)....12
2.3.5.6 รูปแบบของการส่งข้อความของ CAN (Control Area Network)....12
2.3.6 ข้อผิดพลาดที่เกิดขึ้น....12
2.3.6.1 การตรวจจับข้อผิดพลาด (Error Detection)....12
2.3.6.2 ขอบเขตของข้อผิดพลาดที่เกิดขึ้น (Error Confinement)....13
2.3.6.3 การจัดการเมื่อตรวจพบข้อผิดพลาด (Performance of Error Detection)....13
2.3.7 ข้อดีของ CAN (Control Area Network)....14
2.4 SNAP (Simple Network Application Platform)....14
2.4.1 Baseboard....14
2.4.2 คุณสมบัติของ SNAP (Simple Network Application Platform)....16
2.4.2.1 คุณสมบัติทางด้าน Hardware....16
2.4.2.2 สำหรับอุปกรณ์ SNAP มีรายละเอียดของส่วนประกอบและการเชื่อมต่อ....16
2.4.3 การประยุกต์ใช้งานของ SNAP (Simple Network Application Platform)....17
2.5 J2ME (Java 2 Platform, Micro Edition)....17
2.5.1 โครงสร้างสถาปัตยกรรมของ J2ME....17
2.5.1.1 การตั้งค่า (Configurations)....18
2.5.1.2 Profiles....19
2.5.2 ประโยชน์ของ J2ME ในการพัฒนาแอพพลิเคชั่นบนอุปกรณ์ไร้สาย....21
2.6 การสร้างเว็บเพจด้วยภาษา HTML....22
2.6.1 ขั้นตอนในการสร้างเว็บเพจ....22
2.6.2 โครงสร้างภาษา HTML....22

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ (ต่อ)

หน้า

2.6.3 การกำหนดโครงสร้างหลัก....23
2.6.4 โครงสร้างคำสั่งของ HTML....23
2.6.5 โครงสร้างหลักของ HTML....24
2.6.5.1 คำสั่งในหัวข้อของ head (Head Section)....24
2.6.5.2 คำสั่งในส่วนของ (Body Section)....25
2.6.6 การใส่รูปภาพลงไปในเอกสาร (IMAGE)....25

บทที่ 3 วิธีการดำเนินงาน....27
3.1 บทนำ....27
3.2 ไมโครคอนโทรลเลอร์ตระกูล PIC18F458....27
3.3 การออกแบบ....29
3.4 การทำงานของโปรแกรมภายในตู้ควบคุมการเปิด-ปิด ไฟส่องสว่าง....30
3.5 วิธีการดำเนินงาน....31
3.5.1 การติดตั้งและการตั้งค่าโปรแกรม Proteus....31
3.5.2 การติดตั้งและการตั้งค่าโปรแกรม MikroC....36
3.5.3 การติดตั้ง Snap Developer....37
3.5.3.1 เปิด FTP Server บน SNAP Board เตรียมพร้อมสำหรับการ Upload Application file....41
3.5.3.2 ติดตั้งโปรแกรมFTP Cilent ใช้ Upload Application Program....42
3.5.3.3 ทดลองRunโปรแกรม HelloWorld....43
3.5.3.4 การควบคุม SNAP Board ผ่าน network ด้วยโปรแกรม Telnet....44

บทที่ 4 การทดลองและผลการทดลอง....46
4.1 บทนำ....46
4.2 การทดลองและผลการทดลอง....46

บทที่ 5 สรุปผล ปัญหาและข้อเสนอแนะ....49
5.1 สรุปผล....49

ลิขสิทธิ์มหาวิทยาลัยเชียงใหม่ อนุญาตให้ใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ(ต่อ)

หน้า

5.2 ปัญหาและแนวทางแก้ไข.....49

5.3 ข้อเสนอแนะ.....49

บรรณานุกรม.....50

ภาคผนวก ก.....51

ภาคผนวก ข.....88

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

หน้า

รูปที่ 2.1 แสดงตำแหน่งของอุปกรณ์ต่างๆ ในบอร์ด ET-CAN DRIVER.....5
รูปที่ 2.2 แสดงการเลือก Termination และผังการเชื่อมต่อ CAN BUS.....6
รูปที่ 2.3 แผนผังแสดงตัวอย่างการต่อ CAN BUS.....6
รูปที่ 2.4 รูปแบบคำสั่ง (Message Frame) ของ CAN (Control Area Network).....9
รูปที่ 2.5 ความแตกต่างระหว่าง CAN 2.0A และ CAN 2.0B.....10
รูปที่ 2.6 การสื่อสารของแต่ละโหนดบนระบบบัส.....14
รูปที่ 2.7 SNAP (Simple Network Application Platform).....14
รูปที่ 2.8 รูปการจัดวางแบบมาตรฐานของ HTML ทั่วไป.....23
รูปที่ 2.9 คำสั่งในหัวข้อของ Head.....24
รูปที่ 3.1 โครงสร้างการทำงานของระบบควบคุมไฟส่องสว่างผ่าน SNAP.....27
รูปที่ 3.2 ไมโครคอนโทรลเลอร์ PIC18F458.....27
รูปที่ 3.3 ฮาร์ดแวร์การควบคุมเปิด-ปิดไฟส่องสว่าง.....29
รูปที่ 3.4 ด้านหน้าและด้านหลังแสดงในรูปแบบ 3 มิติ.....29
รูปที่ 3.5 ลายวงจร.....29
รูปที่ 3.6 กล่องควบคุม.....30
รูปที่ 3.7 แผนผังการทำงานของตู้ควบคุมการเปิด-ปิดไฟส่องสว่าง.....30
รูปที่ 3.8 แสดงที่เก็บโปรแกรม Proteus.....31
รูปที่ 3.9 แสดงการลงโปรแกรม Proteus.....31
รูปที่ 3.10 แสดงการลงโปรแกรม Proteus (ต่อ).....32
รูปที่ 3.11 แสดงการเลือกพื้นที่ในการติดตั้งโปรแกรม Proteus.....32
รูปที่ 3.12 แสดงการเลือกรูปแบบของโปรแกรม Proteus.....32
รูปที่ 3.13 แสดงการเลือกโฟลเดอร์ของโปรแกรม Proteus.....33
รูปที่ 3.14 แสดงการติดตั้งของโปรแกรม Proteus.....33
รูปที่ 3.15 แสดงการติดตั้งโปรแกรม Proteus เรียบร้อย.....33
รูปที่ 3.16 แสดงไอคอนโปรแกรม Proteus.....34

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

หน้า

รูปที่ 3.17 แสดงการใช้โปรแกรม Proteus....34
รูปที่ 3.18 แสดงการใช้โปรแกรม Proteus (ต่อ)....34
รูปที่ 3.19 แสดงหน้าต่างโปรแกรม ISIS Professional....35
รูปที่ 3.20 แสดงวงจรที่ออกแบบในโปรแกรม ISIS Professional....35
รูปที่ 3.21 แสดงอุปกรณ์ในโปรแกรม ISIS Professional....35
รูปที่ 3.22 แสดงไอคอนโปรแกรม ISIS....36
รูปที่ 3.23 แสดงไอคอนโปรแกรม ARES....36
รูปที่ 3.24 แสดงการตั้งค่าโปรแกรม MikroC....36
รูปที่ 3.25 แสดงการคอมไพล์โปรแกรม MikroC....37
รูปที่ 3.26 แสดงตำแหน่ง directory ของ snap developer 1.0.8....37
รูปที่ 3.27 รายละเอียดในไฟล์ built.bat ว่าด้วยขั้นตอน....38
รูปที่ 3.28 รายละเอียดในไฟล์ built.bat....38
รูปที่ 3.29 เปิดโปรแกรม SnapDev....38
รูปที่ 3.30 กำหนด Comport ที่ต้องการติดต่อกับ SNAP....39
รูปที่ 3.31 แสดงการ Port Setting....39
รูปที่ 3.32 แสดงหน้าจอหลังจากที่จ่ายไฟเข้าที่บอร์ด SNAP....40
รูปที่ 3.33 แสดงการ login เข้าสู่ SNAP board....40
รูปที่ 3.34 แสดงการใช้คำสั่ง help เพื่อโชว์ program ใน root directory....40
รูปที่ 3.35 แสดงสถานะของ SNAP ว่ามีสถานะเป็นอย่างไร....41
รูปที่ 3.36 หน้าจอแสดงผลลัพธ์ของคำสั่ง ipconfig....41
รูปที่ 3.37 แสดงว่ามีการ Active Server ทั้ง 3 Server คือ Telnet, Serial, FTP....42
รูปที่ 3.38 หน้าต่างโปรแกรม FileZilla....43
รูปที่ 3.39 การ upload file....43
รูปที่ 3.40 แสดงการเปลี่ยน Sub Directory ไปยัง Sub Directory Classes....44
รูปที่ 3.41 แสดงผลลัพธ์ของการ RUN โปรแกรม....44

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป (ต่อ)

หน้า

รูปที่ 3.42 หน้าจอของโปรแกรม Telnet.....44
รูปที่ 3.43 แสดงหน้าจอหลังจากผ่านการ login เข้ามาแล้ว.....45
รูปที่ 3.44 แสดงผลลัพธ์ของโปรแกรม HelloWorld ผ่าน Telnet.....45
รูปที่ 4.1 ทำการกดปุ่มและแสดงสถานะหน้ากล่องควบคุม.....46
รูปที่ 4.2 หน้าจอการใช้คำสั่งในการทำงานของโปรแกรม.....47
รูปที่ 4.3 การแสดงผลผ่านทางเว็บบราวเซอร์.....47
รูปที่ 4.4 การส่งข้อมูลมาจากฮาร์ดแวร์เข้าสู่ SNAP.....48
รูปที่ 4.5 การส่งข้อมูลมาจากฮาร์ดแวร์เข้าสู่ SNAP.....48

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

หน้า

ตารางที่ 2.1 เปรียบเทียบความเร็วและระยะทางของการสื่อสาร CAN BUS.....6
ตารางที่ 2.2 โครงสร้างของ CAN Nodes.....8
ตารางที่ 2.3 CAN Message Frame.....12
ตารางที่ 2.4 ตารางเปรียบเทียบระหว่าง CDC และ CLD.....19
ตารางที่ 2.5 ตารางแสดง Profile และ Configuration.....21

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

1.1 ความสำคัญของปริญญานิพนธ์

ในปัจจุบันสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ได้มีการใช้พลังงานไฟฟ้าเพิ่มขึ้นอย่างต่อเนื่องทำให้ทางสถาบันต้องสูญเสียรายจ่ายในเรื่องของค่าไฟเป็นจำนวนมาก จึงได้มีนโยบายรณรงค์ให้คณะต่างๆภายในสถาบันใช้พลังงานไฟฟ้าอย่างประหยัด โดยทั่วไปอาคารของแต่ละภาควิชามักจะมีการเปิดไฟในช่วงเวลากลางคืน หากลืมปิดและเป็นช่วงวันหยุดราชการไฟอาจถูกเปิดทิ้งไว้เป็นเวลาหลายวัน ทำให้สูญเสียพลังงานโดยเปล่าประโยชน์ ทางผู้จัดทำจึงจัดมีแนวคิดที่จะทำโครงงานนี้ขึ้นเพื่อช่วยอำนวยความสะดวกในการตรวจสอบการเปิด-ปิดไฟส่องสว่าง

1.2 วัตถุประสงค์ของปริญญานิพนธ์

1. ศึกษาการเขียนโปรแกรมไมโครซี เพื่อควบคุมการทำงานบน ไมโครคอนโทรลเลอร์
2. ศึกษาการเชื่อมต่อไมโครคอนโทรลเลอร์ กับ SNAP ผ่าน CAN BUS
3. ศึกษาหลักการทำงานและการเขียนโปรแกรมควบคุม SNAP ด้วย ภาษา JAVA

1.3 ขอบเขตของปริญญานิพนธ์

1. สามารถสร้างอุปกรณ์ควบคุมการเปิด – ปิดไฟส่องสว่างได้
2. สามารถแสดงสถานะการเปิด-ปิดไฟส่องสว่างผ่านทางบราวเซอร์ด้วย SNAP ได้

1.4 ขั้นตอนการศึกษา

1. ศึกษาคุณสมบัติและลักษณะการใช้งานของPICและอุปกรณ์ในฮาร์ดแวร์
2. ศึกษาการเขียนโปรแกรมภาษาไมโครซี
3. ศึกษาการเขียนโปรแกรมเพื่อส่ง-รับข้อมูลผ่านระบบ CANBUS
4. ศึกษาหลักการทำงานของ SNAP
5. ศึกษาการเขียนโปรแกรมควบคุมบน SNAP ด้วยภาษา JAVA
6. ศึกษาการเขียนโปรแกรมด้วยภาษา HTML

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 ไมโครคอนโทรลเลอร์ตระกูล PIC

ไมโครคอนโทรลเลอร์ตระกูล PIC ที่นิยมใช้กันอยู่ในปัจจุบันนี้มีอยู่หลายเบอร์ด้วยกันโดยสามารถแยกเป็นกลุ่มต่างๆได้ดังนี้กลุ่มที่ขึ้นต้นด้วย PIC19,PIC12,PIC14,PIC16,PIC17,และPIC18 ซึ่งแต่ละกลุ่มยังแยกเป็นเบอร์ต่างๆอีกหลายเบอร์แต่กลุ่มที่ได้รับความนิยมมีอยู่ด้วยกัน 3 กลุ่มคือกลุ่มที่ขึ้นด้วย PIC16,PIC17,และPIC18 สำหรับทั้ง 3 กลุ่มนี้จะมีคุณสมบัติที่แตกต่างกันออกไปเช่นขนาดของหน่วยความจำโปรแกรมขนาดของหน่วยความจำข้อมูล จำนวนคำสั่งภาษาแอสเซมบลีและจำนวนพอร์ตแต่ละโครงสร้างและสถาปัตยกรรมจะคล้ายกัน สำหรับรายละเอียดสามารถดูได้จาก www.microchip.com ไมโครคอนโทรลเลอร์ตระกูลPICแบ่งตามชนิดของหน่วยความจำโปรแกรมสามารถแบ่งได้ 3 กลุ่ม คือ

กลุ่มที่ 1 PIC ที่มีหน่วยความจำโปรแกรมได้ครั้งเดียวหน่วยโปรแกรมกลุ่มนี้เรียกว่า OTP (one time programmable) เป็นชิปที่มีราคาถูกที่สุดในสามประเภทสาเหตุก็มาจากว่าชิปแบบ OTP จะสามารถทำการโปรแกรมได้แค่ครั้งเดียวเท่านั้น หลังจากชิปได้ถูกโปรแกรมไปแล้วจะไม่สามารถโปรแกรมเข้าไปใหม่ได้อีกดังนั้นชิปประเภทนี้จะเป็นนิยมใช้ หลังจากได้พัฒนาโปรแกรมจนกระทั่งจุดบกพร่องต่างๆในโปรแกรมไม่มีอีกแล้วเพราะจะมีต้นทุนต่ำเมื่อเทียบตัว memory ประเภทอื่นจะมีตัวอักษรCแสดงบนตัวชิปเช่น16C84,16C74

กลุ่มที่ 2 PIC ที่มีหน่วยความจำโปรแกรมได้หลายครั้ง แบบอีพรอมหน่วยโปรแกรมกลุ่มนี้เรียกว่าEPROM(erasableprogramableROM)เป็นชิปที่มีหน่วยความจำโปรแกรมเมื่อเขียนโปรแกรมเข้าไปแล้วสามารถเขียนโปรแกรมใหม่และลบโปรแกรมเดิมโดยให้แสงยูวี (UV:UltraViolet) ส่องผ่านเข้าไปยังชิปประมาณ5-10นาทีดังนั้นที่ด้านบนของชิปจะมีกรอบกระจกเพื่อให้แสงยูวีสามารถส่องผ่านเข้าไปในตัวชิปได้แต่ก็มีจำนวนครั้งในการลบโปรแกรมเช่นกัน เมื่อลบโปรแกรมด้วยแสงยูวีมากๆเข้าก็จะเกิดอาการด้านคือโปรแกรมไม่เข้านั่นเอง ซึ่งจะมีตัวอักษรJWหรือดูว่ามีกรอบกระจกอยู่บนชิปเช่น PIC12C508

กลุ่มที่ 3 PIC ที่มีหน่วยความจำโปรแกรมได้หลายครั้ง แบบแฟรชหรืออีอีพรอมหน่วยโปรแกรมกลุ่มนี้เรียกว่าEEPROM/Flash (electronically erasable programmableROM)เป็นชิปที่ได้รับความนิยมมากที่สุด เนื่องจากส่วนของหน่วยความจำของโปรแกรมสามารถอ่านหรือเขียนด้วยสัญญาณทางไฟฟ้าซึ่งใช้เวลาในการลบข้อมูลไม่กี่วินาที และสามารถลบและเขียนใหม่ได้หลายพันครั้งทำให้สะดวกในการแก้ไขปรับปรุงและเปลี่ยนแปลงโปรแกรมชิป แบบแฟรชจะมีตัวอักษรFเป็นตัวบอกแสดงบนชิปเช่น16F84,16F877

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้มีการนำเนื้อหาไปดัดแปลงต่อเติม หรือเผยแพร่ต่อในรูปแบบอื่นใด

2.1.1 คุณสมบัติของไมโครคอนโทรลเลอร์ตระกูล PIC ในแต่ละกลุ่ม

การแบ่งกลุ่มของ PIC จะแบ่งตามตัวเลข โดยแต่ละกลุ่มก็จะมีเบอร์ต่างๆ แยกออกไปอีกหลายเบอร์ ซึ่งสามารถแบ่งออกเป็น 6 กลุ่ม ได้แก่

1) ไมโครคอนโทรลเลอร์ตระกูล PIC16C5X เป็นชิปที่ผลิตออกมาเป็นยุคแรก เหมาะกับงานที่ไม่ยุ่งยาก และซับซ้อนมีคำสั่งในภาษาแอสเซมบลี 33 คำสั่งมี I/O, Timer และ Watch dog แต่ไม่มี I2C หรือ SERIAL ซึ่งต้องเขียนโปรแกรมขึ้นมาเองชิปในกลุ่มนี้มีหลายเบอร์ แต่ละขาจะมีจำนวนขาและขนาดของหน่วยความจำโปรแกรมและหน่วยความจำข้อมูลต่างกัน
2) ไมโครคอนโทรลเลอร์ตระกูล PIC16CXXXจากข้อจำกัดหลายๆอย่างPIC16C5X บริษัทไมโครชิปจึงได้ทำการพัฒนา และปรับปรุงจนเป็นตระกูลPIC16CXXX โดยเป็นชิปแบบ 18 ขามีคำสั่งในภาษา assembly 35 คำสั่ง มีI/O, มีTimer เพิ่มขึ้น บางเบอร์มีมากกว่า 1 ตัว มีพอร์ต Watch,dog,I2C,USART,SPIและ PWMทำให้สามารถติดต่อกับอุปกรณ์ภายนอกได้สะดวกขึ้น และเขียนโปรแกรมควบคุมง่ายยิ่งขึ้น นอกจากนี้ยังได้ทำการเพิ่มขนาดของหน่วยความจำให้มีขนาดใหญ่ขึ้นด้วย
3) ไมโครคอนโทรลเลอร์ตระกูล PIC12CXXX และ PIC12FXXX เป็นชิปขนาดเล็กที่มีเพียง 8 ขาเท่านั้น ซึ่งเหมาะกับงานเล็กๆมีคำสั่งในภาษาแอสเซมบลี 33และ 35 คำสั่งสำหรับจุดเด่นของ PIC กลุ่มนี้ คือ สัญญาณนาฬิกาหรือออสซิลเลเตอร์ (oscillator) ขนาด 4MHz oscillator อยู่ภายในตัวชิปทำให้ไม่ต้องต่อออสซิลเลเตอร์ภายนอก และมีหน่วยความจำข้อมูลภายในเป็นแบบEEPROM แต่ส่วนของหน่วยความจำโปรแกรมยังเป็นแบบ OTP และ EPROM หลังจากได้ผลิต PIC16FXXX แล้วทางบริษัทไมโครชิปก็ได้ผลิต PIC12FXXX ขึ้นมาซึ่งเป็นแบบแฟรช ทำให้สามารถเขียนและลบโปรแกรมได้หลายครั้ง
4) ไมโครคอนโทรลเลอร์ตระกูล PIC17CXXX เป็นชิปที่ออกมาพร้อมกับ PIC16CXXXแต่แตกต่างกันที่ PIC17CXXX จะมีความสามารถสูงกว่ามีขามากกว่า มีคำสั่งในภาษา Assembly 58 คำสั่งมีคำสั่งการคูณหารรวมทั้งขนาดของหน่วยความจำโปรแกรมหน่วยความจำข้อมูลที่มีขนาดใหญ่ขึ้น และยังสามารถต่อกับหน่วยความจำภายนอกได้ด้วย
5) ไมโครคอนโทรลเลอร์ตระกูล PIC16F8XXX เป็น PIC ที่ได้รับความนิยมมาก เพราะเป็นชิปยุคแรกๆ ที่มีหน่วยความจำโปรแกรมเป็นแบบแฟรชและมีหน่วยความจำข้อมูลแบบEEPROM ทำให้สามารถสร้างได้ง่าย ซึ่ง PIC เบอร์ PIC16F8XXX สนับสนุนการทำงานแบบ อินเซอร์กิตดีบักเกอร์ (In Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Debugger) ทำให้ไม่ต้องซื้ออีพรอมอีมูเลเตอร์ (EPROM Emulator)ซึ่งมีราคาแพงมากมีคำสั่งภาษาแอสเซมบลี 35 คำสั่งและมีวงจรแปลงสัญญาณอะนาล็อคเป็นสัญญาณดิจิตอล (A/D) ขนาด 10 บิตอยู่ภายในด้วย

6) ไมโครคอนโทรลเลอร์ตระกูล PIC18CXXX และ PIC18FXXX (FLASHMCUs) เป็น PIC อีกกลุ่มที่ได้รับความนิยมเนื่องจาก PIC18CXXX และ PIC18FXXX มีความสามารถมากและใหญ่กว่าเบอร์ PIC16F8XXX ไม่ว่าจะเป็นคำสั่งภาษาแอสเซมบลีที่มีถึง 77 คำสั่ง หรือหน่วยความจำโปรแกรมที่มีขนาดใหญ่ขึ้น ทำให้สามารถรองรับการเขียนโปรแกรมภาษาซี

2.2 ET-CAN DRIVER

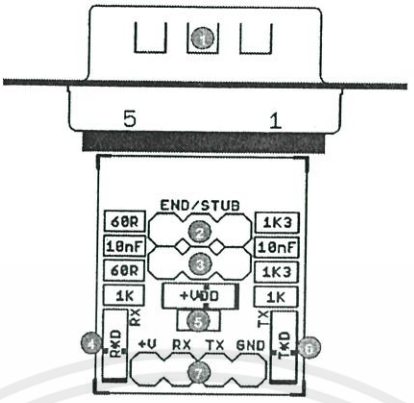
ET-CANDRIVER เป็นบอร์ด CAN TRANSCEIVERS สำหรับการสื่อสาร CAN BUS โดยเลือกใช้ CAN TRANSCEIVERS เบอร์ SN65HVD232D รองรับสัญญาณการสื่อสารแบบ CAN มาตรฐาน “ISO-11898” ใช้สำหรับทำหน้าที่แปลงสัญญาณทางไฟฟ้าของ CAN Logic ให้เป็นสัญญาณ Differential CAN Bus (±25V) ซึ่งบอร์ด ET-CANDRIVER ต้องนำไปใช้งานร่วมกันกับ MCU ที่มีวงจร CAN Controller บรรจุไว้ภายในหรือเชื่อมต่อกับชิพ CAN Controller

2.2.1 คุณสมบัติของบอร์ด ET-CAN DRIVER

1) ใช้ CAN TRANSCEIVERS เบอร์ SN65HVD232D ของ Texas Instruments
2) รองรับการเชื่อมต่อกับ CAN Controller Logic ทั้งระบบ 5V และ 3.3V
3) รองรับข้อกำหนดตามมาตรฐานสัญญาณCAN ISO-11898 (standard physical layer)
4) มีวงจร Termination แบบ Split Termination ภายในบอร์ดสามารถเลือกกำหนดจาก Jumper ได้ทั้งแบบ End Node(120Ω□) และ Stub Node (2.6KΩ)
5) รองรับความเร็วBus ระหว่าง 62.5Kb/s (ระยะทาง 1000 เมตร) – 1Mb/s (ระยะทาง 30 เมตร)
6) สัญญาณเชื่อมต่อด้านLogic ใช้ขั้วต่อ Pin Header ขนาด 1 x 4 ระยะ Pitch 2.54mm
7) สัญญาณเชื่อมต่อด้าน CAN BUS ใช้ Connector DB9 Male จัดเรียงขาสัญญาณตามข้อกำหนดของ J1939,CAN - CIAStandard Connector (Pin2 : CANL,Pin7:CANH,Pin3:CAN-GND)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.2 โครงสร้างบอร์ด ET-CAN DRIVER



รูปที่ 2.1 แสดงตำแหน่งของอุปกรณ์ต่างๆในบอร์ด ET-CAN DRIVER

หมายเลข 1 คือขั้วต่อCAN BUS เป็นแบบDB9 Male จัดขาสัญญาณแบบ J1939,CAN-CIA

หมายเลข 2,3 คือJumper สำหรับเลือกกำหนดรูปแบบของวงจรTermination ให้กับวงจร

หมายเลข 4 คือLED RX แสดงสถานะของการรับข้อมูลจากCAN BUS

หมายเลข 5 คือLED +V แสดงสถานะของแหล่งจ่ายไฟ

หมายเลข 6 คือLED TX แสดงสถานะของการส่งข้อมูลให้CAN BUS

หมายเลข 7 คือขั้วต่อCAN Logic

+V เป็นไฟเลี้ยงวงจรให้บอร์ดใช้ได้กับแรงดัน3-6V ซึ่งปรกติควรต่อกับแหล่งจ่ายของวงจร CAN Controller โดยตรง

RX เป็นขารับข้อมูลCAN Logic ที่ทำการแปลงระดับสัญญาณมาจากCAN BUS

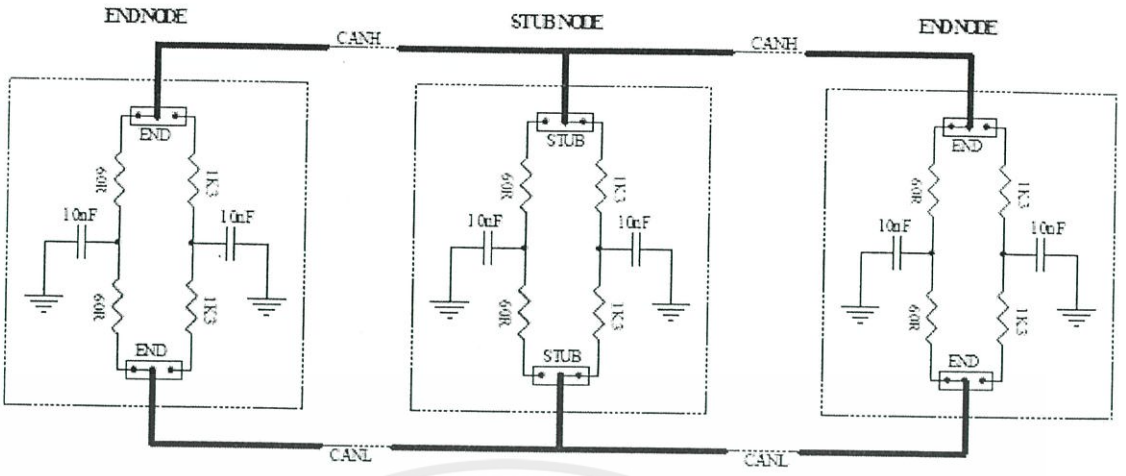
TXเป็นขาส่งข้อมูลCAN Logic เพื่อนำไปแปลงเป็นสัญญาณส่งออกCAN BUS

GND เป็นจุดอ้างอิงระดับสัญญาณ Logic และแหล่งจ่ายไฟของบอร์ด

2.2.3 การเลือก Node สำหรับ CAN BUS

ในระบบ CAN BUS นั้นจำเป็นต้องมีการกำหนดการทำงานของวงจร Termination เพื่อชดเชยความผิดเพี้ยนของสัญญาณ ซึ่งเกิดจากความต้านทานในสายสัญญาณให้กับวงจร CAN Transceiver โดยถ้าเป็น CAN Transceiver ที่อยู่ตำแหน่งต้นสายและปลายสายต้องกำหนดเป็น END Node แต่ถ้าเป็นวงจรของ CAN Transceiver ชุดที่อยู่ระหว่างต้นทางกับปลายทางจะกำหนดเป็น STUB Node ซึ่งในบอร์ด ET-CAN DRIVER เองจะใช้ Jumper (END/STUB) เป็นตัวกำหนด Node ดังตัวอย่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ทั้งนี้ห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

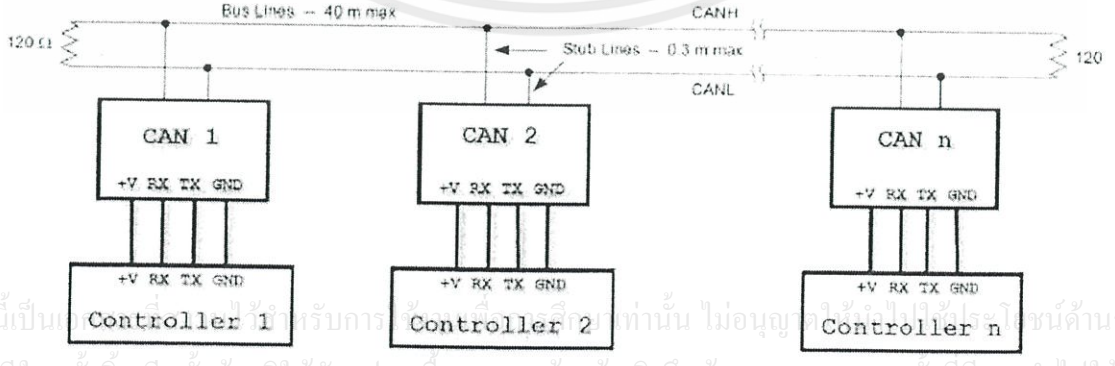


รูปที่ 2.2 แสดงการเลือก Termination และผังการเชื่อมต่อ CAN BUS

ระบบ CAN BUS รองรับการเชื่อมต่อ CAN BUS ได้จำนวนสูงสุด 120 Node และมีระยะทางไกลสุดประมาณ 30 เมตรถึง 1000 เมตร ขึ้นอยู่กับความเร็วที่เลือกใช้โดยใช้สายสัญญาณแบบ STP หรือ UTP โดยระยะทางของการสื่อสารจะแปรผกผันกับความเร็วในการรับส่งข้อมูลดังข้อมูลในตารางที่ 2.1

ตารางที่ 2.1 เปรียบเทียบความเร็วและระยะทางของการสื่อสาร CAN BUS

Bit Rate (kb/s)	Bus Length (เมตร)
1000	30
500	100
250	250
125	500
62.5	1000



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.3 แผนผังแสดงตัวอย่างการต่อ CAN BUS

2.3 Control Area Network (CAN)

2.3.1 ประวัติของ Control Area Network (CAN)

CAN (Control Area Network) หรือ CAN Protocol ถูกพัฒนาโดย Bosch และได้รับมาตรฐานISO 11898 ในปี 1994 ระบบ Control Area Network (CAN) ถูกทำมาใช้ในการควบคุมและส่งข้อมูลระหว่างอุปกรณ์ควบคุมอิเล็กทรอนิกส์ในเครื่องยนต์ นอกจากนี้ CAN (Control Area Network) สามารถนำมาประยุกต์ใช้สำหรับ ไมโครคอนโทรลเลอร์ช่วยควบคุมระบบเครือข่ายในโรงงานอุตสาหกรรม เช่น การควบคุม ระบบบัส (System Bus) ที่อยู่ในเครื่องจักรและอาจมีการควบคุมการทำงานของ เซ็นเซอร์ (Sensor) หรืออุปกรณ์ต่างๆ ได้อย่างมีประสิทธิภาพ ในระบบ CAN (Control Area Network) แต่ละข้อความ (Message) ที่ส่งออกมาจะมีความชัดเจนในการส่งข้อมูลในมาตรฐานของ CAN (Control Area Network) จะมีการส่งข้อความ (Message) ออกมาครั้งละ 11 บิตซึ่งจะทำให้เกิดข้อความ (Message) หรือคำสั่งที่แตกต่างกันได้ถึง 2048 ข้อความ (Message) และนอกจากนี้ได้มีการขยายมาตรฐานของ CAN (Control Area Network) ออกมาอีกโดยมีการกำหนดข้อความ (Message) ครั้งละ 29 บิตซึ่งจะทำเกิดข้อความ (Message) หรือคำสั่งที่แตกต่างกันได้สูงสุดถึง 536 ล้านข้อความ (Message) CAN (Control Area Network) สามารถมีตัวควบคุมหลัก (Master) ได้มากกว่าหนึ่งตัว ในกรณีที่มีตัวควบคุมหลักมากกว่าหนึ่งตัวแต่ละตัวจะมีสิทธิเท่าเทียมกันและจะมีหน้าที่การทำงานเหมือนกันข้อดีที่สำคัญของการใช้ CAN (Control Area Network) แบบตัวควบคุมหลักหลายตัว (Multi-Master) คือ ตัวควบคุมหลัก (Master) แต่ละตัวจะสามารถทำงานและควบคุมได้ด้วยตัวของมันเอง และสามารถส่งข้อความ (Message) หรือรับการร้องขอ (Request)ได้แล้วแต่กรณี การที่ใช้ CAN (Control Area Network) แบบ Multi-Master จะมีข้อดีต่อระบบเครือข่ายคือในกรณีที่มีการเสียหรือเกิดข้อผิดพลาด (Error)ที่ตัวควบคุมหลัก (Master) จะไม่ส่งผลกระทบต่อระบบเครือข่ายและจะไม่ทำให้ระบบบัส (System Bus)ล้มเหลวหรือทำงานไม่ได้

CAN (Control Area Network) ถูกพัฒนาโดยบริษัทในประเทศเยอรมันของ Robert Bosch สำหรับใช้ในอุตสาหกรรมประกอบรถยนต์ เพื่อที่จะช่วยลดปัญหาต้นทุนระบบบัส (System Bus) และการเลือกใช้ระบบ CAN (Control Area Network) เป็นทางเลือกที่เหมาะสม และประหยัดค่าใช้จ่ายที่สุด อุตสาหกรรมรถยนต์ใช้ CAN (Control Area Network) ช่วยเพิ่มความสามารถในการประยุกต์ใช้งานและเนื่องจาก CAN (Control Area Network) สามารถแสดงให้เห็นถึงประสิทธิภาพและความสามารถ ในปัจจุบัน CAN (Control Area Network) ถูกนำมาใช้ในอุตสาหกรรมหลายประเภทเพื่อใช้ควบคุมและเนื่องจากจุดเด่นของ CAN (Control Area Network) คือความเร็วประสิทธิภาพและความปลอดภัย CAN (Control Area Network) ได้รับมาตรฐาน ISO 11898 (สำหรับ High-Speed Application) และ ISO 11519 (สำหรับ Lower-Speed Application)

เอกสารนี้มีลิขสิทธิ์ ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้ ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

2.3.2 โครงสร้างของ Control Area Network (CAN)

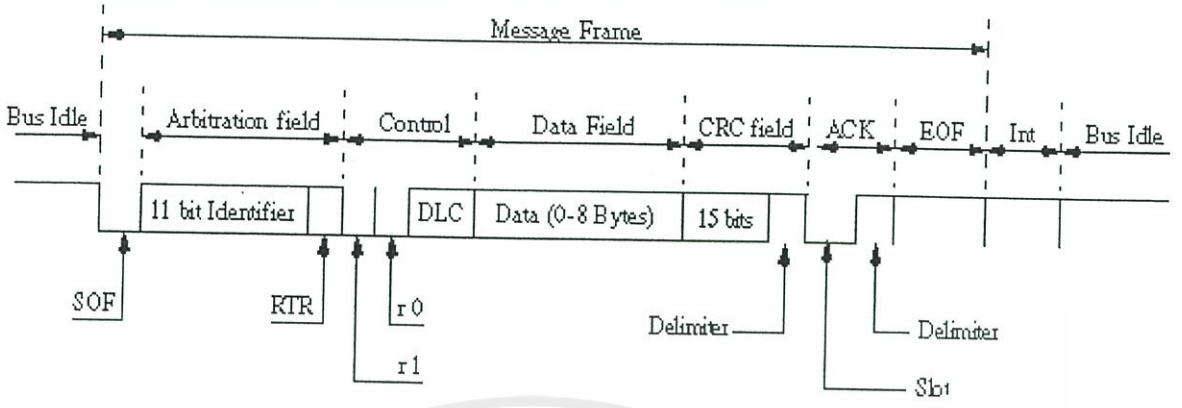
CAN (Control Area Network) คือระบบเครือข่ายการสื่อสารข้อมูลที่มีประสิทธิภาพสูง ทำงานด้วยความรวดเร็วแบบทันทีทันใด (Real Time) และมีการรักษาความปลอดภัยของข้อมูลอยู่ในระดับสูง เป็นระบบเครือข่ายที่มีความเร็วสูงและมีค่าใช้จ่ายในการติดตั้งหรือดูแลรักษาต่ำได้มีการนำเอา CAN (Control Area Network) มาใช้ในรถยนต์ในส่วนของเครื่องยนต์เพื่อช่วยในการควบคุมการทำงานของ เซ็นเซอร์ (Sensor) เช่น การควบคุมกระจกไฟฟ้า การควบคุมระบบไฟในรถยนต์ เป็นต้น โครงสร้างของ CAN Nodesสามารถจัดแบ่ง ได้ 3 ส่วนหลัก ได้แก่ ส่วนที่ 1 CAN Object, ส่วนที่ 2 CAN Transfer Layer, ส่วนที่ 3 CAN Physical Layer แสดงดังตารางที่ 2.2

ตารางที่ 2.2 โครงสร้างของ CAN Nodes

Application Layer
Object Layer - Message Filtering - Message and Status Handling
Transfer Layer - Fault Confinement - Error Detection and Signaling - Message Validation - Acknowledgement - Arbitration - message Framing - Transfer Rate and timing
Physical Layer - Signal Level and Bit Representation - Transmission Medium

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.3 รูปแบบของCAN (Control Area Network)



รูปที่ 2.4 รูปแบบคำสั่ง (Message Frame) ของ CAN (Control Area Network)

2.3.3.1 คำอธิบาย CAN2.0A

มาตรฐานของ CAN 2.0 Aในส่วนของ รูปร่างคำสั่ง (Message Frame) จะประกอบไปด้วย 7 bit field ที่แตกต่างกันStart OF Frame (SOF) Field คือส่วนที่เป็นจุดเริ่มต้นของ รูปร่างคำสั่ง (Message Frame) ซึ่งจะประกอบด้วย โดมิแนนท์ บิต(Dominant bit)เพียงตัวเดียวArbitration Field ประกอบด้วย ข้อความตัวระบุ (Message Identifier) และ Remote Transmission Request (RTR) บิต จะประกอบด้วยตัวระบุ (Identifier) ความยาว 11 บิตโดยจะเริ่มจาก ID10 ถึง ID 0Remote Transmission Request BIT (RTR) ในData Frame จะมีRTR บิต คือ โดมิแนนท์ บิต (Dominant bit) ส่วนใน Remote Frame จะมี RTR บิต คือ รีเซสซีฟ บิต (Recessive Bit) ส่วนของ Control Field จะประกอบด้วย 6 บิต ซึ่งจะแบ่งออกเป็นบิตสำรอง (Reserve Bit) คือ r0 กับ r1 ส่วนบิตที่เหลือจะเรียกว่า Data Length Code (DLC) ซึ่งData Length Code (DLC) จะเป็นตัวที่บ่งบอกจำนวนไบต์ ที่เกิดขึ้นในส่วนของ Data Field Data Field จะเริ่มจาก 0 ถึง 8 ไบต์Cyclic Redundancy Check Field (CRC Field) จะประกอบด้วย 15 บิตจะทำหน้าที่เช็คว่าข้อมูลที่ส่งไปเมื่อถอดรหัสแล้วจะเท่ากับค่าที่ได้ใน Cyclic Redundancy Check (CRC) หรือไม่ถ้าได้เท่ากันแสดงว่าไม่มีข้อผิดพลาดใดๆACKnowledge Field (ACK Field) จะประกอบด้วยบิตจำนวน 2 บิต คือ Slot บิต และ Delimiter บิต ซึ่งจะคอยเช็คดู ว่าบิตที่ส่งมา ค่าของบิตมีการเปลี่ยนแปลงหรือไม่ ถ้าเปลี่ยนแปลงแสดงว่าเกิดข้อผิดพลาดในการส่งEnd Of Frame Field (EOF Field) ประกอบด้วย รีเซสซีฟบิต (Recessive Bit) จำนวน 7 บิต

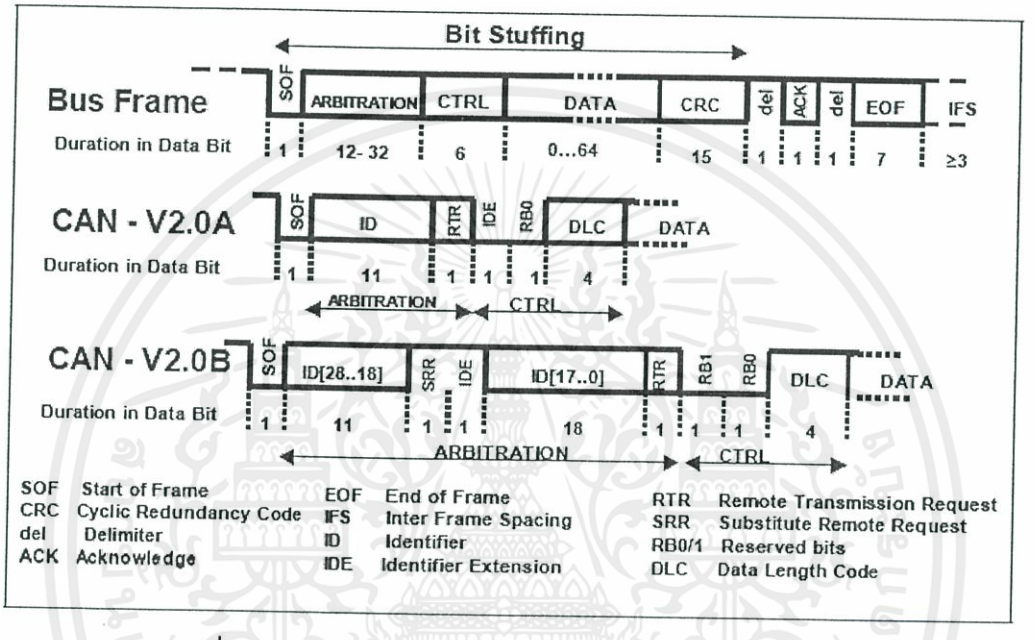
2.3.3.2 คำอธิบายCAN2.0B

CAN 2.0B จะมีตัวระบุ (Identifier) จำนวน 29 บิตและยังสามารถจัดการกับการติดต่อสื่อสารระหว่าง CAN 2.0A ได้อย่างไม่มีปัญหาซึ่ง CAN 2.0B ได้ขยายหรือพัฒนาต่อจาก CAN 2.0A

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น และห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.4 ข้อแตกต่างระหว่างCAN 2.0A และ CAN 2.0B

ใน CAN 2.0B ส่วนของ Arbitration Field ประกอบด้วย 2 Bit field ซึ่ง บิตแรก คือ base ID ซึ่งประกอบด้วย 11 บิต ซึ่งจะสามารถส่งข้อมูลเข้ากับรูปแบบของ CAN 2.0A ได้ ส่วน บิตสอง คือ ID Extension ประกอบด้วย 18 บิต ซึ่งจะทำให้ CAN 2.0B มีขนาด 29 บิต(11+18) ความแตกต่างระหว่าง CAN 2.0A และ CAN 2.0B เกิดจากในส่วนของ IDE (Identifier Extension Bit)



รูปที่ 2.5 ความแตกต่างระหว่าง CAN 2.0A และ CAN 2.0B

2.3.5 หลักการทำงานของCAN

Data Message จะส่งออกมาจากทุกโหนด (node) บนCAN Bus (Control Area Network System Bus) และจะไม่มี address ทั้งโหนด (nodes) ส่งและโหนด (nodes) รับ รายละเอียดของข้อความ (Message) จะถูกกำหนดโดยตัวระบุ (Identifier) ที่มีลักษณะเฉพาะตัวใน เครือข่าย ทุกๆ โหนด (nodes) บนเครือข่ายจะรับ ข้อความ (Message)และคำสั่งการทำงานโดยดู จากตัวระบุ (Identifier) และแต่ละโหนด (nodes) จะตัดสินใจว่าจะรับ ข้อความ (Message) หรือไม่ โดยดูจาก ตัวระบุ ถ้าข้อความ (Message) นั้นตรงกับสิ่งที่โหนด (nodes) นั้นๆต้องการมันจะรับ ข้อความ (Message) นั้นมาประมวลผล ถ้าไม่ใช่ มันก็จะหลีกเลี่ยงโดยการไม่รับ ข้อความ (Message) นั้นมา และ ถ้า มีหลายๆคำสั่งเข้ามาพร้อมๆกันมันจะตัดสินใจในการรับ ข้อความ (Message) ที่เข้า มาโดยการดูว่ามาก่อนหรือหลัง ในบางสถานการณ์ที่มี โหนด (nodes) มากกว่าสองโหนด (nodes) ที่ พยายามที่จะส่งข้อมูลในเวลาเดียวกัน ซึ่ง CAN (Control Area Network) จะมีเทคนิคในการจัดการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

ข้อมูล โดย CAN (Control Area Network) มีการให้ลำดับก่อน-หลัง อย่างชัดเจนซึ่งรับประกันได้ว่าไม่มีข้อมูลใดสูญหายขณะที่ส่งพร้อมกันในเวลาเดียวกัน

2.3.5.1 การเข้ารหัสบิต(Bit Encoding)

CAN (Control Area Network) ใช้ NRZ (Non Return to Zero) เพื่อเข้ารหัสสำหรับข้อมูลที่สื่อสารบนสายการสื่อสาร 2 สายที่แตกต่างกันโดยใช้ NRZ (Non Return to Zero) เข้ารหัสเพื่อให้แน่ใจว่า ข้อความ (Message) กะทัดรัด กระชับ และมีขนาดเล็ก ยืดหยุ่นต่อการถูกรบกวนจากภายนอก

2.3.5.2 ความมีเสถียรภาพในการทำงาน

CAN (Control Area Network) สามารถที่จะจัดการกับข้อมูล (Data) ที่ส่งหรือรับในสภาวะที่สภาวะแวดล้อมไม่ปกติและมีการขยายการตรวจจับข้อผิดพลาด (Error Checking) เพื่อให้แน่ใจว่า การสื่อสารข้อมูล (Data) นั้นๆ ค่าความผิดพลาด (Error) ถูกตรวจพบ ตามมาตรฐาน ISO 11898 ได้กล่าวสนับสนุนว่า Control Area Network (CAN) ยังคงทำงานอยู่ได้เมื่อเกิดเหตุการณ์เหล่านี้คือ สายสื่อสารระหว่างกันมีปัญหาเกิดการเสียหาย สายสื่อสารมีการช็อตกับไฟขั้วบวกหรือสายสื่อสารมีการช็อตกันกับไฟขั้วลบ

2.3.5.3 ความยืดหยุ่นของระบบเครือข่าย(Network Flexibility and Expansion)

โดยธรรมชาติของ CAN Message จะมีแบบแผนและมีความยืดหยุ่นสูงสำหรับสภาวะแวดล้อมที่แตกต่างกัน โหนด (nodes) ใหม่ๆสามารถที่จะรับหรือส่งข้อมูลและสามารถเพิ่มเข้ามาในระบบเครือข่ายโดยไม่ต้องมีการ เปลี่ยนแปลงทั้งทางด้าน ฮาร์ดแวร์ หรือซอฟแวร์

2.3.5.4 การจัดการลำดับข้อความ

ในระบบต่างๆค่า พารามิเตอร์ บางค่า อาจเปลี่ยนแปลงอย่างรวดเร็ว เช่น การเปลี่ยนแปลงของเครื่องยนต์ในรถยนต์ หรือการเปลี่ยนแปลงชั้นของลิฟท์ (Lift) แต่ค่าพารามิเตอร์บางค่าอาจเปลี่ยนแปลงอย่างช้า เช่น ค่าอุณหภูมิในเครื่องทำความเย็นของรถยนต์ ดังนั้นเมื่อมีการเปลี่ยนแปลงอย่างรวดเร็วจึงต้องมีการส่งค่าพารามิเตอร์ บ่อยขึ้นจึงต้องมีการจัดการลำดับก่อน-หลังเป็นอย่างดีเพื่อลดความซ้ำซ้อนและความสับสนของข้อมูลการจัดลำดับก่อน-หลัง ใน CAN Message นั้นใช้หลักในการตัดสินใจโดยใช้ตัวระบุ (Identifier) ซึ่งตัวระบุ (Identifier) จะมีการกำหนดค่าในระหว่างช่วงเริ่มต้น (Initial Phase)ของระบบ ตัวระบุ (Identifier)จะมีค่าไม่มากแต่จะมีการกำหนดสิทธิ์ลำกับการมาก่อน - หลังของข้อความ (Message) ที่ดีมาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.5.5 การหยุดพักหรือการเริ่มต้นทำงานใหม่ (Sleep Mode / Wake-up)

เพื่อช่วยในการลดการสิ้นเปลืองพลังงาน ในการทำงานในกรณีที่โหนด (nodes) ไม่ได้ทำการติดต่อกับระบบ ไม่ได้ทำงานนะเวลานั้น หรือไม่ได้มีการทำงานภายในโหนด (nodes) นั้น โหนด (nodes)นั้นๆ จะเข้าสู่สภาวะหยุดพัก (Sleep Mode) และหยุดการติดต่อกับระบบบัส (Bus) สภาวะการหยุดพักจะตื่นตัว (Wake-up) เมื่อมีการกระตุ้นจากระบบบัส (Bus) หรือเงื่อนไขอื่นๆ ในโหนด (nodes) ที่สั่งเอง

2.3.5.6 รูปแบบของการส่งข้อความของCAN (Control Area Network)

ในระบบ CAN ข้อมูลการสื่อสาร และการรับส่งใช้ รูปร่างคำสั่ง (Message Frame) รูปร่างคำสั่ง (Message Frame) จะนำเอาข้อมูลจากโหนด (nodes) ไปสู่ โหนด (nodes) อื่นๆ แสดงดังตารางที่ 2.3มาตรฐานของ CAN 2.0A ทำให้ทราบว่า Base Frame Format ซึ่งสนับสนุนการทำงานของ ข้อความ (Message) ขนาด 11 บิต CAN ที่ขยายออกจาก CAN2.0A คือ CAN 2.0B ซึ่ง ข้อความ (Message) ขนาด 29 บิต ทำให้ทราบว่า Extended Frame Format สนับสนุนการทำงานของข้อความ (Message) ทั้งสองแบบคือแบบ CAN 2.0A และ CAN 2.0B

ตารางที่ 2.3 CAN Message Frame

SOF	Identifier	RTR	Control	Data	CRC	ARC	EOF

2.3.6 ข้อผิดพลาดที่เกิดขึ้น

CAN 2.0B สามารถที่จะส่งข้อมูลผ่านไปยัง CAN 2.0A เพราะมีความเข้ากันได้ของระบบ แต่อย่างไรก็ตาม การส่งสัญญาณหรือการสื่อสารระหว่าง CAN 2.0B ไปยัง CAN 2.0A แบ่งเป็น 2 กรณี คือ

1) เมื่อมีการส่งข้อมูลผ่านระหว่าง CAN 2.0B ไปยัง CAN 2.0A จะเกิดความผิดพลาด (Error)
2) เมื่อมีการส่งข้อมูลผ่านระหว่าง CAN 2.0B ไปยัง CAN 2.0A เกิดความผิดพลาด (Error) แต่จะไม่มีผลต่อการทำงานเนื่องจากในส่วนของ CAN 2.0A จะไม่สนใจในความผิดพลาด (Error) ที่เกิดขึ้นในกรณีนี้

2.3.6.1 การตรวจจับข้อผิดพลาด (Error Detection)

CAN จะแบ่งการตรวจจับ Error ออกเป็น 5 อย่าง ซึ่งจะจัดอยู่ใน 2 หมวด

1) การตรวจสอบข้อความ (Message)

- CRC (Cyclic Redundancy Check) ซึ่งทุกๆ ข้อความที่ส่งไปจะประกอบด้วย CRC (Cyclic Redundancy Check) ซึ่งจะเกิดจาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เข้ารหัสข้อความ เพื่อเป็นการตรวจเช็คเมื่อถึงผู้รับ จะมีการเข้ารหัสข้อความเพื่อ เช็คกับค่าของ CRC (Cyclic Redundancy Check) เพื่อตรวจสอบว่าตรงกันหรือไม่ ถ้าไม่ตรงกันแสดงว่าเกิดการ Error ของข้อมูล

- Frame Checkคือ มีการเช็คที่บิตของข้อมูลทั้ง Frame ว่าถูกต้องเหมือนตอนที่ส่งมาหรือไม่ ถ้าไม่ถูกต้องแสดงว่า เกิดข้อผิดพลาดหรือข้อมูลเสียหายกับ รูปร่างคำสั่ง (Message Frame) นั้นๆ
- ACK (Acknowledgement) Error Checkจะมีการตรวจสอบที่ผู้รับจะเช็คบิตในจุดACK (Acknowledgement) ว่า มีการเปลี่ยนแปลงหรือไม่ ถ้ามีการเปลี่ยนแปลงได้ค่าไม่ตรงแสดงว่า เกิดข้อผิดพลาด

2) หมวดการตรวจสอบบิต (bit)

- BitMonitoring จะมีการตรวจสอบทุกๆบิตว่าบิตที่ออกไปถูกต้องหรือไม่
- Bit Stuffing จะมีการแทรกบิตเข้าไปและจะมีการเช็คที่ปลายทางว่าบิตที่แทรกเข้าไป ว่าตรงกับใส่ไปหรือไม่ถ้าไม่แสดงว่าเกิดข้อผิดพลาด (Error)

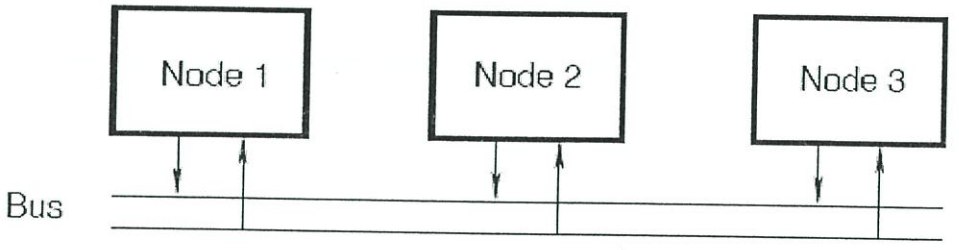
2.3.6.2 ขอบเขตของข้อผิดพลาดที่เกิดขึ้น(Error Confinement)

- Error Counts จะมีการนับเรื่อยๆเมื่อมีการ Error และจะกลับคืนค่าเป็นศูนย์เมื่อหาย Error
- Error Active Mode เมื่อมีการนับ Error ได้ไม่เกิน 127
- Error Passive Mode เมื่อมีการนับ Error ได้เกิน 127 ขึ้นไป
- Bus Off Mode เมื่อ มีการ นับ Error ได้เกิน 255 จะทำการปิดการทำงานของระบบบัส(System Bus) ข้อมูล

2.3.6.3 การจัดการเมื่อตรวจพบข้อผิดพลาด (Performance of Error Detection)

เมื่อตรวจพบข้อผิดพลาดระบบจะทำการหยุดการรับข้อความ (Message) และระบบจะทำการส่งข้อความ (Message) นั้นใหม่อีกครั้งในทันทีและจะตรวจสอบอีกว่าเกิดข้อผิดพลาดหรือไม่ ถ้าไม่มีระบบก็จะดำเนินต่อไปแต่ถ้าเกิดข้อผิดพลาดระบบก็จะทำการส่งใหม่อีกครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



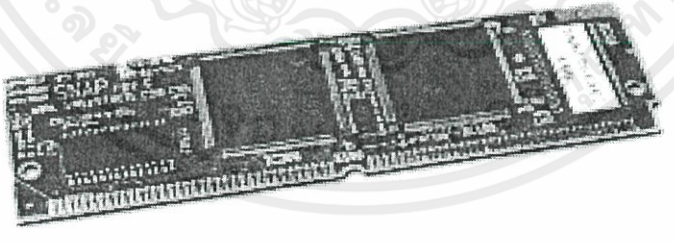
รูปที่ 2.6 การสื่อสารของแต่ละโหนดบนระบบบัส

2.3.7 ข้อดีของCAN (Control Area Network)

CAN (Control Area Network) หรือ CAN BUSคือการสื่อสารแบบต่อเนื่อง (Serial) สำหรับควบคุมอุปกรณ์ต่างๆแบบทันทีทันใด (Real Time) ซึ่งCAN (Control Area Network) สามารถจัดการกับอัตราการขนถ่ายข้อมูลได้สูงสุดถึง 1 Megabits ต่อวินาที มีความสามารถในการตรวจจับข้อผิดพลาด (Error Detection) ได้อย่างมีประสิทธิภาพและช่วยในการประหยัดค่าใช้จ่าย

2.4 SNAP (Simple Network Application Platform)

Simple Network Application Platform (SNAP) คือ Embedded Board ที่ได้รับการพัฒนาให้ทำงานร่วมกับภาษาจาวา (J2ME-CLDC) ซึ่งได้รับการรับรองโดยบริษัท Sun Microsystem พร้อมกับการพัฒนา Class ต่างๆไว้อย่างหลากหลาย เพื่อให้ง่ายต่อการติดต่อสื่อสารในระบบเครือข่าย (Network) และการควบคุมอุปกรณ์ผ่านอินพุตเอาต์พุตพอร์ตต่างๆบนตัว SNAPเช่น I2C, Serial port, SPI port, GPIO เป็นต้น แสดงดังรูปที่ 2.7



รูปที่ 2.7 SNAP (Simple Network Application Platform)

2.4.1 Baseboard

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เนื่องจาก SNAP ได้จัดรูปแบบการ Interface เป็นขามาตรฐานแบบ 72 pins ทำให้ไม่สะดวกต่อการนำเอา SNAP ไปใช้งาน บริษัทจึงได้ผลิต Baseboard ขึ้นมาเพื่อ อำนวยความสะดวก

ต่อการนำเอา SNAP ไปใช้งาน หรือ เรียนรู้ได้รวดเร็วขึ้น เนื่องจาก SNAP มีการออกแบบอินเตอร์เฟรส (Interface) ต่างๆ ไว้หลายประเภทดังนี้

1) Standard RS-232 Baseboardทางบริษัทได้ออกแบบให้มี Serial Port 1 Port (DB-9(F)) สำหรับใช้ติดต่อสื่อสารระหว่างตัว SNAP กับ อุปกรณ์ต่างๆ ได้โดยตรงซึ่งอัตราความเร็วสูงสุดอยู่ที่ 115200 Bits/S พร้อมทั้งสัญญาณควบคุมต่างๆ
2) CAN Controller Area Network เป็นมาตรฐานการส่งข้อมูลพร้อมๆกันบน Serial bus ได้รับการออกแบบมาสำหรับ ECU (Electronic Control Units) ซึ่งสามารถทนต่อสัญญาณรวบกวนสูงได้ สูงกว่าการทำงานRS-485 ซึ่งเป็นการทำงานแบบ Balanced Line ซึ่งเหมาะสำหรับการใช้งานรถยนต์ ขนส่ง และ ในโรงงานที่มีสัญญาณรบกวนสูง อัตราการส่งอยู่ที่ 1Mbits/S จะได้ระยะทางประมาณ 40M.และ ที่ 125Kbits/S
3) I2C port ถูกออกแบบโดยบริษัทPhilip สำหรับติดต่อสื่อสารกับอุปกรณ์ต่างๆ ผ่านสายสัญญาณสองเส้น คือ SCL, SDA มีการนำไปใช้งานกันอย่างกว้างขวาง เช่น Masterboard, Embedded System, Mobile Phone เป็นต้น โดยที่ Maximum Voltage = 5 Volts และ ต่ำสุดที่ 3.3 Volts โดยการสื่อสารจะเป็นแบบสองทางทั้ง in/out บน SDA ความเร็วสูงสุดที่ทำได้อยู่ที่ 400Mbits/S
4) Expansion portถูกออกแบบสำหรับบุคคล ที่ต้องการออกแบบส่วนต่อขยายสำหรับงานโดยเฉพาะ เช่น ใช้งานกับเครื่องคอมพิวเตอร์พื้นฐาน เป็นต้น
5) MMC port SNAPได้ออกแบบให้สามารถใช้งานMMC (Multi Media Card)หรือ SD (Secure Digital Cards) ได้ โดยกำหนดโครงสร้างของการเก็บข้อมูลอยู่ในรูปแบบ FAT16 ซึ่งคุณสามารถนำไปใช้เก็บ Application Program หรือ Data ได้ตามความต้องการ
6) Local Area Network ใช้สำหรับการติดต่อสื่อสารบนระบบโครงข่าย เป็นแบบ 10/100 Base-T Ethernet รวมถึงใช้ในการส่ง Application Program ไปยัง Memory ของ SNAP ผ่านช่องทาง FTP ซึ่ง SNAP สามารถรองรับการทำงานของ FTP Server, HTTP Server, Telnet Server, Serial Server ได้ในเวลาเดียวกัน
7) Terminal Port ใช้ในการควบคุมการทำงานของSNAP เช่น การสั่ง execute Application Program หรือการใช้ Command ต่างของOS เช่น การใช้คำสั่ง ipconfig เพื่อกำหนดค่าของ IP Address, Subnet Mask, Gateway หรือคำสั่ง mkdir เพื่อสร้างSubdirectory

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.2 คุณสมบัติของSNAP (Simple Network Application Platform)

2.4.2.1 คุณสมบัติทางด้าน Hardware

- รองรับแหล่งจ่ายไฟมาตรฐาน 5V
- บอร์ดถูกออกแบบให้ใช้พลังงานต่ำ 3.3V
- 72 pin SIMM board, 31.8 mm. x 102.9 mm.
- 10/100 Base T Ethernet
- 1x RS232, 115.2 kbit/s
- I2C, High speed 3.4 Mbit/s
- 2 Mbyte flash memory
- 8 Mbyte DRAM
- Real time clock and calendar
- RoHs compliant

2.4.2.2 สำหรับอุปกรณ์ SNAP มีรายละเอียดของส่วนประกอบและการเชื่อมต่อ

- Build บน Socket Standard 72-pins SIMM from Factor
- Plugs into TINI compatible socket boards
- SUN certified J2ME-CLDC
- SNAP runtime environment
- Native Java execution
- IEEE 454 floating point acceleration
- Extensive I/O capabilities through Java APIs
- TCP/IP stack
- General-purpose digital I/O
- 10/100 Base-T Ethernet
- Real time clock and Calendar
- 2 Mbytes flash memory
- Three serial ports
- Dual 1-Wire interfaces
- CAN (Controller Area Network)
- High-speed I2C bus and SPI
- Single +5V Power supply
- Connector for optional C/Java/Assembler debug SW

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.3 การประยุกต์ใช้งานของSNAP (Simple Network Application Platform)

- การควบคุมเครื่องหยอดเหรียญ
- ตรวจสอบติดตามควบคุมGPRS และ GPS
- การวินิจฉัยทางการแพทย์ (Medical Diagnostics)

2.5 J2ME (Java 2 Platform, Micro Edition)

J2ME (Java 2 Platform, Micro Edition) เป็นเทคโนโลยีที่ทำให้โปรแกรมเมอร์สามารถ ใช้จาวาและเครื่องมือที่เกี่ยวข้องในการพัฒนาโปรแกรมสำหรับอุปกรณ์ไร้สาย (Mobile Devices) เช่น โทรศัพท์มือถือและพีดีเอJ2ME ประกอบด้วยเครื่องจักรเสมือน (Virtual Machine) ขนาดเล็กที่เรียกว่า KVirtual Machine ทำให้โปรแกรมที่พัฒนาขึ้นโดย J2ME สามารถรันบนอุปกรณ์ไร้สายได้ สามารถแบ่งส่วนการทำงานเป็น 2 ส่วน คือ CLDC (Connected , Limited Device Configuration) และ MIDP (Mobile Information Device Profile) โดยจะมีการแบ่งหน้าที่กันทำงาน โดย CLDC ดูแลเกี่ยวกับ API และคุณสมบัติของVirtual Machine ที่จำเป็นต้องใช้ในการรันโปรแกรมบนอุปกรณ์ไร้สาย และสำหรับ MIDP จะเป็นส่วนที่เสริม CLDC ดูแลเรื่องส่วนติดต่อผู้ใช้ (UI), เน็ตเวิร์ค และคำสั่งที่ติดต่อกับอุปกรณ์ไร้สาย MIDP ยังครอบคลุมถึงแนวคิดของ midletหรือโปรแกรมจาวาขนาดเล็กที่คล้ายกับ applet แต่ใช้คุณสมบัติของ CLDC และ MIDP และคิดค้นขึ้นเพื่อจุดประสงค์ของงานอุปกรณ์ไร้สายเท่านั้นอุปกรณ์ที่สามารถใช้เทคโนโลยี J2ME ได้แบ่งออกเป็น 2 กลุ่มดังนี้ คือ Personal, mobile, connected, information devices เช่น โทรศัพท์มือถือ และเพจเจอร์เป็นต้น ซึ่งเป็นอุปกรณ์ที่มีการอินเตอร์เฟสแบบง่ายๆ มีหน่วยความจำประมาณ 128 – 512 กิโลไบต์และมีแบนด์วิดธ์ที่ต่ำ ส่วนการเชื่อมต่อของอุปกรณ์ไม่ได้ใช้โปรโตคอล TCP/IP และ Shared, fixed, connected, information devices เช่น อุปกรณ์เซตท็อป (set-top boxes) อินเทอร์เน็ตทีวี ความบันเทิงในรถยนต์และระบบการเดินเรือ เป็นต้น อุปกรณ์จำพวกนี้มี USER อินเทอร์เฟสให้ใช้หลายแบบมีหน่วยความจำประมาณ 2 – 16 เมกะไบต์ ส่วนการเชื่อมต่อกับระบบเครือข่ายนั้น โดยมากแล้วใช้โปรโตคอล TCP/IP ที่มีแบนด์วิดธ์ค่อนข้างสูง จุดมุ่งหมายที่สำคัญของการออกแบบภาษาจาวาคือ โปรแกรมต้องทำงานบนเครื่องต่างระบบกันได้ โดยเรียกคุณสมบัตินี้ว่า "ไม่ขึ้นกับระบบ" ซึ่งสามารถทำงานได้ทุกระบบปฏิบัติการ

2.5.1 โครงสร้างสถาปัตยกรรมของJ2ME

การออกแบบโครงสร้างเทคโนโลยีจาวาตระกูล J2ME มีการจัดแบ่งหน้าที่การทำงานออกเป็น 4 ระดับชั้นด้วยกัน เพื่อรองรับกับการใช้งานกับอุปกรณ์ต่างๆ คือ Operating System Java Virtual Machine Layer Configuration และ Layer Profile Layer

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้เพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระดับชั้นที่หนึ่ง Host Operating System จะเป็นส่วนของระบบปฏิบัติการ เช่นเดียวกับบนเครื่อง Desktop หรือ Laptop ที่มี Windows เป็นระบบปฏิบัติการ บนโทรศัพท์มือก็มีด้วยเช่นกัน อาทิ Nokia 7650, 3650 จะมี Symbian OS เป็นระบบปฏิบัติการ, เครื่อง Palm จะมี Palm OS เป็นระบบปฏิบัติการ

ระดับชั้นที่สอง Java Virtual Machines (JVM) จะเป็นส่วนของระบบจัดการ ที่ควบคุม และทำงาน ให้สามารถทำงานร่วมกันได้ ระหว่าง Java กับ Host Operating System โดยมากจะเป็นการแปลงจาก code Java ไปเป็นคำสั่ง ที่ Host Operating System เข้าใจ และทำงานร่วมกันได้

ระดับชั้นที่สาม Configuration เป็นกลุ่มของ Class Library ที่ครอบคลุมถึงอุปกรณ์ต่างๆ ที่อยู่ในกลุ่ม

ระดับชั้นที่สี่ Profiles เป็นกลุ่มของ คำสั่ง API (Application Programming Interface) ที่ใช้สำหรับอุปกรณ์ แต่ละประเภทโดยเฉพาะหัวใจหลักของโครงสร้าง J2ME สามารถแบ่งออกเป็น 2 ส่วน คือ Configurations และ Profiles ซึ่งมีลักษณะเป็นโมดูลที่มีความยืดหยุ่นพร้อมกับ สามารถนำไปใช้งานกับอุปกรณ์ที่ต่างกันได้ โดยสามารถปรับแต่งได้ตามความต้องการของผู้ใช้งานและนอกจากนี้ ผู้ผลิตหรือผู้พัฒนาซอฟต์แวร์บนอุปกรณ์ ยังสามารถขยายหรือเพิ่มความสามารถใน Virtual Machine ได้ ดังนั้นจึงไม่ได้เป็นแพลตฟอร์มเฉพาะกับอุปกรณ์ชนิดใดชนิดหนึ่ง

2.5.1.1 การตั้งค่า (Configurations)

สำหรับการตั้งค่าของ J2ME ที่ใช้งานกับอุปกรณ์ใด ๆ ต้องสามารถทำงานกับมาตรฐานที่กำหนดไว้ในกลุ่ม JCP และต้องสามารถใช้งาน Runtime Classes ตามข้อกำหนดได้ นอกจากนี้การตั้งค่ายังจะเป็นตัวกำหนดฟิวเจอร์หรือไลบารีมาตรฐานซึ่งจะมีเหมือนกันในทุกอุปกรณ์ที่จัดอยู่ในประเภทเดียวกันการจัดแบ่งกลุ่มคอนฟิกกูเรชันปัจจุบันมี 2 ประเภท คือ Connected Device Configuration (CDC) และ Connected, Limited Device Configuration (CLDC) โดยแต่ละตัวจะใช้ VM (Virtual Machine) ที่ต่างกันด้วย นอกจากนี้แล้วสำหรับอุปกรณ์ที่มีข้อจำกัดใน การทำงานจะเป็นบรรทัดฐานในการแบ่งกลุ่ม โดยคุณสมบัติพื้นฐานทั้งหมดนี้จะใช้ในการจัดแบ่งกลุ่มของอุปกรณ์ซึ่งมีความคล้ายกันในเรื่องของหน่วยความจำ หน้าจอ เครือข่ายในการเชื่อมต่อ และพลังงาน

1) Connected Device Configuration (CDC) ใช้หน่วยความจำอย่างน้อย 512 กิโลไบต์ สำหรับจาวา ใช้หน่วยความจำอย่างน้อย 256กิโลไบต์ ในขณะรันไทม์ (runtime) สามารถเชื่อมต่อสัญญาณที่มีแบนด์วิดธ์สูง (bandwidth) ตัวอย่างอุปกรณ์เหล่านี้ได้แก่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ ดัดแปลง เผยแพร่ หรือนำไปใช้ประโยชน์ในการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

อุปกรณ์เซตท็อป อินเทอร์เน็ตทีวี เครื่องซักผ้า ตู้เย็น ระบบเครื่องเสียงแบบ ไฮเอนต์ ระบบการเดินเรือ และความบันเทิงในรถยนต์ เป็นต้น

2) Connected, LimitedDevice Configuration(CLDC) ใช้หน่วยความจำ 128 กิโลไบต์สำหรับจาวา และใช้หน่วยความจำ 32 กิโลไบต์สำหรับรันไทม์ (runtime) มีข้อจำกัดเกี่ยวกับส่วนผู้ใช้งาน (user interface) ใช้พลังงานแบตเตอรี่ต่ำ (battery) สามารถเชื่อมต่อกับอุปกรณ์ไร้สายชนิดที่มีแบนด์วิดธ์ต่ำ(bandwidth)และเข้าถึงแบบไม่ต่อเนื่อง ตัวอย่างอุปกรณ์เหล่านี้ได้แก่ โทรศัพท์มือถือและเพจเจอร์ เป็นต้น

ตารางที่ 2.4 ตารางเปรียบเทียบระหว่าง CDC และ CLD

CDC	CLDC
- ทุกๆส่วนของ features และ APIs ของ Java	- บางส่วนของ features และ APIs ของ Java
- สำหรับอุปกรณ์ ในกลุ่ม High-end devices	- สำหรับอุปกรณ์ ในกลุ่ม Low-end devices
- หน่วยประมวลผล 32 bit	- หน่วยประมวลผล 16 และ 32 bit
- หน่วยความจำเครื่อง อย่างน้อย 2 MB	- หน่วยความจำเครื่อง 160-512 KB
- ใช้ CVM (Compact Virtual Machine)	- ใช้ KVM (Kilo Virtual Machine)

2.5.1.2 Profiles

เป็นหัวใจหลักอันสำคัญของเทคโนโลยี J2ME เนื่องจากลักษณะของการใช้งานในแต่ละงานนั้นแตกต่างกัน โดยถูกสร้างไว้อยู่เหนือระดับการตั้งค่าจึงเป็นตัวกำหนดฟังก์ชัน ในการทำงานพร้อมทั้งเป็นตัวกลางระหว่างแอพพลิเคชันกับสภาวะแวดล้อมของ J2ME ดังนั้น Profile จึงเกี่ยวข้องกับคุณลักษณะทางด้านฮาร์ดแวร์ของอุปกรณ์แต่ละตัว เช่น Profile ของอุปกรณ์ประเภทมือถือ (Mobile Information Device Profile) แสดงตามตารางที่ 2.2 สำหรับชุดของ API (Application Programming Interface) นี้ใช้สำหรับงานในแต่ละ vertical market ส่วนผู้ใช้งานคอมโพแนนซ์ คือ Input/Output, Event handling, Persistent storage, Networking และ Timers Profiles เป็นส่วนของ API และ Class ที่ใช้งานได้บนตัวของอุปกรณ์ แต่ละประเภทเป็นการขยายความสามารถของ Connected Device Configuration หรือ Connected, Limited Device Configuration มากขึ้น และมีส่วนของการทำงานที่เป็นลักษณะเฉพาะของอุปกรณ์นั้นๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

ตัวอย่างของ Profile ที่ใช้งานการพัฒนา เช่น MIDP (Mobile Information Device Profile) เป็น ประเภทของ Device ที่มีคุณสมบัติต่อไปนี้ small display (min. 96 x 54 pixels) ,มี touch screen หรือ keypad, สามารถ connect mobile network ด้วย bandwidth ที่จำกัด MIDP ประกอบด้วย APIs ที่ทำหน้าที่ดังต่อไปนี้

1) Mobile Information Device Next Generation (MIDP_NG) เป็นGeneration ที่จะพัฒนาเป็นตัวถัดไปของ MIDP ซึ่งจะเพิ่มคุณสมบัติต่าง ๆ ดังนี้
   - Domain security mode
   - HTTPS Support
   - Socket , datagrams support
   - มี OTA Provisioning
   - Push architectureโดยข้อความ หรือ เหตุการณ์ภายนอกถูกส่ง MIDlet
   - เพิ่มความสามารถในการ Control large screen
   - มี XML parser
   - มี Sound API Personal Digital Assistant Profile
   - (PDA Profile) สำหรับอุปกรณ์ประเภท Organizer
2) Foundation Profileสำหรับอุปกรณ์ในกลุ่มของ High-end device, เป็นส่วนขยายเพิ่มเติมเฉพาะด้านให้กับ CDC ซึ่งประกอบด้วย API และ Function พื้นฐาน เป็น Profile ที่พัฒนาบน CDC เหมาะสำหรับ Device ที่มีคุณสมบัติลักษณะต่อไปนี้
   - 1024k minimum ROM
   - 512k minimum RAM
   - Connectivity กับระบบ Network ได้
   - ไม่มี GUI ยกเว้นจะใช้ additional profile อื่น ๆ เพิ่มเติมเข้ามาเพื่อทำ GUI
3) Personal Profileสำหรับอุปกรณ์ในกลุ่มของHigh-end device, เป็นส่วนขยายเพิ่มเติม เฉพาะด้านให้กับ Foundation Profileจะประกอบด้วย การจัดการด้าน GUI
4) RMI Profile สำหรับอุปกรณ์ในกลุ่มของ High-end device, เป็นส่วนขยายเพิ่มเติมเฉพาะด้านให้กับ Foundation Profile จะประกอบด้วย การจัดการด้าน RMI (Remote Method Invocation)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.5 ตารางแสดง Profile และ Configuration

CDC	CLDC
- ทุกๆส่วนของ features และ APIs ของ Java	- บางส่วนของ features และ APIs ของ Java
- สำหรับอุปกรณ์ ในกลุ่ม High-end devices	- สำหรับอุปกรณ์ ในกลุ่ม Low-end devices
- หน่วยประมวลผล 32 bit	- หน่วยประมวลผล 16 และ 32 bit
- หน่วยความจำเครื่อง อย่างน้อย 2 MB	- หน่วยความจำเครื่อง 160-512 KB
- ใช้ CVM (Compact Virtual Machine)	- ใช้ KVM (Kilo Virtual Machine)

2.5.2 ประโยชน์ของJ2ME ในการพัฒนาแอพพลิเคชั่นบนอุปกรณ์ไร้สาย

เป็นแพลตฟอร์มอิสระ เนื่องจาก อุปกรณ์ไร้สายแต่ละชนิดนั้นมีขนาด และ หน้าที่การทำงานที่แตกต่างกัน แอพพลิเคชั่นใช้งานที่ติดตั้งพร้อมกับอุปกรณ์สื่อสารไร้สายส่วนใหญ่ได้รับการพัฒนา โดยใช้ไลบรารี เฉพาะระบบที่ผู้ผลิตจัดเตรียมไว้ให้ ภาษาที่ใช้ในการเขียนแอพพลิเคชั่นมีหลากหลายภาษา ตั้งแต่ C++ , Virtual , Basic ไปจนถึงภาษาที่มีสคริปต์เฉพาะตัวแอพพลิเคชั่นที่เขียนขึ้นเพื่อใช้กับอุปกรณ์ชนิดใดชนิดหนึ่งไม่สามารถนำไปใช้กันอุปกรณ์อื่นๆได้ ในขณะที่ J2ME นั้นนำปรัชญาเก่าแก่ในการออกแบบโปรแกรม เขียนเพียงครั้งเดียว แล้วสามารถนำไปใช้ที่ใดก็ได้ (WriteOnce ,Run Anywhere) มาสู่โลกอุปกรณ์ไร้สาย แอพพลิเคชั่นบนอุปกรณ์ไร้สายที่พัฒนาด้วยจาวา สามารถทำงานบนอุปกรณ์ที่ผลิตโดยผู้ผลิตรายอื่นได้ช่วยให้เคลื่อนย้ายไปใช้งานอื่นได้สะดวกยิ่งขึ้น

สามารถเขียนแอพพลิเคชั่นด้วย ภาษาที่ง่ายขึ้นเทคโนโลยีจาวาช่วยประหยัดเวลา และต้นทุนในการพัฒนาแอพพลิเคชั่น ทำให้ผลงานที่ได้มีประสิทธิภาพมากขึ้น สำคัญในตลาดที่มีการแข่งขันสูงและก้าวหน้าอย่างรวดเร็ว ประกอบด้วย ฟังก์ชั่นด้านเครือข่ายจำนวนมาก แอพพลิเคชั่นบนอุปกรณ์ไร้สายจะทำงานเกี่ยวเนื่องกับ ระบบเครือข่าย เพื่อให้ผู้ใช้งานสามารถติดต่อสื่อสารกับโลกภายนอกได้จากทุกๆที่ในทันที จาวานั้นได้รับการออกแบบให้มีความสามารถด้านเครือข่ายในตัวเอง โดยมีไลบรารีระบบเครือข่ายมากมายทำให้การเขียนแอพพลิเคชั่นด้านเครือข่ายทำได้ง่ายกว่าเดิม

มีระบบรักษาความปลอดภัยในตัวจาวา มีระบบรักษาความปลอดภัยหลายระดับตั้งแต่ class loader byte code verifier ไปจนถึงโปรแกรมรักษาความปลอดภัย เช่น Security Manager ซึ่งสามารถป้องกัน client จากแอพพลิเคชั่นที่ไม่น่าไว้วางใจ นอกจากนี้ ยังมี APIs รักษาความปลอดภัย การรับส่งข้อมูลผ่านเว็บอีกด้วยวิธีนี้ช่วยให้การทำธุรกรรมการค้า และการเงินผ่านโทรศัพท์เคลื่อนที่มีความปลอดภัยมากยิ่งขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นแอพพลิเคชั่นที่แพร่หลาย จาวาได้รับความนิยมอย่างสูง เมื่ออินเตอร์เน็ตกลายเป็นแพลตฟอร์มหลักสำหรับแอพพลิเคชั่น จาวาถือเป็นตัวเลือกในการพัฒนาแอพพลิเคชั่น นอกจากนี้แอพพลิเคชั่นที่พัฒนาด้วย J2ME ยังสามารถประสานเข้ากับแอพพลิเคชั่นที่พัฒนาด้วย J2EEได้ง่าย ช่วยสนับสนุนการออกแอพพลิเคชั่นบนอุปกรณ์ไร้สาย สำหรับองค์กรโดยแอพพลิเคชั่นบนอุปกรณ์ไร้สายมักเป็น clientขนาดเล็ก การที่ XMLได้รับการสนับสนุนในจาวา ทำให้แอพพลิเคชั่นแบบ client/server หรือแอพพลิเคชั่นติดต่อสื่อสารระหว่างกันเกิดขึ้นได้ในอุปกรณ์ไร้สาย

2.6 การสร้างเว็บเพจด้วยภาษา HTML

HTML ย่อมาจากคำว่า "HyperText Markup Language" เป็นภาษาที่ใช้ในการเขียนโปรแกรมภาษาหนึ่งของคอมพิวเตอร์ที่แสดงผลในลักษณะของเว็บเพจ ซึ่งสามารถแสดงผลได้ในรูปแบบต่างๆ ไม่ว่าเป็นภาพกราฟิกภาพนิ่ง ภาพเคลื่อนไหว เสียง หรือการเชื่อมโยงไปยังเว็บไซต์อื่นๆ ภาษาHTML เป็นภาษาที่มีลักษณะของโค้ดกล่าวคือ จะเป็นไฟล์ที่เก็บข้อมูลที่เป็นตัวอักษรในมาตรฐานของรหัสแอสกี (ASCII Code) โดยเขียนอยู่ในรูปแบบของเอกสารข้อความจึงสามารถกำหนดรูปแบบ และโครงสร้างได้ง่าย

2.6.1 ขั้นตอนในการสร้างเว็บเพจ

- เปิดโปรแกรมTextEditorแล้ว ทำการพิมพ์คำสั่ง HTML แล้วบันทึกเป็นไฟล์นามสกุล .htmหรือ .html
- เปิดโปรแกรมWebBrowserเพื่อใช้ในการดูผลลัพธ์ที่ได้จากการเขียนภาษา HTML จากที่ได้เขียนจาก TextEditor

2.6.2 โครงสร้างภาษา HTML

การเขียนภาษา HTML นั้นมีส่วนประกอบหลักอยู่ 2 ส่วนคือ ส่วนที่เป็นเนื้อหา และส่วนที่เป็นคำสั่ง ส่วนที่เป็นคำสั่งนั้นจะอยู่ในรูปของ Tag ซึ่งจะเขียนอยู่ในเครื่องหมาย มากกว่า และน้อยกว่า<>แต่ละ Tag มีหน้าที่ที่แตกต่างกันออกไป Tag แบ่งออกเป็น 2 ประเภท คือ

- แท็กเดี่ยว คือ คำสั่งที่มีคำสั่งเพียงอย่างเดียวซึ่งสามารถใช้และสิ้นสุดคำสั่งได้ด้วยตัวของมันเองเช่น

  ข้อความ...

  <hr>

  <! - ข้อความ ->
- แท็กคู่ คือคำสั่งที่ต้องมีส่วนเริ่มต้นและส่วนจุดจบของคำสั่งนั้นๆโดยแท็กที่เป็นส่วนจบจะมีเครื่องหมาย slash / ติดไว้ เช่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

<html>ส่วนของเนื้อหา .....</html>

ข้อความ.....

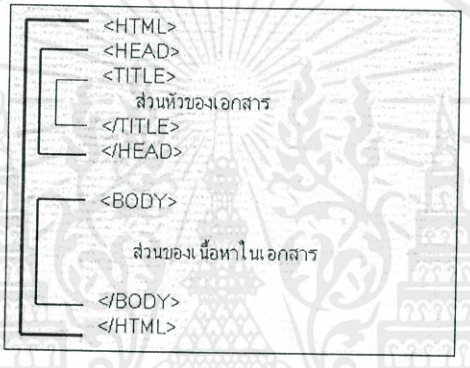
ข้อความ....

ถ้าหากมีการใช้แท็กคู่หลายๆ คำสั่ง เช่นคำสั่งตัวขีดเส้นใต้<U> .... </U>และตามด้วยคำสั่ง ตัวเอียง<I>....</I>จะต้องปิดคำสั่งตัวเอียงก่อนแล้วจึงจะมาปิดคำสั่งตัวหนา***

<I> U>ข้อความ....</U></I>

2.6.3 การกำหนดโครงสร้างหลัก

การจัดวางที่เห็นเป็นการจัดวางแบบมาตรฐานซึ่งประกอบไปด้วย 4 ส่วน เริ่มเขียนภาษา HTML ควรเริ่มจากส่วนนี้ก่อนทุกครั้ง



รูปที่ 2.8 รูปการจัดวางแบบมาตรฐานของ HTML ทั่วไป

- คำสั่งหลัก<HTML> ....</HTML>เป็นคำสั่งที่ไว้กำหนดจุดเริ่มต้นและจุดของเอกสาร
- คำสั่งหลัก<HEAD> ....</HEAD>เป็นคำสั่งที่ทำหน้าที่กำหนดส่วนหัวเรื่องโดยภายในคำสั่งนี้จะประกอบไปด้วย
- คำสั่งหลัก<TITLE> .... </TITLE>เป็นคำสั่งที่ใช้กำหนดข้อความที่ต้องการให้ขึ้นอยู่ในส่วนของ Title Bar โดยสามารถพิมพ์ได้ยาว 64 ตัวอักษร
- คำสั่งหลัก<BODY> .... </BODY>เป็นคำสั่งที่ใช้ในการกำหนดรูปแบบของเอกสารทั้งหมดว่าจะให้มีลักษณะอย่างไร

2.6.4 โครงสร้างคำสั่งของ HTML

การใช้งานรายละเอียดคำสั่งของ HTML โดยการใช้งานหลักจะมีดังนี้

1) คำสั่ง หรือ Tag

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Tag เป็นลักษณะเฉพาะของภาษา HTML ใช้ในการระบุรูปแบบคำสั่ง หรือการลงรหัสคำสั่ง HTML ภายในเครื่องหมาย less-than bracket (<) และ greater-than bracket (>) โดยที่ Tag HTML แบ่งได้ 2 ลักษณะ คือ Tag เดี่ยว เป็น Tag ที่ไม่ต้องมีการปิดรหัส เช่น <HR>,
 เป็นต้น

Tag เปิด/ปิด รูปแบบของ tag นี้จะเป็นแบบ <tag> .... </tag> โดยที่

<tag> เราเรียกว่า tag เปิด

</tag> เราเรียกว่า tag ปิด

2) Attributes

Attributes เป็นตัวบอกรายละเอียดของ tag นั้นเช่น   เป็นการบอกว่าให้อักษรที่อยู่ใน tag นี้ชิดซ้าย

3) not case sensitive

หมายถึง คุณจะพิมพ์
 หรือ
 ก็ได้ ผลลัพธ์ออกมาไม่ต่างกัน

2.6.5 โครงสร้างหลักของ HTML

โครงสร้างหลักของ HTML ก็จะเริ่มด้วย <html> และจบด้วย </html> เสมอ ซึ่งชุดคำสั่งที่ใช้จะแยกเป็น 2 ส่วนคือ

1) head คำสั่งที่อยู่ในส่วนนี้จะใช้บรรยายรายละเอียดเกี่ยวกับ web page ซึ่งจะไม่แสดงผลที่ web page โดยตรง
2) body คำสั่งที่อยู่ในส่วนนี้จะใช้ในการจัดรูปแบบตัวอักษร จัดหน้า ใส่รูปภาพ ซึ่งตัวอักษรในส่วนนี้จะแสดงที่ web browser โดยตรง

2.6.5.1 คำสั่งในหัวข้อของ head (Head Section)

Head Section เป็นส่วนที่ใช้อธิบายเกี่ยวกับข้อมูลเฉพาะของหน้าเว็บนั้นๆ เช่น ชื่อเรื่องของหน้าเว็บ (Title), ชื่อผู้จัดทำเว็บ (Author), คีย์เวิร์ดสำหรับการค้นหา (Keyword) โดยมี Tag สำคัญ คือ

```
<HEAD>
    <TITLE>ข้อความอธิบายชื่อเรื่องของเว็บ</TITLE>
    <META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=utf-8">
    <META NAME="Author" CONTENT="ชื่อผู้พัฒนาเว็บ">
    <META NAME="KeyWords" CONTENT="ข้อความ 1, ข้อความ 2 ">
</HEAD>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.9 คำสั่งในหัวข้อของ Head

TITLE ข้อความที่ใช้เป็น TITLE ไม่ควรพิมพ์เกิน 64 ตัวอักษร, ไม่ต้องใส่ลักษณะพิเศษ เช่น ตัวหนา, เอียง หรือสี โดยข้อความในส่วนนี้จะแสดงผลใน title bar ของ web browser META Tag META จะไม่ปรากฏผลบนเบราเซอร์ แต่จะเป็นส่วนสำคัญ ในการจัดอันดับบัญชีเว็บ สำหรับผู้ให้บริการสืบค้นเว็บ (Search Engine เช่น google , yahoo) charset=TIS-620 ใช้บอกว่าใช้ชุดตัวอักษรแบบใดในการแสดงผล ภาษาไทยเราใช้ charset=TIS-620 หรืออาจเป็น charset=windows-874 ก็ได้ ตอนนี้แนะนำให้ใช้เป็น charset=utf-8 keyword ดังภาพด้านบน จะเห็นว่าเราสามารถใช่ keywords มากกว่า 1 คำได้โดยใช้เครื่องหมาย (,) ในการคั่นระหว่างคำ การพิมพ์ชุดคำสั่ง HTML สามารถพิมพ์ได้ทั้งตัวพิมพ์เล็ก ตัวพิมพ์ใหญ่ หรือผสม การย่อหน้า เว้นบรรทัด หรือช่องว่าง สามารถกระทำได้อิสระ โปรแกรมเบราเซอร์จะไม่สนใจเกี่ยวกับระยะเว้นบรรทัด หรือย่อหน้า หรือช่องว่าง

2.6.5.2 คำสั่งในส่วนของ (Body Section)

Body Section เป็นส่วนเนื้อหาหลักของหน้าเว็บ ซึ่งการแสดงผลจะต้องใช้ Tag จำนวนมาก ขึ้นอยู่กับลักษณะของข้อมูล เช่น ข้อความ, รูปภาพ, เสียง, วีดิโอ หรือไฟล์ต่างๆ

ส่วนเนื้อหาเอกสารเว็บ เป็นส่วนการทำงานหลักของหน้าเว็บ ประกอบด้วย Tag มากมายตามลักษณะของข้อมูล ที่ต้องการนำเสนอ การป้อนคำสั่งในส่วนนี้ ไม่มีข้อจำกัดสามารถป้อนติดกัน หรือ 1 บรรทัดต่อ 1 คำสั่งก็ได้ แต่มักจะยึดรูปแบบที่อ่านง่าย คือ การทำย่อหน้าในชุดคำสั่งที่เกี่ยวข้องกัน ทั้งนี้ให้ป้อนคำสั่งทั้งหมดภายใต้ Tag <BODY> </BODY> และแบ่งกลุ่มคำสั่งได้ดังนี้

- กลุ่มคำสั่งเกี่ยวกับการจัดรูปแบบเอกสาร
- กลุ่มคำสั่งจัดแต่ง/ควบคุมรูปแบบตัวอักษร
- กลุ่มคำสั่งการทำเอกสารแบบรายการ (List)
- กลุ่มคำสั่งเกี่ยวกับการทำลิงค์
- กลุ่มคำสั่งจัดการรูปภาพ
- กลุ่มคำสั่งจัดการตาราง (Table)
- กลุ่มคำสั่งควบคุมเฟรม
- กลุ่มคำสั่งอื่นๆ

2.6.6 การใส่รูปภาพลงไปในเอกสาร (IMAGE)

ใช้ในการแสดงรูปภาพกราฟฟิกโดยจะต้องเป็นไฟล์กราฟฟิกที่เว็บบราวเซอร์รู้จัก เช่น GIF,JPEG,XPM,XBM เป็นต้น

ตัวอย่าง code <imgsrc="url">

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<img align="top | middle | bottom | left | right">การกำหนดตำแหน่งให้กับรูปภาพ

<img width="pixels">การกำหนดความกว้างรูปภาพ

<img height="pixels">การกำหนดความสูงรูปภาพ

<img border="number">การกำหนดความสูงรูปภาพ

<imghspace="number">การกำหนดช่องว่างแนวนอนระหว่างรูปภาพกับข้อความ

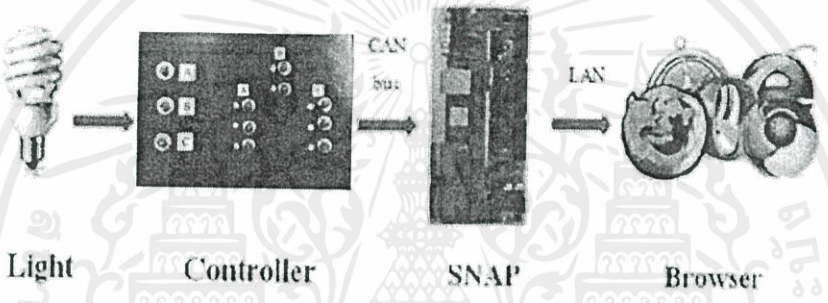
<imgvspace="number">การกำหนดช่องว่างแนวตั้งระหว่างรูปภาพกับข้อความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
วิธีการดำเนินงาน

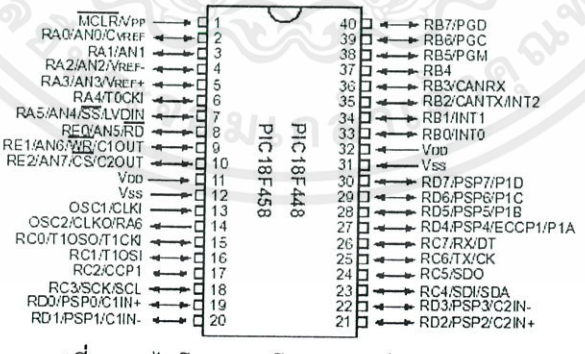
3.1 บทนำ

การออกแบบระบบไฟส่องสว่างนั้น จะมีโหมดฟังก์ชันการสั่งงานคือ โหมดที่สั่งการด้วยตนเอง เมื่อมีการกดปุ่มก็จะส่งค่าผ่าน CAN bus ไปยัง SNAPโดยมีเว็บบราวเซอร์เป็นตัวแสดงสถานะของหลอดไฟแต่ละดวง อีกทั้งกล่องควบคุมการเปิด-ปิดไฟส่องสว่างนี้ก็จะมีฟังก์ชันการใช้งาน 2 แบบดังนี้ สามารถที่จะเปิด-ปิดได้ทีละดวงและสามารถเปิด-ปิดเป็นกลุ่มได้



รูปที่ 3.1 โครงสร้างการทำงานของระบบควบคุมไฟส่องสว่าง

3.2 ไมโครคอนโทรลเลอร์ตระกูล PIC18F458



รูปที่ 3.2 ไมโครคอนโทรลเลอร์PIC18F458

ไมโครคอนโทรลเลอร์ตระกูล PIC18F458 ซึ่งเป็นตัวไมโครคอนโทรลเลอร์ที่ใช้กันในตระกูล PIC (Peripherral Interface Controller) ในปัจจุบันซึ่งมีศักยภาพในการทำงานสูงและในคุณสมบัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

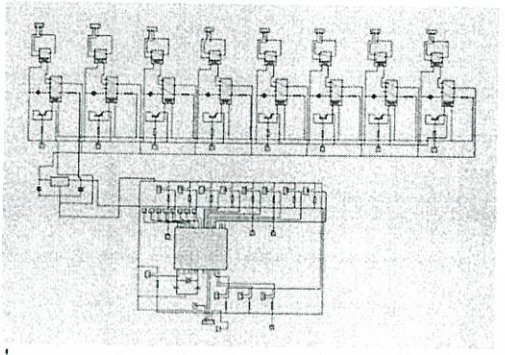
ของไมโครคอนโทรลเลอร์เบอร์นี้คือเพียบพร้อมไปด้วยทรัพยากรหรือฟังก์ชัน(Function) การใช้งานต่างๆไว้ภายในตัวมันเอง เช่น มีโมดูล(Module)แปลงสัญญาณแอนะลอกเป็นดิจิตอล (Analog To Digital Converter),USART,SPI,I^2C,PWMอื่นๆมีโมดูลที่เพิ่มเติมขึ้นมาใหม่คือโมดูล Can, ECCP ซึ่งเป็นคุณสมบัติใหม่ของไมโครคอนโทรลเลอร์ตระกูล PIC โดยในไมโครคอนโทรลเลอร์PIC18F458และเบอร์อื่นๆของ PIC18FXXX และยังเหมาะต่อการใช้งานตรงที่ หน่วยความจำโปรแกรมเป็นหน่วยความจำแบบเฟลช (Flash Program Memory) ซึ่งสามารถเขียนและลบข้อมูลด้วยสัญญาณไฟฟ้าได้นับหลายพันครั้งข้อเด่นอีกประการหนึ่งของไมโครคอนโทรลเลอร์เบอร์นี้ ในเรื่องของความเร็ว PIC18F458 สามารถทำงานได้ที่ความถี่สัญญาณนาฬิกาสูงถึง 40 เมกะเฮิรตซ์และมีวงจร PLL (Phase Lock Loop) ซึ่งเป็นวงจรคูณความถี่อยู่ภายในโดยสามารถเลือกโดยการโปรแกรมทางซอฟต์แวร์ ซึ่งสามารถคูณค่าความถี่ที่รับเข้ามาได้ถึง4เท่าของสัญญาณนาฬิกาจากภายนอกทั้งยังทำงานในลักษณะไปป์ไลน์ทำให้มีความเร็วในการทำงานมากกว่าซีพียูทั่วไปที่มีความถี่เดียวกัน ในลักษณะการทำงานจะใช้สัญญาณนาฬิกาเพียง 1 หรือ 2 ไซเคิลต่อคำสั่งเท่านั้นและหน่วยความจำไม่ถูกแบ่งเป็นเพจ) ไมโครคอนโทรลเลอร์ PIC18F458 มีคุณสมบัติสำคัญดังนี้

- ทำงานได้สูงสุดที่ 40 MHz.
- หน่วยความจำโปรแกรมเป็นแบบFlash มีขนาด 32 กิโลไบต์
- มีแรมขนาด 1536 ไบต์
- มีEEPROM ขนาด 256 ไบต์
- มีระบบ Code Protection
- มี CAN คอนโทรลเลอร์ในตัว
- สามารถเปิดโปรแกรมด้วยไฟ +5 VDC ได้
- ทำงานที่ไฟเลี้ยงตั้งแต่ 2 VDC ถึง 5.5 VDC
- มีระบบ UART สำหรับต่อกับการสื่อสารแบบ RS232
- มี I/O พอร์ตทั้งหมด 5 พอร์ต

จากคุณสมบัติที่สำคัญข้างต้นจึงได้มีการเลือกใช้ไมโครคอนโทรลเลอร์ PIC18F458 เนื่องจากมีคุณสมบัติและพอร์ตที่ใช้เพียงพอต่อการทำโครงงานและมีพอร์ต CAN ซึ่งจำเป็นต้องใช้ในการจัดทำโครงงานโดยจะใช้การส่งข้อมูลผ่านทาง CAN จึงมีความประสงค์ที่จะใช้ไมโครคอนโทรลเลอร์ที่มี CAN อยู่ภายในและในการจัดทำโครงงานนี้มีความต้องการที่จะใช้พอร์ต I/O มากกว่า 3 พอร์ต

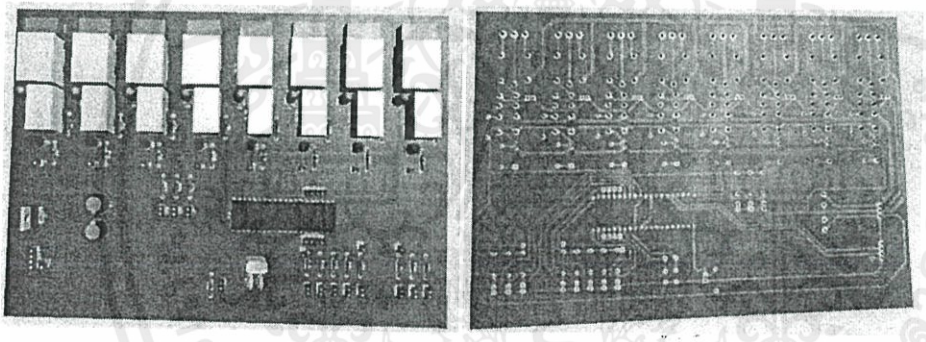
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 การออกแบบ



รูปที่ 3.3 ฮาร์ดแวร์การควบคุมเปิด-ปิดไฟส่องสว่าง

ในการออกแบบอุปกรณ์ควบคุมไฟเริ่มแรกด้วยการศึกษาอุปกรณ์ต่างๆที่ใช้ หลังจากนั้น ออกแบบและทดลองเขียนวงจรลงในโปรแกรม ต่อขาของอุปกรณ์ให้ครบ และเขียนแบบ PCB

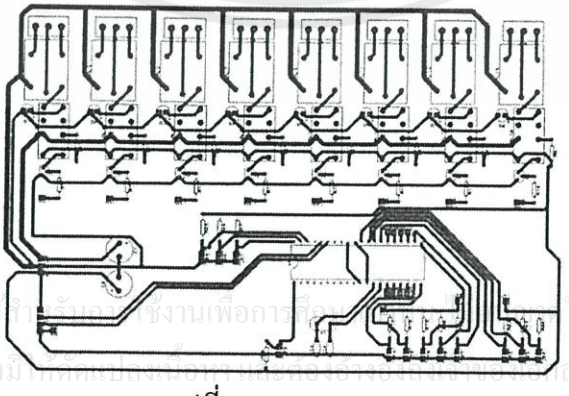


(ก) (ข)

รูปที่ 3.4 ด้านหน้าและด้านหลังแสดงในรูปแบบ 3 มิติ

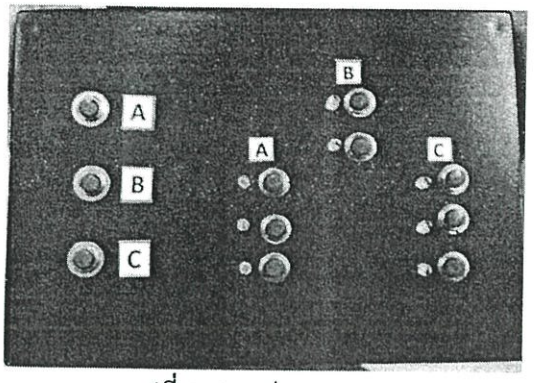
(ก) ด้านหน้า

(ข) ด้านหลัง



รูปที่ 3.5 ลายวงจร

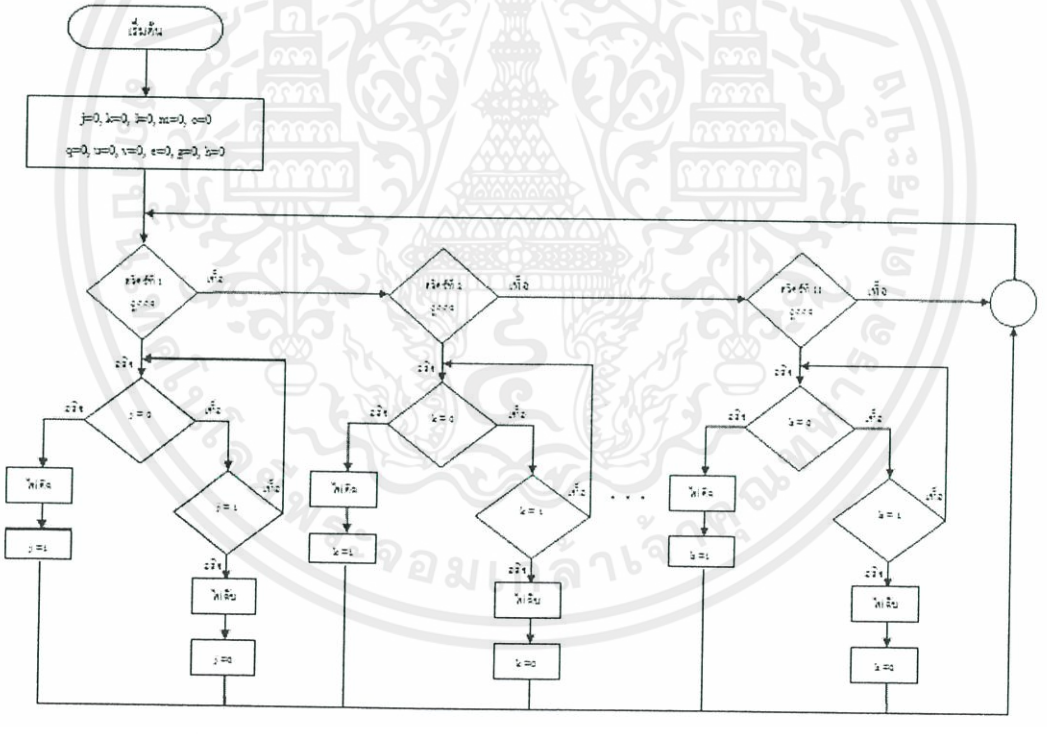
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.6 กล่องควบคุม

ตัวอย่างกล่องที่ใช้ควบคุมระบบไฟในอาคาร

3.4 การทำงานของโปรแกรมภายในตู้ควบคุมการเปิด-ปิดไฟส่องสว่าง



รูปที่ 3.7 แผนผังการทำงานของตู้ควบคุมการเปิด-ปิดไฟส่องสว่าง

แผนผังการทำงานของตู้ควบคุมการเปิด - ปิดไฟส่องสว่างมีกระบวนการการทำงานที่สามารถตรวจเช็คจากการกดปุ่ม เมื่อมีการกดปุ่มก็จะรับค่าของข้อมูลเข้ามาแล้วแสดงสถานะของไฟส่องสว่างตามค่าที่ได้รับมาโดยค่าที่ส่งเข้ามานั้นจะทำงานในลักษณะของท็อกเกิ้ล

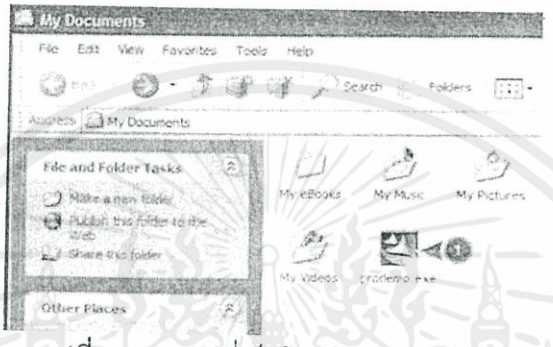
เอกสารนี้มีลิขสิทธิ์ ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ต้องอ้างอิงเอกสารทุกครั้งที่มีการนำไปใช้

3.5 วิธีการดำเนินงาน

สามารถแบ่งวิธีการดำเนินงานได้เป็นการติดตั้งการใช้งานและการเขียนโปรแกรมควบคุม

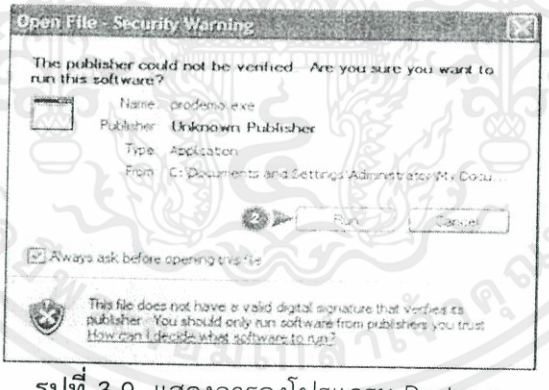
3.5.1 การติดตั้งและการตั้งค่าโปรแกรม Proteus

เมื่อดาวน์โหลดไฟล์มาแล้วให้เข้าไปยังที่เก็บไฟล์ไว้โดยตรงในที่นี้เก็บไว้ในไฟล์เอกสาร แล้วดับเบิ้ลคลิกที่ไอคอนหมายเลข 1



รูปที่ 3.8 แสดงที่เก็บโปรแกรม Proteus

จากนั้นคลิกที่ปุ่ม RUN เพื่อลงโปรแกรม Proteus



รูปที่ 3.9 แสดงการลงโปรแกรม Proteus

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

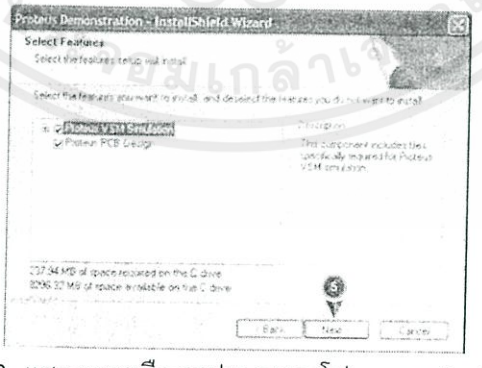
คลิกที่ปุ่ม Next

รูปที่ 3.10 แสดงการลงโปรแกรม Proteus(ต่อ)

คลิกที่ปุ่ม Browser เพื่อเลือกตำแหน่งเก็บโปรแกรมตามต้องการในที่นี้เลือกติดตั้งตามที่โปรแกรมจัดไว้ให้แล้วคลิกที่ปุ่ม Next

รูปที่ 3.11 แสดงการเลือกพื้นที่ในการติดตั้งโปรแกรม Proteus

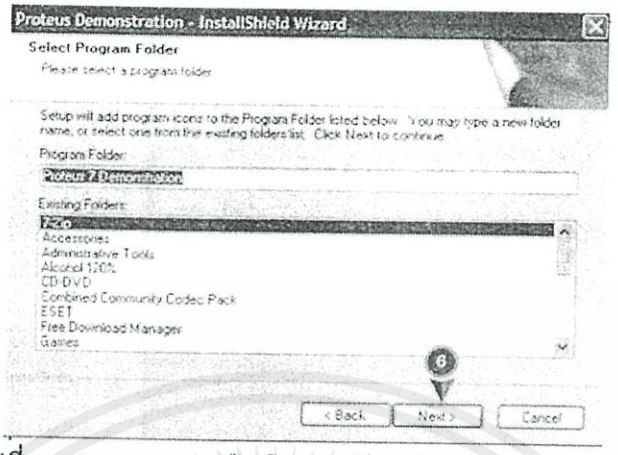
คลิกที่ปุ่ม Next



รูปที่ 3.12 แสดงการเลือกรูปแบบของโปรแกรม Proteus

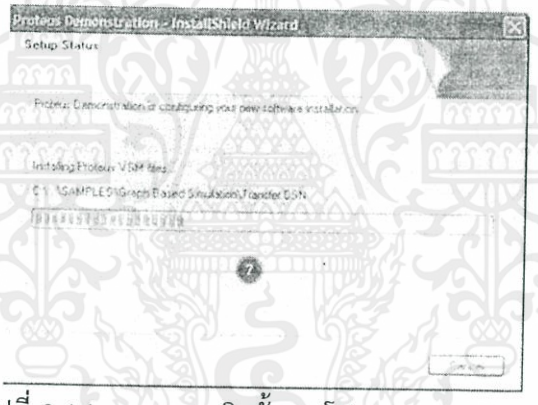
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คลิกที่ปุ่ม Next อีกครั้ง



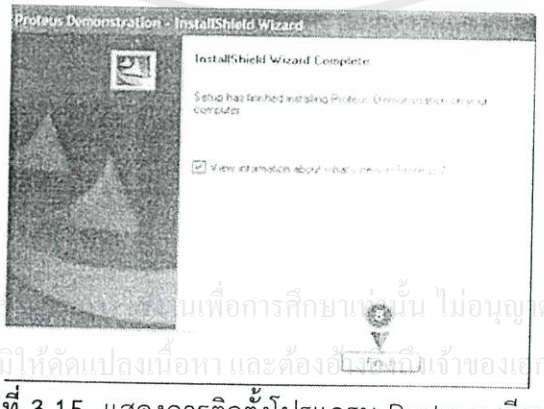
รูปที่ 3.13 แสดงการเลือกโฟลเดอร์ของโปรแกรม Proteus

โปรแกรมก็จะทำการติดตั้งลงบนคอมพิวเตอร์ทันที



รูปที่ 3.14 แสดงการติดตั้งของโปรแกรม Proteus

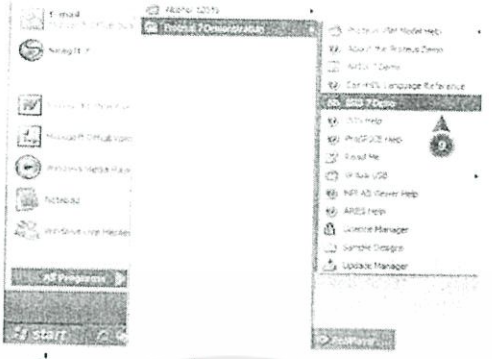
จากนั้นคลิกที่ปุ่ม Finish สิ้นสุดขั้นตอนการติดตั้ง



รูปที่ 3.15 แสดงการติดตั้งโปรแกรม Proteus เรียบร้อย

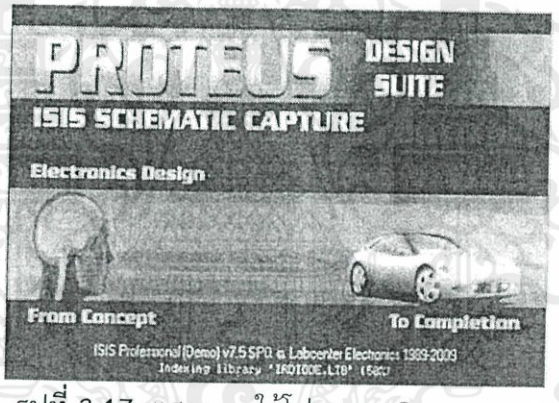
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อลงโปรแกรมแล้วให้ลองเปิดใช้โดยไปคลิกปุ่ม Start โปรแกรมทั้งหมด Proteus 7



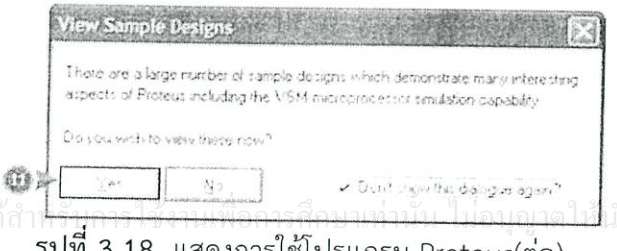
รูปที่ 3.16 แสดงไอคอนโปรแกรม Proteus

จะปรากฏ Proteus ISIS schematic ขึ้นมา ซึ่งเป็นโปรแกรมที่ใช้ออกแบบและจำลองการทำงานของวงจร รอสักครู่



รูปที่ 3.17 แสดงการใช้โปรแกรม Proteus

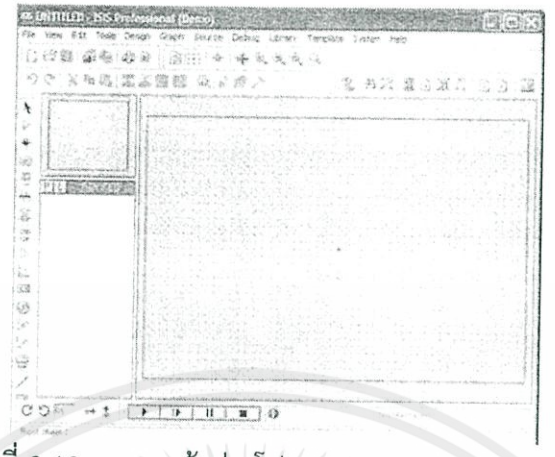
จะปรากฏไดอะล็อกบ็อกซ์ View Sample Designs ขึ้นมา ให้คลิกเครื่องหมายถูกที่ช่อง Don't show this dialogue again ? เพื่อครั้งต่อไป เมื่อเปิดโปรแกรมขึ้นมาใหม่ไดอะล็อก View Sample Designs จะไม่แสดงขึ้นมา จากนั้นคลิกที่ปุ่ม Yes



รูปที่ 3.18 แสดงการใช้โปรแกรม Proteus(ต่อ)

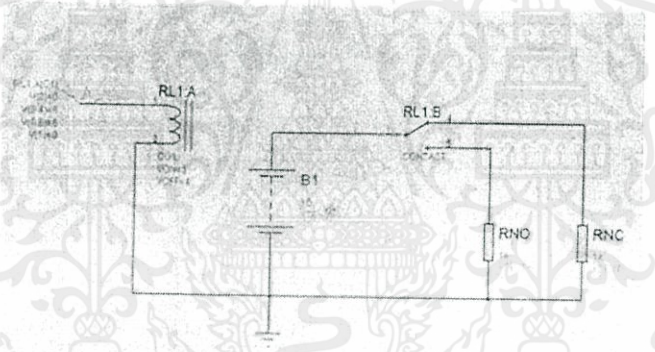
เอกสารนี้เป็นเอกสารที่สงวนไว้สำ... ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะพบหน้าต่างของโปรแกรม ISIS Professional แสดงขึ้นมา



รูปที่ 3.19 แสดงหน้าต่างโปรแกรม ISIS Professional

วงจรที่เราออกแบบ จะประกอบไปด้วยอุปกรณ์อิเล็กทรอนิกส์ต่างๆ มาวางบนพื้นทำงาน แล้วเชื่อมสายสัญญาณเข้าด้วยกัน จนเกิดเป็นวงจรดังรูป



รูปที่ 3.20 แสดงวงจรที่ออกแบบในโปรแกรม ISIS Professional

กลุ่มของอุปกรณ์ที่แยกออกไว้แต่ละประเภท เพื่อให้ง่ายต่อการค้นหาเช่นเมื่อเลือกไลบรารี Capacitors ก็จะพบกับอุปกรณ์จำพวก Capacitors หลายชนิดหลายขนาดมากมายซึ่งเราสามารถเลือกใช้ได้ตามต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.21 แสดงอุปกรณ์ในโปรแกรม ISIS Professional

โปรแกรม Proteus จะแยกย่อยออกเป็นสองโปรแกรมหลักๆ คือ

1) โปรแกรม ISIS คือ โปรแกรมที่ใช้ออกแบบและจำลองการทำงานของวงจร

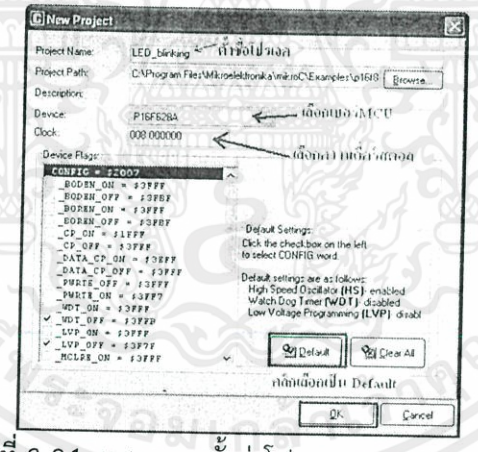
รูปที่ 3.22 แสดงไอคอนโปรแกรม ISIS

2) โปรแกรม ARES คือ โปรแกรมที่ใช้ออกแบบลายวงจรพิมพ์ (PCB) โดยนำวงจรจากโปรแกรม ISISเข้ามาออกแบบ ให้อยู่ในรูปเหมือนรูปอุปกรณ์จริงทุกอย่าง

รูปที่ 3.23 แสดงไอคอนโปรแกรม ARES

3.5.2 การติดตั้งและการตั้งค่าโปรแกรมMikroC

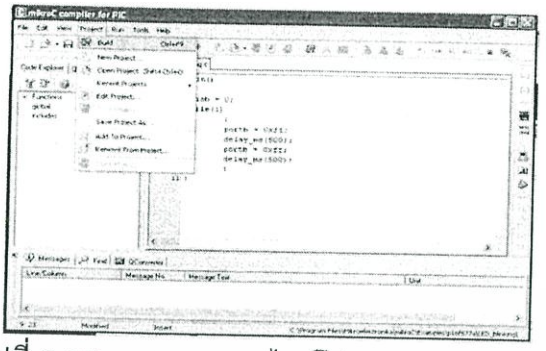
1) เปิดโปรแกรมไมโครซีขึ้นมาสร้างโปรเจคใหม่กรอกรายละเอียดแล้วคลิกตกลง



รูปที่ 3.24 แสดงการตั้งค่าโปรแกรม MikroC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

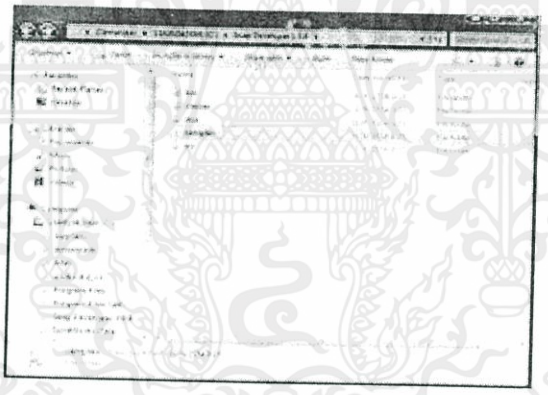
2) เขียนโค้ดโปรแกรมแล้วบันทึก ทำการคอมไพล์ไฟล์ที่เราเขียนโดยคลิกที่ Build



รูปที่ 3.25 แสดงการคอมไพล์โปรแกรม MikroC

3.5.3 การติดตั้ง Snap Developer

1. สามารถทำได้ด้วยการ copy subdirectory "Snap Developer 1.0.8" ไปยัง Drive ที่ต้องการ จะได้ sub directory ดังแสดงในรูปที่ 3.26



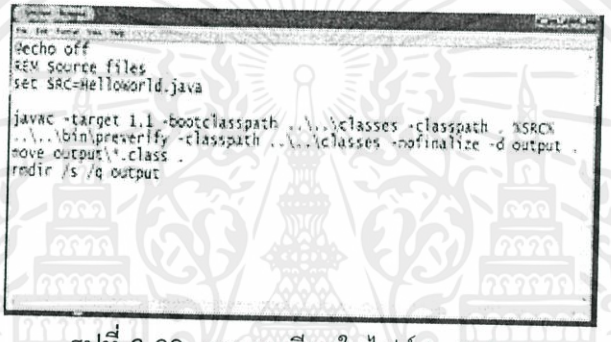
รูปที่ 3.26 แสดงตำแหน่ง directory ของ snap developer 1.0.8

2. ทดลอง Compile โปรแกรม HelloWorld โดยเปลี่ยน sub directory ไปที่ x://Snap Developer 1.0.8/samples/HelloWorld จะเห็นแฟ้มbuild.bat ให้ใช้ mouse เลือกที่ built.baจากนั้นจะปรากฏแฟ้ม HelloWorld.class ขึ้นมา เป็นอันเสร็จสิ้นการ Compile โปรแกรม HelloWorld ดังรูปที่ 3.27

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

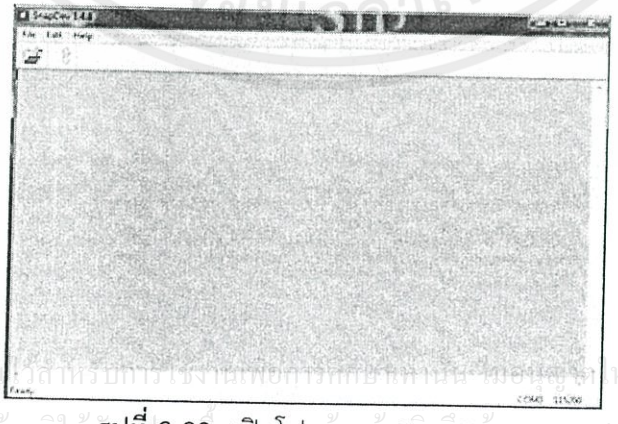
รูปที่ 3.27 รายละเอียดในไฟล์ built.bat ว่าด้วยขั้นตอน

3. รายละเอียดโค้ดภายในไฟล์แสดงดังรูปที่ 3.4



รูปที่ 3.28 รายละเอียดในไฟล์ built.bat

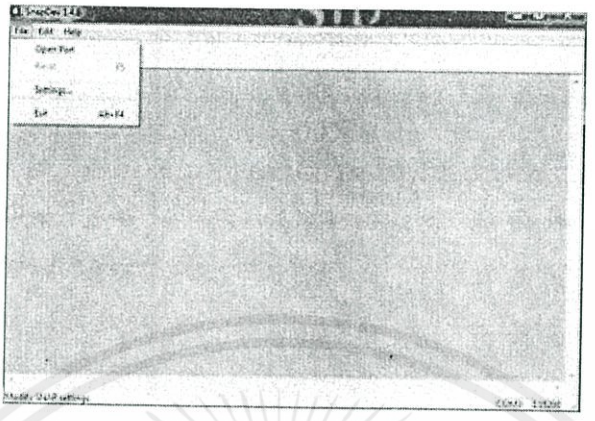
4. การติดต่อ SNAP Board กับ Complie ผ่าน SNAPDev ให้ประกอบ SNAP เข้ากับ Base board จากนั้นต่อสาย Terminal console โดยให้ใช้หัว RJ-11 ต่อเข้ากับ Connector RJ-11 ของ Base board ด้านหัว DB-9 ให้ต่อเข้ากับ Com1 ของ Computer จากนั้นเปิดโปรแกรม SnapDev (x:\Snap Developer 1.0.8\bin\snapdev.exe)



รูปที่ 3.29 เปิดโปรแกรม SnapDev

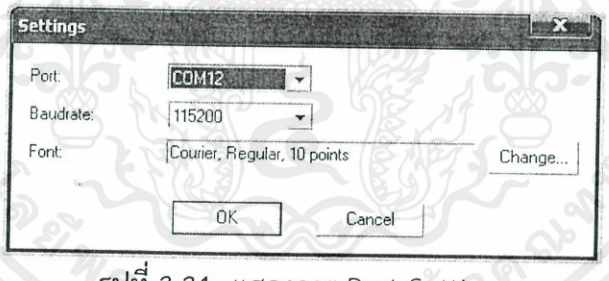
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5. กำหนด Configuration ให้กับโปรแกรม SnapDev โดย File แล้วเลือกไปที่ Settings ดังรูปที่ 3.6



รูปที่ 3.30 กำหนด Comport ที่ต้องการติดต่อกับ SNAP

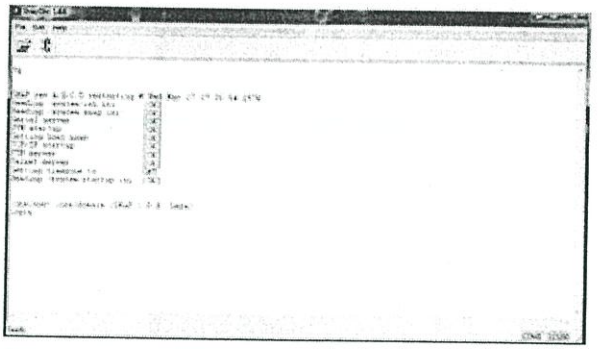
6. เลือก Comport ที่ต้องการใช้ติดต่อกับ SNAP จากนั้นให้กดปุ่ม Configuration เพื่อกำหนดค่า โดยกำหนด Baudrate = 115200 เมื่อกำหนดเสร็จ เรียบร้อยแล้วให้กดปุ่ม OK การกำหนดค่า Comport ดังรูปที่ 3.31



รูปที่ 3.31 แสดงการ Port Setting

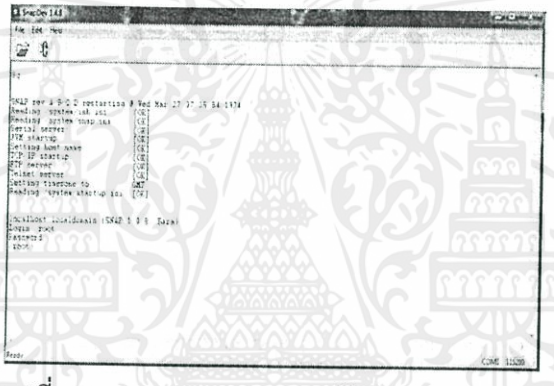
7. เลือกหัวข้อ File >> Open Port เพื่อให้โปรแกรม SnapDev พร้อมที่จะรับข้อมูลจาก SNAP board จากนั้นจ่าย Power ให้กับ SNAP board หรือไม่ก็ให้กดปุ่ม Hardware Reset ในกรณีที่จ่ายไฟให้กับ SNAP แล้ว ก่อนจะเปิด โปรแกรม SnapDev ดังรูปที่ 3.32

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



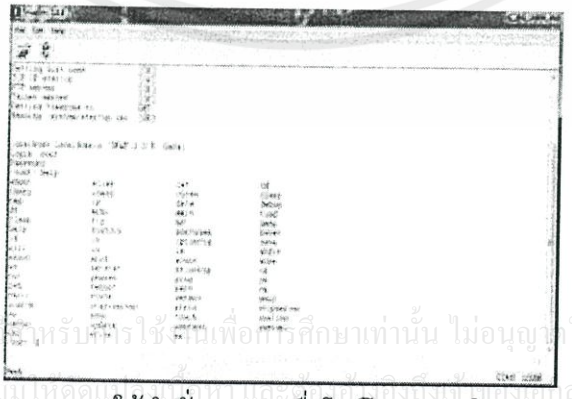
รูปที่ 3.32 แสดงหน้าจอหลังจากที่จ่ายไฟเข้าที่บอร์ด SNAP

8. ลอง login เข้าสู่ SNAP board พิมพ์ username = root, password = snap หลังจาก Login สำเร็จแล้วจะได้ prompt/root> ดังรูปที่ 3.33



รูปที่ 3.33 แสดงการ login เข้าสู่ SNAP board

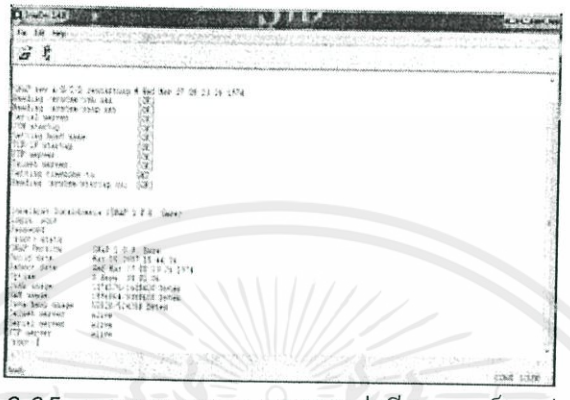
9. ทดสอบคำสั่ง จะได้เข้าใจระบบของ SNAP กันมากขึ้น พิมพ์ help เพื่อแสดงคำสั่งที่เก็บใน Sub Directory root แต่ถ้าเป็น Sub Directory อื่น ให้ใช้คำสั่ง ls แทน ตามรูปที่ 3.34



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

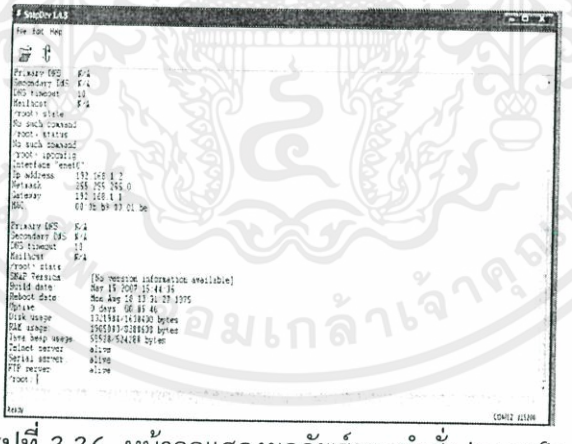
รูปที่ 3.34 แสดงการใช้คำสั่ง help เพื่อโชว์โปรแกรมใน root directory

10.ใช้คำสั่ง status สำหรับแสดงผลสถานะของ SNAPboard เช่น Version Snap, RAM usage, Disk usage, Uptime, Java heap usage, Server ตัวไหน Active บ้าง ดังรูปที่ 3.35



รูปที่ 3.35 แสดงสถานะของ SNAP ว่ามีสถานะเป็นอย่างไร

11.คำสั่ง ipconfig เป็นคำสั่งที่แสดงสถานะของ Network ต่างๆ ไม่ว่าจะเป็น IP, Netmask, Gateway, Mac address, Primary DSN, Secondary DNS ดังรูปที่ 3.36



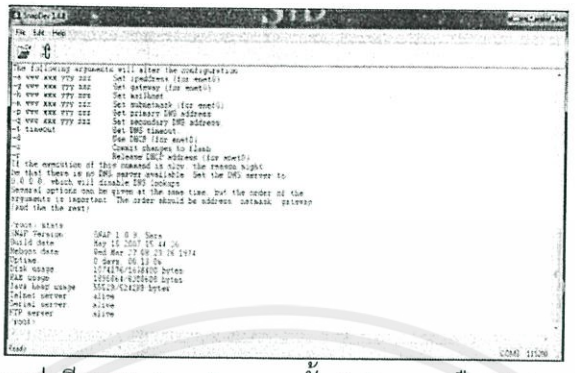
รูปที่ 3.36 หน้าจอแสดงผลลัพธ์ของคำสั่ง ipconfig

3.5.3.1 เปิด FTP Server บน SNAP Board เตรียมพร้อมสำหรับการ Upload Application file

1. ให้ต่อสาย Terminal console กับ SNAP Board เข้ากับ Serial Port จากนั้นเปิดโปรแกรม SnapDev จากนั้นให้ login เข้าสู่ Admin mode ตรวจสอบว่าได้มีการเปิด FTP Server แล้วหรือยังด้วยการพิมพ์คำสั่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

stats ที่ root prompt (/root>stats<enter>) จะได้ผลลัพธ์ดังรูปที่ 3.37



รูปที่ 3.37 แสดงว่ามีการActive Server ทั้ง 3 Server คือ Telnet, Serial, FTP

จากรูปจะเห็นได้ว่ามีการ Active FTP server ไว้ให้เรียบร้อยแล้วแต่ถ้า FTP server แสดงค่าเป็น dead หมายความว่า FTP server ยังไม่ได้ start ให้ทำการ start ด้วยคำสั่ง startserver –f ที่ root prompt จากนั้นให้ใช้คำสั่ง stats อีกครั้งเพื่อตรวจสอบว่า FTP server ได้มีค่า status เป็น Active การเตรียมความพร้อมทางด้านของSNAP Board แล้ว

3.5.3.2 ติดตั้งโปรแกรมFTP Cilent ใช้ Upload Application Program

1. เปิดโปรแกรม FileZilla จากนั้นกำหนดค่า Parameter ให้กับ FTP โดยกรอกหมายเลข IP ของ SNAP ที่ Hostaddress ในที่นี้กำหนดเป็น 192.168.1.2 จากนั้นให้ใส่ Username พร้อม Password ที่กำหนดไว้ใน SNAP Board กดปุ่ม Connect เพื่อติดต่อกับ SNAP ถ้าติดต่อสำเร็จ จะให้ผลลัพธ์ ดังรูปที่ 3.38 ในกรณีที่ไม่สามารถติดต่อกับ SNAP Board ได้ให้ตรวจสอบ IP ของ SNAP Board ดังนี้ให้ต่อ SNAP Board ผ่านโปรแกรม SnapDev จากนั้นใช้คำสั่ง ipconfig เพื่อตรวจสอบ IP Address, Netmask, Gateway ว่าถูกต้องหรือเปล่าถ้าต้องการเปลี่ยนแปลงให้ใช้คำสั่ง ipconfig –a ipaddress และค่าต่างๆตามที่เราต้องการ เสร็จแล้วให้ใช้คำสั่ง ipconfig –c เพื่อ commit ข้อมูลที่เราเปลี่ยนแปลงลงไปเก็บที่ Flash ไม่เช่นนั้น การ Rebot หรือ ปิดเครื่องใหม่จะได้ค่าเดิมก่อนเปลี่ยนแปลง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.38 หน้าต่างโปรแกรม FileZilla

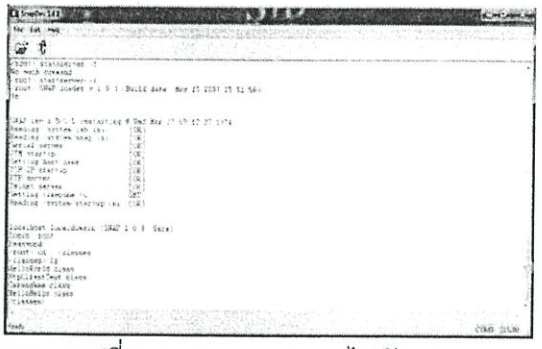
2. ให้ทดลองส่ง HelloWorld.class ที่ได้เขียนและ Compile ไว้แล้ว ในหัวข้อที่แล้ว ขึ้นไปยัง SNAP Board โดยการ Upload ผ่าน FileZilla โดยให้เปลี่ยน Upload Folder ของ SNAP ไปที่ "classes" จากนั้นให้เปลี่ยน Location Drive ไปที่ C:\Snap Developer 1.0.8\samples\HelloWorld\ ซึ่งเป็นที่เก็บ class ไฟล์ของ Application ให้คลิกขวาที่ HelloWorld.class แล้วเลือกหัวข้อ Upload หลังจากนั้น โปรแกรมจะ Upload แฟ้ม Helloworld.class ไปที่ SNAP Board เป็นการส่ง Application โปรแกรมของเราไปที่ SNAP ดังรูปที่ 3.39

รูปที่ 3.39 การ upload file

3.5.3.3 ทดลองRunโปรแกรม HelloWorld

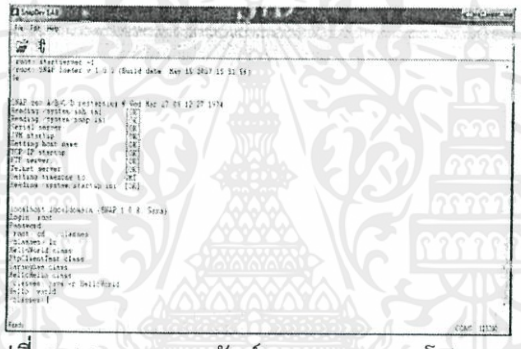
1. ต่อ SNAP Board เข้ากับ เครื่องคอมพิวเตอร์ผ่านสาย Terminal Console เปิดโปรแกรม SnapDev จากนั้น login เข้าสู่ user root เปลี่ยน Sub Directory ไปที่ classes โดยใช้คำสั่ง cd ../classes จะได้ผลลัพธ์ดังรูปที่ 3.40

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.40 แสดงการเปลี่ยน Sub Directory ไปยัง Sub Directory Classes

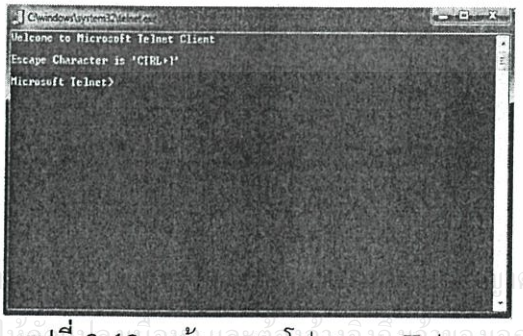
2. ใช้คำสั่ง ls เพื่อแสดงชื่อแฟ้มที่เก็บอยู่ใน Sub Directory classes จากนั้นให้ใช้ คำสั่ง java –r HelloWorld เพื่อสั่งโปรแกรม HelloWorld ทำงานหลังจากกด <enter> ก็จะได้ผลลัพธ์ดังรูปที่ 3.41



รูปที่ 3.41 แสดงผลลัพธ์ของการ RUN โปรแกรม

3.5.3.4 การควบคุม SNAP Board ผ่าน network ด้วยโปรแกรม Telnet

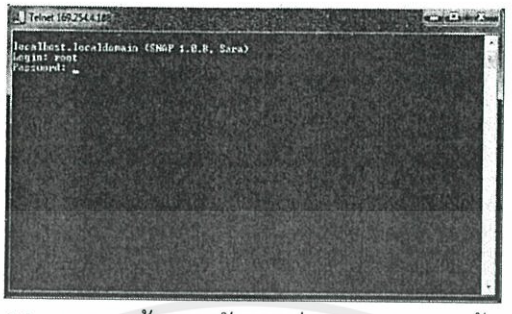
1. เปิดโปรแกรม Telnet จากนั้นใช้คำสั่ง open แล้วตามด้วย ip address ของ SNAP Board จะได้ผลลัพธ์ตามรูปที่ 3.42 หน้าจอของโปรแกรม Telnet



รูปที่ 3.42 หน้าจอของโปรแกรม Telnet

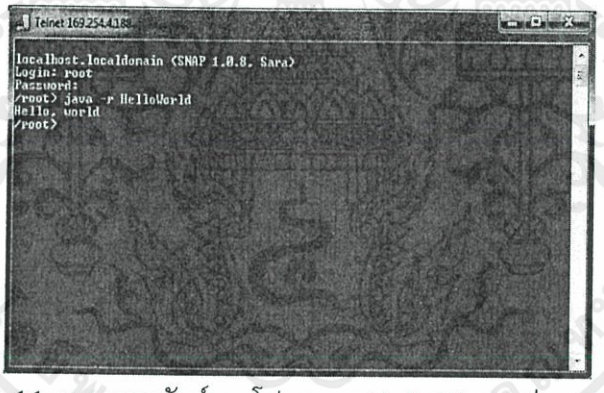
เอกสารนี้เป็นเอกสารที่สงวนไว้สำ ตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

2. ใช้คำสั่ง open ตามด้วย ip address ของ SNAP Board จะได้หน้าจอสำหรับหรอก login และ password ดังรูปที่ 3.43 แสดงหน้าจอหลังจากผ่านการ login เข้ามาแล้ว



รูปที่3.43 แสดงหน้าจอหลังจากผ่านการ login เข้ามาแล้ว

3. ทีนี้ก็สามารถสั่ง RUN Application หรือ Configuration SNAP ผ่าน Network เสมือน SNAP Board อยู่ใกล้ตัวลองทดสอบการ Run โปรแกรม HelloWorldให้เปลี่ยน Sub Directory ไปที่ classes จากนั้นใช้คำสั่ง java –r HelloWorldหลังจาก Run จะได้ดังรูปที่ 3.44



รูปที่ 3.44 แสดงผลลัพธ์ของโปรแกรม HelloWorld ผ่าน Telnet

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

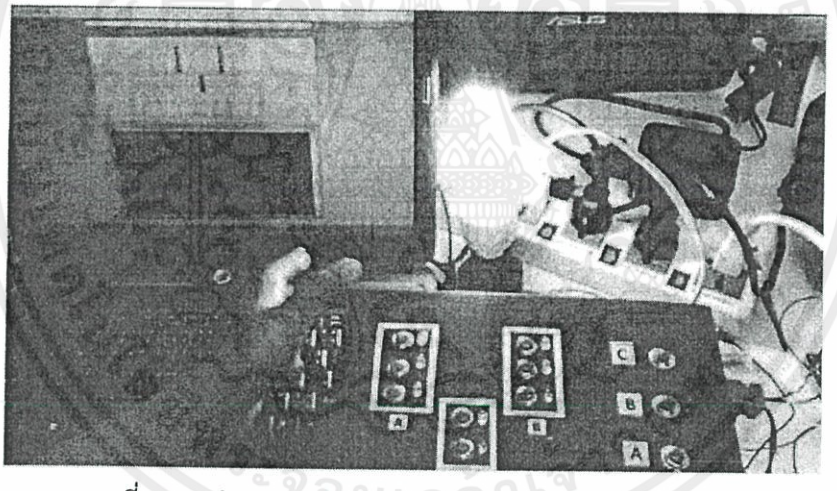
บทที่ 4
การทดลองและผลการทดลอง

4.1 บทนำ

ในบทนี้จะกล่าวถึงรายละเอียดการทดลองและผลของโครงงาน โดยระบบการเปิด-ปิดไฟส่องสว่างนี้ จะมีการสั่งงานโหมดที่สั่งงานด้วยตนเองจากที่ได้ทำการทดลองนั้น สามารถที่จะควบคุมการเปิด-ปิดไฟส่องสว่างได้ในโหมดที่สั่งการด้วยตนเอง และยังสามารถที่จะควบคุมได้สองแบบคือแบบแยกควบคุมการเปิด-ปิดไฟส่องสว่างทีละดวง และแบบควบคุมการเปิด-ปิดไฟส่องสว่างเป็นกลุ่มได้

4.2 การทดลองและผลการทดลอง

- ทำการกดปุ่มค่าก็จะส่งผ่าน CAN bus ไปยัง SNAP แสดงสถานะผ่านหน้าเว็บบราวเซอร์ และสามารถที่จะตรวจสอบสถานะที่กดได้ ณ หน้ากล่องควบคุม



รูปที่ 4.1 ทำการกดปุ่มและแสดงสถานะหน้ากล่องควบคุม

- ค่าที่ได้จากฮาร์ดแวร์ก็จะส่งไปยัง SNAP โดยใช้ telnet

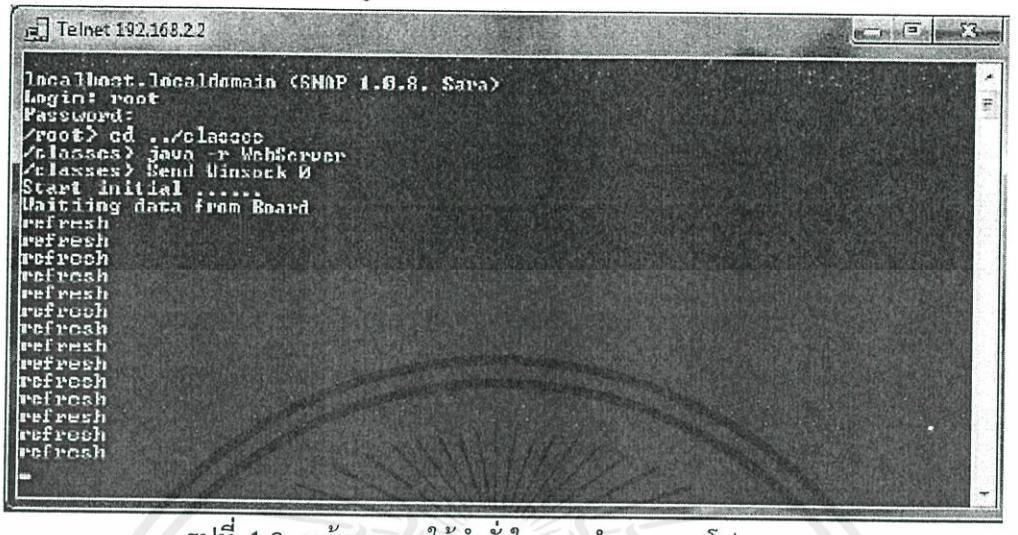
  telnet 198.168.2.2

  login = root

  password = snap

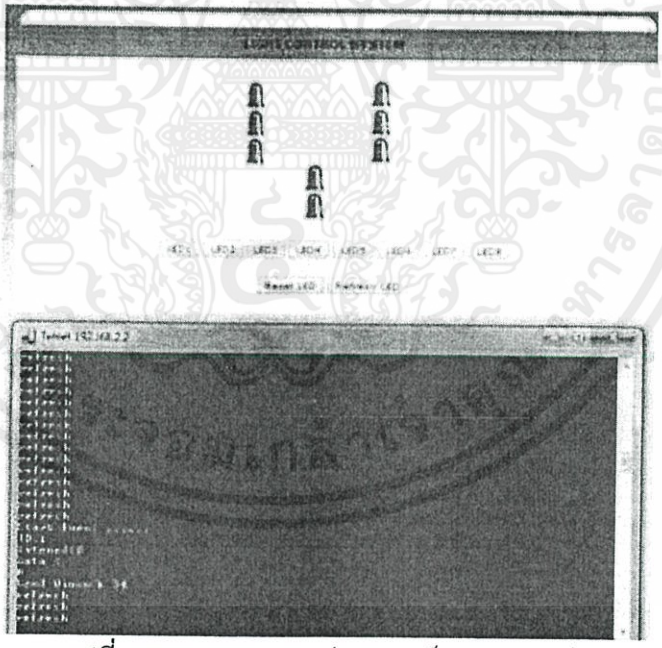
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสั่งในการรัน Snap คือ java-r WebServerถ้าการทำงานของโปรแกรมถูกต้องโปรแกรมจะทำงานตามรูปที่ 4.2



รูปที่ 4.2 หน้าจอการใช้คำสั่งในการทำงานของโปรแกรม

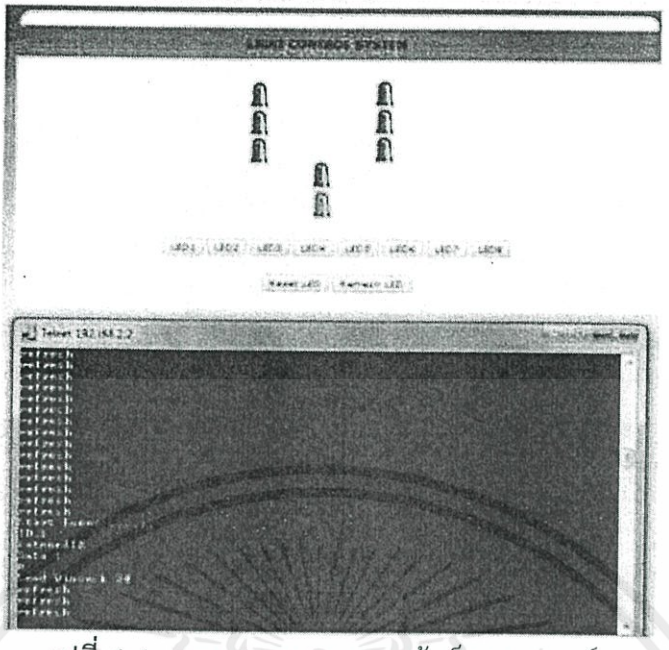
- เมื่อมีการรับข้อมูลจาก Hardware เข้าสู่ snap โปรแกรมส่วนของ SNAP ทำงานได้ตามที่ต้องการ จะส่งค่าสถานะของไฟส่องสว่างไปยังเว็บบราวเซอร์ดังรูปที่ 4.3



รูปที่ 4.3 การแสดงผลผ่านทางเว็บบราวเซอร์

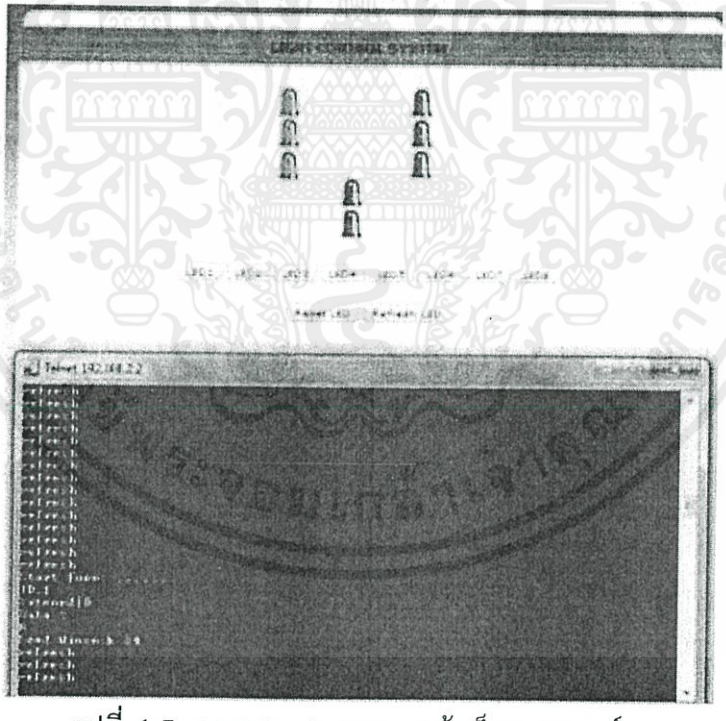
- โดยรูปแบบในการสั่งงานควบคุมไฟส่องสว่างนั้นมี 2 แบบดังนี้
  - การสั่งงานแบบควบคุมไฟทีละดวง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.4 การแสดงสถานะบนหน้าเว็บบราวเซอร์

- การสั่งงานแบบควบคุมไฟเป็นกลุ่ม



รูปที่ 4.5 การแสดงสถานะบนหน้าเว็บบราวเซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
สรุปผล ปัญหาและข้อเสนอแนะ

5.1 สรุปผล

จากการทำการทดลองโครงงานระบบควบคุมไฟส่องสว่างนี้ สามารถนำฮาร์ดแวร์มาใช้ในการควบคุมการเปิด-ปิดไฟส่องสว่างได้โดยโหมดเปิด-ปิดไฟส่องสว่างด้วยตนเอง และมีฟังก์ชั่นการใช้งานของฮาร์ดแวร์แบบเปิด-ปิดไฟส่องสว่างที่ละดวงและแบบเปิด-ปิดเป็นกลุ่มด้วย โดยจะแสดงสถานะให้เห็นผ่านเว็บบราวเซอร์

5.2 ปัญหาและแนวทางแก้ไข

ปัญหาในการทดลองและแนวทางแก้ไขมีดังนี้

- ในการเขียนโปรแกรมลงในไมโครคอนโทรลเลอร์ PIC18F458 นั้น มีข้อควรระวังในการใช้งานPORT I/O เนื่องจากในแต่ละPORT ไม่ได้ทำหน้าที่เป็น PORT I/O เพียงอย่างเดียวจึงควรทำการเขียนคำสั่งปิด PORT I/O ที่ไม่ได้ใช้เสียก่อน
- การเขียนโปรแกรมการส่งข้อมูลผ่านทาง CAN BUS ควรศึกษารูปแบบวิธีการส่งและคอมไพล์เลอร์ที่ใช้ในการเขียนโปรแกรมการส่งข้อมูล เนื่องจากคอมไพล์เลอร์แต่ละตัวจะมีรูปแบบการเขียนโปรแกรมการส่งข้อมูลผ่าน CAN BUS ไม่เหมือนกัน
- เนื่องจาก SNAP เป็น UNIX และออกแบบให้เขียนโปรแกรมควบคุมได้เพียงภาษาเดียวเท่านั้นซึ่งยากต่อการใช้งาน

5.3 ข้อเสนอแนะ

- สามารถที่จะนำเอา SNAP ไปประยุกต์ใช้ในโครงงานต่างๆได้ เช่น การควบคุมเครื่องหยอดเหรียญ ตรวจสอบติดตามควบคุม GPRSและGPS หรือการวินิจฉัยทางการแพทย์
- CAN (Control Area Network)หรือ CANbus นั้น ใช้สำหรับการควบคุมอุปกรณ์แบบReal Time มีประสิทธิภาพในการตรวจจับข้อผิดพลาดและช่วยประหยัดงบประมาณได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

"คู่มือการใช้งาน SNAP board"บริษัท เอ็มเบ็ดเด็ด เจ จำกัด.

ณัฏฐพล วงศ์สุนทรชัย และชัยวัฒน์ ลิ้มพรจิตรวิไล. เรียนรู้และปฏิบัติการไมโครคอนโทรลเลอร์ PIC16F877. กรุงเทพฯ: อินโนเวตีฟ เอ็กเพอริเมนต์ จำกัด, 2521.

ณัฐพล สีนาก,ปฏิพัทธ์ แปลกโฉมฉิน,ปิยาวี อันโต : "ระบบตอบรับความพึงพอใจแบบหลายจุด" ปริญญานิพนธ์สำหรับปริญญาวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมการวัดคุม สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ปีการศึกษา 2555

สุดา เธียรมนตรี. คู่มือเรียนเขียนโปรแกรมภาษา Java ฉบับสมบูรณ์. นนทบุรี: ไอดีซี พรีเมียร์ จำกัด ,2555

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก.

โค้ดโปรแกรม

โค้ด PIC18F458

```
unsigned char Can_Init_Flags, Can_Send_Flags, Can_Rcv_Flags; // can flags
unsigned char Rx_Data_Len;                    // received data length in bytes
char RxTx_Data[8];                            // can rx/tx data buffer
char Msg_Rcvd;                                // reception flag
const long ID_1st = 12111, ID_2nd = 3;        // node IDs
long Rx_ID;
char data_TX[1];
/************************************************************
*** STARTING PROGRAM
************************************************************/
unsigned int j=0,k=0,l=0,m=0,o=0,q=0,u=0,v=0,e=0,g=0,h=0;
void main()
{
Can_Init_Flags = 0;                           //
Can_Send_Flags = 0;                           // clear flags
Can_Rcv_Flags  = 0;                           //
Can_Send_Flags = CAN_TX_PRIORITY_0 &          // form value to be used
                 CAN_TX_XTD_FRAME &           //     with CANWrite
                 CAN_TX_NO_RTR_FRAME;
Can_Init_Flags = CAN_CONFIG_SAMPLE_THRICE &   // form value to be used
                 CAN_CONFIG_PHSEG2_PRG_ON &   // with CANInit
                 CAN_CONFIG_XTD_MSG &
                 CAN_CONFIG_DBL_BUFFER_ON &
                 CAN_CONFIG_VALID_XTD_MSG;
CANInitialize(1,2,3,3,1,Can_Init_Flags);      // Initialize CAN module
CANSetOperationMode(CAN_MODE_CONFIG,0xFF);    // set CONFIGURATION mode
CANSetMask(CAN_MASK_B1,-1,CAN_CONFIG_XTD_MSG);    // set all mask1 bits to ones
CANSetMask(CAN_MASK_B2,-1,CAN_CONFIG_XTD_MSG);    // set all mask2 bits to ones
CANSetFilter(CAN_FILTER_B2_F4,ID_2nd,CAN_CONFIG_XTD_MSG);// set id of filter B2_F4
to 2nd node ID
CANSetOperationMode(CAN_MODE_NORMAL,0xFF);    // set NORMAL mode
RxTx_Data[0] = 9;                             // set initial data to be sent
CANWrite(ID_1st, RxTx_Data, 1, Can_Send_Flags);   // send initial message
Rx_ID = 0;
  CMCON |= 0xFF;              // turn off comparators
  ADCON1 |= 0xFF;             // turn off analog inputs
  TRISC = 0 ;                 // port C is output only
  PORTC = 0 ;                 // chip enable, reverse on, 8x8 font
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    TRISD=0xFF;
    PORTD=0xFF;
//    PORTB=0xE0;
//    TRISB=0xE0;
while(1)
{
int x;
   /////light 1/////
     if(PORTD.F0!=1)
     {
     while(PORTD.F0!=1 )
      {
     }
         if(j==0)
         {
          x=1;
              while (x==1) {
                if(PORTD.F0==1){
                PORTC.F0=1;

                x=0;
                }
                else {}
              }
          j=1;
        }
        else if(j==1)
        {
          x=1;
              while (x==1) {
                if(PORTD.F0==1){
                PORTC.F0=0;

                x=0;
                }
                else {}
              }
         j=0;
        }
        else {}
CANWrite(1, "0", 1, Can_Send_Flags);
delay_ms(500);
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        /////light 2/////
        else if(PORTD.F1!=1)
        {
          while(PORTD.F1!=1 )
          {
          }
             if(k==0)
             {
               x=1;
                  while (x==1) {
                     if(PORTD.F1==1){
                     PORTC.F1=1;

                     x=0;
                     }
                     else {}
                  }
               k=1;
             }
             else if(k==1)
             {
              x=1;
                  while (x==1) {
                     if(PORTD.F1==1){
                     PORTC.F1=0;
                    x=0;
                     }
                      else {}
                  }
              k=0;
            }
            else {}
CANWrite(1, "1", 1, Can_Send_Flags);
delay_ms(500);
      }
      /////light 3/////
      else if(PORTD.F2!=1)
      {
        while(PORTD.F2!=1 )
        {
        }
           if(l==0)
           {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            x=1;
               while (x==1) {
                  if(PORTD.F2==1){
                  PORTC.F2=1;
                 x=0;
                  }
                  else {}
               }
             l=1;
           }
           else if(l==1)
           {
            x=1;
               while (x==1) {
                  if(PORTD.F2==1){
                  PORTC.F2=0;
                  x=0;
                  }
                  else {}
               }
            l=0;
           }
           else {}
CANWrite(1, "2", 1, Can_Send_Flags);
delay_ms(500);
      }
      ///light 4/////
      else if(PORTD.F3!=1)
      {
       while(PORTD.F3!=1 )
       {
       }
          if(m==0)
          {
           x=1;
              while (x==1) {
                 if(PORTD.F3==1){
                 PORTC.F3=1;
                x=0;
                 }
                 else {}
              }
           m=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            }
            else if(m==1)
            {
              x=1;
                 while (x==1) {
                    if(PORTD.F3==1){
                    PORTC.F3=0;
                    x=0;
                    }
                    else {}
                 }
              m=0;
            }
            else {}
CANWrite(1, "3", 1, Can_Send_Flags);
delay_ms(500);
       }
        /////light 5/////
      else if(PORTD.F4!=1)
      {
        while(PORTD.F4!=1 )
        {
        }
           if(o==0)
           {
            x=1;
               while (x==1) {
                  if(PORTD.F4==1){
                  PORTC.F4=1;
                  x=0;
                  }
                  else {}
               }
            o=1;
          }
          else if(o==1)
          {
            x=1;
               while (x==1) {
                  if(PORTD.F4==1){
                  PORTC.F4=0;
                  x=0;
                  }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                else {}
              }
             o=0;
            }
            else {}
CANWrite(1, "4", 1, Can_Send_Flags);
delay_ms(500);
      }
       /////light 6/////
      else if(PORTD.F5!=1)
      {
        while(PORTD.F5!=1 )
        {
        }
          if(q==0)
          {
            x=1;
              while (x==1) {
                if(PORTD.F5==1){
                PORTC.F5=1;
                x=0;
                }
                else {}
              }
            q=1;
          }
          else if(q==1)
          {
            x=1;
              while (x==1) {
                if(PORTD.F5==1){
                PORTC.F5=0;
                x=0;
                }
                else {}
              }
            q=0;
          }
          else {}
CANWrite(1, "5", 1, Can_Send_Flags);
delay_ms(500);
      }
       /////light 7/////
```

เอกสารนี้มีไว้เพื่อประโยชน์ในการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        else if(PORTD.F6!=1)
        {
          while(PORTD.F6!=1 )
          {
          }
             if(u==0)
             {
              x=1;
                 while (x==1) {
                    if(PORTD.F6==1){
                    PORTC.F6=1;
                    x=0;
                    }
                    else {}
                 }
              u=1;
            }
            else if(u==1)
            {
             x=1;
                while (x==1) {
                    if(PORTD.F6==1){
                    PORTC.F6=0;

                    x=0;
                    }
                   else {}
                 }
              u=0;
            }
            else {}
CANWrite(1, "6", 1, Can_Send_Flags);
delay_ms(500);
      }
       ///light 8/////
      else if(PORTD.F7!=1)
      {
        while(PORTD.F7!=1 )
        {
        }
            if(v==0)
            {
             x=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                while (x==1) {
                    if(PORTD.F7==1){
                    PORTC.F7=1;
                    x=0;
                    }
                    else {}
                }
              v=1;
            }
            else if(v==1)
            {
              x=1;
                while (x==1) {
                    if(PORTD.F7==1){
                    PORTC.F7=0;
                    x=0;
                    }
                    else {}
                }
              v=0;
            }
            else {}
CANWrite(1, "7", 1, Can_Send_Flags);
delay_ms(500);
      }
      /////group A/////
      else if(PORTB.F5!=1)
      {
        while(PORTB.F5!=1 )
        {
        }
          if(e==0)
          {
            x=1;
              while (x==1) {
                if(PORTB.F5==1){
                PORTC.F0=1;
                PORTC.F1=1;
                PORTC.F2=1;
                x=0;
                }
                else {}
              }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            e=1;
           }
           else if(e==1)
           {
            x=1;
               while (x==1) {
                  if(PORTB.F5==1){
                  PORTC.F0=0;
                  PORTC.F1=0;
                  PORTC.F2=0;
                  x=0;
                  }
                  else {}
               }
            e=0;
          }
          else {}
CANWrite(1, "A", 1, Can_Send_Flags);
delay_ms(500);
      }
      ///group B/////
      else if(PORTB.F6!=1)
      {
       while(PORTB.F6!=1 )
       {
       }
          if(g==0)
          {
           x=1;
              while (x==1) {
                 if(PORTB.F6==1){
                 PORTC.F3=1;
                 PORTC.F4=1;
                 PORTC.F5=1;
                 x=0;
                 }
                 else {}
              }
           g=1;
          }
          else if(g==1)
          {
           x=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                while (x==1) {
                   if(PORTB.F6==1){
                   PORTC.F3=0;
                   PORTC.F4=0;
                   PORTC.F5=0;
                   x=0;
                   }
                   else {}
                }
             g=0;
          }
          else {}
CANWrite(1, "B", 1, Can_Send_Flags);
delay_ms(500);
      }
      ///group C/////
      else if(PORTB.F7!=1)
      {
       while(PORTB.F7!=1 )
       {
       }
          if(h==0)
          {
           x=1;
              while (x==1) {
                 if(PORTB.F7==1){
                 PORTC.F6=1;
                 PORTC.F7=1;
                 x=0;
                 }
                 else {}
              }
           h=1;
          }
          else if(h==1)
          {
           x=1;
              while (x==1) {
                 if(PORTB.F7==1){
                 PORTC.F6=0;
                 PORTC.F7=0;
                 x=0;
                 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    else {}
                }
              h=0;
            }
            else {}
  CANWrite(1, "C", 1, Can_Send_Flags);
  delay_ms(500);
      }
    else {}
  }
  }//void main();
```

โค้ดใน SNAP

Index.txt

```
import se.imsys.comm.CAN;
import se.imsys.comm.CANEventListener;
import se.imsys.comm.CANException;
import se.imsys.comm.CANFrame;
import java.net.*;
import java.io.*;
public class DataReader implements CANEventListener{
        private static final int BRP =5;
        private static boolean DEBUG = true;
        private CAN cp;
        private CANFramecf;
        public  CANFramecff;
        private static int COM_TO_CAN_ID =0x01;
        private static int CAN_TO_COM_ID =0x02;
        //private static int CAN_TO_COM_ID2 =0x02;
        private static int CAN_SUBMASK = 863;
        static public intData_SW[]={0,0,0,0,0,0,0,0,0,0,0,0};
        static public int  Data_SW2[]={0,0,0,0,0,0,0,0,0,0,0,0};
  //public  int status
  //LEDcgild = new LEDcgi();
public void CANEvent (int event){
System.out.println("Start Event .....");
try{
if ((event & CAN.IR_RI) != 0)
{
while (cp.receivePoll(cf))
{
if (DEBUG)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
//System.out.println("Receive Event .....");
System.out.println("ID."+Integer.toHexString(cf.ID));
if (cf.extendedID) System.out.println("ExtenedID");
if (cf.remoteFrameRequest) System.out.println("remoteFrameRequest");
System.out.println("Data :");
for (int i=0; i<cf.length ; i++ ) System.out.print((char) cf.data[i]);
System.out.println();
if (cf.data[0]=='0')
{Data_SW[0]=1;
if(Data_SW2[0]==1)
{
Data_SW2[0]=0;
send_Socket(75);
}
else
{
Data_SW2[0]=1;
send_Socket(76);
}
}
else if (cf.data[0]=='1')
{
Data_SW[1]=1;
if(Data_SW2[1]==1)
{
Data_SW2[1]=0;
send_Socket(10);
}
else
{
Data_SW2[1]=1;
send_Socket(11);
}
}
else if (cf.data[0]=='2')
{
Data_SW[2]=1;
if(Data_SW2[2]==1)
{
Data_SW2[2]=0;
send_Socket(20);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 else
 {
 Data_SW2[2]=1;
 send_Socket(21);
 }
 }else if (cf.data[0]=='3')
 {
 Data_SW[3]=1;
 if(Data_SW2[3]==1)
 {
 Data_SW2[3]=0;
 send_Socket(30);
 }
 else
 {
 Data_SW2[3]=1;
 send_Socket(31);
 }
 }
 else if (cf.data[0]=='4')
 {
 Data_SW[4]=1;
 if(Data_SW2[4]==1)
 {
 Data_SW2[4]=0;
 send_Socket(40);
 }else
 {
 Data_SW2[4]=1;
 send_Socket(41);
 }
 }
 else if (cf.data[0]=='5')
 {
 Data_SW[5]=1;
 if(Data_SW2[5]==1)
 {
 Data_SW2[5]=0;
 send_Socket(50);
 }
 else
 {
 Data_SW2[5]=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 send_Socket(51);
 }
 }
 else if (cf.data[0]=='6')
 {
 Data_SW[6]=1;
 if (Data_SW2[6]==1)
 {
 Data_SW2[6]=0;
 send_Socket(60);
 }
 else
 {
 Data_SW2[6]=1;
 send_Socket(61);
 }
 }
 else if (cf.data[0]=='7')
 {
 Data_SW[7]=1;
 if(Data_SW2[7]==1)
 {
 Data_SW2[7]=0;
 send_Socket(70);
 }
 else
 {
 Data_SW2[7]=1;
 send_Socket(71);
 }}
 else if (cf.data[0]=='A')
 {
 Data_SW[8]=1;
 if (Data_SW2[8]==1)
 {
 Data_SW2[8]=0;
 send_Socket(33);
 }
 else
 {
 Data_SW2[8]=1;
 send_Socket(34);
 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
else if (cf.data[0]=='B')
{
Data_SW[9]=1;
if (Data_SW2[9]==1)
{
Data_SW2[9]=0;
send_Socket(35);
}
else
{
Data_SW2[9]=1;
send_Socket(36);
}
}
else if (cf.data[0]=='C')
{
Data_SW[10]=1;
if (Data_SW2[10]==1)
{
Data_SW2[10]=0;
send_Socket(37);
}
else
{
Data_SW2[10]=1;
send_Socket(38);
}
}
}
if ( (!cf.extendedID) && (cf.ID == COM_TO_CAN_ID))
{
System.out.println("Start send to board ");
cf.ID = 2;
cf.data[0] = 0x31;
cp.sendFrame(cf);
}
}
}
}catch (CANException e){
System.out.println(e.getMessage());
} catch (Throwable e){
System.out.println(e.getMessage());
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
e.printStackTrace();
}
}
//=========================================
/*public void test_send_can(){
try {
System.out.println("Start send to board ");
cf.length =1;
cf.ID = 2;
cf.data[0] = 0x31;
cp.sendFrame(cf);
cf.ID = 2;
cf.data[0] = 0x32;
cp.sendFrame(cf);
}
catch(CANException e) {
System.out.println("error1"+e.getMessage());
}
catch (Throwable e) {                    /// <==================== Error in this
System.out.println("error2"+e.getMessage());
e.printStackTrace();
}
}
*/public booleaninit() {
booleancanOpen = false;
booleancanEnabled = false;
send_Socket(0);
try{
cp = new CAN();
canOpen = true;
cf = new CANFrame();
cf.extendedID = false;
cf.remoteFrameRequest = false;
cp.setSampleRate(3);
cp.setTSEG1(4);
cp.setTSEG2(3);
cp.setSynchronizationJumpWidth(1);
cp.setBaudRatePrescaler(BRP);
cp.set29BitGlobalIDMask (COM_TO_CAN_ID ^ CAN_SUBMASK);
//cp.set29BitGlobalIDMask (1551);
cp.set29BitGlobalArbitrationID(COM_TO_CAN_ID);
cp.addEventListener(this);
cp.writeRegister(cp.REG_IER,(byte)(cp.readRegister(cp.REG_IER)| CAN.IER_DOIE));
```

เอกสารนี้มีไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//cp.writeRegister((byte)4,(byte)(cp.readRegister((byte)4 | 8));
cp.enableController();
canEnabled=true;
System.out.println("Start initial .....");
//send_Socket(11); // test
} catch (CANException e){
System.out.println(e.getMessage());
} catch (Throwable e){
System.out.println(e.getMessage());
e.printStackTrace();
}
if (!(canOpen&&canEnabled)){
if (canOpen){
try{
cp.close();
}catch (CANException e) {
System.out.println(e.getMessage());
}
}
return false;
}
return true;
}
//**********************************************************************
public void send_Socket(int a)
{
String serverName = "192.168.2.3";
String DataR;
intri=0;
int port = 8686;
//a=80;
try
{
Socket client = new Socket(serverName, port);
System.out.println("Send Winsock "+a);
OutputStreamoutToServer = client.getOutputStream();
DataOutputStream out =  new DataOutputStream(outToServer);          DataInputStream in  = new
DataInputStream(client.getInputStream());
if (a==0)  out.writeUTF("0");
else if (a==11)  out.writeUTF("11");
 else if (a==21)  out.writeUTF("21");
else if (a==31)  out.writeUTF("31");
else if (a==41)  out.writeUTF("41");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  else if (a==51)  out.writeUTF("51");
  else if (a==61)  out.writeUTF("61");
  else if (a==71)  out.writeUTF("71");
  else if (a==75)  out.writeUTF("75");
  else if (a==10)  out.writeUTF("10");
 else if (a==20)  out.writeUTF("20");
  else if (a==30)  out.writeUTF("30");
  else if (a==40)  out.writeUTF("40");
  else if (a==50)  out.writeUTF("50");
  else if (a==60)  out.writeUTF("60");
  else if (a==70)  out.writeUTF("70");
  else if (a==76)  out.writeUTF("76");
  else if (a==33)  out.writeUTF("33");
  else if (a==34)  out.writeUTF("34");
  else if (a==35)  out.writeUTF("35");
  else if (a==36)  out.writeUTF("36");
  else if (a==37)  out.writeUTF("37");
  else if (a==38)  out.writeUTF("38");
  else if (a==9)  out.writeUTF("99");
client.close();
}catch(IOException e)
{
e.printStackTrace();
}
}
public void readDataViaCAN(){
CANFrameccf = new CANFrame();
try{
ccf.ID = COM_TO_CAN_ID;
ccf.extendedID = true;
ccf.remoteFrameRequest = false;
while(true)
{
}
} catch (Throwable e){
System.out.println(e.getMessage());
}
try{
cp.close();
} catch (CANException e) {
System.out.println(e.getMessage());
}
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
```

Datareader.java

```
import se.imsys.comm.CAN;
import se.imsys.comm.CANEventListener;
import se.imsys.comm.CANException;
import se.imsys.comm.CANFrame;
import java.net.*;
import java.io.*;
public class DataReader implements CANEventListener{
        private static final int BRP =5;
        private static boolean DEBUG = true;
        private CAN cp;
        private CANFramecf;
        public  CANFramecff;
        private static int COM_TO_CAN_ID =0x01;
        private static int CAN_TO_COM_ID =0x02;
        //private static int CAN_TO_COM_ID2 =0x02;
        private static int CAN_SUBMASK = 863;
        static public intData_SW[]={0,0,0,0,0,0,0,0,0,0,0};
        static public int  Data_SW2[]={0,0,0,0,0,0,0,0,0,0,0};
//public  int status
//LEDcgild = new LEDcgi();
        public void CANEvent (int event){
System.out.println("Start Event .....");
try{
if ((event & CAN.IR_RI) != 0)
{
while (cp.receivePoll(cf))
{
if (DEBUG)
{
//System.out.println("Receive Event .....");
System.out.println("ID."+Integer.toHexString(cf.ID));
if (cf.extendedID) System.out.println("ExtenedID");
if (cf.remoteFrameRequest) System.out.println("remoteFrameRequest");
System.out.println("Data :");
for (int i=0; i<cf.length ; i++ ) System.out.print((char) cf.data[i]);
System.out.println();
if (cf.data[0]=='0')
{
Data_SW[0]=1;
if (Data_SW2[0]==1)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
Data_SW2[0]=0       send_Socket(75);
}
else
{
Data_SW2[0]=1;
send_Socket(76);
}
}
else if (cf.data[0]=='1')
{
Data_SW[1]=1;
if(Data_SW2[1]==1)
{
Data_SW2[1]=0;
send_Socket(10);
}
else
{
Data_SW2[1]=1;
send_Socket(11);
}
}
else if (cf.data[0]=='2')
{
Data_SW[2]=1;
if (Data_SW2[2]==1)
{
Data_SW2[2]=0;
send_Socket(20);
}
else
{
Data_SW2[2]=1;
send_Socket(21);
}
}
else if (cf.data[0]=='3')
{
Data_SW[3]=1;
if(Data_SW2[3]==1)
{
Data_SW2[3]=0;
send_Socket(30);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
else
{
Data_SW2[3]=1;
send_Socket(31);
}
}
else if (cf.data[0]=='4')
{
Data_SW[4]=1;
if(Data_SW2[4]==1)
{
Data_SW2[4]=0;
send_Socket(40);
}
else
{
Data_SW2[4]=1;
send_Socket(41);
}
}
else if (cf.data[0]=='5')
{
Data_SW[5]=1;
if(Data_SW2[5]==1)
{
Data_SW2[5]=0;
send_Socket(50);
}
else
{
Data_SW2[5]=1;
send_Socket(51);
}
}
else if (cf.data[0]=='6')
{
Data_SW[6]=1;
if(Data_SW2[6]==1)
{
Data_SW2[6]=0;
send_Socket(60);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
else
{
Data_SW2[6]=1;
send_Socket(61);
}
}
else if (cf.data[0]=='7')
{
Data_SW[7]=1;
if (Data_SW2[7]==1)
{
Data_SW2[7]=0;
send_Socket(70);
}
else
{
Data_SW2[7]=1;
send_Socket(71);
}
}
else if (cf.data[0]=='A')
{
Data_SW[8]=1;
if (Data_SW2[8]==1)
{
Data_SW2[8]=0;
send_Socket(33);
}
else
{
Data_SW2[8]=1;
send_Socket(34);
}
}
else if (cf.data[0]=='B')
{
Data_SW[9]=1;
if (Data_SW2[9]==1)
{
Data_SW2[9]=0;
send_Socket(35);
}
else
```

เอกสารนี้มีลิขสิทธิ์ การที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
Data_SW2[9]=1;
send_Socket(36);
}
}
else if (cf.data[0]=='C')
{
Data_SW[10]=1;
if (Data_SW2[10]==1)
{
Data_SW2[10]=0;
send_Socket(37);
}
else
{
Data_SW2[10]=1;
send_Socket(38);
}
}
}
if(!cf.extendedID) && (cf.ID == COM_TO_CAN_ID))
{
System.out.println("Start send to board ");
cf.ID = 2;
cf.data[0] = 0x31;
cp.sendFrame(cf);
}
}
}
}catch (CANException e){
System.out.println(e.getMessage());
} catch (Throwable e){
System.out.println(e.getMessage());
e.printStackTrace();
}
}
//=========================================
/*
public void test_send_can(){
try {
System.out.println("Start send to board ");
cf.length =1;
cf.ID = 2;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
cf.data[0] = 0x31;
cp.sendFrame(cf);
cf.ID = 2;
cf.data[0] = 0x32;
cp.sendFrame(cf);
}
catch(CANException e) {
System.out.println("error1"+e.getMessage());
}
catch (Throwable e) {                    /// <==================== Error in this
System.out.println("error2"+e.getMessage());
e.printStackTrace();
}
}
*/public booleaninit() {
booleancanOpen = false;
booleancanEnabled = false;
send_Socket(0);
try{
cp = new CAN();
canOpen = true;
cf = new CANFrame();
cf.extendedID = false;
cf.remoteFrameRequest = false;
cp.setSampleRate(3);
cp.setTSEG1(4);
cp.setTSEG2(3);
cp.setSynchronizationJumpWidth(1);
cp.setBaudRatePrescaler(BRP);
cp.set29BitGlobalIDMask (COM_TO_CAN_ID ^ CAN_SUBMASK);
//cp.set29BitGlobalIDMask (1551);
cp.set29BitGlobalArbitrationID(COM_TO_CAN_ID);
cp.addEventListener(this);
cp.writeRegister(cp.REG_IER,(byte)(cp.readRegister(cp.REG_IER)| CAN.IER_DOIE));
//cp.writeRegister((byte)4,(byte)(cp.readRegister((byte)4 | 8));
cp.enableController();
canEnabled=true;
System.out.println("Start initial .....");
//send_Socket(11); // test
} catch (CANException e){
System.out.println(e.getMessage());
} catch (Throwable e){
System.out.println(e.getMessage());
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
e.printStackTrace();
}
if (!(canOpen&&canEnabled)){
if (canOpen){
try{
cp.close();
}catch (CANException e) {
System.out.println(e.getMessage());
}
}
return false;
}
return true;
}
//**********************************************************************
public void send_Socket(int a)
{
String serverName = "192.168.2.3";
String DataR;
intri=0;
int port = 8686;
//a=80;
try
{
Socket client = new Socket(serverName, port);
System.out.println("Send Winsock "+a);
OutputStreamoutToServer = client.getOutputStream();
DataOutputStream out =  new DataOutputStream(outToServer);          DataInputStream in  = new
DataInputStream(client.getInputStream());
if (a==0)  out.writeUTF("0");
else if (a==11)  out.writeUTF("11");
else if (a==21)  out.writeUTF("21");
else if (a==31)  out.writeUTF("31");
else if (a==41)  out.writeUTF("41");
else if (a==51)  out.writeUTF("51");
else if (a==61)  out.writeUTF("61");
else if (a==71)  out.writeUTF("71");
else if (a==75)  out.writeUTF("75");
else if (a==10)  out.writeUTF("10");
else if (a==20)  out.writeUTF("20");
else if (a==30)  out.writeUTF("30");
else if (a==40)  out.writeUTF("40");
else if (a==50)  out.writeUTF("50");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 else if (a==60)  out.writeUTF("60");
 else if (a==70)  out.writeUTF("70");
 else if (a==76)  out.writeUTF("76");
 else if (a==33)  out.writeUTF("33");
 else if (a==34)  out.writeUTF("34");
 else if (a==35)  out.writeUTF("35");
 else if (a==36)  out.writeUTF("36");
 else if (a==37)  out.writeUTF("37");
 else if (a==38)  out.writeUTF("38");
 else if (a==9)  out.writeUTF("99");
 client.close();
 }catch(IOException e)
 {
 e.printStackTrace();
 }
 }
 public void readDataViaCAN(){
 CANFrameccf = new CANFrame();
 try{
 ccf.ID = COM_TO_CAN_ID;
 ccf.extendedID = true;
 ccf.remoteFrameRequest = false;
 while(true)
 {
 }
 } catch (Throwable e){
 System.out.println(e.getMessage());
 }
 try{
 cp.close();
 } catch (CANException e) {
 System.out.println(e.getMessage());
 }
 }
 }
```

LEDcgi.java

```
import se.imsys.net.*;
import java.net.*;
import java.io.*;
import java.util.*;
import com.dalsemi.system.*;
public class LEDcgi implements HttpCgi {
private DataPort port = new DataPort();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 private intcurrentdata = 0x00; // On all
 public int status[] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0};
 public String output[] = {"of.gif","of.gif","of.gif","of.gif","of.gif","of.gif","of.gif","of.gif"};
 public  String hhh = "Hello";

 //public void off();
 //DataReader DR = new DataReader();
 public LEDcgi() {
 //off();
 }
 public String getName() {
 return("led.cgi");
 }
 //set led on

 public void checkstatus(int ii) {
    ii--;
    //System.out.println("check in "+ii);
if(ii==0){
if (status[0]== 0)
{
status[0]=1;
output[0]="on.gif";
}
else if (status[0]== 1)
{
output[0]="of.gif";
status[0]=0;
}
}
if(ii==1){
if (status[1]== 0)
{
output[1]="on.gif";
status[1]=1;
}
else if (status[1]== 1)
{
output[1]="of.gif";
status[1]=0;
}
}
if(ii==2){
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 if (status[2]== 0)
 {
 status[2]=1;
 output[2]="on.gif";
 }
 else if(status[2]== 1)
 {
 status[2]=0;
 output[2]="of.gif";
 }
 }
 if(ii==3){
 if (status[ii]== 0)
 {
 status[3]=1;
 output[3]="on.gif";
 }
 else if (status[ii]== 1)
 {
 status[3]=0;
 output[3]="of.gif";
 }
 }
 if(ii==4){
 if (status[ii]== 0)
 {
 status[4]=1;
 output[4]="on.gif";
 }
 else if (status[ii]== 1)
 {
 status[4]=0;
 output[4]="of.gif";
 }
 }
 if(ii==5){
 if (status[ii]== 0)
 {
 status[5]=1;
 output[5]="on.gif";
 }
 else if (status[ii]== 1)
 {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
 status[5]=0;
 output[5]="of.gif";
 }
 }
 if(ii==6){
 if (status[ii]== 0)
 {
 output[6] ="on.gif";
 status[6]=1;
 }
 else if (status[ii]== 1)
 {
 output[6] ="of.gif";
 status[6]=0;
 }
 }
 if(ii==7){
 if (status[ii]== 0)
 {
 output[7] ="on.gif";
 status[7]=1;
 }
 else if (status[ii]== 1)
 {
 output[7] ="of.gif";
 status[7]=0;
 }
 }
 if(ii==8){
 if (status[ii]== 0)
 {
 status[8]=1;
 output[0] ="on.gif";
 output[1] ="on.gif";
 output[2] ="on.gif";
 }
 else if (status[ii]== 1)
 {
 status[8]=0;
 output[0] ="of.gif";
 output[1] ="of.gif";
 output[2] ="of.gif";
 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
if(ii==9){
if (status[ii]== 0)
{
status[9]=1;
output[3] ="on.gif";
output[4] ="on.gif";
output[5] ="on.gif";
}
else if (status[ii]== 1)
{
status[9]=0;
output[3] ="of.gif";
output[4] ="of.gif";
output[5] ="of.gif";
}
}
if(ii==10){
if (status[ii]== 0)
{
status[10]=1;
output[6] ="on.gif";
output[7] ="on.gif";
}
else if (status[ii]== 1)
{
status[10]=0;
output[6] ="of.gif";
output[7] ="of.gif";
}
}
/*
if (status[ii]== 0)
{
output[ii] ="on.gif";

status[ii]=1;
}
else if (status[ii]== 1)
{
output[ii] ="of.gif";
status[ii]=0;
}*/
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//System.out.println("output  "+ii+"="+output[ii]);
}
//cgi process
public void execute(Hashtable variables, OutputStreamos) throws IOException,HttpException {
String s;
String input;
input = (String) variables.get("data");
int a = Integer.parseInt(input);
   //System.out.println("Data from Click Web = "+(a));
   if (a==0)
   {System.out.println("refresh  ");
for (int i = 0;i<=10;i++)
{
if (DataReader.Data_SW[i]==1)
{
DataReader.Data_SW[i]=0;
checkstatus(i+1);
}
}
/*for (int i = 0;i<=7;i++)
{

if(i==1){

if (status[1] == 0)

{

   output[1] ="of.gif";
   status[1] = 1;
}
else
{
   output[1] ="on.gif";
   status[1] = 0;
}
}
if(i==2){
if (status[2] == 0)
{
   output[2] ="of.gif";
}
else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
    output[2] ="on.gif";
}
}
if(i==3){
if (status[3] == 0)
{
    output[3] ="of.gif";
}
else
{
    output[3] ="on.gif";
}
}
if(i==4){
if (status[4] == 0)
{
    output[1] ="of.gif";
    output[2] ="of.gif";
    output[3] ="of.gif";
}
else
{
output[1] ="on.gif";
    output[2] ="on.gif";
    output[3] ="on.gif";
}
}
}*/


   }
else if (a<9)
{
checkstatus(a);
}
else
{
//off();
//DataReaderdd = new DataReader();
//dd.send_Socket(9);
for (int i = 0;i<=7;i++)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
output[i] ="of.gif";
status[i] = 0;
}
}
  // System.out.println("out   "+output[0]+" "+output[1]+" "+output[2]+" "+output[3]+" "+output[4]+" "+output[5]+"
"+output[6]+" "+output[7]+" ");
   // System.out.println("status"+status[0]+" "+status[1]+" "+status[2]+" "+status[3]+" "+status[4]+" "+status[5]+"
"+status[6]+" "+status[7]+" ");
//System.out.println(hhh);
    s = "Content-Type: text/html\r\n\r\n\r\n";
s += "<table width='200' border='0' cellspacing='0' cellpadding='0'>";
s +="<tr>\n";
s += "<td><imgsrc='../../"+output[0]+"' width='30' height='30' /></td>";
s += "<td></td>";
s += "<td><imgsrc='../../"+output[3]+"' width='30' height='30' /></td>";
s +="</tr>";
s +="<tr>\n";
s += "<td><imgsrc='../../"+output[1]+"' width='30' height='30' /></td>";
s += "<td></td>";
s += "<td><imgsrc='../../"+output[4]+"' width='30' height='30' /></td>";
s +="</tr>";
s +="<tr>\n";
s += "<td><imgsrc='../../"+output[2]+"' width='30' height='30' /></td>";
s += "<td></td>";
s += "<td><imgsrc='../../"+output[5]+"' width='30' height='30' /></td>";
s +="</tr>";
s +="<tr>\n";
s += "<td></td>";
s += "<td><imgsrc='../../"+output[6]+"' width='30' height='30' /></td>";
s += "<td></td>";
s +="</tr>";
s +="<tr>\n";
s += "<td></td>";
s += "<td><imgsrc='../../"+output[7]+"' width='30' height='30' /></td>";
s += "<td></td>";
s +="</tr>";
//for (int i=0;i<=7;i++){
//s +="<td><imgsrc='../../"+output[i]+"' width='30' height='30' /></td>";
//s +="<td><imgsrc='"+output[i]+"' width='30' height='30' /></td>";
//}
//s +="</tr>";
s +="</table>";
os.write(s.getBytes());
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}
}
```

WebServer .java

```
import se.imsys.net.*;
  public class WebServer {
    public static void main(String arg[]) {
                    HttpServerhs = new HttpServer(); // Create webserver
                    Thread t = new Thread(hs); // Create webserver thread
                    hs.addCgi(new LEDcgi()); // Add the Form handler
                    t.start(); // Start thread

                    ThreadCan can = new ThreadCan();
                    Thread c = new Thread(can);
                    c.start();
    }
  }
```

ThreadCan.java

```
import se.imsys.comm.*;
class  ThreadCan implements Runnable
{       DataReader app = new DataReader();
        public void run()
        {
                if (app.init())
                {
                        System.out.println("Waitiing data from Board");
                        app.readDataViaCAN();
                }
                else  System.out.println("Can't app initial");
                        }
}
```

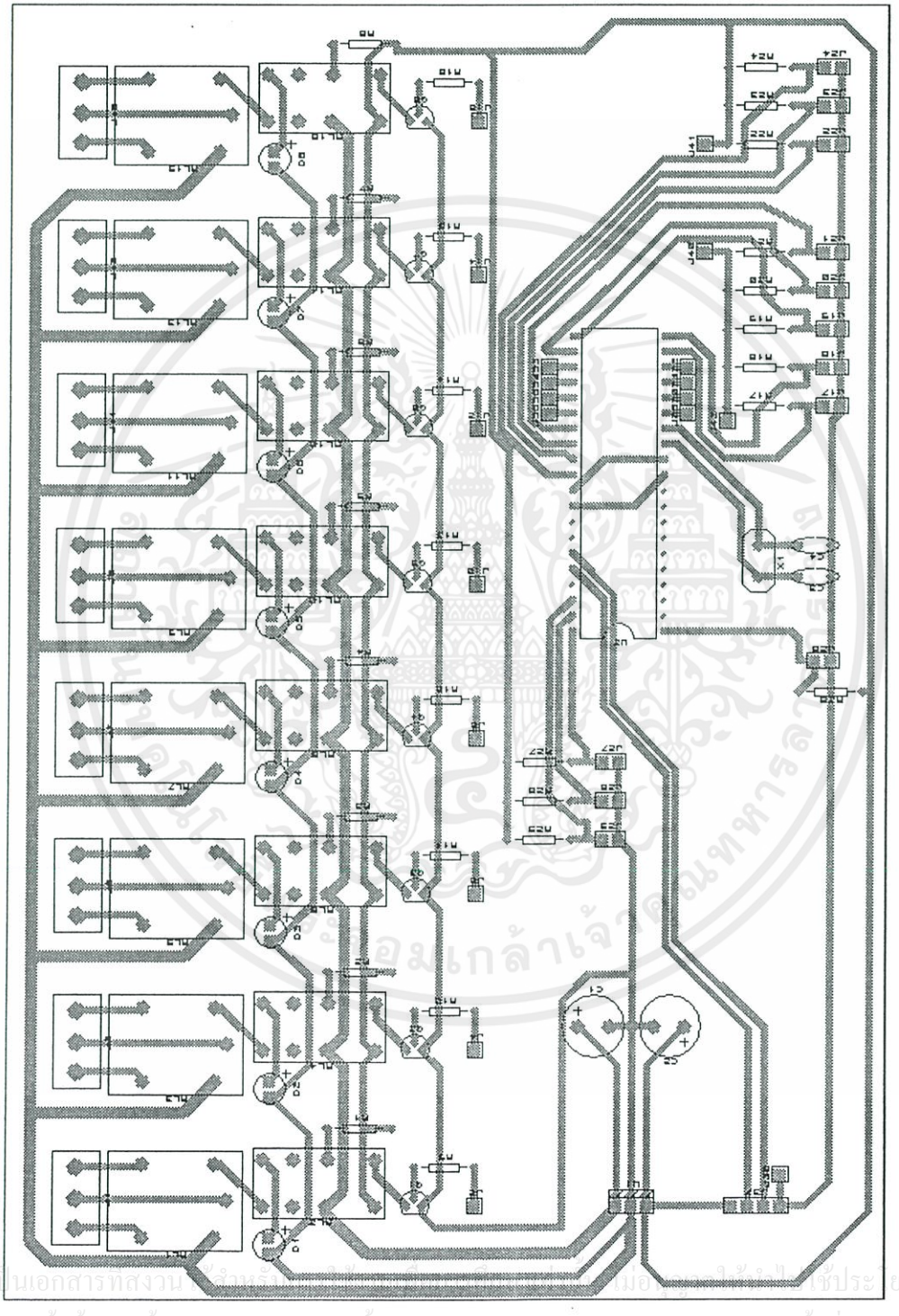
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข

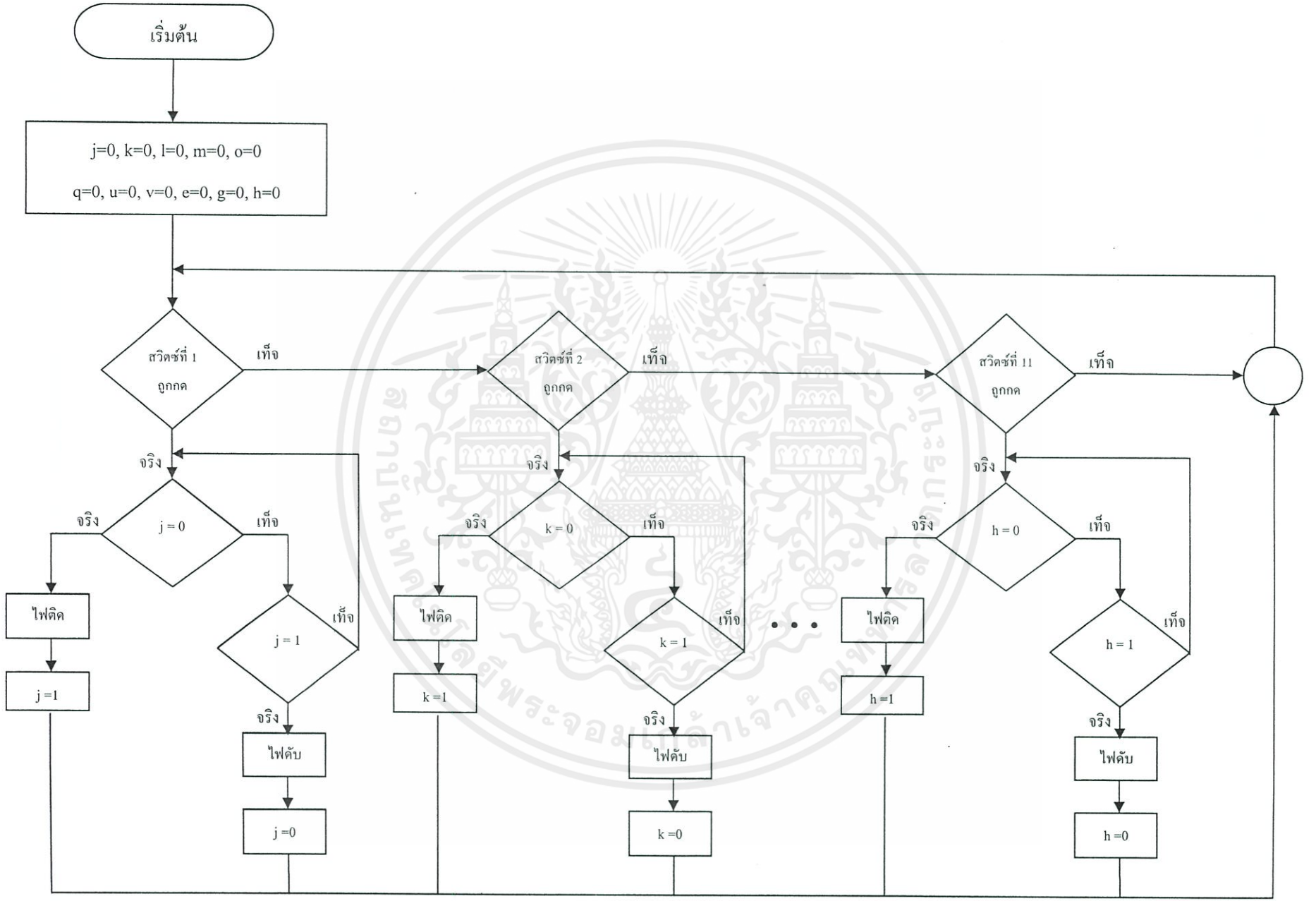
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข

ลายวงจร



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


เริ่มต้น
j=0, k=0, l=0, m=0, o=0
q=0, u=0, v=0, e=0, g=0, h=0
สวิตช์ที่ 1 ถูกกด
เท็จ
สวิตช์ที่ 2 ถูกกด
เท็จ
สวิตช์ที่ 11 ถูกกด
เท็จ
จริง
จริง
จริง
จริง
j = 0
เท็จ
จริง
k = 0
เท็จ
จริง
h = 0
เท็จ
ไฟติด
j =1
j = 1
เท็จ
จริง
ไฟดับ
j =0
ไฟติด
k =1
k = 1
เท็จ
จริง
ไฟดับ
k =0
ไฟดิด
h =1
h = 1
เท็จ
จริง
ไฟดับ
h =0