ระบบการควบคุมอุปกรณ์ไฟฟ้าและรักษาความปลอดภัยภายใน บ้านผ่านเครือข่ายคอมพิวเตอร์

HOME-APPLIANCE CONTROL AND SECURITY SYSTEM VIA COMPUTER NETWORK

โดย

นายภูวเนศวร์ ตาไฝ รหัส 48015163

นายภัทรพงศ์ ไชยศิริ รหัส 48015174

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาอิเล็กทรอนิกส์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

ระบบการควบคุมอุปกรณ์ไฟฟ้าและรักษาความปลอดภัยภายใน บ้านผ่านเครือข่ายคอมพิวเตอร์

HOME-APPLIANCE CONTROL AND SECURITY SYSTEM VIA COMPUTER NETWORK

โดย

นายภูวเนศวร์ ตาไฝ รหัส 48015163

นายภัทรพงศ์ ไชยศิริ รหัส 48015174

อาจารย์ที่ปรึกษา

รศ.ดร.มนัส สังวรศิลป์

ปริญญานิพนธ์สำหรับปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาอิเล็กทรอนิกส์

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2550

ปริญญานิพนธ์ ปีการศึกษา 2550

ภาควิชา อิเล็กทรอนิกส์

คณะ วิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบการควบคุมอุปกรณ์ไฟฟ้าและรักษาความปลอดภัยภายใน บ้านผ่านเครือข่ายคอมพิวเตอร์

HOME-APPLIANCE CONTROL AND SECURITY SYSTEM VIA COMPUTER NETWORK

ผู้จัดทำ

1. นายภูวเนศวร์ ตาไฝ รหัส 48015163

2. นายภัทรพงศ์ ไชยศิริ รหัส 48015174

.....อาจารย์ที่ปรึกษา

(รศ.ดร. มนัส สังวรศิลป์)

ระบบการควบคุมอุปกรณ์ไฟฟ้าและรักษาความปลอดภัยภายในบ้านผ่านเครือข่ายคอมพิวเตอร์

นายภูวเนศวร์ ตาไฝ รหัส 48015163
นายภัทรพงศ์ ไชยศิริ รหัส 48015174
รศ.ดร.มนัส สังวรศิลป์ อาจารย์ที่ปรึกษา
ปีการศึกษา 2550

บทคัดย่อ

รายงานฉบับนี้จัดทำขึ้นเพื่ออธิบายการออกแบบและสร้าง เครื่องควบคุมอุปกรณ์ไฟฟ้าและรักษาความปลอดภัยภายในบ้านผ่านเครือข่ายคอมพิวเตอร์ โดยใช้โมดูลอินเตอร์เน็ตไมโครคอนโทรเลอร์ RABBIT 3000 RCM 3720 CORE MODULE ซึ่งเป็นอุปกรณ์ที่สามารถโปรแกรมข้อมูลที่ใช้ในการควบคุมการเปิด-ปิดอุปกรณ์ไฟฟ้า ควบคุมการเคลื่อนไหวของกล้องและการตรวจจับของเซ็นเซอร์อินฟาเรด ผ่านทางเครือข่ายอินเตอร์เน็ตโดยอาศัยโพรโตคอล ทีซีพี/ไอพี (TCP/IP Protocol) ในการสื่อสารข้อมูล โดยพัฒนาด้วยโปรแกรม Dynamic C และใช้ Program C++Builder6 ช่วยในการเขียนหน้าต่างควบคุมและตรวจสอบสถานะต่างๆภายในบ้าน ซึ่งในส่วนของการ เปิด-ปิดอุปกรณ์ไฟฟ้าจะใช้ รีเลย์ (Relay)ในการควบคุม และใช้ เซอร์โวมอเตอร์ควบคุมการเคลื่อนไหวของกล้อง โดยรับคำสั่งจากไมโครคอนโทรเลอร์ RABBIT 3000 และใช้เซ็นเซอร์อินฟาเรดตรวจจับการเคลื่อนไหว เพื่อตรวจสอบผู้บุกรุกที่เข้ามาในบ้าน ทำให้เราสามารถควบคุมอุปกรณ์ไฟฟ้าและตรวจสอบสถานะของบ้านได้จากทุกที่ที่สามารถเข้าถึงอินเตอร์เน็ตได้โดยผ่าน IP Address

HOME-APPLIANCE CONTROL AND SECURITY SYSTEM VIA COMPUTER NETWORK

Mr.Phuwaneth Tafai ID. 48015163

Mr.Pattarapong Chaisiri ID. 48015174

Assoc. Prof. Dr. Manas Sangworasilp Advisor

Educational Year 2007

Abstract

This project is conducted in order to design and build a device for controlling household appliances and home security system via computer network using RABBIT 3000 RCM 3720 Core Module. This internet and microcontroller module can be programmed to control appliances and the movement of a camera and infrared sensors through internet network using TCP/IP Protocol to send and receive data. The following programs and apparatus are used in building this device (1) Dynamic C and C++ Builder6 to develop the control box and inspect the status of the house (2) Relay to control the switching of the appliances (3) servo to control movement of a camera (4) promise order from the RABIT 3000 microcontroller and (5) Infrared sensor to detect movement of home intruders .This device allows us to control the appliances and check the status of our home from anywhere with internet access.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดีเนื่องจากได้รับการช่วยเหลือจาก รศ.ดร.มนัส สังวรศิลป์ อาจารย์ที่ปรึกษาโปรเจค อาจารย์ ชินภัทร นันทจิวากรชัย และนายธีระศักดิ์ จันทร์วิเมลือง พี่นักศึกษาปริญญาโท ที่ได้คอยให้คำปรึกษาในเรื่องต่าง ๆ ที่พวกเราไม่เข้าใจ ทั้งยังช่วยสนับสนุนในเรื่องของสถานที่ในการทำโปรเจคและ เครื่องมือต่าง ๆ เป็นอย่างดี พวกเราจึงขอกราบขอบพระคุณบุคคลที่ได้กล่าวไปข้างต้นนี้เป็นอย่างสูงไว้ ณ โอกาสนี้ และสุดท้ายขอกราบขอบพระคุณบิดามารดาที่ได้คอยอุปการะในเรื่องการเรียนด้วยดีตลอดมา ทั้งคอยดูแลและให้ความช่วยเหลือในทุก ๆ เรื่อง

ผู้จัดทำ

นายภูวเนศวร์ ตาไฝ

นายภัทรพงศ์ ไชยศิริ

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญรูปภาพ

สารบัญรูปภาพ (ต่อ)

หน้า

สารบัญตาราง

หน้า

บทที่ 1
บทนำ

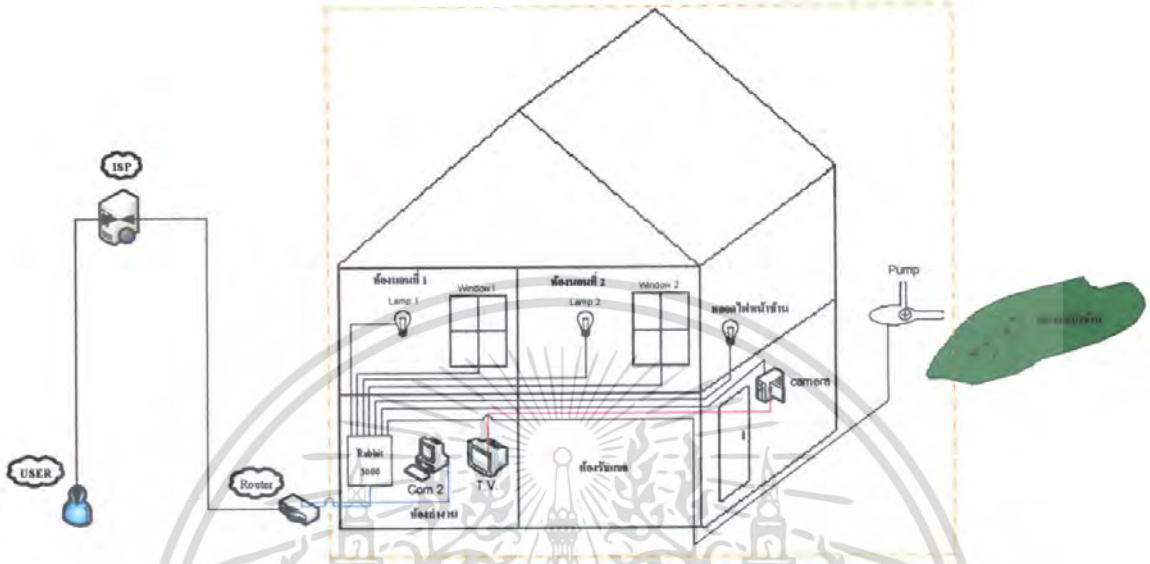
1.1 ความเป็นมาและความสำคัญ

ในยุคปัจจุบันนี้มนุษย์ได้พัฒนาเทคโนโลยีเพื่ออำนวยความสะดวกในการใช้ชีวิตประจำวันมากมายจนเรียกได้ว่าแทบจะตามไม่ทัน โดยเฉพาะเทคโนโลยีทางด้าน การสื่อสาร เช่น โทรศัพท์มือถือ Internet ที่นับวันจะเป็นส่วนหนึ่งของชีวิตประจำวันไปเลยก็ได้ เมื่อ Internet ได้เข้ามามีบทบาทกับมนุษย์เรามากขึ้น ผู้จัดทำโครงงานจึงเลงเห็นว่าควรที่จะประยุกต์ใช้ประโยชน์จาก การติดต่อสื่อสารทางด้าน Networkให้มากที่สุดก็คงจะเป็น การรักษาความปลอดภัยและการควบคุมระยะไกล และเนื่องจากว่าทรัพย์สินที่มนุษย์หามาได้ด้วยความยากลำบากนั้น ถ้าเกิดความเสียหายหรือถูกลักขโมยไปก็จะต้องเกิดความเสียใจ อย่างที่สุดเพื่อแก้ไขปัญหานี้ จึงเกิดโครงงานนี้ขึ้นมา โดยจะใช้ Rabbit 3000 Core Modules RCM 3720 ซึ่งมีความสามารถที่เป็นได้ทั้งไมโครคอลโทรลเลอร์ และเป็น Server ให้บริการแก่เครื่องคอมพิวเตอร์อื่น ๆ ในการออกแบบนั้นผู้จัดทำโครงงานจะใช้จะใช้ Modules นี้เป็นเครื่องที่คอยควบคุมอุปกรณ์เครื่องใช้ไฟฟ้า และคอยรักษาความปลอดภัยอยู่ที่บ้าน โดยเมื่อใดที่เจ้าของบ้านอยากดูว่าบ้านของตนเป็นอย่างไรบ้างก็สามารถดูได้จากโปรแกรมที่เขียนขึ้นโดย C++ Builder 6

1.2 วัตถุประสงค์

1) เพื่อศึกษาวิธีการสื่อสารผ่านเครือข่ายอินเตอร์เน็ต
2) เพื่อศึกษาการทำงานของ Rabbit 3000 Core Modules RCM 3720
3) เพื่อประยุกต์ใช้งานเครือข่ายอินเตอร์เน็ต ให้เกิดประโยชน์ทางด้านการรักษาความปลอดภัย
4) เพื่อนำความรู้จากการเรียน วิชา Micro Controller มาประยุกต์ใช้งานด้านการควบคุม
5) เพื่อนำความรู้จากการเรียน วิชา Data Communication มาประยุกต์ใช้งานด้านการควบคุม
6) เพื่อฝึกทักษะการแก้ปัญหาเฉพาะหน้าในการทำงานจริง
7) เพื่อเตรียมความพร้อมก่อนที่จะได้ออกไปทำงาน ทางด้านวิศวกรรม

1.3 ขอบเขตของโครงงาน



รูปที่ 1.1 แสดงขอบเขตของโครงงาน

จากรูปที่ 1.1 แสดงภาพรวมของระบบควบคุมและรักษาความปลอดภัยภายในบ้านผ่านระบบอินเตอร์เน็ต โดยใช้ Rabbit 3000 Core Module RCM 3720 ที่เป็นทั้ง Microcontroller และเป็น Server ที่คอยให้บริการแก่ผู้ใช้จากภายนอก จากรูปเราออกแบบให้ Module ตัวนี้สามารถควบคุมการปิดเปิดและตรวจสอบสถานะของ Lamp 1, Lamp 2, หลอดไฟหน้าบ้าน, Pump, เซ็นเซอร์แสงที่หน้าต่างที่ 1และ หน้าต่างที่ 2 บนห้องนอน, และสามารถหมุนปรับมุมมองของกล้องหน้าบ้านซึ่งก็จะสามารถทราบได้ว่ามีใครมาหา โดยมองภาพผ่าน จอ T.V ในห้องทำงานได้

ผู้ใช้งานที่ติดต่อเข้ามาจากภายนอกจะสามารถทราบสถานะต่าง ๆ ของหลอดไฟ, เซ็นเซอร์หน้าต่างที่ 1, เซ็นเซอร์หน้าต่างที่ 2, และปั๊มน้ำ ว่าเปิดหรือ ปิดอยู่ และสามารถสั่งปิดเปิดโดยใช้โปรแกรมที่เขียนขึ้นด้วย C++ Builder6

บทที่2
ทฤษฎีที่เกียวข้อง

2.1 TCP/IP (Transmission Control Protocol/internet Protocol)

หลายเทคโนโลยีที่เราท่านใช้อยู่ทั่วไปมีจุดกำเนิดจากเทคโนโลยีการสงคราม IP เน็ตเวิร์กก็เป็น หนึ่งในนั้น เมื่อครั้งสงครามเย็นระหว่างสหรัฐอเมริกาและสหภาพโซเวียต กระทรวง กลาโหม ภายใต้รัฐบาลกลางสหรัฐฯ จ้างมหาวิทยาลัยต่าง ๆ ทำวิจัยเพื่อสร้างเครือข่ายที่ทนต่อความล้มเหลว (ด้วยระเบิดนิวเคลียร์) สิ่งที่ได้คือโพรโตคอล TCP/IP เครือข่ายคอมพิวเตอร์ที่ใช้โพรโตคอลนี้เรียกสั้น ๆว่า TCP/IP (Transmission Control Protocol/Internet Protocol) คือชุดของโพรโตคอลที่ รวมกันเป็นกลุ่มให้ใช้งานเช่น Internet Protocol (IP) , Address Resolution Protocol (ARP), Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP) ฯลฯ แต่โพรโตคอลที่ มีบทบาทสำคัญคือ Internet Protocal (IP) โดยมีหลักการทำงานคือ แบ่งเนื้อข้อมูลที่ต้องการส่งเป็นชิ้นเล็ก ๆ เรียกว่าแพ็กเก็ต ส่งแพ็กเก็ตไปยังเส้นทางที่เหมาะสมเป็นทอดจนกว่า จะถึงปลายทาง แต่ละแพ็กเก็ตอาจใช้เส้นทางคนละทิศขึ้นกับการพิจารณาของเราเตอร์ในช่วงต่าง ๆ หากเกิด ข้อผิดพลาด ณ ช่วงการส่งใด เราเตอร์ที่รับผิดชอบการส่งช่วงนั้นจะจัดส่งแพ็กเก็ตชิ้นนั้นใหม่โดยอัตโนมัติเมื่อถึงจุดหมายระบบปลายทางจะรวบรวมแพ็กเก็ตกลับให้เป็นเนื้อข้อมูลดังเดิมซึ่งถ้าจะว่ากันตามทฤษฎีแล้ว TCP/IP นั้นจะประกอบด้วยส่วนสำคัญอยู่ 2 ส่วน ด้วยกันก็คือ TCP หรือ Transmission Control Protocol และอีกส่วนก็คือ IP หรือ Internet Protocol นั่นเอง การแบ่งลักษณะในการทำงานก็จะแบ่งเป็น TCP มีหน้าที่ในการตรวจสอบการรับส่งข้อมูลระหว่าง เครื่องคอมพิวเตอร์ผู้รับ และเครื่องคอมพิวเตอร์ผู้ส่ง ให้ได้รับขอมูลที่ถูกต้องและครบ ถ้วนหรือว่าหากมีการสูญหายของข้อมูลก็จะมีการแจ้งให้ต้นทางที่ส่งข้อมูลมารับทราบแล้วให้ทำการส่งข้อมูลมาให้ใหม่

2.2 ลักษณะการทำงาน

ลักษณะการทำงานของ IP นั้น จะทำหน้าที่ในการเลือกเส้นทางที่จะใช้ในการรับส่งข้อมูลใน ระบบเครือข่าย และทำการตรวจสอบที่อยู่ของผู้รับโดยการใช้ข้อมูลขนาด 4 Byte เป็นตัวกำหนด แอดเดรสหรือที่เราเรียกกันว่า IP Address ซึ่งโพรโตคอล TCP จะทำงานอยู่ในชั้น Transport Layer ตัวแพ็กเก็ต TCP จะประกอบด้วย ส่วนหัว (Header) และสวนข้อมูล (Data) และโพรโตคอล IP จะ ทำงานอยู่ในชั้น Network Layer ตัวแพ็กเก็ต IP ประกอบด้วย 2 ส่วนใหญ่ ๆ คือ ส่วนหัว (IP Header) จะประกอบด้วย IP แอดเดรสของเครื่องต้นทางและปลายทาง และส่วนข้อมูล

(IP Data) จะ เป็นที่เก็บโพรโตคอล TCP เนื่องจากโพรโตคอล TCP/IP จะถูก Encapsulate ให้มาอยู่ในส่วนของ แพ็กเก็ต IP

2.3 จุดเด่นของโพรโตคอล TCP/IP

2.3.1 สามารถนำส่งข้อมูลไปถึงจุดหมายได้แม้เส้นทางบางที่เสียหาย: เป็นจุดประสงค์หลักที่ ช่วยให้ทนต่อความล้มเหลว โดยหากระหว่างการสื่อสารข้อมูลและมีเส้นทางใดเสียหายหรือล้มเหลว IP เน็ตเวิร์กจะปรับใช้เส้นทางอื่นที่ทดแทนได้เพื่อนำส่งข้อมูลให้ไปถึงปลายทางอย่างอัตโนมัติ ผู้ส่งและผู้รับข้อมูลไม่จำเป็นต้องรับรู้หรือปรับตัวแต่ประการใด

2.3.2 ไม่ขึ้นกับแพลตฟอร์มใด ๆ: ไม่ว่าเครือขายนั้นเป็นเครือขายท้องถิ่นหรือเครือข่ายระหว่าง ภูมิภาค เป็นไฟล์ / พรินต์เซิร์ฟเวอร์หรือไคลเอ็นต์/เซิร์ฟเวอร์ เป็นระบบปฏิบัติการใด เน็ตเวิร์ก อินเทอร์เฟซเป็นแบบใดก็ตาม ในมุมมองของโพรโตคอล TCP/IP ก็คือ IP เน็ตเวิร์ก

2.4 จุดอ่อนของโพรโตคอล TCP/IP

2.4.1 รับส่งโดยไม่มีการรักษาความปลอดภัยเนื้อข้อมูล: การรับส่งข้อมูลด้วย IP แพ็กเก็ตไม่มี ทั้งการเข้ารหัสข้อมูลและป้องกันการปลอมแปลงใด ๆ การไม่เข้ารหัสข้อมูลอาจทำให้ผู้ไม่ประสงค์ ดีระหว่างเส้นทางที่ IP แพ็กเก็ตผานดักลอบดูเนื้อขอมูลอย่างง่ายดาย แม้ว่าเราอาจสามารถบังคับ เส้นทางของ IP แพ็กเก็ตได้ก็ไม่อาจมั่นใจได้ว่าระหว่างทางมีการดักลอบดูหรือไม่

ในเรื่องปัญหาการปลอมแปลงแบ่งออกเป็นสองกรณีคือ การปลอมแปลงหรือดัดแปลงเนื้อข้อ มูล และการปลอมแปลงส่วนหัวของ IP แพ็กเก็ต ทั้งสองกรณีให้ผลเหมือนกันคือผู้รับได้ขอมูลที่ ผิดจากความเป็นจริง ว่าจุดประสงค์ต่างกัน หากเป็นกรณีแรกนั้น ผู้ไม่หวังดีต้องการหลอกหรือกลั่นแกล้ง ให้ได้ข้อมูลผิด ๆ หากเป็นกรณีหลัง ผู้ไม่หวังดีต้องการแอบอ้างว่าข้อมูลนั้นมากจากแหล่งที่ผู้รับไว้ใจหรือแหลงอื่นที่กลายเป็นเหยื่อของการแอบอ้างโดยไม่รู้ตัว

2.4.2 รับส่งโดยไม่คำนึงถงคุณภาพการให้บริการ: การรับส่งต่อ IP แพ็กเก็ตระหว่างเครือข่าย ย่อยไปเป็นทอดนั้นใช้หลักการใครมาก่อนได้ก่อน ฉะนั้นจึงคาดเดาไม่ได้ว่าข้อมูลที่นำสงไปจะไป ถึงปลายทางเมื่อใด แม้ว่า IP เน็ตเวิร์กใช้หลักการเลือกเส้นทางที่เหมาะสมที่สุดในขณะนั้นก็ตาม หากแต่ความเหมาะสมนั้นผู้ส่งและผู้รับไม่อาจคาดการณ์หรือมีส่วนร่วมตัดสินใจได้เลยว่าจะช้าเร็ว หรือมีโอกาสที่ข้อมูลผิดพลาดมากน้อยเพียงไร

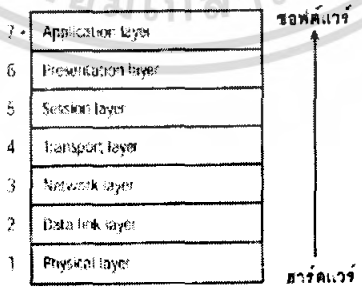
สำหรับเรื่องการรับส่งโดยไม่มีการรักษาความปลอดภัยเนื้อข้อมูลนั้น องค์กรกลางของอินเทอร์เน็ตได้ออกมาตรฐานที่ช่วยแก้ไขปัญหานี้คือ IPSec โดยมีทั้งการเข้าและถอดรหัสเนื้อข้อมูลในระดับ IP แพ็กเก็ตการตรวจสอบความถูกต้องเนื้อข้อมูลและการพิสูจน์ตนของ IP แพ็กเก็ตเพื่อป้องกันการปลอมแปลง

2.5 มาตรฐานการสื่อสารข้อมูล

การกำหนดมาตรฐานของการสื่อสารข้อมูลนั้น นับว่ามีความจำเป็นอย่างมากสำหรับระบบเครือข่ายที่มี องค์ประกอบของอุปกรณ์ต่างๆ หลากหลายผู้ผลิต ซึ่งอุปกรณ์ทั้งหมดเหล่านั้นจะต้องทำงานเข้ากันได้อย่างราบรื่น การกำหนดมาตรฐานต่างๆ นั้นจะเริ่มตั้งแต่โครงสร้างพื้นฐานของฮาร์ดแวร์ระบบเครือข่าย ได้แก่ ระบบสายเคเบิล อุปกรณ์ในการส่งสัญญาณข้อมูล ตลอดจนถึงเครื่องเซิร์ฟเวอร์ และซอฟต์แวร์ในการสื่อสารบนระบบเครือข่าย เพื่อเป็นการรับประกันว่า ส่วนประกอบต่างๆ จะสามารถทำงานร่วมกันได้ ผู้ผลิตฮาร์ดแวร์และซอฟต์แวร์ระบบเครือข่าย จะต้องทำตามคำแนะนำตามมาตรฐานการออกแบบและสร้างผลิตภัณฑ์ ซึ่งกำหนดขึ้นโดย องค์กรมาตรฐานสากล (International Organization for Standardization - ISO) โดยมาตรฐานที่กำหนดขึ้นและได้ประกาศใช้ตั้งแต่ปี คศ.1984 เรียกว่า Open Systems Interconnection Reference Model เรียกสั้นๆ ว่า OSI Reference Model หรือ ISO/OSI Model

2.6 แบบจำลอง OSI

OSI Reference Model เป็นการกำหนดชุดของคุณลักษณะเฉพาะที่ใช้อธิบายโครงสร้างของระบบเครือข่าย โดยมีวัตถุประสงค์ เพื่อให้ผู้ผลิตฮาร์ดแวร์หรือซอฟต์แวร์ใดๆ ใช้เป็นโครงสร้างอ้างอิงในการสร้างอุปกรณ์ให้สามารถทำงานร่วมกันได้อย่างดีบนระบบเครือข่าย โดยมีการ จัดแบ่งเลเยอร์ของ OSI ออกเป็น 7 เลเยอร์ แต่ละเลเยอร์จะมีการ โต้ตอบหรือรับส่งข้อมูลกับเลเยอร์ ที่อยู่ข้างเคียงเท่านั้น โดยเลเยอร์ที่อยู่ชั้นล่างจะกำหนดลักษณะของอินเตอร์เฟซ เพื่อให้บริการกับเลเยอร์ที่อยู่เหนือขึ้นไปตามลำดับขั้น เริ่มตั้งแต่ส่วนล่างสุดซึ่งเป็นการจัดการลักษณะทางกายภาพของฮาร์ดแวร์และการส่งกระแสของข้อมูลในระดับบิต ไปสิ้นสุดที่แอพพลิเคชั่นเลเยอร์ในส่วนบนสุด



รูปที่ 2.1 OSI Reference Model

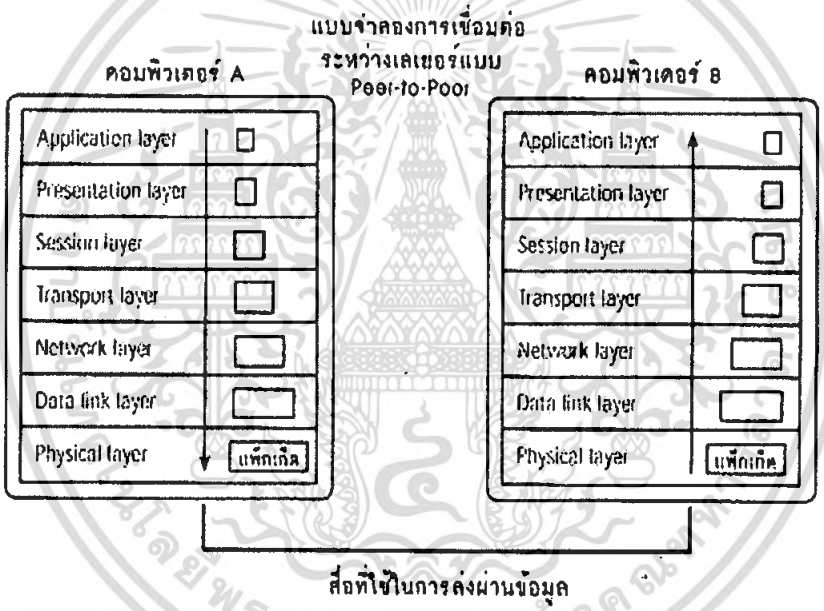
2.6.1 หลักการออกแบบเลเยอร์

- แต่ละเลเยอร์จะมีการกำหนดการทำงานอย่างละเอียดโดยมีการทำงานเป็นอิสระไม่ขึ้นต่อกัน

- ฟังก์ชันภายในเลเยอร์จะพยายามมุ่งไปสู่ข้อกำหนดมาตรฐาน (Standard Protocol)
- ขอบเขตของเลเยอร์จะถูกเลือกและจำกัดให้มีปริมาณการเชื่อมต่อระหว่างเลเยอร์ให้น้อยที่สุด
- จำนวนของเลเยอร์ต้องมากพอที่จะแยกฟังก์ชั่นที่จำเป็นและแตกต่างกันไม่ให้อยู่ในเลเยอร์เดียวกัน

2.6.2 การทำงานของ OSI Reference Model

การที่แพ็กเก็ตข้อมูลเดินทางจากเครื่องคอมพิวเตอร์ A ไปยังเครื่องคอมพิวเตอร์ B นั้น มีกระบวนการทำงานดังนี้



รูปที่ 2.2 การส่งแพ็กเก็ตใน OSI Reference

จากแผนผัง คอมพิวเตอร์ A และคอมพิวเตอร์ B มีโครงสร้างเป็น OSI ซึ่งมี 7 เลเยอร์ เมื่อเครื่อง คอมพิวเตอร์ A พร้อมที่จะส่งสัญญาณข้อมูลไปยังเครื่องคอมพิวเตอร์ B นั้น แต่ละเลเยอร์ในเครื่อง คอมพิวเตอร์ A จะเสมือนกับว่ามีการส่อสารกับเลเยอร์ในระดับเดียวกันบนเครื่องคอมพิวเตอร B ถึงแม้ ว่าจะไม่มีการสื่อสารระหว่าง เลเยอร์เหล่านี้เกิดขึ้นจริง แต่เลเยอร์ในระดับต่างๆ บนเครื่อง คอมพิวเตอร์ทั้งคู่นั้นจะทำตามกฎเกณฑ์หรือโพรโตคอล (protocol) อย่างเดียวกัน เพื่อให้มั่นใจได้ว่าแต่ละเลเยอร์บนเครื่องคอมพิวเตอร์ฝ่ายผู้รับจะได้รับแพ็กเก็ตข้อมูล แบบเดียวกันกับแพ็กเก็ต ข้อมูลที่รวบรวม โดยแต่ละเลเยอร์บนเครื่องคอมพิวเตอร์ฝ่ายผู้ส่ง โดยแพ็กเก็ตข้อมูลจะเริ่มที่ระดับ สูงสุดคือ Application Layer บนเครื่องคอมพิวเตอร์ A และเคลื่อนลงมาทีละระดับชั้นจนมาถึงชั้น ล่างสุดคือ Physical Layer การที่แพ็กเก็ตเคลื่อนผ่านจากระดับหนึ่งไปยังระดับถัดไป

นั้น จะมีการ กำหนดที่อยู่ การจัดรูปแบบของข้อมูลและอื่นๆ ซึ่งแต่ละเลเยอร์จะเป็นตัวจัดการและมีกระบวนการ ของตนเอง เมื่อแพ็กเก็ตเคลื่อนตัวลงมาถึง Physical Layer ก็จะถูกแปลงให้เป็นกระแสข้อมูลแบบ อนุกรมและส่งผานสื่อกลางคือสาย สัญญาณ ซึ่งเป็นเลเยอร์เดียวที่เครื่องคอมพิวเตอร์ A สื่อสารกับ เครื่องคอมพิวเตอร์ B และเมื่อสัญญาณ ข้อมูลมาถึงเครื่องคอมพิวเตอร์ B กระบวน การก็จะเริ่มทำใน ทางตรงข้าม คือจะทำการแยกแพ็กเก็ตออกผ่าน OSI ทั้ง 7 เลเยอร์ ส่งย้อนกลับขึ้นไปยัง Application Layer ของเครื่องคอมพิวเตอร์ B เมื่อแพ็กเก็ตเดินทางผ่านเลเยอร์ระดับต่างๆ แต่ละเลเยอร์ จะแยก ข้อมูลข่าวสารตามกำหนดที่อยู่ และการจัดรูป แบบของแพ็กเก็ต จนเมื่อมาถึงเลเยอร์ระดับสูงสุดคือ Application Layer ก็จะเหลือเฉพาะข้อมูลที่เหมือนกับบน Application Layer ของเครื่อง คอมพิวเตอร์ A

เลเยอร์ 1: Appplication Layer

Layer นี้เป็นการแลกเปลี่ยนข้อมูลระหว่างโปรแกรมที่ทำงานอยู่บนคอมพิวเตอร์และบริการ อื่นๆ ในเครือข่าย เช่น Database

เลเยอร์ 2: Data Link Layer

เลเยอร์นี้มีจุดประสงค์หลักคือพยายามควบคุมการส่งข้อมูลให้เสมือนกับว่าไม่มีความผิดพลาด เกิดขึ้น เพื่อให้เลเยอร์สูงขึ้นไปสามารถนำข้อมูลไปใช้ได้อย่างถูกต้อง วิธีการคือฝ่ายผู้ส่งจะทำการ แตก ข้อมูลออกเป็นเฟรมข้อมูล (data-frame) โดยจะต้องมีการกำหนดขอบเขตของเฟรม (frame boundary) โดยการเติมบิทเข้าไปยังจุดเริ่มต้นและจุดสิ้นสุดของเฟรม จากนั้นทำการส่งเฟรมข้อมูล ออก ไปทีละชุดและรอรับการตอบรับ (acknowledge frame) จากผู้รับ ถ้าหากมีการสูญหายของเฟรม ข้อมูล ซึ่งอาจเนื่องมาจากสัญญาณรบกวนจากภายนอกหรือข้อผิดพลาดอื่นๆ ในกรณีนี้ฝ่ายผู้ส่ง จะต้องส่ง เฟรมข้อมูลเดิมออกมาใหม่

เลเยอร์ 3: Network Layer

เป็นเลเยอร์ที่ทำหน้าที่หลักเกี่ยวข้องกับการหาเส้นทาง (routing) ในการส่งแพคเก็ตจากต้นทาง ไปยังปลายทาง ซึ่งจะมีการสลับช่องทางในการส่งข้อมูลหรือที่เรียกว่า แพ็กเกตสวิตชิ่ง (packet switching) มีการสร้างวงจรเสมือน (virtual circuit) ซึ่งคล้ายกับว่ามีเส้นทางเชื่อมโยงกันระหว่าง คอมพิวเตอร์ 2 เครื่องให้ติดต่อสื่อสารถึงกันได้โดยตรง การกำหนดเส้นทางการส่งข้อมูลนั้น คอมพิวเตอร์ฝ่ายผู้ส่งอาจทำหน้าที่พิจารณาหาเส้นทางที่เหมาะสมในการส่งข้อมูล ตั้งแต่ต้น หรือ อาจใช้วิธีแบบไดนามิกส์ (dynamic) คือแต่ละแพคเก็ตสามารถเปลี่ยนแปลงเส้นทางได้ตลอดเวลา นอก จากนี้เครื่องคอมพิวเตอร์ฝ่ายผู้ส่งยังมีหน้าที่ในการจัดการเรื่องที่อยู่ของเครือข่ายปลายทางโดย จะมีการแปลงที่อยู่แบบตรรกะ (logical address) ให้เป็นที่อยู่แบบกายภาพ (physical address) ซึ่งถูก กำหนดโดยการ์ดเชื่อมต่อระบบเครือข่าย

เลเยอร์ 4: Transport Layer

Transport Layer ทำหน้าที่เสมือนบริษัทขนส่งที่รับผิดชอบการจัดส่งข้อมูลโดยปราศจากความ ผิดพลาด ซึ่งมีหน้าที่หลักคือ การตรวจสอบและแก้ไขความผิดพลาดที่เกิดขึ้นในข้อมูล คอย แยกแยะ และจัดระเบียบของแพ็กเก็ต ข้อมูลให้จัดเรียงลำดับอย่างถูกต้อง และมีขนาดที่เหมาะสม นอกจากนี้ ยังทำการผนวกข้อมูลทั้งหลายให้อยู่ในรูปของ วงจรเดียวหรือเรียกว่าการมัลติเพล็กซ์ (multiplex) และมีกลไกสำหรับควบคุมการไหลของข้อมูลให้มีความสม่ำเสมอ

เลเยอร์ 5: Session Layer

จากเลเยอร์ที่ผ่านมาจะเห็นว่าการทำงานต่างๆ จะเกี่ยวพันอยู่เฉพาะกับบิตและข้อมูลเท่านั้น โดยไม่ได้สนใจเกี่ยวกับสถานะภาพการใช้งานจริงของผู้ใช้แต่อย่างใด ซึ่งหน้าที่ดังกล่าวนี้จะเกิดขึ้นที่ Session Layer ในเลเยอร์นี้จะมีการให้บริการสำหรับการใช้งานเครื่องที่อยู่ห่างไกลออกไป (remote login) การถ่ายโอนไฟล์ระหว่างเครื่อง โดยจะมีการจัดตั้งการสื่อสารระหว่าง 2 ฝ่าย เรียกว่า Application Entities หรือ AE ซึ่งเทียบได้กับบุคคล 2 คนที่ต้องการสนทนากันทางโทรศัพท์ โดย Session Layer จะมีหน้าที่จัดการให้การสนทนาเป็นไปอย่างราบรื่น โดยการเฝ้า ตรวจสอบการไหลของข้อมูลอย่างเป็นจังหวะ ดูแลเรื่องความปลอดภัยเช่น ตรวจสอบอายุการใช้งานของรหัสผ่าน จำกัดช่วงระยะเวลาในการติดต่อ ควบคุมการถ่ายเทข้อมูลรวมถึงการกู้ข้อมูลที่เสียหายอันเกิดมาจากเครือข่ายทำงานผิดปกติ นอก จากนี้ยังสามารถตรวจดูการใช้งานของระบบและจัดทำบัญชีรายงานช่วงเวลาการใช้งานของผู้ใช้ได้

เลเยอร์ 6: Presentation Layer

หน้าที่หลักคือการแปลงรหัสข้อมูลที่ส่งระหว่างเครื่องคอมพิวเตอร์ 2 เครื่องให้เป็นอักขระแบบเดียวกัน เครื่องคอมพิเตอร์ส่วนใหญ่จะใช้รหัส ASCII (American Standard Code for Information Interchange) แต่ในบางกรณีเครื่องที่ใช้รหัส ASCII อาจจะต้องสื่อสารกับเครื่อง เมนเฟรมของ IBM ที่ใช้รหัส EBCDIC (Extended Binary Coded Decimal Interchange Code) ดังนั้น Presentation Layer จะทำหน้าที่แปลงรหัสเหล่านี้ให้เครื่องคอมพิวเตอร์เข้าใจได้ตรงกัน นอกจากนี้ยังสามารถทำการลดขนาดของข้อ มูล (data compression) เพื่อเป็นการประหยัดเวลาใน การรับส่งและสามารถเข้ารหัสเพื่อเป็นการป้องกันการโจรกรรมข้อมูลได้อีกด้วย

เลเยอร์ 7: Application Layer

เป็นเลเยอร์บนสุดที่ทำงานใกล้ชิดกับผู้ใช้ การทำงานของเลเยอร์นี้จะเกี่ยวข้องกับโพรโตคอล ต่างๆ มากมาย ซึ่งจะมีการใช้งานที่เฉพาะตัวแตกต่างกันออกไป มีบริการทางด้านโปรแกรม ประยุกต์ต่างๆ ได้แก่ Email, file transfer, remote job entry, directory services นอกจากนี้

ยังมีการ จัดเตรียมฟังก์ชั่นในการเข้าถึงไฟล์และเครื่องพิมพ์ ซึ่งเป็นการแบ่งปันการใช้ทรัพยากรณ์บนระบบ เครื่อข่าย

2.7 แบบจำลอง TCP/IP

TCP/IP Model มีแนวคิดพื้นฐานแตกต่างจาก OSI Model คือไม่ได้มีพื้นฐานของการสื่อสาร แบบการสนทนา TCP/IP Model เป็นภาพแสดงถึงโลกของระบบเครื่อข่ายสากล (Internetworking) ที่ทำการเคลื่อนย้ายและกำหนดเส้นทางให้กับข้อมูลระหว่างเครือข่ายและระหวางเครื่อง คอมพิวเตอร์ต่างๆ เมื่อเปรียบเทียบความสัมพันธ์ระหว่างทั้ง 2 โมเดล จะพบว่ามีบางเลเยอร์ที่มีการ กำหนดคุณสมบัติที่เทียบได้ใกล้เคียงกัน แต่บางเลเยอร์ก็ไม่สามารถเทียบหาความสัมพันธ์กันได้เลย

รูปที่ 2.3 เลเยอร์ต่างๆ ใน TCP/IP และ ISO/OSI Model

TCP/IP (Transmitsion Control Protocol/ InternetProtocol) เป็นชุดของโพรโตคอลที่ถูกใช้ใน การสื่อสารผ่านเครือข่ายอินเทอร์เน็ต ได้รับการพัฒนามาตั้งแต่ปี 1960 ซึ่งถูกใช้เป็นครั้งแรกในเครือ ข่าย ARPANET ซึ่งต่อมาได้ขยายการเชื่อมต่อไปทั่วโลกเป็นเครือข่ายอินเตอร์เน็ต ทำให้ TCP/IP เป็นที่ยอมรับอย่างกว้างขวางจนถึงปัจจุบัน

2.7.1 การแบ่งชั้นของ TCP/IP

TCP/IP แบ่งออกเป็น 4 เลเยอร์ ดังรูปที่ 2.4

รูปที่ 2.4 การแบ่งชั้นของ TCP/IP

ในแต่ละเลเยอร์จะมีหน้าที่ดังนี้

• **Link Layer** - เลเยอร์นี้มีหน้าที่ควบคุมการรับส่งข้อมูลในระดับฮาร์ดแวร์ของเครือข่าย รับผิดชอบการรับส่งข้อมูลในระดับกายกาพ จนถึงการแปลความจะกลับสัญญาณไฟฟ้าเป็นข้อมูลทางคอมพิวเตอร์

• **Network Layer** - ทำหน้าที่รับข้อมูลจากชั้น Transport Layer และค้นหาและเลือกเส้นทาง ระหว่างผู้รับและผู้ส่ง เทียบได้กับ Network Layer ของ OSI Model โพรโตคอลในเลเยอร์นี้ได้แก่ IP, ICMP, IGMP

• **Transport Layer** - รับผิดชอบการรับส่งข้อมูลระหว่างปลายด้านส่งและด้านรับข้อมูล และส่ง ข้อมูลขึ้นไปให้ Application Layer นำไปใช้งาน ต่อ เทียบได้กับ Session Layer และ Transport Layer ของ OSI Model

• **Application Layer** - เป็นเลเยอร์ที่แอพพลิเคชั่นเรียกโพรโตคอลระดับล่างๆลงไป เพื่อให้บริการ ต่างๆ เช่น FTP, SMTP, Telnet, HTTP, POP

2.7.2 โครงสร้างของโพรโตคอล TCP/IP

เนื่องจาก TCP/IP เป็นชุดของโปรโตคอลประกอบด้วยโพรโตคอลหลายตัวทำงานร่วมกันในเล เยอร์ต่างๆ และมีหน้าที่แตกต่างกันออกไป ได้แก่

• **TCP: (Tranmission Control Protocol)** - อยู่ใน Transport Layer ทำหน้าที่จัดการและควบคุม การรับส่งข้อมูล และมีกลไกความคุมการ รับส่งข้อมูลให้มีความถูกต้อง (reliable) และมีการ สื่อ
สารอย่างเป็นกระบวนการ (Connection-orient)

• **UDP: (User Datagram Protocol)** - อยู่ใน Transport Layer ทำหน้าที่จัดการและควบคุมการรับ ส่งข้อมูลแ ต่ไม่มีกลไกความคุม การรับส่ง ข้อมูลให้มีเสถียรภาพและเชื่อถือได้ (unreliable, connectionless) โดยปล่อยให้เป็นหน้าที่ของแอพพลิเคชันเลเยอร์ แต่ UDP มีข้อ ได้เปรียบในการส่งข้อ มูลได้ทั้งแบบ unicast, multicast และ broadcast อีกทั้งยังทำการ ติดต่อสื่อสารได้เร็วกว่า TCP เนื่องจาก TCP ต้องเสีย overhead ให้กับขั้นตอนการสื่อสารที่ทำให้ TCP มีความน่าเชื่อถือในการรับส่งข้อมูลนั่นเอง

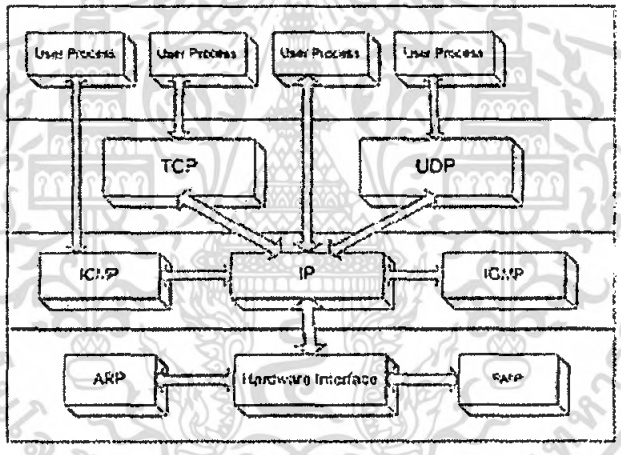
• **IP: (Internet Protocol)** - อยู่ใน Internetwork Layer เป็นโพรตคอลหลักในการสื่อสารข้อมูล มี หน้าที่ค้นหาเส้นทางระว่างผู้รับและผู้ส่ง โดยใช้ IP Address ซึ่งมีลักษณะเป็นเลขสี่ชุด แต่ละชุดมีค่า ตั้งแต่ 0-255 เช่น 172.17.3.12 ในการอ้างอิงโฮสต์ต่างๆ และกลไกการ Route เพื่อส่งต่อข้อมูลไป จนถึงจุดหมายปลายทาง

• **ICMP: (Internet Control Message Protocol)** - อยู่ใน *Internetwork Layer* มีหน้าที่ส่งข่าวสาร และแจ้งข้อผิดพลาดให้แก่ IP

• **IGMP: (Internet Group Management Protocol)** อยู่ในเน็ตเวิร์กเลเยอร์ ทำหน้าที่ในการ ส่ง UDP ดาต้าแกรมไปยัง กลุ่มของโฮสต์ หรือ โฮสต์หลายๆตัวพร้อมกัน

• **ARP : (Address Resolution Protocol)** - อยู่ใน Link Layer ทำหน้าที่เปลี่ยนระหว่าง IP แอดเดรส ให้เป็นแอดเดรสของ Network Interface เรียกว่า MAC Address ในการติดต่อระหว่างกัน MAC Address คือหมายเลขประจำของ Hardware Interface ซึ่งในโลกนี้จะไม่มี MAC Address ที่ซ้ำกัน มีลักษณะเป็นเลขฐาน 16 ยาว 6 ไบต์ เช่น 23:43:45:AF:3D:78 โดย 3 ไบต์แรกจะเป็นรหัสของผู้ผลิต และ 3 ไบต์หลังจะเป็นรหัสของผลิตภัณฑ์

• **RARP: (Reverse ARP)** - อยู่ในลิงค์เลเยอร์เช่นกัน แต่ทำหน้าที่กลับกันกับ ARP คือ เปลี่ยน ระหว่างแอดเดรสของ Network Interface ให้เป็นแอดเดรสที่ใช้โดย IP Address

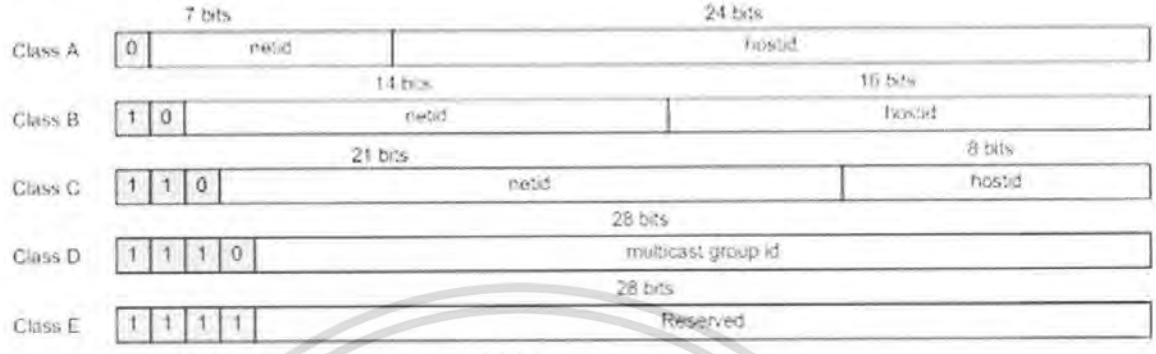


รูปที่ 2.5 แสดงให้เห็นถึงความสัมพันธ์ระหว่างโพรโตคอลต่างๆใน TCP/IP

2.8 IP Addressing

ทุกอินเตอร์เฟซที่ต่ออยู่บนอินเตอร์เน็ตจะต้องมีหมายเลขประจำตัวเพื่อใช้ในการสื่อสารข้อมูล เรียกว่า *Internet* Address หรือเรียกย่อๆว่า IP Address โดยค่า IP Address นี้จะเป็นหมายเลขจำนวน 32 บิต แต่แทนที่จะกำหนดให้เลขทั้ง 32 บิตนั้นถูกนับต่อเนื่องกันไป ก็จะใช้วิธีการแบ่งหมายเลข ดังกล่าวออกเป็นกลุ่มของเลขขนาด 8 บิตจำนวน 4 ชุด และคั่นแต่ละชุดด้วยจุด ตัวอย่าง เช่น 172.17.3.12 นอกจากนี้ใน IP Address นั้นยังถูกแบ่งออกเป็น 2 ส่วนคือ ส่วนที่ เป็นแอดเดรสของเน็ตเวิร์ก (Network ID) และส่วนที่เป็นแอดเดรสของโฮสต์ (Host ID) ซึ่งข้อมูลในส่วนนี้จะถูกใช้สำหรับ ค้นหาเส้นทางของ IP ในการที่จะขนส่งข้อมูลจากต้นทางให้ถึงปลายทางอย่างถูกต้อง เพื่อเป็นการกำหนดขนาดของเน็ตเวิร์ก สำหรับ IP Address ต่างๆดังนั้นจึงมีการจัด IP Address ใน

แต่ละช่วงออกเป็นคลาส (class) ต่างๆกันจาก A ถึง E เพื่อจะได้ทำการจัดสรร IP Address ได้อย่างเหมาะสมกับขนาดของเน็ตเวิร์ก



รูปที่ 2.6 การกำหนด IP Address ในคลาสต่างๆ

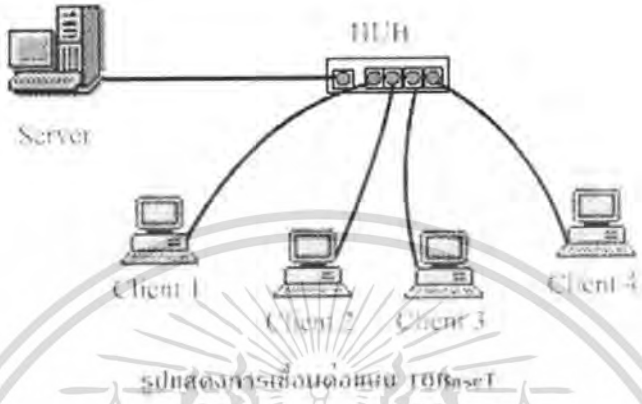
จากข้อกำหนดในการแบ่งคลาสของ IP Address หากลองนำบิตที่อยู่ในตอนต้นของ ip address ในแต่ละคลาสมาแปลงเป็น ip address ในเลขฐานสิบ จะเห็นว่าแต่ละคลาสครอบคลุม ip address ช่วง ต่างๆ ดังตารางที่ 2.1

Class	IP Range		
A	0.0.0.0	-	127.255.255.255
B	128.0.0.0	-	191.255.255.255
C	192.0.0.0	-	223.255.255.255
D	224.0.0.0	-	239.255.255.255
E	240.0.0.0	-	255.255.255.255

ตารางที่ 2.1 แสดงช่วงของ IP Adress ในแต่ละคลาส

2.9 อุปกรณ์เครือข่าย

2.9.1 เซอร์เวอร์ (Server)



รูปที่ 2.7 เซอร์เวอร์

เซอร์เวอร์ หรือเรียกอีกอย่างหนึ่งว่า เครื่องแม่ข่าย เป็นเครื่องคอมพิวเตอร์หลักในเครือขาย ที่ ทำหน้าที่จัดเก็บและให้บริการไฟล์ข้อมูลและทรัพยากรอื่นๆ กับคอมพิวเตอร์เครื่องอื่น ๆ ใน เครือข่าย โดยปกติคอมพิวเตอร์ที่นำมาใช้เป็นเซอร์เวอรมักจะเป็นเครื่องที่มีสมรรถนะสูง และ มีฮาร์ดดิสส์ความจำสูงกว่าคอมพิวเตอร์เครื่องอื่น ๆ ในเครือข่าย

ไคลเอนต์ (Client) หรือเรียกอีกอย่างหนึ่งว่า เครื่องลูกข่าย เป็นคอมพิวเตอร์ในเครือข่ายที่ ร้อง ขอ บริการและเข้าถึงไฟล์ข้อมูลที่จัดเก็บในเซอร์เวอร์ หรือพูดง่าย ๆ ก็คือ ไคลเอนต์ เป็น คอมพิวเตอร์ ของผู้ใช้แต่ละคนในระบบเครือข่ายนั่นเอง

2.9.2 ฮับ (HUB)

รูปที่ 2.8 ฮับ

ฮับ (HUB) หรือเรียก รีพีทเตอร์ (Repeater) คืออุปกรณ์ที่ใช้เชื่อมต่อกลุ่มคอมพิวเตอร์ฮับมี หน้าที่รับส่งเฟรมข้อมูลทุกเฟรมที่ได้รับจากพอร์ตใดพอร์ตหนึ่ง ไปยังพอร์ตที่เหลือ คอมพิวเตอร์ที่

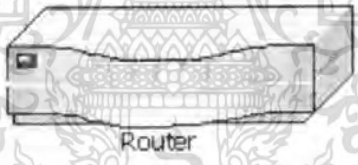
เชื่อมต่อเข้ากับฮับจะแชร์แบนด์วิธหรืออัตราข้อมูลของเครือข่าย เพราะฉะนั้นถ้ามีคอมพิวเตอร์เชื่อมต่อมากจะทำให้อัตราการส่งข้อมูลลดลง

2.9.3 สวิตซ์ (Switch)

รูปที่ 2.9 สวิตซ์

คืออุปกรณ์เครือข่ายที่ทำหน้าที่ในเลเยอร์ที่ 2 และทำหน้าที่ส่งข้อมูลที่ได้รับมาจากพอร์ตหนึ่งไปยังพอร์ตเฉพาะที่เป็นปลายทางเท่านั้น และทำให้คอมพิวเตอร์ที่เชื่อมต่อกับพอร์ตที่เหลือส่งข้อมูลถึงกันในเวลาเดียวกัน ดังนั้น อัตราการรับส่งข้อมูลหรือแบนด์วิธจึงไม่ขึ้นอยู่กับคอมพิวเตอร์ ปัจจุบันนิยมเชื่อมต่อแบบนี้มากกว่าฮับเพราะลดปัญหาการชนกันของข้อมูล

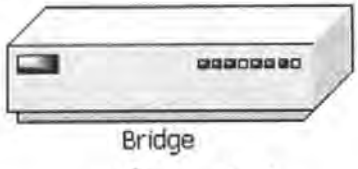
2.9.4 เราเตอร์ (Router)



รูปที่ 2.10 เราเตอร์

เป็นอุปกรณ์ที่ทำหน้าที่ในเลเยอร์ที่ 3 เราท์เตอร์จะอ่านที่อยู่ (Address) ของสถานีปลายทางที่ ส่วนหัว (Header) ข้อแพ็กเก็ตข้อมูล เพื่อที่จะกำหนดและส่งแพ็กเก็ตต่อไป เราท์เตอร์จะมีตัวจัดเส้นทางในแพ็กเก็ต เรียกว่า เราติ้งเทเบิล (Routing Table) หรือตารางจัดเส้นทางนอกจากนี้ยังส่ง ข้อมูลไปยังเครือข่ายที่ให้โพรโตคอลต่างกันได้ เช่น IP (Internet Protocol) , IPX (Internet Package Exchange) และ AppleTalk นอกจากนี้ยังเชื่อมต่อกับเครือข่ายอื่นได้ เช่น เครือข่ายอินเตอร์เน็ต

2.9.5 บริดจ์ Bridge



รูปที่ 2.11 บริดจ์

บริดจ์ เป็นอุปกรณ์ที่มักจะใช้ในการเชื่อมต่อวงแลน (Lan Segments) เข้าด้วยกัน ทำให้สามารถขยายขอบเขตของ LAN ออกไปได้เรื่อยๆ โดยที่ประสิทธิภาพรวมของระบบ ไม่ลดลงมากนัก เนื่องจากการติดต่อของเครื่องที่อยู่ในเซกเมนต์เดียวกันจะไม่ถูกส่งผ่าน ไปรบกวนการจราจรของเซกเมนต์อื่น และเนื่องจากบริดจ์เป็นอุปกรณ์ที่ทำงานอยู่ในระดับ Data Link Layer จึงทำให้สามารถใช้ในการเชื่อมต่อเครือขายที่แตกต่างกันในระดับ Physical และ Data Link ได้ เช่น ระหว่าง Eternet กับ Token Ring เป็นต้น.

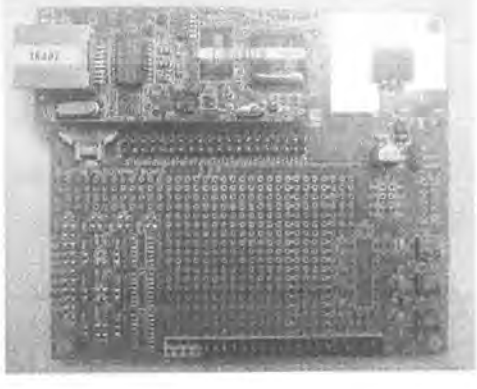
บริดจ์ มักจะถูกใช้ในการเชื่อมเครือข่ายย่อย ๆ ในองค์กรเข้าด้วยกันเป็นเครือข่ายใหญ่ เพียงเครือข่ายเดียว เพื่อให้เครือขายย่อยๆ เหลานั้นสามารถติดต่อกับเครือขายย่อยอื่นๆ ได้

2.9.6 เกตเวย์ (Gateway)

รูปที่ 2.12 เกตเวย์

เกตเวย์ เป็นอุปกรณ์ฮารดแวร์ที่เชื่อมต่อเครือข่ายต่างประเภทเข้าด้วยกัน เช่น การใช้เกตเวย์ใน การเชื่อมต่อเครือข่าย ที่เป็นคอมพิวเตอร์ประเภทพีซี (PC) เข้ากับคอมพิวเตอร์ประเภทแมคอินทอช (MAC) เป็นต้น

2.10 คุณสมบัติของไมโครคอนโทรลเลอร์ Rabbit3000 Core Module RCM 3720

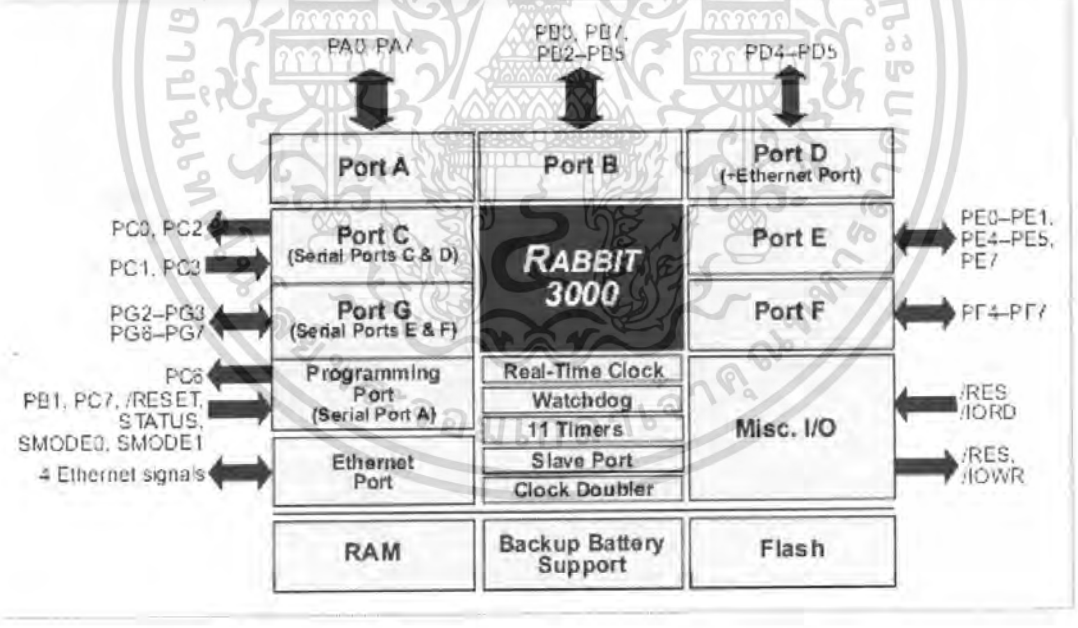


รูปที่ 2.13 บอร์ดวงจร RCM 3720

- ขนาดเล็ก 1.20" x 2.95" x 0.89" หรือ (30 mm x 75 mm x 23 mm)
- ไมโครโปรเซสเซอร์ Rabbit 3000 ใช้ความถี่ในการทำงานที่ 22.1 MHz
- มีขาที่เป็น Input/Output ทั้งหมด 33 ขา เป็นแบบ Parallel มีโหมดการทำงาน หลากหลาย
- มี Data Bus 8 เส้น
- มี Address Bus 5 เส้น
- มี Flash Memory ขนาด 512 K และ Static Ram ขนาด 512K
- มีสัญญาณนาฬิกาให้จังหวะการทำงาน
- ใช้หัวต่อสัญญาณแบบ RJ - 45 เป็น Ethernet Port
- Port Serial เป็นชนิด CMOS ซึ่งเป็นอัตราบอร์ดแบบ Asynchronous ได้ความถี่ สูงสุดถึง 2.76 Mbps

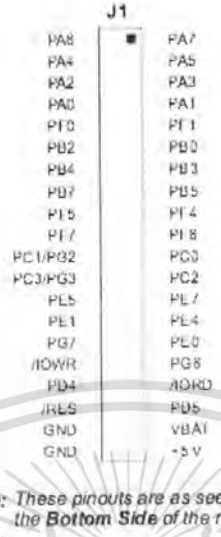
2.11 ส่วนประกอบและอุปกรณ์ย่อยของ ไมโครคอนโทรลเลอร์ Rabbit3000

2.11.1 อินพุต/เอาต์พุตพอร์ต



รูปที่ 2.14 โครงสร้างภายในของ ไมโครคอนโทรลเลอร์Rabbit3000

Rabbit3000 มี Input / Output ที่แบ่งเป็น Port 7 Port แต่ละ Port มีขนาด 8 บิต อยู่ที่ หัวข้อต่อสัญญาณ J1 โดยที่เส้น Input / Output จะอยู่ที่ PA0 - PA7, PB0 - PB7, PC0 -PC5, PD4, PD5, PE1, PE3 -PE7, PF0 -PF7 และ PG0 -PG7 ถูกแสดงในรูปที่ 2.15



*Note: These pinouts are as seen on the **Bottom Side** of the module.*

รูปที่ 2.15 จุดต่อสัญญาณใช้งานภายนอกของ Rabbit3000 Core RCM3720

PIN		Pin Name	Default Use	Alternate Use	Notes
J1	1 - 8	PA[0:7]	Parallel I/O	External data bus & Slave port data bus	External data bus
	9	PF1	Input / Output	QD1A , CLKC	
	10	PF0	Input / Output	QD1B , CLKD	
	11	PB0	Input / Output	CLKB	
	12	PB2	Input / Output	IA0 / SWR	External Address 0 Slave port write
	13	PB3	Input / Output	IA1 / SRD	External Address 1 Slave port read
	14	PB4	Input / Output	IA2 / SA0	Address 2 Slave port

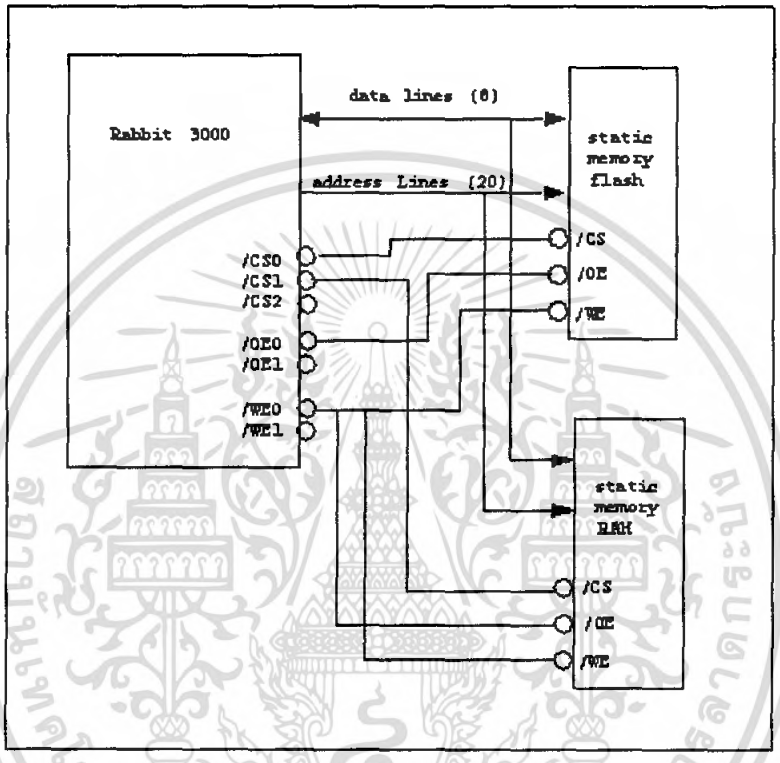
	15	PB5	Input / Output	IA3 / SA1	External Address 3 Slave port Address 1
	16	PB7	Input / Output	IA5 / SLAVEATIN	External Address 5 Slave port Attention
J1	17	PF4	Input / Output	AQD1B PWM0	
	18	PF5	Input / Output	AQD1A PWM1	
	19	PF6	Input / Output	AQD2B PWM2	
	20	PF7	Input / Output	AQD1A PWM3	
	21	PC0	Output	TXD	Serial Port D
	22	PC1/PG2	Input / Output	RXD/TXF	Serial Port D Serial Port F
	23	PC2	Output	TXC	Serial Port C
	24	PC3/PG3	Input / Output	TXC/RXF	Serial Port C Serial Port F
	25	PE7	Input / Output	I7/SCS	External Address7 Slave Port Chip Select
	26	PE5	Input / Output	I5 INT1B	
	27	PE4	Input / Output	I4 INT0B	

	28	PE1	Input / Output	I1 INT1A	I/O Strobe 1 Interrupt1 A
	29	PE0	Input / Output	I0 INT0A	I/O Strobe 0 Interrupt0 A
	30	PG7	Input / Output	RXE	Serial Port E
	31	PG6	Input / Output	TXE	
	32	/IOWR	Output		External write strobe
	33	/IORD	Input		External read strobe
	34	PD4	Input / Output	ATXB	Alternate Serial Port B
	35	PD5	Input / Output	ARXB	
	36	/RES	Reset output	Reset Input	
	37	VBAT			
	38	GND			
	39	+5V			
	40	GND			

ตารางที่ 2.2 ความสามารถในการนำไปใช้งานของหัวต่อสัญญาณ J1

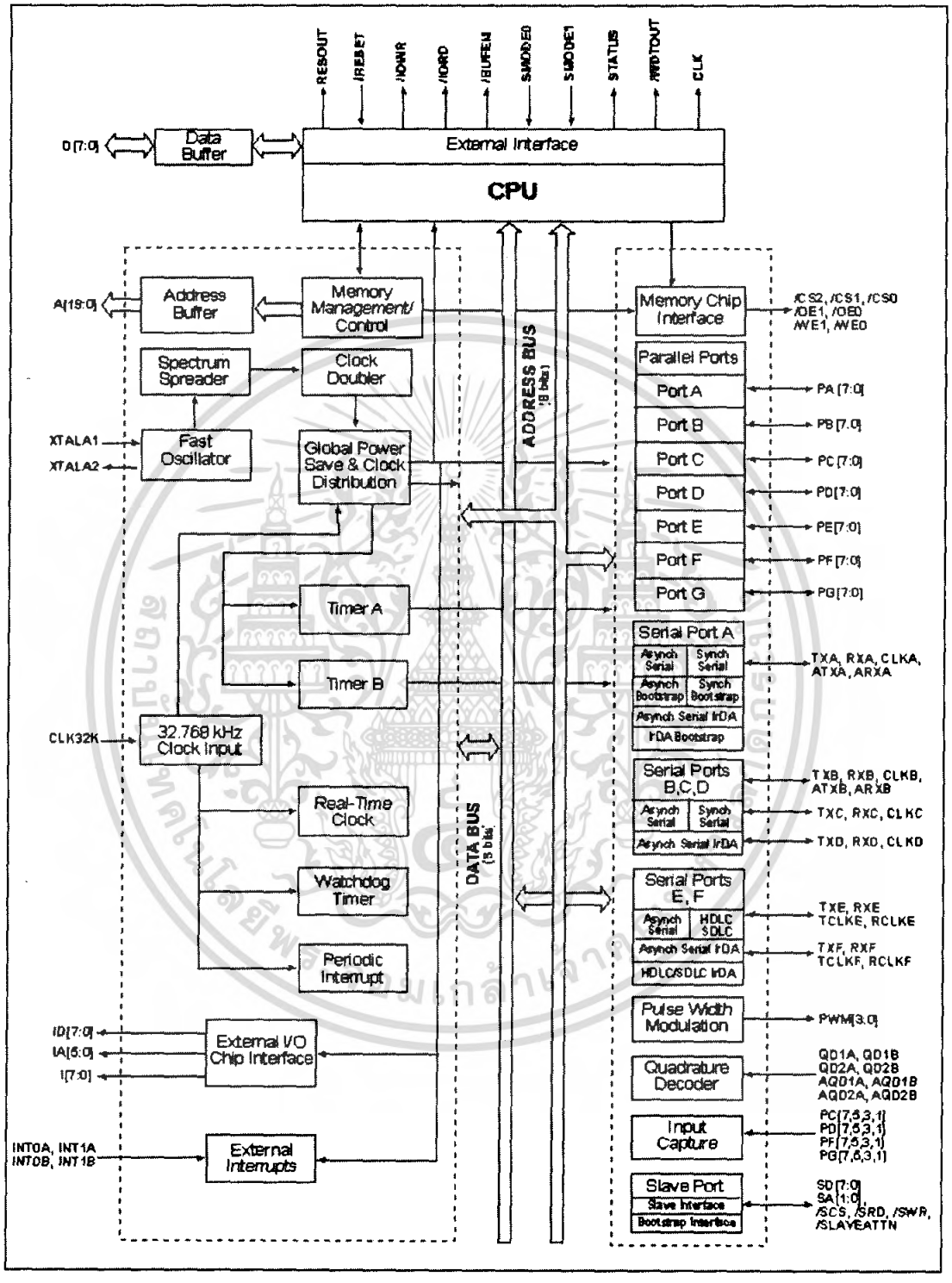
2.11.2 อินพุต / เอาต์พุตที่ติดต่อหน่วยความจำ

จะประกอบด้วยบัสตำแหน่ง (Address bus) จาก A0 - A19 และบัสข้อมูล (data bus) จาก D0 - D7 สำหรับการติดต่อกับอุปกรณ์ภายนอก จะใช้ตำแหน่งขาสัญญาณ (IOWR) และ (IORD) เป็น อินพุท/เอาท์พุท ในการอ่านและเขียนข้อมูล เพื่อใช้ติดต่อข้อมูล



รูปที่ 2.16 การติดต่อกับหน่วยความจำ (RAM)

2.12 รายละเอียดของไมโครโปรเซสเซอร์ Rabbit 3000



รูปที่ 2.17 โครงสร้างภายในของ Rabbit3000

ไมโครโปรเซสเซอร์ Rabbit3000 เป็นไมโครโปรเซสเซอร์ที่ผลิตโดยบริษัท Rabbit Semiconductor Corporation เป็นไมโครโปรเซสเซอร์ขนาด 8 บิตมีความเร็วของสัญญาณนาฬิกา

22.1 MHz ถูกออกแบบมาเพื่อให้ใช้กับงานที่เกี่ยวข้องกับระบบ Ethernet โดยเฉพาะและ ไมโครโปรเซสเซอร์ Rabbit 3000 ก็เป็นเหมือนหัวใจหลักของ Rabbit Core Module RCM3720 ซึ่ง คอยควบคุมการทำงานต่างๆไม่ว่าจะเป็นอุปกรณ์ภายในและภายนอกของ Rabbit Core Module RCM 3720

2.13 หน้าที่ของขาอินพุท/เอาท์พุทบนไมโครโปรเซสเซอร์ Rabbit3000

- CLK เป็นขาชนิดเอาท์พุทที่ป้อนสัญญาณนาฬิกาที่ใช้ในการประมวลผล
- RESET เป็นขาชนิดอินพุทที่ใช้ในการกำหนดค่าเริ่มต้นหรือการทำงานใหม่
- XTALA1, XTALA2 เป็นขาชนิดอินพุท/เอาท์พุทที่ใช้สำหรับการเชื่อมต่อกับ Oscillator quartz crystal 11.0592 MHz
- CLK32K เป็นขาชนิดอินพุทใช้สำหรับการเชื่อมต่อกับ Oscillator Quartz Crystal 32.768 KHz
- A0-A19 เป็นขาชนิดเอาท์พุททำหน้าที่กำหนดการเลือกใช้และควบคุมหน่วยความจำภายนอก
- D0-D7 เป็นขาชนิดอินพุท/เอาท์พุทใช้เป็นทางผ่านของข้อมูลระหว่างพอร์ตต่างๆ ของไมโครโปรเซสเซอร์
- WDTOUT เป็นขาชนิดเอาท์พุทใช้ตรวจเช็คว่าสัญญาณลูกคลื่นมีค่าต่ำากว่า 30 us หรือไม่ภายในระยะเวลาที่กำหนด
- STATUS เป็นขาชนิดเอาท์พุทใช้ตรวจดูสภาวะการทำงานของ Rabbit 3000
- SMODE0, SMODE1 เป็นขาชนิดอินพุทใช้กำหนดเริ่มการทำงาน
- CS0, OE0, WEO เป็นขาชนิดเอาท์พุทตามปกติใช้เป็นการ Enable เพื่อให้ทำการเขียนข้อมูลลง Flash Memory ใน Program Code
- CS1, OE1, WE1 เป็นขาชนิดเอาท์พุทตามปกติใช้เป็นการ Enable เพื่อให้ทำการเขียนข้อมูลลง Static RAM และทำให้ความเร็วและการสูญเสียพลังงานเหมาะสมกัน
- CS2 เป็นขาชนิดเอาท์พุทจุดประสงค์ทั่วไปใช้ในการ Memory Decode
- BUFFEN เป็นขาชนิดเอาท์พุทปกติตั้งไว้เป็น Enable ทำงานกับ I/O ภายนอกจะทำงานเมื่อไมโครโปรเซสเซอร์ Rabbit 3000 ขอมา
- IORD, IOWR เป็นขาชนิดเอาท์พุทใช้ให้ระบุการอ่านและเขียนข้อมูลกับส่วน I/O
- PA0-PA7 เป็นขาชนิดอินพุท/เอาท์พุทของพอร์ต A (Slave Port) ทำหน้าที่ Reset พอร์ต
- PB0-PB7 เป็นขาชนิดอินพุท/เอาท์พุทของพอร์ต B (Slave Port) ทำหน้าที่ควบคม Slave Port, Serial Port Clocks

- PC0-PC7 เป็นขาชนิดอินพุท/เอาท์พุทของพอร์ต C (Slave Port) โดยทั่วไปทำหน้าที่เป็น Serial Data ของ Slave Port
- PD0-PD7 เป็นขาชนิดอินพุท/เอาท์พทของพอร์ต D (Slave Port)
- PE0-PE7 เป็นขาชนิดอินพุท/เอาท์พุทของพอร์ต E (Slave Port) โดยทั่วไปทำหน้าที่เป็น Interrupts จากภายนอก
- PG0-PG7 เป็นขาชนิดอินพุท/เอาท์พุทของพอร์ต E (Slave Port)
- VBAT ใช้จ่ายพลังงานชนิด Real-Time Clock (25uA at 2.3V)
- VDD ใช้จ่ายพลังงานให้กับไมโครโปรเซสเซอร์ Rabbit3000 ที่แรงดัน 2.7-5 volts
- VSS คือ Ground

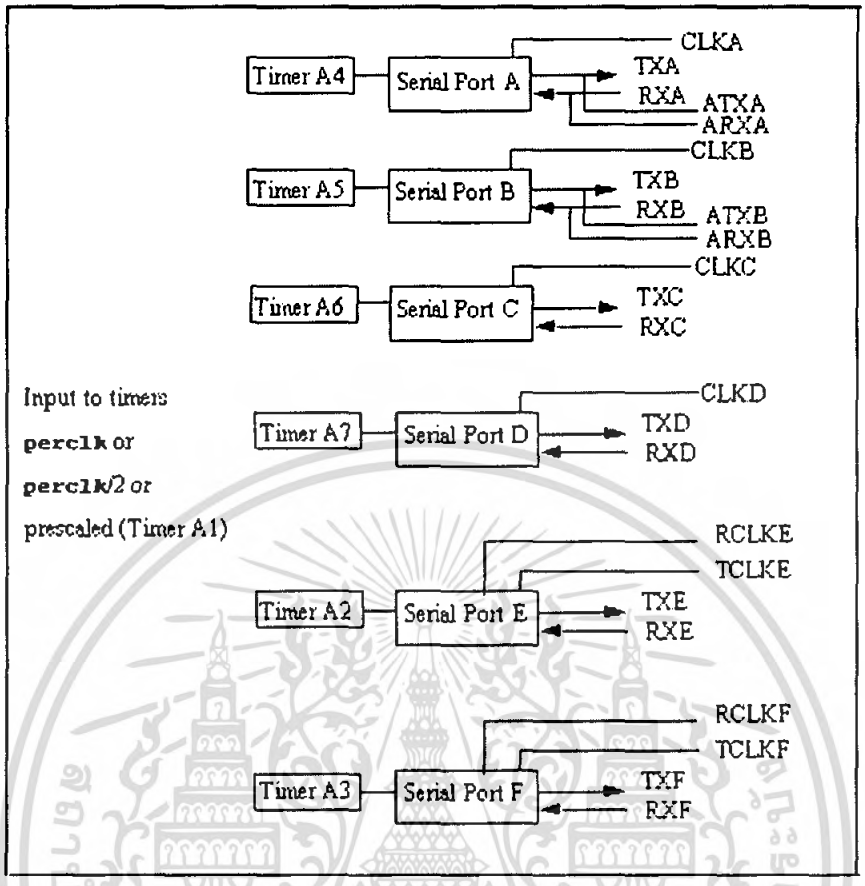
2.14 พอร์ตในการเชื่อมต่อ RabbitCore RCM3720

พอร์ตในการเชื่อมต่อ RabbitCore RCM3720 ที่บอร์ดมีสาย RS - 232 หรือสาย RS - 485 สามารถติดต่อผานบอร์ดได้โดยตรง

2.14.1 พอร์ตอนุกรม (serial ports)

พอร์ตอนุกรม (serial ports) มีอยู่ 6 พอร์ตคือ พอร์ต A , B ,C ,D ,E และ F พอร์ต อนุกรมทั้งหมด 6 พอร์ต สามารถทำงานอะซิงโคนัส เพื่อให้อัตราการส่วนข้อมูลของระบบสัญญาณนาฬิกาแบ่ง จาก 16 พอร์ตอะซิงโคนัสสามารถจัดการที่ 7 -8 บิตข้อมูล บิตที่ 9 เป็นบิตแอดเดรสซึ่งถูกส่งเป็นไบต์แรกของข้อความ(message) พอรตอนุกรม A ,B ,C และ D จะทำงานในโหมด สัญญาณนาฬิกาในแบบอนุกรม ในโหมดนี้จะรับสัญญาณนาฬิกาเข้ามาเพื่อซิงโคนัสข้อมูลเข้าและ ออกตามจังหวะของสัญญาณนาฬิกาในการสื่อสารของอุปกรณ์ 2 ตัว ที่สนับสนุนการใช้สัญญาณ นาฬิกา เมื่อ Rabbit 3000 ใช้สัญญาณนาฬิกาอัตราการส่งข้อมูล จะสูงถึง 80% ของระบบสัญญาณ นาฬิกาที่ถูกแบ่งจาก 128 ถึง 183, 750 bps ที่ความเร็วของสัญญาณนาฬิกาที่ 22.1 MHz

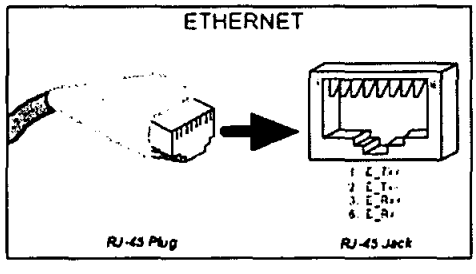
พอร์ทอนุกรม E และ F สามารถตั้งค่าให้เป็น SDLC/HDLC ซึ่งเป็นรูปแบบของ SDLC จะสนับสนุนโปรโตคอล Irda ทั้ง 2 พอร์ต



รูปที่ 2.18 บล็อกไดอะแกรม ของพอร์ตอนุกรม

2.14.2 อีเทอร์เนต พอรต (Ethernet port)

อีเทอร์เนต พอร์ต (Ethernet port) ที่ใช้บนบอร์ด Rabbit RCM3720 จะใช้หัวต่อแบบ RJ-45 ซึ่งจะมี LED อยู่สองดวงมีหน้าที่บอกสถานะการทำงานของบอร์ด Rabbit RCM3720 โดย ขาสัญญาณ (LNK) มีหน้าที่บอกสถานะการเชื่อมต่อ ส่วนขาสัญญาณ (ACT) จะบอกสถานะการ ถ่ายโอนข้อมูลโดย อีเทอร์เนต พอร์ต (Ethernet port) จะอยู่ที่คอนเน็คเตอร์ (connecter) J2



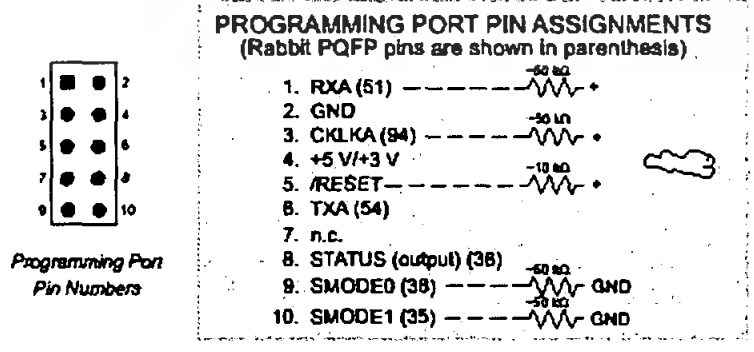
รูปที่ 2.19 พอร์ท Ethernet RJ-45

2.14.3 มาตรฐานพอร์ตอนุกรมแบบ RS-232

มาตรฐานการสื่อสารของพอร์ตอนุกรม ได้ระบบช่วงระดับแรงดันการทำงานของพอร์ตอนุกรมไว้ว่า ที่ลอจิก "0" จะมีระดับสัญญาณ +3 ถึง +15 v ส่วนลอจิก "1" จะมีระดับสัญญาณ -3V ถึง -15 v ระดับสัญญาณนี้ทำให้ไม่สามารถที่จะนำขาเอาต์พุตใด ๆ ต่อเข้ากับลอจิกเกตเพื่อใช้งานได้โดยตรง จะต้องผ่านวงจรเพื่อเปลี่ยนระดับแรงดันเสียก่อน เพื่อให้มีการใช้งานเส้นสัญญาณหรือรูปแบบตัวเชื่อมต่อที่สอดคล้องกัน จะได้ช่วยลดปัญหาการเข้ากันไม่ได้ระหว่างสัญญาณของอุปกรณ์ที่มาเชื่อมต่อกันทั้งสองด้านให้น้อยลง โดยระดับสัญญาณแบบ TTL จากขาสัญญาณ TxD และ RxD จะต้องถูกปรับเปลี่ยนไปเป็นระดบสัญญาณ RS-232 ก่อนที่จะทำการส่งออก ไปในสายนำสัญญาณต่อไป ในที่นี้ใช้ไอซีวงจรรวมซึ่งประกอบด้วยวงจรรับและส่งแบบ RS-232 อยู่ภายในตัวและใช้ไฟเลี้ยง +5 v เท่านั้น โดยภายในยังมีวงจรเปลี่ยนระดับโวลท์เตจ (dc-to converter) ประกอบอยู่ด้วย ซึ่งทำให้การสร้างวงจรับ/ส่งข้อมูลตามมาตรฐาน RS-232 กระทำได้ง่ายและสะดวกมากยิ่งขึ้น ไอซีวงจรรวมดังกล่าว คือ MAX232 มันจะทำหน้าที่แปลงแรงดันของ RS232 ให้อยู่ในระดับทีทีแอล โดยลอจิก "0" ซึ่งเดิมมีระดับสัญญาณ + 3 ถึง +5v จะถูกแปลงเป็น 0v ส่วนลอจิก "1" ซึ่งมีระดับสัญญาณ -3 ถึง -5v จะแปลง เป็น +5v ทั้งนี้เพื่อให้สามารถติดต่ออุปกรณ์ ดิจิตอลอื่นที่ใช้ระดับแรงดันทีทีแอลได้

2.14.4 พอร์ตการเขียนโปรแกรม (Programming Port)

พอร์ตการเขียนโปรแกรม สื่อสารกับ Rabbit RCM3720 โดยทาง Chip serial port พอรต a มี ค่วามสามารถพิเศษที่ใช้สำหรับการพัฒนาซอฟต์แวรภายใต้โปรแกรมไดนามิค C ขาสัญญาณ (SMODE0, SMODE1) แสดงการเขียนโปรแกรม เชื่อมต่อกับอุปกรณ์ภายนอกตำแหน่งที่ใช้ในการเขียนโปรแกรมสำหรับ Rabbit Core RCM3720 ในโหมดที่จะดาวน์โหลดโปรแกรมจากคอมพิวเตอร์ให้แก่ RCM3720 จนจบโปรแกรม พร้อมทั้งทำการตรวจสอบตรวจสอบจุดบกพร่องในโปรแกรม



รูปที่ 2.20 พอร์ตการเขียนโปรแกรม

2.15 Parallel Ports

Rabbit3000 มีพอร์ตขนานทั้งหมด 7 พอร์ต โดยแต่ละพอร์ตจะมีพอร์ตละ 8 บิตซึ่งพอร์ตขนานจะแบ่งออกเป็น A ,B ,C ,D ,E ,F และ G โดยแต่ละพอร์ตมีขนาด 8 บิต เช่น A ,B ,C ,D,E ,F และ Gโดยแต่ละขาจะใช้สำหรับพอร์ตขนานซึ่งใช้กับฟังก์ชั่นต่าง โดยมีคุณสมบติของแต่ละพอร์ตดังนี้

พอร์ต A - จะทำการแชร์กับส่วนที่ติดต่อข้อมูลของพอร์ต Slaveและบัสข้อมูลของI/O

พอร์ต B - จะทำการแชร์กับเส้นทางของหน่วยควบคุมของพอร์ต Slave , จะเป็นส่วนประกอบของI/O ,และเป็น clock I/O สำหรับclockของพอร์ตอนุกรม A และ B

พอร์ต C - จะทำการแชร์กับพอร์ตที่เป็นชุดข้อมูลของชุดอินพุท/เอาท์พุท

พอร์ต D - จะมีบิตอยู่ 4 บิตที่จะทำการแชร์และสลับกันของขาสัญญาณอินพุท/เอาท์พุท สำหรับพอร์ตอนุกรมของ A และ B ส่วนอีก 4 บิตจะไม่มีการแชร์ โดยพอร์ต D มีความสามารถที่จะแก้ไข ให้มีการออกของข้อมูลพอร์ต D ยังมี Output Preload registers ที่สามารถสร้าง clock ไปยังเอาท์พุท รีจิสเตอร์ อยู่ใน timer Control สร้าง pulse ได้

พอร์ต E- พอร์ต E ทุกบิตของพอร์ต E สามารถปรับแก้เป็น I/O Strobe 4 บิต ของพอร์ต E สามารถใช้กับอินเตอร์รัพ ข้อมูลภายนอก 1 บิตของพอร์ต E สามารถแชร์กับชิพ พอร์ตลูกที่เลือกเอาไว้ พอรต E มี output preload register อยู่ภายใต้การควบคุมของ timer control สำหรับสร้างพัลส์

พอร์ต F - พอรต F เป็นเอาท์พุท สามารถรับค่าให้เปิดทางของเอาท์พุท พอร์ตขนาน F เอาท์พุทสามารถนำ 4 Pulse- Width Modulation เอาท์พุทในด้านอินพุท พอร์ตขนาน F อินพุท พอร์ตขนาน F อินพุทสามารถนำอินพุทไปยัง 2 ช่อง ของตัวถอดรหัสควอแรนด์ พอร์ต F สามารถใช้ขาสัญญาณนาฬิกาสำหรับพอร์ตอนุกรม C และ D

พอร์ต G -พอรต G เป็นพอร์ตที่มีแต่เอาท์พุต พอร์ต G สามารถแก้ไขให้เอาท์พุทออกได้ พอร์ตอินพุทและเอาท์พุทและเอาท์พุทของ G ที่ถูกใช้สำหรับการเข้าถึงจุดอื่นแบบอนุกรมรอบข้างบนชิพแต่ละอย่างใช้อะซิงโคนัสหรือ การติดต่อแบบ SDLC/HDLC

พอร์ต D - G โดยมีวิธีการเข้าถึงแบบเดียวกันโดยใช้ดิจิตอล I/O

2.15.1 Parallel Port A

Parallel Port A จะมีหน้าที่การอ่าน / เขียนรีจิสเตอร์ได้อย่างเดียว โดย จะมีข้อมูล PADR (ADR = 030h) รีจิสเตอร์นี้ไม่ควรจะใช้ถ้าพอร์ต Slave มีการใช้งานรีจิสเตอร์ของหน่วยควบคุม พอร์ต Slave ถูกใช้ให้หน่วยควบคุมพอร์ตขนาน A เป็นไปได้ทั้งอินพทและเอาท์พุท เพื่อทำให้พอร์ต A เป็นอินพุทต้องให้เก็บค่า 080h ใน SPCR (slave port control register) ส่วนการทำให้พอร์ต A เป็น เอาท์พุทต้องให้เก็บค่า 084h ใน SPCR พอร์ต A จะทำการเซ็ทค่าอินพุทโดยการ รีเซท

2.15.2 Parallel Port B

พอร์ตขนาน B โดยมี 8 ขาสามารถเขียนโปรแกรมได้และสามารถทำเป็นขาอินพุทและเอาท์พุทได้พร้อมกัน หลังจากการเซ็ทอัพแล้ว พอร์ตขนาน B จะมีขาที่ 0-5 จะเป็นอินพุท แล้วขาที่ PB6 และ PB7 จะเป็นเอาท์พุท ซึ่งจะมีค่าเป็น 0

ในขณะที่ส่วนประกอบของ DATA BUS มีการทำงานอยู่นี้ พอร์ตที่ 2-7 จัดการ 6 แถวข้อมูล โดยที่มีผลคือ 6 แถวจาก 16 แถว หมายความว่า เต็มไปด้วยที่ว่างของ I/O

ในขณะที่พอร์ตลูกทำงาน โดยที่พอร์ตขนาน PB2-PB7 ได้กำหนดค่าของฟังก์ชั่นต่างๆ ของพอร์ตลูก แต่อย่างไรก็ตามเป็นสิ่งที่เป็นไดเมนการอ่าน PB-5 ใช้พอร์ตB DATA Registers ในขณะที่ PB2-PB7 จะถูกใช้สำหรับพอร์ตลูก พอร์ตที่ 6-7 สามารถอ่านสัญญาณที่เข้ามา

PB0 - จะทำหน้าที่เป็น Clock ของพอร์ตอนุกรมแบบ A, PB1 จะทำหน้าที่เป็น Clock ของพอร์ตอนุกรมแบบ B

PBDR - Parallel Port B Register Read/Write

PBDDR - Parallel Port B Data Direction Register. A"1" จะทำให้ขานั้นเป็นเอาท์พุท ซึ่งจะเป็นรีจิสเตอร์ที่ไว้สำหรับเขียนเท่านั้น

2.15.3 Parallel Port C

Parallel Port C จะมีอินพุทอยู่ 4 บิตและเอาท์ทอยู่ 4 บิต โดยจะแบ่งออกเป็นบิตที่เป็น เลขคู่ จะประกอบไปด้วย PC0, PC2, PC4 และ PC6 จะเป็นเอาท์พุท และบิตที่เป็นเลขคี่จะประกอบ ไปด้วย PC1, PC3, PC5 และ PC7 จะเป็นอินพุท เมื่อรีจิสเตอร์ของข้อมูลทำการอ่าน บิตที่ 1 ,3 ,5 ,7 จะทำการคืนค่าของแรงดันไฟฟ้าบนขาของสัญญาณ ส่วนบิตที่ 0 ,2 ,4 ,6 จะทำการคืนค่าโดยการ ส่งสัญญาณบัพเฟอร์ไปที่เอาท์พุท โดยในการส่งสัญญาณบัพเฟอร์ผลลัพธ์ที่ได้และค่าของขาของสัญญาณจะเหมือนกัน

Parallel Port C จะมีการแชร์กันของขาสัญญาณของพอร์ตอนุกรมทั้งหมด 4 บิต โดยขาสัญญาณของพอร์ตขนาน C สามารถเป็นอินพุทและยังเป็นขาสัญญาณของพอร์ตอนุกรมที่สามารถเป็นอินพุทได้ด้วย (พอร์ตอนุกรม A และ B สามารถใช้สลับบิต 7 และ5 ตามลำดับในพอร์ต

C ที่เป็นอินพุทและ Source ของอินพุทที่พอร์ตอนุกรมจะขึ้นอยู่กับหน่วยควบคุมพอร์ต โดยตรงของรีจิสเตอร์ควบคุมของพอร์ตอนุกรม) เส้นทางของข้อมูลสามารถอ่านได้จากพอร์ตขนาน C โดยรีจิสเตอร์ของข้อมูลเอาท์พุทของพอร์ตขนานสามารถถูกนำไปเลือกใช้พอร์ตอนุกรม โดยจะ ทำการเก็บในตำแหน่งที่ตรงกันของพอร์ต C ในรีจิสเตอร์ฟังก์ชัน (PCFR) เมื่อขาสัญญาณของ พอร์ตขนานถูกเลือกใช้โดยผลลัพธ์ของพอร์ตอนุกรมค่าที่เก็บในรีจิสเตอร์ของข้อมูลจะไม่ถูกสนใจ

2.15.4 Parallel Port *D*

พอร์ตขนาน D มีขาสัญญาณทั้งหมด 8 บิตซึ่งสามารถที่จะโปรแกรมเฉพาะไม่ว่าจะเป็นอินพุทหรือเอาท์พุท เมื่อโปรแกรมเป็นเอาท์พุทขาสัญญาณก็สามารถที่จะเลือกเอาท์พุทหรืออินพุทที่เป็นมาตรฐาน ขาสัญญาณของพอร์ต D สามารถเป็นบิตที่เป็นตำแหน่ง Address รีจิสเตอร์ของเอาท์พุทจะมีรูปแบบและมี Timers ควบคุมตามลำดับขั้นตอน การทำงานจะมีการ สร้างสัญญาณพัลส์ในการจับเวลา นอกจากนี้เอาท์พุทของพอร์ต D มีความสามารถในการส่ง สัญญาณที่สูง พอร์ต D จะมีบิต 4 และ 5 ทสามารถที่จะทำการสลับขาสัญญาณกันของพอร์ตอนุกรม B และบิต 6 และ 7 ยังสามารถถูกใช้ในการสลับขาสัญญาณกันของพอร์ตอนุกรม A การสลับกัน ของบิตพอร์ตอนุกรมจะกระทำในพอร์ตอนุกรมโดยจะมีการเชื่อมต่อกับขาสัญญาณในการติดต่อที่ แตกต่างกันในเวลาเดียวกัน ในกรณีการรีเซ็ท ข้อมูลในรีจิสเตอร์โดยตรงจะมีค่าเท่ากับ “0” การ ทำงานของขาสัญญาณทั้งหมดจะเป็นอินพุท นอกจากนี้ในการส่งค่าให้กับรีจิสเตอร์หน่วยควบคุมมี ค่าเท่ากับ “0” (บิต 0,1,4,5) ก็เพื่อที่จะให้แน่ใจว่าข้อมูลได้ถูกใช้สัญญาณนาฬิกาเข้าไปในรีจิสเตอร์ เมื่อมีการบันทึกรีจิสเตอร์อื่นๆ ทั้งหมดที่เกี่ยวข้องกับพอร์ต D จะไม่ถูกทำงานตอนเริ่มต้นเมื่อเรา ทำการรีเซ็ท

PDDR	รีจิสเตอร์ข้อมูลของพอร์ตขนาน D จะทำการ อ่าน/เขียน
PDDDR	รีจิสเตอร์โดยตรงข้อมูลของพอร์ตขนาน D มีค่าเท่ากับ"1" ทำให้ขาสัญญาณเป็น เอาท์พุทใช้ในการเขียนเท่านั้น
PDDCR	รีจิสเตอร์ของหน่วยควบคุมของพอร์ตขนาน D มีค่าเท่ากับ"1" ทำให้ ขาสัญญาณเป็นการเปิดสัญญาณเอาท์พุทใช้ในการเขียนเท่านั้น
PDFR	รีจิสเตอร์หน่วยควบคุมฟังค์ชั่นของพอร์ตขนาน D พอร์ตนี้อาจจะถูกใช้เพื่อหา ตำแหน่งของพอร์ต 4 และ 6 ของเอาท์พุทอนุกรม ใช้ในการเขียนเท่านั้น
PDBXR	จะมีรีจิสเตอร์ทั้งหมด 8 ตัวโดยจะถูกใช้เพื่อเซ็ทตำแหน่งของพอร์ตที่จะใช้งาน

PDCR รีจิสเตอร์หน่วยควบคุมของพอร์ตขนาน D รีจิสเตอร์นี้ถูกใช้ในการควบคุม สัญญาณนาฬิกาของด้าน High และ Low ของเอาท์พุทตัวสุดท้ายของรีจิสเตอร์ของ พอร์ต ส่วนการรีเซ็ท บิต 0, 1, 4, และ 5 จะเท่ากับ 0

2.15.5 Parallel Port E

มีขาสัญญาณอินพุท/เอาท์พุททั้งหมด 8 บิตที่สามารถถูกโปรแกรมเฉพาะอย่าง ไม่ว่าจะเป็นอินพุทและเอาท์พุท พอร์ต E มีความสามารถในการส่งสัญญาณที่สูงกว่าพอร์ตอื่นๆ PE7 จะ ถูกใช้เมื่อพอร์ต Slave ถูกใช้งานเป็นเอาท์พุทของพอร์ต E สามารถที่จะแก้ไข I/O strobe นอกจากนี้ ยังมีขาสัญญาณจำนวน 4 บิตของพอร์ต E สามารถเกิดการอินเตอร์รัพท์เมื่อมีการร้องขอการ อินเตอร์รัพท์ที่อินพุท รีจิสเตอร์ของเอาท์พุทมีรูปแบบและ Timers อย่างมีแบบแผนการทำงาน เป็นไปตามการสร้างสัญญาณของเวลา

PEDR รีจิสเตอร์ข้อมูลของพอร์ต E มีหน้าที่อ่านค่าจากขาสัญญาณ

PEDDR รีจิสเตอร์โดยตรงข้อมูลของพอร์ต E หากเซ็ทเป็น "1" จะทำให้ขาสัญญาณ
เป็นเอาท์พุท รีจิสเตอร์นี้จะรีเซ็ทก็ต่อเมื่อมีค่าเท่ากับ 0

PEFR รีจิสเตอร์ฟังค์ชั่นของพอร์ต E หากเซ็ทเป็น "1" จะทำให้เอาท์พุทตรงกับ I/O strobe โดยปรกติ I/O strobe จะถูกควบคุมโดย รีจิสเตอร์ I/O ของหน่วยควบคุม (IBxCR) ต้องมีการเซ็ทเป็นเอาท์พุทสำหรับ I/O strobe ถึงจะทำงาน

PEBxR รีจิสเตอร์นี้จะเป็นส่วนของรีจิสเตอร์ที่ใช้ในการเซ็ทให้เป็นเอาท์พุทของแต่ละบิต

PECR รีจิสเตอร์หน่วยควบคุมของพอร์ตขนาน E เป็นรีจิสเตอร์ที่ถูกใช้เพื่อควบคุม สัญญาณนาฬิกาของด้าน High และ low ของเอาท์พุทตัวสุดท้ายของรีจิสเตอร์
ของพอร์ต ในการรีเซ็ท บิต 0, 1, 4, และ 5 จะมีค่าเท่ากับ 0

2.15.6 Parallel Port F

เป็นพอร์ตที่สามารถกำหนดแต่ละบิตสำหรับกำหนดเส้นทางและไดร์วเป็นเรื่องปกติของอินพุทและควบคุมเอาท์พุทและบันทึกลงใน PFDR เป็นเอาท์พุท บิตของพอร์ตที่เก็บในบัฟเฟอร์ด้วยข้อมูลที่ถูกเขียนสำหรับ PFDR (Port C Data Register) ส่งไปยังขาเอาท์พุทที่ขอบขา G เอาท์พุทของ A1 Timer B1 และ Timer B2 สามารถใช้สำหรับฟังก์ชั่น โดยแต่ละช่วงของพอร์ตจะมีการเลือกบริเวณในการควบคุมไทเมอร์

ซึ่งอินพุทและเอาท์พุทจะใช้สำหรับการเข้า- ออก ส่งไปยังสวนต่างๆ บริเวณชิพ เช่น เอาท์พุท PPF(Parallel Port F) ของพอร์ตเอาท์พุท สามารถนำค่าเอาท์พุทที่มีค่าเท่ากับ 4 Pulse - Wide Modulationm เอาท์พุท อินพุท จะมีข้อมูลส่งไปยังควอแรนท์ ถอดรหัส ในขณะที่ SPC(Serial Port C) และ SPD (Serial Port D) ถูกใช้ในโหมดสัญญาณนาฬิกาแบบอนุกรม 2 ขา ของ PDF ซึ่งใช้ ในการส่งสัญญาณ Serial Clock ในขณะที่ clock ภายในจะถูกเลือกในพอร์ตอนุกรม เป็นผลตอบสนองของ PPF จะถูกกำหนดเป็นเอาท์พุท

PFDR Port F Data Register อ่านค่าต่างๆที่ขา เขียนไปยังพอร์ต F Preload Register

PFCR Parallel Port F Control Register เป็นรีจิสเตอร์ที่ถูกใช้สำหรับการควบคุมสัญญาณ นาฬิกาของส่วนบนและส่วนล่างแบ่งออกเป็นส่วนๆของเอาท์พุทรีจิสเตอร์ตัวสุดท้าย มีการเซตค่าที่ บิต 0 ,1 ,4 และ 5 จะมีค่าเป็น 0

PFFR Port F Function Register ตั้งค่าเป็น 1 สำหรบการทำงานหลายฟังก์ชั่น เอาท์พุท บิต ที่ 4 - 7 ทำงานเอาท์พุท PWM และบิตที่ 0 - 1 ทำงานเป็นอะซิงโคนัสของพอร์ต C และ D สัญญาณ เอาท์พุทสำหรับพอร์ตอนุกรมที่ต้องเปลี่ยนค่าสำหรับการสร้างสัญญาณนาฬิกาภายใน

PFDCR Parallel Port F Dirve Control Register ค่า A “0” ทำให้ขาที่คล้ายคลึงกับเอาท์พุท A “1” ทำขาให้เปิดพอร์ตให้เอาท์ออกไปได้ เขียนเท่านั้น
PFDDR - Port F DataDirection Register มีค่าเท่ากับ “1” ทำให้มีการติดต่อที่ขาคล้ายกับเอาท์พุทรีจิสเตอร์ที่มีค่าเท่ากับ 0 หลังจากการเซ็ท

2.15.7 Parallel Port G

เป็นพอร์ตที่สามารถกำหนดแต่ละบิตสำหรับกำหนดเส้นทางและไดร์วเป็นเรื่องปกติของอินพุทและควบคุมเอาท์พุทและบันทึกลงใน PGDR เป็นเอาท์พุท บิตของพอร์ตที่เก็บในบัฟเฟอร์ด้วยข้อมูลที่ถูกเขียนสำหรับ PGDR (Port G Data Register) ส่งไปยังขาเอาท์พุทที่ขอบขา เอาท์พุทของ A1 Timer B1 และ Timer B2 สามารถใช้สำหรับฟังก์ชั่น โดยแต่ละช่วงของพอร์ตจะมีการเลือกบริเวณในการควบคุมไทเมอร์

ซึ่งอินพุทและเอาท์พุทจะใช้สำหรับการเข้า- ออกส่งไปยังสวนต่างๆ บริเวณชิพ เหมือนกับเอาท์พุท โดยที่พอร์ต G ได้ส่งข้อมูลและสัญญาณนาฬิกาเอาท์พุทจาก Serial Port E (SPE) และ F (SPF) อินพุท โดยที่พอร์ต G ได้ส่งข้อมูลและสัญญาณนาฬิกา อินพุทสำหรับ 2 พอร์ตขนาน

PFDR Port G Data Register อ่านค่าต่างๆที่ขา เขียนไปยังพอร์ต G Preload Register

PGCR Parallel Port G Control Register เป็นรีจิสเตอร์ที่ถูกใช้สำหรับการควบคุมสัญญาณนาฬิกาของส่วนบนและส่วนล่างแบ่งออกเป็นส่วนๆของเอาท์พุทรีจิสเตอร์ตัวสุดท้าย มีการ เซตค่าที่บิต 0 , 1, 4 และ 5 จะมีค่าเป็น 0

PGFR Port G Function Register ตั้งค่าเป็น 1 เพื่อให้ฟังก์ชั่นเอาท์พุทสลับกันใช้ได้ บิตที่ 2 และ 6 ทำให้พอร์ตอนุกรมอะซิงโคนัส หรือ SDLC/HDLC ใช้ได้ E และเอาท์พุท F และบิต 4 - 5 และ 0 - 1 ทำให้ SDLC/HDLC ใช้ได้ส่งและรับเอาท์พุทของสัญญาณนาฬิกาสำหรับพอร์ตเป็นชุดE และ F

PGDCR Parallel Port G Drive Control Register "0" ทำขาที่คล้ายคลึงอินพุทที่ออกประจำ "1" ทำขาที่คล้ายเอาท์พุทเปิด เขียนเท่านั้น

PGDDR Port G Data Direction Register มีค่าเท่ากับ "1" ทำให้มีการติดต่อที่ขาคล้ายกับเอาท์พุท รีจิสเตอร์ที่มค่าเท่ากับ 0 หลังจากการเซ็ท Data Register เท่า กับ 0 ทำทุกขาที่เป็นขาอินพุท บิตที่รู้แน่นอนจะอยู่ที่ Register Control เท่า กับ 0 (bits 0 ,1 ,4 ,5) ข้อมูลจะเป็นสัญญาณนาฬิกาที่ถูกส่งไปยังเอาท์พุท Register ในขณะที่มี การโหลด Register ที่เกี่ยวข้องกับพอร์ต G จะไม่ทำการ Reset

2.16 โปรแกรม Dynamic C

Dynamic C คือระบบการพัฒนารวมของการเขียนซอฟท์แวร์ สร้างมาจากระบบคอมพิวเตอร์ของ IBM ซึ่งออกแบบให้เข้ากับคอมพิวเตอรทั่วไป ให้ใช้งานได้

Dynamic C มีการใช้ในเครือข่ายตั้งแต่ปี คศ. 1989 ถูกออกแบบสำหรับการเขียนโปรแกรมฝังระบบ และมีความสามารถตรวจสอบในตัวของมันเองอย่างรวดเร็ว สำหรับไมโครโปรเซสเซอร์ Rabbit 3000 ที่ใช้งานกันทั่วไป สามารถติดต่อกันโดย สายแพ 10 สายที่ พอร์ต B โปรแกรมพื้นฐานของระบบ มีข้อมูลประมาณ 1,000 ไบท์ ที่ใช้ในการจัดเตรียม Debugging และการติดต่อข้อมูลต่างๆ Dynamic C ต้องการ BIOS เพื่อใช้ในการตรวจสอบ โปรแกรม เพื่อที่จะใช้งานได้สะดวก ถ้าผู้ใช้งานได้สะดวกถ้าผู้ใช้หยุดการ Run โปรแกรมและใช้ โปรแกรมใหม่ BIOS ก็จะเช็คการทำงานใหม่ตลอด Dynamic C ออกแบบให้เข้ากับภาษา Assembly หรือใช้ได้กับโปรแกรมภาษาซี

2.16.1 Using Dynamic C

ผู้ใช้โปรแกรม มีตัวเลือกในการที่จะพัฒนาซอฟท์แวร์ ภาษาเขียนใน Flash Memory ขนาด 512 Kbyte หรือ ใน Static Ram ขนาด 512 Kbyte ผลการทำงานในหน่วยความจำ คือการ บันทึกข้อมูลสามารถบันทึกได้ถึง 100,000 ครั้งของการเขียน

ข้อเสียของการใช้ Flash Memory เมื่อมีการดีบักโปรแกรมเพื่อขจัดจังหวะการทำงาน จะทำให้ Interrupt เกิดข้อผิดพลาด การทำงานของโปรแกรมก็จะหยุดตามไปด้วย

ไดนามิค C เป็นภาษาที่ใช้ในการสนับสนุน TCP/IP โดยไดนามิค C จะประกอบไปด้วย Libraries ต่างๆ โดยจะมี Libraries หลัก คือ DCRTCP.LIB อีกทั้งไดนามิค C ยังมี Libraries ส่วนของ DNS (Domain Name Server), IP, TCP, and UDP (User Datagram Protocol) คือ DNS.LIB, IP.LIB, NET.LIB, TCP.LIB and UDP.LIB ส่วนในการติดต่อหรือในส่วนของชั้นเครือขายของโปรโตคอล TCP/IP จะมี Libraries ที่ชื่อ ARP.LIB และ ICMP.LIB

ในส่วนของ Libraries หลัก DCRTCP.LIB จะประกอบไปด้วย macros ที่ทำการตั้งค่าไว้ ส่วนโครงสร้างข้อมูลและฟังค์ชั่นที่ใช้กับ IP เวอร์ชัน 4 ได้มีการสนับสนุนโดย DCRTCP.LIB ในการคอมไพล์ TCP/IP จะต้องให้ส่วนของบอร์ดควบคุมรู้ค่า IP address, netmask and default gateway

2.16.2 การเซตค่า IP Addresses

ค่า IP Address มีความจำเป็นที่จะต้องทำการเซ็ทค่าในช่วงของการคอมไพล์ โดยจะกำหนดการตั้งค่าไว้ที่ MY_IP_ADDRESS, MY_NETMASK, MY_GATEWAY และ MY_NAMESERVER ตามลำดับในช่วงของการคอมไพล์ในส่วนของฟังค์ชั่น tcp_config sethostid, sethostname โดยสามารถที่จะควบคุมโดย Macros

2.16.3 IP Addresses Set Dynamically

Library BOOTP.LIB จะยอมให้บอร์ดในส่วนของ BOOTP หรือ DHCP ของ Client ให้เป็น ส่วนที่จะนำไปใช้ โดยโปรโตคอลจะยึดหลักของขนาดของเครื่องเซิร์ฟเวอรที่ทำการติดตั้งบนเครือข่ายภายใน เครื่องเซิร์ฟเวอร์ BOOTP และ DHCP จะติดตั้งในส่วนกลางของระบบเครือข่ายภายในและจะคอยจัดการในการวางแผนงานของระบบเครือข่าย

โปรโตคอลทั้ง 2 นี้ จะมีค่าพารามิเตอร์ที่ใช้ในการส่งไปยัง Client รวมถึง

- IP address ของ Client
- Net mask
- รายชื่อ Gateway
- Host และ รายชื่อโดเมนพื้นฐาน
- รายชื่อของเครื่องเซิร์ฟเวอร์ ในการนำไปใช้งาน ตองโปรแกรมดังนี้

```
#define USE_DHCP
#use DCRTCP.LIB
```

2.16.4 BOOTP/DHCP Control Macros

มาโคร ต่างๆ สามารถที่จะทำการควบคุมการ DHCP หากมีการเซ็ทค่าก่อนบรรทัด#use "dcrtcp.lib" ในส่วนโปรแกรมแอพพลิเคชั่น USE_DHCP ถ้ามาโครนี้ถูกกำหนดจุดมุ่งหมาย ในการใช้ BOOTP หรือDHCP เพื่อแก้ไขตัวแปรที่ต้องการ ถ้า USE_DHCP ไม่ถูกกำหนดจะทำให้ MY_IP_ADDRESS, MY_NETMASK, MY_GATEWAY และ MY_NAMESERVER อาจจะถูกกำหนดโดยในส่วนของโปรแกรมแอพพลิเคชั่น

2.16.5 Sizes for TCP/IP I/O Buffers

ในการเริ่มทำงานของไดนามิค C เวอร์ชั่น 8.01 บัฟเฟอร์ TCP และ UDP I/O C ได้มีการแบ่งแยกออกมาเป็น

TCP_BUF_SIZE ได้มีการกำหนดขนาดบัฟเฟอร์ของ TCP ไว้ที่ 4096 ไบท์

UDP_BUF_SIZE ได้มีการกำหนดขนาดบัฟเฟอร์ของ UDP ไว้ที่ 4096 ไบท์

ถ้า SOCK_BUF_SIZE มีการกำหนดจะทำให้ค่า TCP_BUF_SIZE และ UDP_BUF_SIZE จะตรงกับ SOCK_BUF_SIZE แต่ถ้า SOCK_BUF_SIZE ไม่มีการกำหนด จะทำให้ค่า TCP_BUF_SIZE และ UDP_BUF_SIZE จะเท่ากับ tcp_MaxBufSize * 2

2.16.6 Number of Sockets

การเริ่มต้นการทำงาน Dynamic C เวอร์ชั่น 8.01 จะมีการกำหนดมาโคร 2 ตัวให้กับ หมายเลขซ็อกเกตได้ดังนี้

MAX_TCP_SOCKET_BUFFERS จะมีการกำหนดหมายเลขสูงสุดให้กับซ็อกเกตของ TCP โดยจะมีการเรียก tcp_open () หรือ tcp_listen ()

MAX_UDP_SOCKET_BUFFERS จะมีการกำหนดหมายเลขสูงสุดให้กับซ็อกเกตของ UDP udp_open ()

2.16.7 Passive Open

จะมีอยู่ 2 เส้นทางในการเปิดซ็อกเกตของ TCP คือ passive และ active ซึ่งการเปิดซ็อกเกตนี้จะต้องทำการเรียก tcp_listen (); โดยจะทำการรอการติดต่อกับอุปกรณ์และชนิดของการเปิดจะใช้กับเซิร์ฟเวอร์ของอินเตอร์เนตหรืออาจจะให้ tcp_listen () เป็นตัวชี้ตำเหน่งของข้อมูลของ tcp_Socket ถ้าคุณต้องการที่จะติดต่อโดยคุณได้ยอมรับข้อตกลงของหมายเลขพอร์ตและ IP Address โดยทำการเซ็ทค่าให้เป็นศูนย์หรือหนึ่งทั้งคู่

2.16.8 Active Open

เมื่อมีการเรียกหน้าเว็บเบราเซอร์ก็จะทำการเชื่อมต่อกับเซิร์ฟเวอร์เพื่อทำการเปิด โดยจะทำการเรียกใช้ tcp_open () และยังใช้พารามิเตอร์คล้ายๆกับการใช้ tcp_listen () โดยหลักพารามิเตอรนี้

จะเกี่ยวกับ IP Address และหมายเลขพอร์ตคุณสามารถทำการติดต่อ โดยทำการเซ็ทค่าพารามิเตอร์ lport ให้เป็นศูนย์ส่วน DCRTCP.LIB เป็นการแสดงถึงการเลือกพอร์ตระหว่าง 1024 และ 65535 หาก tcp_open () กลับค่ามาเป็นศูนย์อีกครั้งจะไม่สามารถทำการติดต่อได้

2.16.9 Delay a Connection

การยอมรับการร้องขอการติดต่อเมื่อกระบวนการวิธีการร้องขอไม่เหมาะสมก็จะทำการเรียกฟังค์ชั่น tcp_reserveport () เมื่อมีการตอบรับการติดต่อพารามิเตอร์ในส่วนหัวของ TCP จะทำการเซ็ทค่าเป็นศูนย์ในช่วงเวลานี้จะทำการรอค่าจากพารามิเตอร์ tcp_clearreserve (port number) เพื่อทำการดูแลในการติดต่อนอกจากนี้ยังมีมาโคร USE_RESERVEDPORTS จะทำการกำหนดเพื่อให้เกิดการทำงานขึ้นของสองฟังค์ชั่นคือเมื่อมีการใช้ tcp_reserveport, 2MSL (Maximum Segment Lifetime)

2.16.10 Skeleton Program

โปรแกรมข้างล่างนี้ เป็นโครงสร้างพื้นฐานของ Dynamic C TCP/IP โดยส่วนแรกจะทำการกำหนดการเซ็ทค่า IP ของข้อมูลในบรรทัด "memmap" จะเป็นการให้โปรแกรมทำการคอมไพล์และตรวจสอบค่าส่วนในบรรทัด "use" จะเป็นตัวส่งให้ตัวคอมไพล์ทำการคอมไพล์ของข้อมูลที่มีการตั้งค่าไว้ในส่วน TCP/IP ของ Dynamic C

```
#define MY_IP_ADDRESS "10.10.6.101"
#define MY_NETMASK "255.255.255.0"
#define MY_GATEWAY "10.10.6.19"
#memmap xmem
#use dcrtcp.lib
main ()
{
        sock_init ();
        for (;;) {
                tcp_tick (NULL);
                }
}
```

2.16.11 TCP/IP Stack Initialization

ใน TCP/IP จะมีฟังค์ชั่น Main () เป็นส่วนเริ่มแรกและจะมีการเรียก sock_init () เพื่อ ใช้กับโครงสร้างข้อมูลภายในและชิป Ethernet หรือชิป RealTek ส่วน DCRTCP.LIB จะคอยจัดการกับแพ็คเกตที่เข้ามา

2.16.12 Packet Processing

เมื่อไรก็ตามเมื่อแพ็คเกตมีการนำเข้ามาจะมีการเรียก tcp_tick(),tcp_open, udp_open, sock_read, sock_write, sock_close และ sock_abort ซึ่งเป็นวิธีที่ดีที่ tcp_tick() จะรู้ระยะเวลาในการประมวลผลโปรแกรมที่ส่งมาในรูปของแพ็คเกต

2.16.13 Function Reference

ในส่วนนี้จะกล่าวถึงฟังค์ชั่นที่ใช้ใน DCRTCP.LIB ในเริ่มต้นนั้น DCRTCP.LIB ของ Dynamic C 7.05 จะมีฟังค์ชั่นที่มีการใช้อยู่ภายนอก DNS.LIB, IP.LIB, NET.LIB, TCP.LIB และ UDP.LIB และยังมีฟังค์ชั่นอีกส่วนหนึ่งที่สามารถเลือกใช้ได้ เช่น ARP.LIB, ICMP.LIB, BSDNAME.LIB และ XMEM.LIB

2.16.14 Macros

- MAX_SOCKETS มาโครนี้จะเป็นตัวกำหนดหมายเลขซ็อกเกตเพื่อทำการจัดสรรพื้นที่
- MY_GATEWAY มาโครนี้จะทำการส่งค่านี้เพื่อไปควบคุม Gateway ในช่วงเวลา Runtime
- MY_NETMASK มาโครนี้เป็นพื้นฐานของ NETMASK ที่ใช้ในการควบคุม
- MY_NAMESERVER มาโครนี้เป็นการแสดงชื่อของเซิร์ฟเวอร์ในช่วง Runprogram
- MY_DOMAIN มาโครนี้จะทำการควบคุมตำแหน่งเริ่มต้นของโดเมนในช่วงเวลา Runtime
- MY_IP_ADDRESS มาโครนี้จะทำการระบุตำแหน่ง IP Address เพื่อใช้ในการควบคุมในช่วง

เวลา Runtime

2.17 การควบคุมการเคลื่อนที่เซอร์โว

เซอร์โว(Servo)เป็นอุปกรณ์ขนาดเล็กที่ประกอบด้วยวงจรอิเล็คทรอนิกส์และเครื่องกลไกทำหน้าที่แปลงสัญญานควบคุมไฟฟ้าจากเครื่องรับวิทยุไปเป็นการเคลื่อนที่เพื่อนำไปควบคุมอุปกรณ์บังคับการบินต่างๆ เช่น บังคับ หางเสือ เพื่อเลี้ยว เป็นต้น โดยที่เซอร์โวจะมีแกนหมุนติดอยู่แกนหมุนนี้จะสามารถเปลี่ยนตำแหน่งได้ด้วยการส่งสัญญาณควบคุมเข้าไปที่เซอร์โวและเซอร์โว

จะยังคงตำแหน่งของแกนหมุนไว้จนกว่าสัญญาณควบคุมจะมีการเปลี่ยนแปลงไป มุมของแกนหมุนนี้สามารถเปลี่ยนแปลงได้ตั้งแต่ 0-210 องศา แต่โดยทั้วไปจะใช้งานอยู่ในช่วง 0-180 องศา สายสัญญาณที่ต่อเข้าไปที่เซอร์โวจะมีสามเส้นคือ สายสีดำจะต่อกับไฟลบ สายสีแดงจะต่อกับไฟบวก และสายสีขาวเป็นสายสัญญาณควบคุม (สีของสายไฟสำหรับเซอร์โวยี่ห้อ FUTABA ถ้าเป็นยี่ห้ออื่นจะแตกต่างจากนี้)

2.17.1 Pulse

pulse เป็นสัญญาณไฟฟ้าที่มีการเปลี่ยนแปลงในช่วงเวลาสั้นๆ ตัวอย่างเช่น ถ้าเราเปิดสวิทช์ไฟ (มีสถานะเป็น 1) เป็นเวลาหนึ่งวินาที แล้วปิดสวิทช์ไฟ (มีสถานะเป็น 0) อีกหนึ่งวินาที สลับกันไปเรื่อยๆก็จะเป็นการสร้างสัญญาณ pulse ที่มีช่วงเวลาเปิดและปิดรวมเป็น 2 วินาที หรือเรียกว่ามีคาบเวลาเท่ากับ 2 วินาที เป็นต้น ดังนั้นการวัดคาบเวลาของ pulse (ใช้สัญลักษณ์ T) จะเริ่มวัดตั้งแต่ช่วงของการเปิดสวิทช์ไฟครั้งแรกจนถึงการเปิดสวิทช์ไฟครั้งต่อไป คาบเวลายังมีความสัมพันธ์กับความถี่ (ใช้สัญลักษณ์ f) ของการเปิดและปิดไฟด้วย โดยเราสามารถคำนวณหาความถี่ของ pulse ได้จากสูตร f = 1/T duty cycle จะวัดเป็นเปอร์เซ็นต์ (%) โดยถ้าช่วงเวลาของการเปิดและปิดไฟเท่ากันเราจะเรียกว่าสัญญาณ pulse นี้มีค่า duty cycle เป็น 50% ถ้าช่วงเวลาในการเปิดไฟน้อยกว่าช่วงเวลาในการปิดไฟ ค่า duty cycle ก็จะลดลง เช่นถ้าเราเปิดไฟครึ่งวินาที แล้วปิดไฟหนึ่งวินาทีครึ่ง ค่า duty cycle จะเป็น 25% เป็นต้น ดังนั้นค่า duty cycle หาได้จาก

Duty cycle (D) = (ระยะเวลาของการเปิดไฟ x 100) / คาบเวลาของ Pulse

2.17.2 ความถี่ (Frequency)

ความถี่ (Frequency หรือ f) คือจำนวน pulse ต่อหนึ่งวินาที มีหน่วยเป็นเฮิรท์ (Hz) เช่น ถ้าเราเปิดไฟและปิดไฟ 2 ครั้งในเวลาหนึ่งวินาที เราจะนับจำนวน pulse ได้เท่ากับ 2 นั้นก็คือความถี่ 2 เฮิรท์นั้นเอง เราสามารถคำนวณหาคาบเวลาของ pulse ได้จากสูตร T = 1/fโดยปกติเซอร์โวจะใช้ไฟเลี้ยงอยู่ระหว่าง 4 - 6 โวลท์ การควบคุมตำแหน่งของแกนหมุนจะใช้สัญญาณเป็น pulse สั้นๆ โดยมีความกว้างของสัญญาณอยู่ระหว่าง 1 - 2 ms (มิลลิวินาที) โดยถ้าสัญญาณควบคุมมี Pulse เป็น 1 ms แกนหมุนของเซอร์โวจะอยู่ที่ตำแหน่ง 0 องศา ถ้าเป็น 1.5 ms แกนหมุนจะอยู่ที่ตำแหน่ง 90 องศา และ 2 ms แกนหมุนจะอยู่ที่ตำแหน่ง 180 องศา หลังจากส่ง pulse ควบคุมเข้าไปที่เซอร์โวแล้วจะต้องรออีก 20 ms (0.02 วินาที) (แต่เซอร์โวสามารถรับคาบเวลาของ pulse ได้ตั้งแต่ 10 ms จนถึง 30 ms) จึงจะส่ง pulse อีกอันตามเข้าไปได้อีก จากข้อกำหนดนี้ทำให้เราสามารถคำนวนหาค่า duty cycle ได้ดังนี้

ช่วงเวลาเปิดสัญญาณต่ำสุด	= 1ms
ช่วงเวลาของสัญญาณควบคุมทั้งหมด	= 20 ms
ค่า Duty cycle ต่ำสุด	= (1 x 100) / 20 = 5%

ช่วงเวลาเปิดสัญญาณสูงสุด = 2 ms

- ช่วงเวลาของสัญญาณควบคุมทั้งหมด = 20 ms

ค่า Duty cycle สูงสุด = (2 x 100) / 20 = 10%

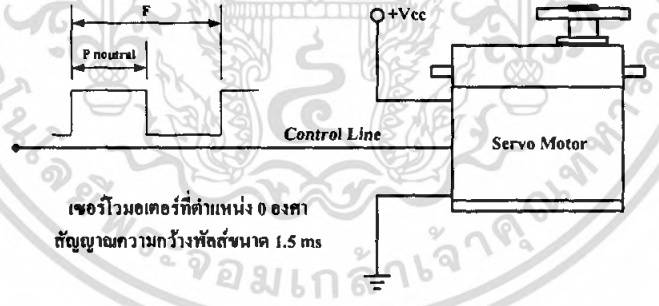
ดังนั้นเราต้องการค่า Duty cycle ที่เซอร์โวต้องการอยู่ระหว่าง 5 - 10%

การเคลื่อนที่ของเซอร์โวมอเตอร์ การทำงานในส่วนของเซอร์โวมอเตอร์ที่ใช้งานในระบบการจับภาพเคลื่อนไหวโดยใช้กล้องดิจิตอล ในการควบคุมการทำงานของเซอร์โวมอเตอร์ทำได้โดยการป้อนสัญญาณความกว้างพัลส์ให้กับมอเตอร์ ซึ่งตำแหน่ง และทิศทางการหมุนของมอเตอร์นี้ จะขึ้นอยู่กับขนาดของความกว้างของพัลส์นั้น ๆ โดยทั่วไปแล้ว ความกว้างของสัญญาณพัลส์ จะมีจุดให้อ้างอิง 3 จุด ดังนี้

- สัญญาณความกว้างพัลส์ขนาด 1.5 ms จะควบคุมให้เซอร์โวมอเตอร์หมุนไปอยู่ที่ตำแหน่งมุม 0 องศา หรือจุดกึ่งกลางของมอเตอร์

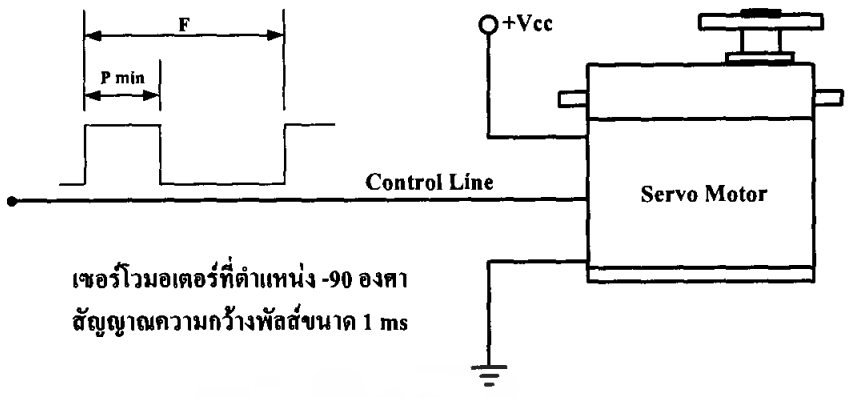
- สัญญาณความกว้างพัลส์ขนาด 1 ms จะควบคุมให้เซอร์โวมอเตอร์หมุนไปอยู่ที่ตำแหน่งมุม -90 องศา หรือในทิศทางทวนเข็มนาฬิกา

- สัญญาณความกว้างพัลส์ขนาด 2 ms จะควบคุมให้เซอร์โวมอเตอร์หมุนไปอยู่ที่ตำแหน่งมุม +90 องศา หรือในทิศทางทวนเข็มนาฬิกา



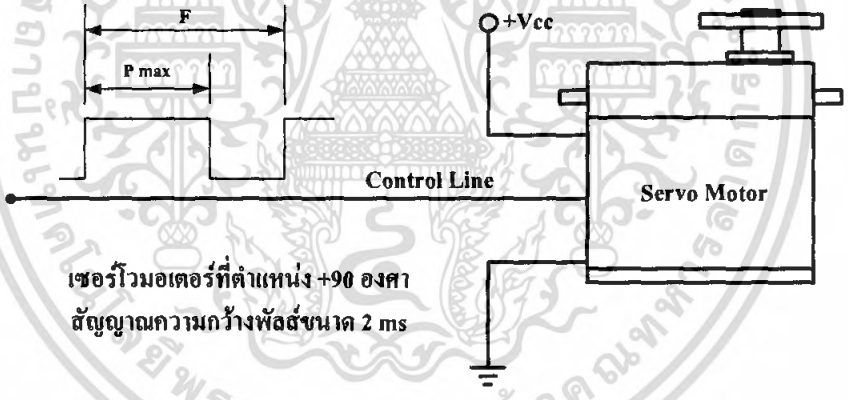
รูปที่ 2.21 การควบคุมเซอร์โวที่ตำแหน่ง 0 องศา

รูปที่ 2.21 แสดงสัญญาณความกว้างพัลส์ขนาด 1.5 ms จะควบคุมให้เซอร์โวมอเตอร์หมุนไปอยู่ที่ตำแหน่งมุม 0 องศา หรือจุดกึ่งกลางของมอเตอร์ ส่วนการควบคุมการทำงานของเซอร์โวมอเตอร์จะใช้หลักการสร้างสัญญาณพัลส์ขนาดความกว้าง 1.5 ms ส่งไปควบคุมการทำงานของมอเตอร์



รูปที่2.22 การควบคุมเซอร์โวที่ตำแหน่ง -90 องศา

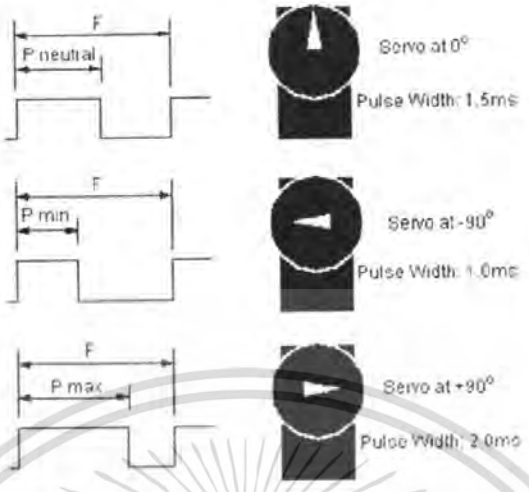
รูปที่ 2.22 แสดงสัญญาณความกว้างพัลส์ขนาด 1 ms จะควบคุมให้เซอร์โวมอเตอร์หมุนไปอยู่ที่ตำแหน่งมุม -90 องศา หรือในทิศทางทวนเข็มนาฬิกา ส่วนการควบคุมการทำงานของเซอร์โวมอเตอร์ จะใช้หลักการสร้างสัญญาณพัลส์ขนาดความกว้าง 1 ms ส่งไปควบคุมการทำงานของมอเตอร์



รูปที่2.23 การควบคุมเซอร์โวที่ตำแหน่ง +90 องศา

รูปที่ 2.23 แสดงสัญญาณความกว้างพัลส์ขนาด 2 ms จะควบคุมให้เซอร์โวมอเตอร์หมุนไปอยู่ที่ตำแหน่งมุม +90 องศา หรือในทิศทางทวนเข็มนาฬิกา ส่วนการควบคุมการทำงานของเซอร์โวมอเตอร์ จะใช้หลักการสร้างสัญญาณพัลส์ขนาดความกว้าง 2 ms ส่งไปควบคุมการทำงานของมอเตอร์

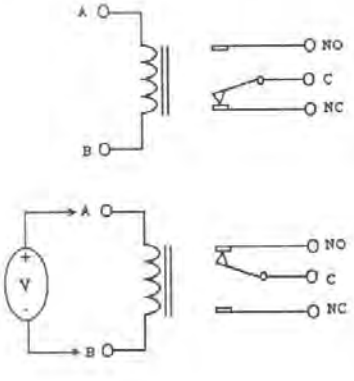
ส่วนการที่จะควบคุมให้มอเตอร์หมุนเป็นมุมอื่นๆนั้น ก็สามารถทำได้โดยการป้อนสัญญาณพัลส์เป็นระดับความกว้างต่าง ๆ โดยอ้างอิงจากจุดทั้ง 3 จุด ที่กล่าวมานี้ ตัวอย่างเช่น ถ้าต้องการให้มอเตอร์หมุนไปที่มุม -45 องศา จะต้องป้อนสัญญาณพัลส์ที่มีความกว้าง 1.25 ms เป็นต้น และสัญญาณพัลส์นี้จะต้องจ่ายให้มอเตอร์ทุก ๆ 20 ms (Period) เพื่อรักษาสภาพตำแหน่งของมอเตอร์ไว้



รูปที่ 2.24 การหมุนในตำแหน่งต่างของเซอร์โว

2.18 รีเลย์ (The Relay)

เป็นสวิชต์แม่เหล็กชนิดหนึ่งที่การทำงานหน้าสัมผัสสวิทช์เคลื่อนที่เปิด-ปิดได้จาดผลของแม่เหล็กไฟฟ้า ซึ่งเกิดจากการจ่ายกระแสไฟฟ้าเข้าไปในขดลวดของรีเลย์ (Coil)และจะหยุดทำงานเมื่อหยุดจ่ายกระแสไฟฟ้าเข้าไป โครงสร้างเบื้องต้นประกอบ ไปด้วยขดลวดรีเลย์หนึ่งชุดและหน้าสัมผัสรีเลย์อย่างน้อยหนึ่งชุดหรือมากกว่า เช่น อาจจะมีหน้าสัมผัสปกติเปิด (no) หนึ่งชุด หรือหน้าสัมผัสปกติปิด (nc) หนึ่งชุดก็ได้ แสดงในรูป 2.27 เมื่อไม่จ่ายกระแสไฟฟ้าให้กับขดลวดของรีเลย์ หน้าสัมผัส NC คือ ขั้ว 1 และ 2 จะปิดอยู่ สภาวะนี้คือ สภาวะที่รีเลย์ไม่ทำงาน เมื่อจ่ายกระแสไฟฟ้าเข้าไปในขดลวด จะเกิดเส้นแรงแม่เหล็กไฟฟ้าของขดลวดดึงดูดให้หน้าสัมผัสของรีเลย์เคลื่อนที่ลงมาทำให้หน้าสัมผัสของขั้ว 1และ 2 ซึ่งเคยปิดอยู่จากกัน และต่อหน้าสัมผัสของขั้ว 1 และ 3ให้ติดกันแทน ดังรูป 2.27

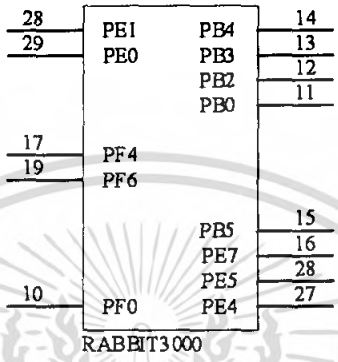


รูป 2.27 แสดงโครงสร้างเบื้องต้นและการทำงานของรีเลย์

บทที่ 3

การออกแบบระบบควบคุมและรักษาความปลอดภัยภายในบ้าน

3.1 กำหนดพอร์ตใช้งาน Rabbit 3000 Core Module RCM 3720



รูปที่ 3.1 พอร์ตใช้งาน Rabbit 3000

จากโครงสร้างของ Rabbit3000 และข้อมูล หน้าที่ของพอร์ตต่าง ๆ จากตารางที่ 2.2 เราสามารถเลือกพอร์ตใช้งาน โดยพอร์ตที่เราทำการเลือกนั้นจะต้อง สามารถทำหน้าที่เป็น Input/Output พอร์ตได้ด้วย แต่ในการใช้งานเป็น Module PWM เพื่อใช้ในการควบคุมความกว้างของพัลส์ที่จะส่งไปควบคุมการหมุนของ Servo Motor นั้นเราจะเลือกใช้ PF4 และ PF6 ซึ่งพอร์ตที่เราทำการเลือกแสดงได้ดังรูปที่ 3.1 และตารางที่ 3.1

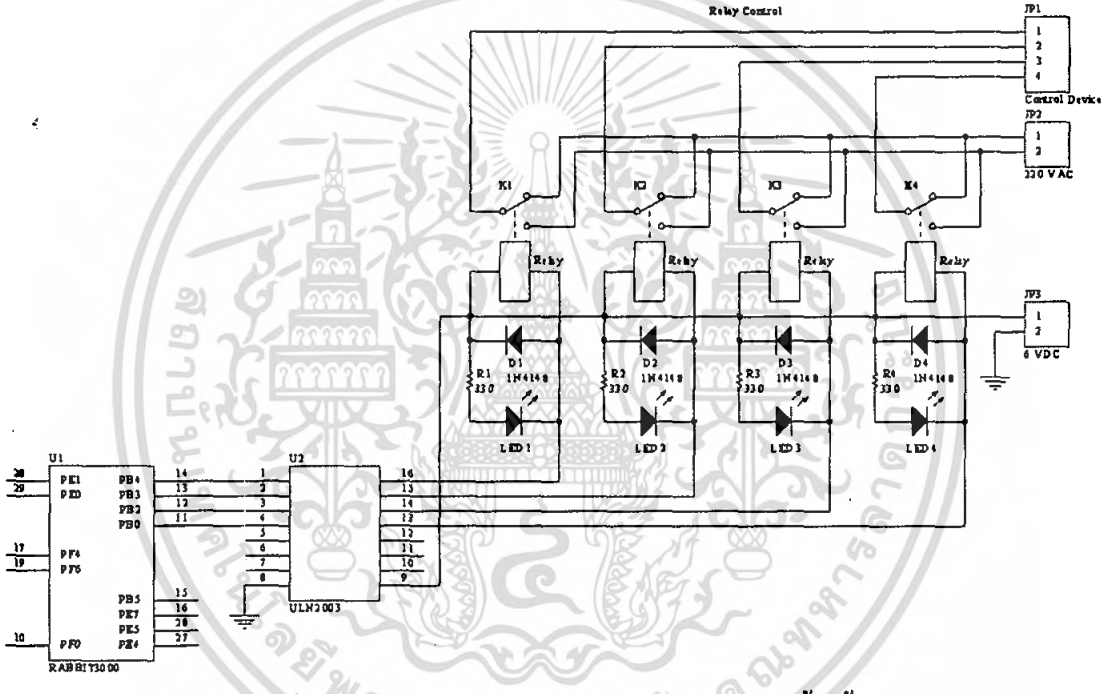
ตารางที่ 3.1 แสดงหน้าที่การทำงานของพอร์ตต่าง ๆ

ตำแหน่งขา	ชื่อพอร์ต	ชนิดของพอร์ต	หน้าที่	หมายเหตุ
14	PB4	เอาท์พุต	ควบคุมหลอดไฟหลอดที่ 1	-
13	PB3	เอาท์พุต	ควบคุมหลอดไฟหลอดที่ 2	-
12	PB2	เอาท์พุต	ควบคุมหลอดไฟหลอดที่ 3	-
11	PB0	เอาท์พุต	ควบคุมปั๊มน้ำ	-
15	PB5	อินพุต	ตรวจสอบสถานะหลอด 1	-
16	PE7	อินพุต	ตรวจสอบสถานะหลอด 2	-
26	PE5	อินพุต	ตรวจสอบสถานะหลอด 3	-
27	PE4	อินพุต	ตรวจสอบสถานะปั๊มน้ำ	-
28	PE1	อินพุต	เช็คเซนเซอร์แสงหน้าต่างที่ 1	-

ตำแหน่งขา	ชื่อพอร์ต	ชนิดของพอร์ต	หน้าที่	หมายเหตุ
29	PE0	อินพุต	เช็คเซนเซอร์แสงหน้าต่างที่ 2	-
17	PF4	PWM0	ควบคุมขากล้องแนวตั้ง	-
19	PF6	PWM2	ควบคุมขากล้องแนวนอน	-
10	PF0	เอาท์พุต	สร้างเสียงเตือนภัย	-

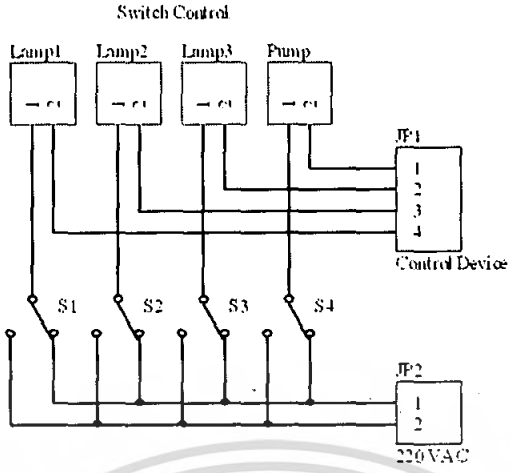
3.2 การควบคุม Lamp1, Lamp2, Lamp3 และ Pump

3.2.1 วงจรควบคุม



รูปที่ 3.2 วงจรควบคุมการเปิด-ปิดหลอดไฟ และปั๊มน้ำ

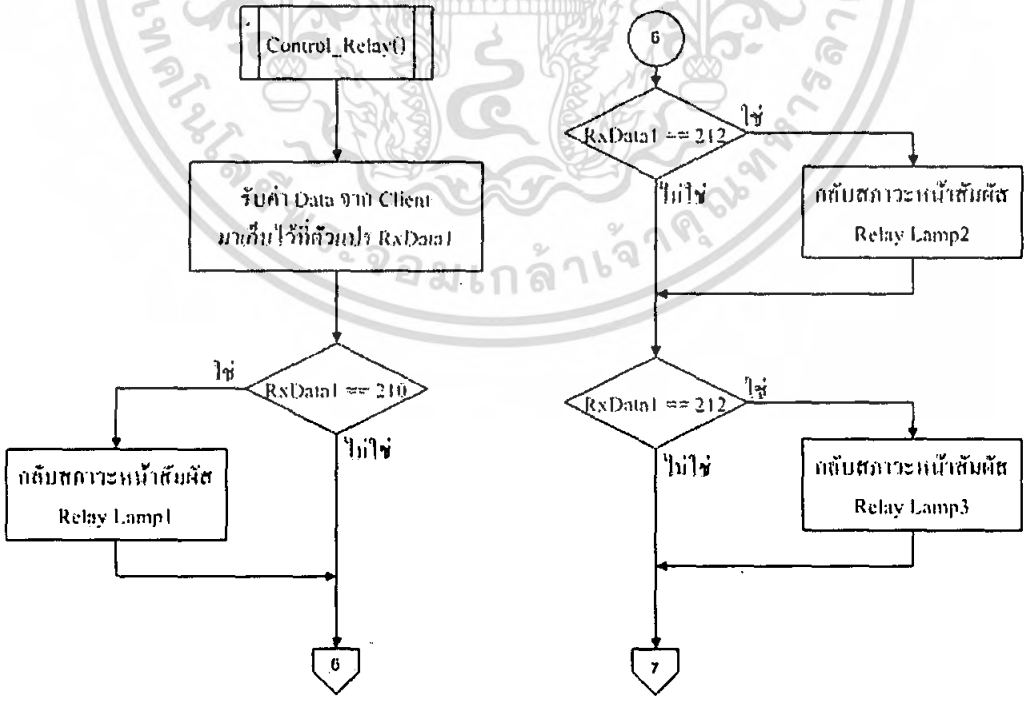
จากวงจรในรูปที่ 3.2 เราจะใช้ Port PB4, PB3, PB2, PB0, ของ Rabbit3000 เป็นพอร์ตควบคุมการ ON – OFF หลอดไฟและปั๊มน้ำ โดยจะใช้งานร่วมกันกับ IC U2#ULN2003 เป็นตัวทำหน้าที่ ขับกระแสเลี้ยง อุปกรณ์ Relay ที่จะคอยตัดต่อระหว่างกราวด์ กับไฟ AC 220 โวลต์ การทำงานก็คือเมื่อ พอร์ตใดพอร์ตหนึ่งมีสถานะเป็น Logic "1" IC#ULN2003 จะทำการตัดต่อวงจรภายในให้พอร์ตทางด้าน Output ต่อลงกราวด์ ส่งผลให้แรงดันไฟ 6 VDC ไหลผ่านขดลวด Relay ลงกราวด์ครบวงจร เกิดสนามแม่เหล็กเหนี่ยวนำให้หน้าสัมผัสของ Relay เปลี่ยนแปลงสภาวะไปจากเดิม การที่จะให้หน้าสัมผัสคงค้างสภาวะครั้งล่าสุดไว้ได้นั้นจะต้อง ป้อน Logic เดิมทางด้าน Input ตลอดเวลา

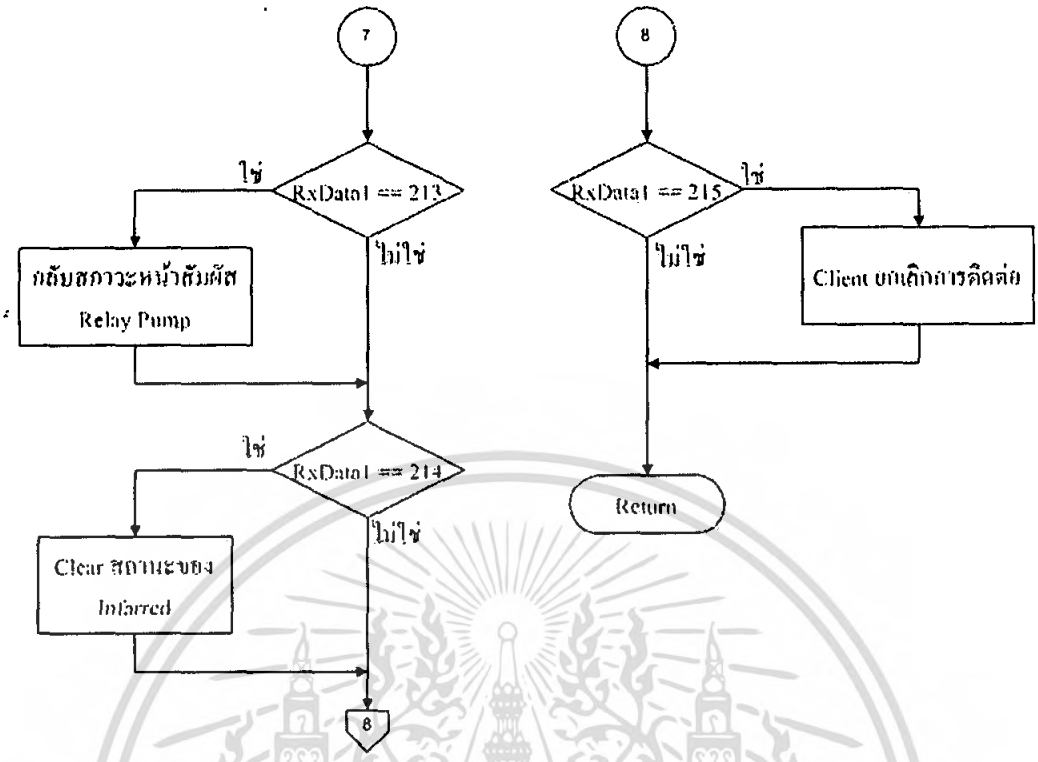


รูปที่ 3.3 วงจร Switch ปิด – เปิด อุปกรณ์จากที่บ้านใช้งานรวมกับ Relay

แต่เนื่องจากว่าการควบคุมจากระยะไกลเพียงอย่างเดียวไม่มีความสะดวกในการเปิด-ปิดตอนอยู่ที่บ้าน จึงออกแบบให้ Switch ที่ใช้ในการควบคุมอุปกรณ์ต่าง ๆ เป็น Switch 2 ทาง เพื่อความสะดวกในการควบคุมจากระยะไกลและจากที่บ้านด้วยดังแสดงเป็นวงจรได้ในรูปที่ 3.3

3.2.2 การเขียนโปรแกรมควบคุม Server (Rabbit 3000)





รูปที่ 3.4 Flowchart ของโปรแกรมควบคุมการปิด – เปิด Lamp1 ถึง Lamp 3 และ Pump

โปรแกรม Control_Relay () เป็นฟังก์ชั่นย่อย ที่คอยรับค่าจาก Client ที่ติดต่อเข้ามาเพื่อนำค่าที่ได้รับเข้ามาประมวลผลแล้วสั่ง ปิด – เปิด หลอดไฟและปั๊มน้ำที่อยู่ที่บ้านซึ่งในการเขียนโปรแกรมนั้นเราจะใช้ โปรแกรม Dynamic C 9.21 ที่ให้มาพร้อมโมดูล Rabbit3000

ตัวอย่างการอ่านค่าที่รับมาจาก Client

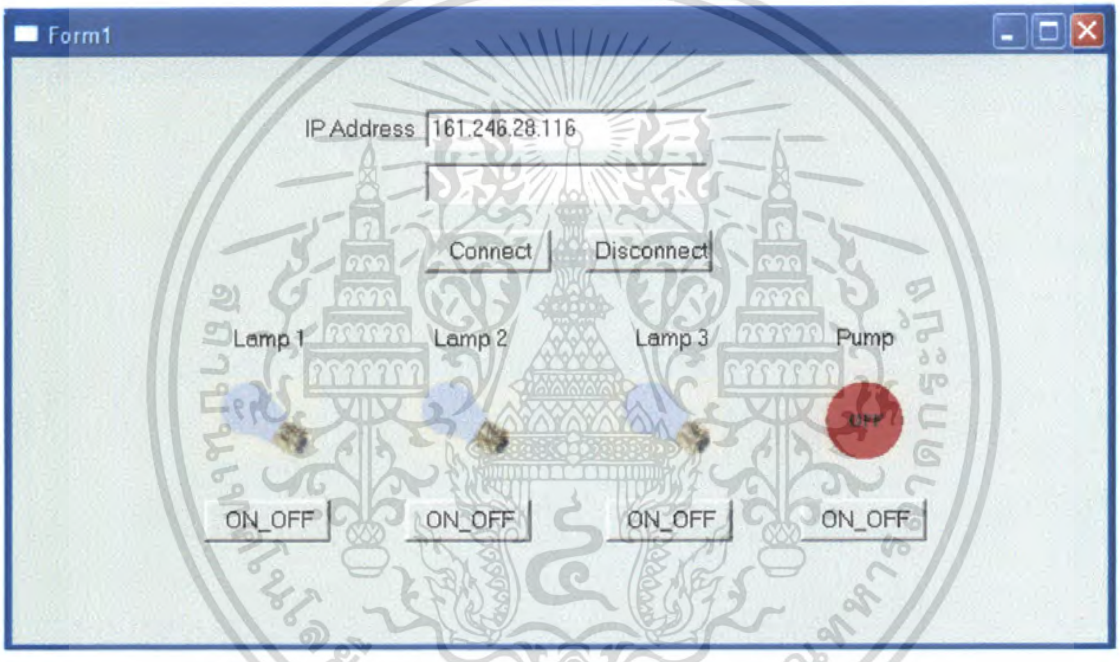
```
byte_read=sock_fastread (&socket, &RxData1, 1);
Switch (RxData1)
    {
    Case 210:
        BitWrPortI (PBDR, &PBDRShadow, Relay1=Relay1? 0:1, 4);
        Break;
```

RxData1 จะมีCode ต่าง ๆ ที่ส่งมาจาก Client แล้วเราก็ใช้คำสั่ง Switch () เพื่อตรวจสอบว่า Code ที่ได้รับเข้ามานั้นเป็นคำสั่งที่ต้องการทำอะไรในที่นี้ คำสั่ง BitWrPortI คือต้องการกลับหน้าสัมผัสของ Relay1 หรือ PB4 ให้มีค่าตรงข้ามกับค่าเดิมเช่นจากเดิมเป็น “0” เปลี่ยนเป็น “1”

ตารางที่ 3.2 ความหมายของ Code ต่าง ๆ ที่รับเข้ามา

Relay1 (Lamp1)	Relay2 (Lamp2)	Relay3 (Lamp3)	Relay4 (Pump)	Clear Infrared	Disconnect
210	211	212	213	214	215
PB4	PB3	PB2	PB0	-	-

3.2.3 การเขียนโปรแกรมควบคุมผ่านระบบ Network โดย C++Builder6

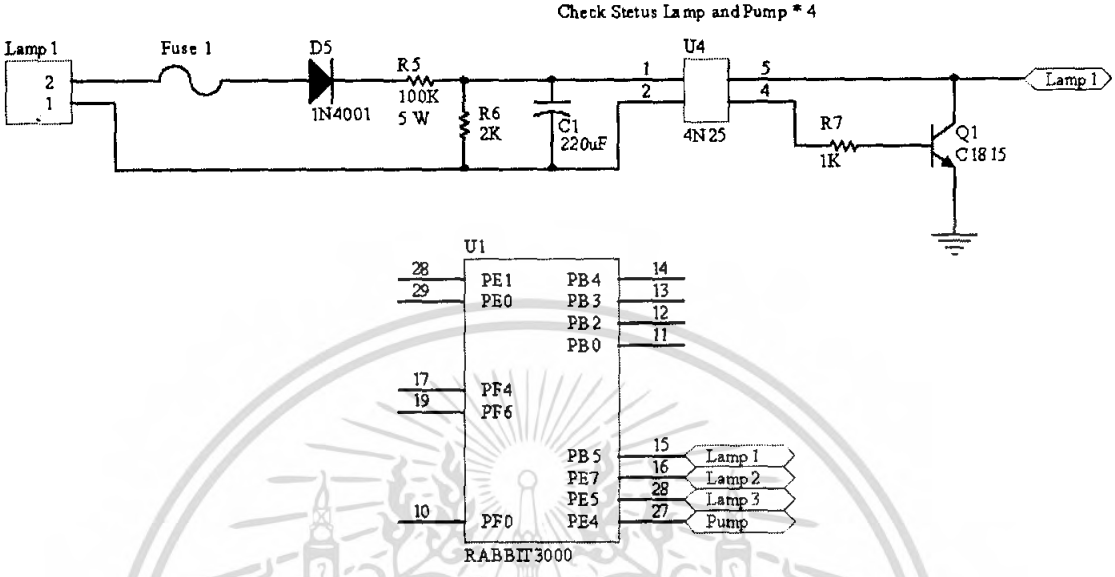


รูปที่ 3.5 แสดงหน้าต่างโปรแกรมควบคุมระยะไกลผ่านระบบ Network

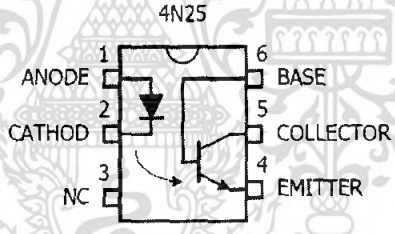
ใช้โปรแกรม C++Builder6 ในการเขียนหน้าต่าง ควบคุมอุปกรณ์ระยะไกลผ่านระบบ Network ในการเขียนโปรแกรมนั้นจะใช้ภาษา C ส่วนของการทำการติดต่อกับโมดูลเราจะระบุหมายเลข IP Address ในช่องสำหรับ IP Address และเบื้องต้นได้ทดลองทำการเขียนโปรแกรมเพื่อควบคุมหลอดไฟ 3 หลอดและ ปั๊มน้ำ โดยเมื่อเราทำการกดเพื่อเปิด Lamp1 โปรแกรมจะส่ง Code 210 ออกมาให้ Server (Rabbit 3000) เพื่อควบคุมให้หลอดไฟ Lamp1 ติด ทำนองเดียวกันเมื่อเราทำการกด Lamp2, Lamp3, และ Pump ก็จะส่งค่า Code ต่าง ๆ ออกมาตามตารางที่ 3.2 และจะทำงานเช่นเดียวกันกับ Lamp1 การทำงานของโปรแกรมนี้ไม่เหมาะที่จะเขียนเป็น Flowchart ในการอธิบายการทำงาน เนื่องจากมันจะทำงานเป็นเหตุการณ์หรือคล้าย ๆ กับการบริการ อินเตอร์รัพของไมโครคอลโทรลเลอร์ ไม่ได้มีการวนลูปเหมือนโปรแกรมอื่น ๆ

3.3 การตรวจสอบสถานะของ Lamp1, Lamp2, Lamp3, และ Pump

3.3.1 วงจรตรวจสอบสถานะของอุปกรณ์



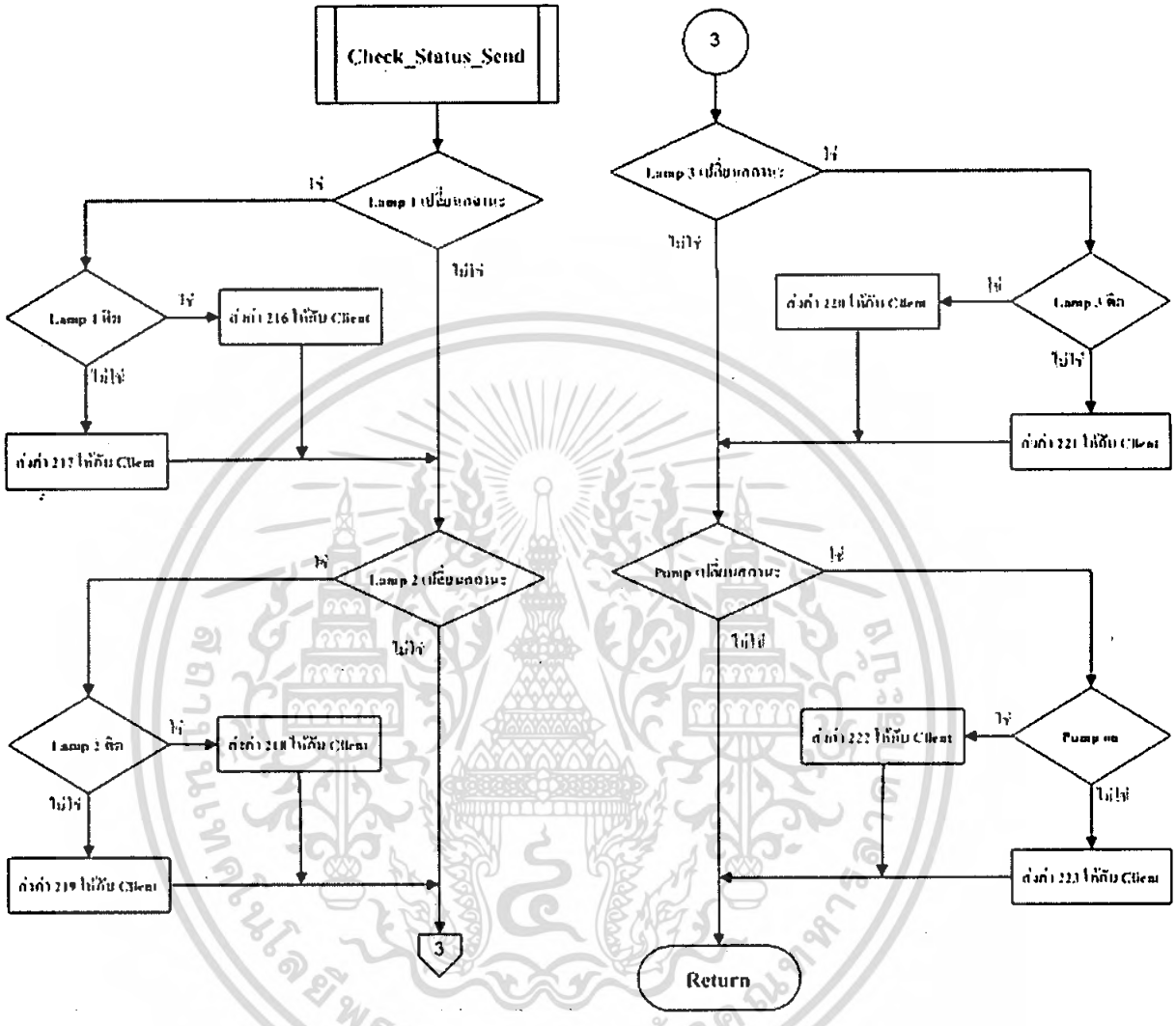
รูปที่ 3.6 วงจรตรวจจับสถานะของ Lamp1, Lamp2, Lamp3, Pump



รูปที่ 3.7 วงจรภายใน OPTO 4N25

โครงงานนี้คือ ระบบรักษาความปลอดภัยภายในบ้านผ่านระบบ Network ซึ่งก็หมายความว่า เมื่อเราทำการสั่ง ปิด – เปิด อุปกรณ์ไฟฟ้า เราจะต้องทราบสถานะของอุปกรณ์ ที่เราสั่งปิด – เปิด ด้วยว่าทำงานตามที่เราสั่งงานหรือไม่ จากวงจรนี้ Rabbit3000 จะรับค่าสถานะของอุปกรณ์ต่าง ๆ ผ่านพอร์ต PB5, PE7, PE5, PE4, โดยใช้ OPTO U4#4N25 เป็นตัวส่งผ่านทางแสงจากวงจรทาง AC 220V ที่ต่อขนานกับหลอดไฟ เมื่อหลอดไฟติดจะมีไฟ AC 220V เข้ามาที่ D5 แล้วทำการ Rectifier เป็นไฟ DC จากนั้นใช้ R100K 5W ทำการ Drop แรงดันให้ต่ำลงและใช้ C1 220 uF ทำหน้าที่กรอง Ripple ให้แรงดัน DC มีความเรียบมากยิ่งขึ้น จากรูปที่ 3.7 เมื่อ Diode ภายใน OPTO ได้รับ Forward Bias จะทำงานกำเนิดแสงให้ ทรานซิสเตอร์ NPN "ON" Drive กระแสให้ C1815 ทำงาน ทำให้แรงดันที่ Port ของ Rabbit3000 กลายเป็น Logic "0" และเพื่อความปลอดภัยของ Rabbit3000 เราจึงใส่ Fuse 1 เป็นตัวป้องกันไฟ AC 220V อีกทีหนึ่ง

3.3.2 การเขียนโปรแกรม ตรวจสอบสถานะ Server (Rabbit3000)



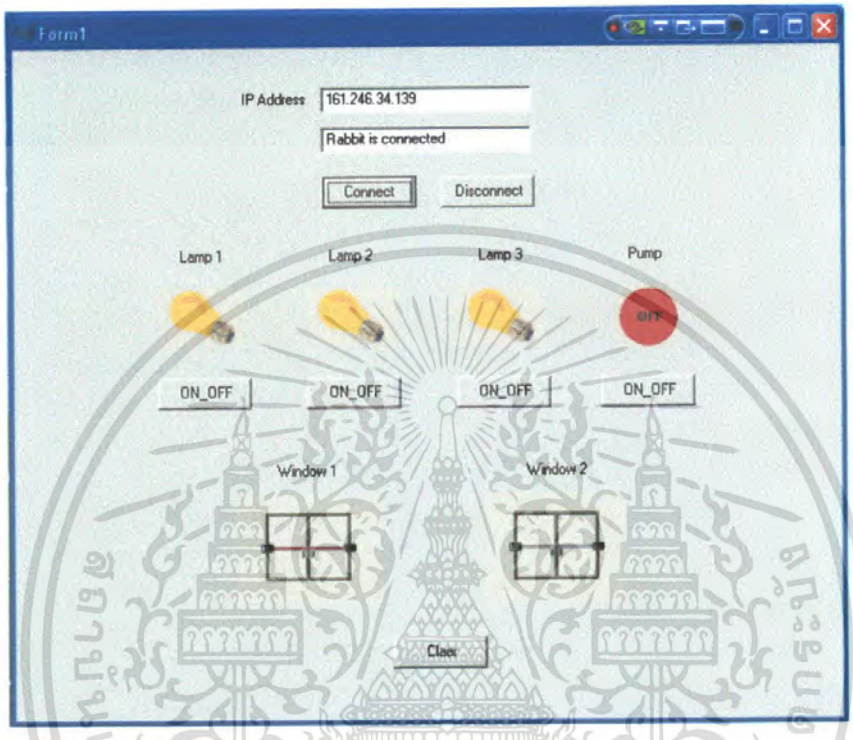
รูปที่ 3.8 Flowchart ตรวจสอบสถานะของหลอดไฟและปั๊มน้ำของ Rabbit3000

การทำงานของโปรแกรม Check_Status_Send คือ เมื่อมีการกดเปิด Switch จากที่บ้านหรือที่หน้าต่างควบคุมในรูปที่ 3.5 โปรแกรมนี้ก็จะตรวจจับได้จากพอร์ต PB5, PE7, PE5, PE4, ด้วยวงจรในรูปที่ 3.6 แล้วจะส่ง Code ตามตารางที่ 3.3 นี้ออกไปให้ Client ที่ขอรับบริการจาก Rabbit3000 เมื่อมีการ Connect

ตารางที่ 3.3 แสดงความหมายของ Code ต่าง ๆ ที่Rabbit3000 จะส่งเพื่อแจ้งสถานะอุปกรณ์

สถานะ	Lamp1	Lamp2	Lamp3	Pump
ON	216	218	220	222
OFF	217	219	221	223

3.3.3 การเขียนโปรแกรม แสดงสถานะหลอดไฟและปั๊มน้ำจาก Network โดย C++Builder6

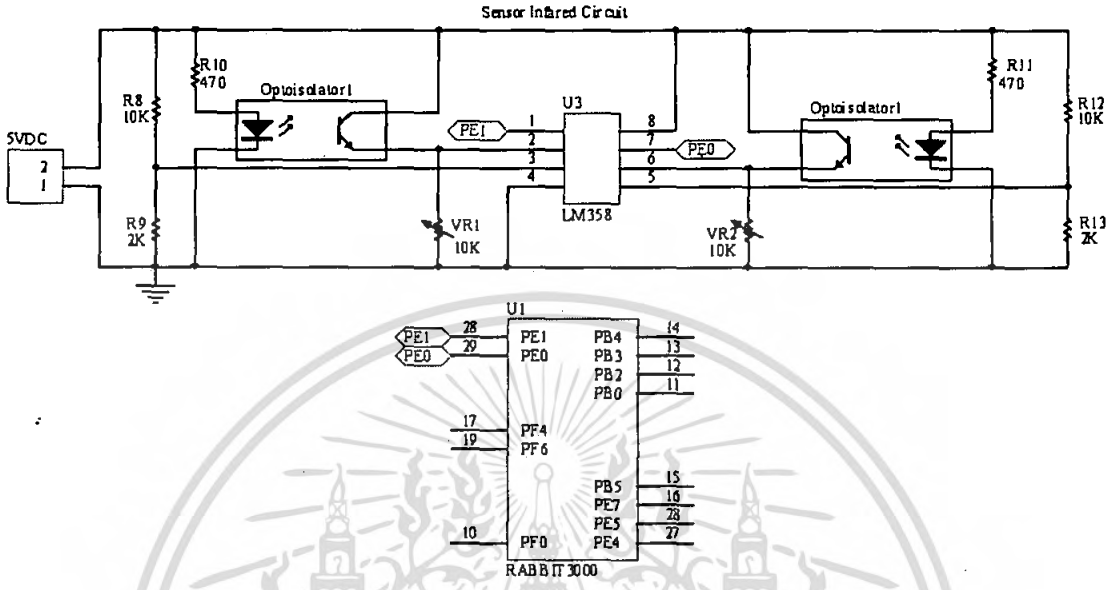


รูปที่ 3.9 แสดงภาพหน้าต่างควบคุมระยะไกล บอกสถานการณ์ทำงานของอุปกรณ์ปลายทาง

จากรูปเป็นรูปที่แสดงผลของการส่งค่าต่าง ๆ ที่ทาง Rabbit 3000 (Server) ส่งมาให้เมื่อมีการ Connect และเมื่อมีการเปลี่ยนสถานะของหลอดไฟ ในรูปนี้อยู่ในสถานะ Connect และหลอดไฟหลอดที่ 1, หลอดที่ 2, หลอดที่ 3, และสถานะของเซนเซอร์ที่หน้าต่างที่ 1 ซึ่งจริง ๆ แล้วจากรูปการที่หลอดไฟทั้งสามหลอดติดหมายความว่า ขณะนี้มีผู้บุกรุก ทางหน้าต่างที่ 1 ซึ่งการทำงานก็คือ เมื่อมีคนตัดผ่านสำแสงอินฟาเรดที่หน้าต่างที่ 1 ขั้นตอนแรกของการรักษาความปลอดภัยก็คือ หลอดไฟทั้งสามหลอดจะติดพร้อมกันทั้งหมด หมายความว่าเจ้าของบ้านทราบว่ามีขโมยขึ้นบ้าน ทำให้ขโมยอาจจะตกใจแล้ววิ่งหนีไปก็ได้ ขั้นตอนที่ สองคือจะมีเสียงสัญญาณกันขโมยดังขึ้นเพื่อให้เพื่อนบ้านทราบจะได้ช่วยกันสอดส่องว่าเกิดอะไรขึ้น ขั้นตอนที่ สาม Server จะส่งค่ากลับมาที่หน้าต่างควบคุมทราบว่ามีการตัดผ่านลำแสงอินฟาเรดที่หน้าต่างที่เท่าไหร่ เจ้าของบ้านที่อยู่ที่อื่นจะได้ดำเนินการแก้ไขต่อไปได้ เมื่อเหตุการณ์สงบลงเจ้าของบ้านสามารถที่จะ กดปุ่ม Clear เพื่อกลับสู่การทำงานปกติของหน้าต่างควบคุม

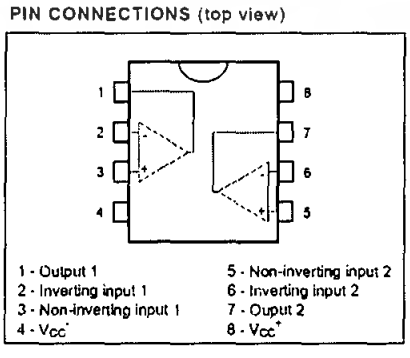
3.4 การตรวจจับผู้บุกรุกด้วยเซนเซอร์แสง

3.4.1 วงจรตรวจจับผู้บุกรุกด้วยลำแสงอินฟาเรด



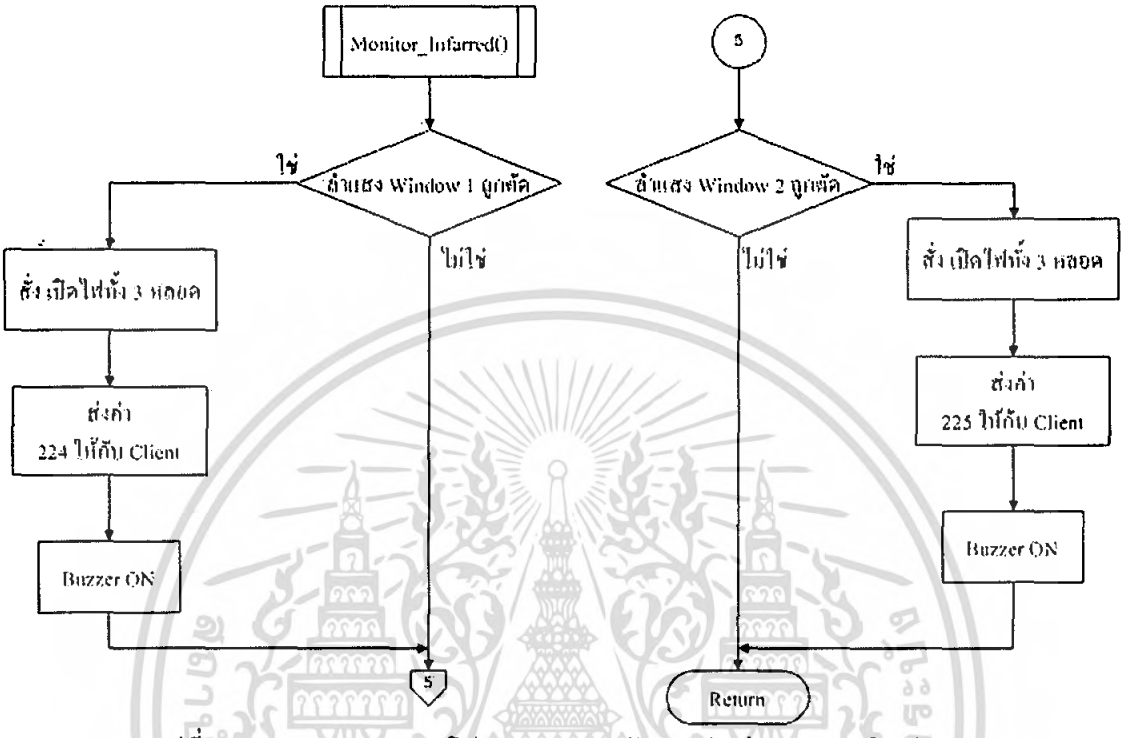
รูปที่ 3.10 วงจรตรวจสอบการตัดผ่าน ลำแสง Infrared ที่หน้าต่าง

ใช้รูปวงจรเสมือนของ Optoisolator1, 2 เพื่อแสดงให้เห็นถึงลักษณะการต่อวงจร ซึ่งในความเป็นจริงเราจะใช้ LED ในย่านรังศี อินฟาเรด เป็นตัวกำเนิดแสงให้กับ Opto Transistor NPN เพราะจะทำให้ได้ระยะการจรวจจับที่ไกลมากขึ้น โดยเมื่อ Transistor ได้รับแสงมาตกกระทบที่บริเวณรอยต่อ ระหว่างขา E กับขา C จะทำให้ Transistor อยู่ในสถานะ ON และมีแรงดันมาตกคร่อมที่ VR1 10K ค่า ๆ หนึ่งซึ่งต่ออยู่กับขาที่ 2 (Non inverting) ของ LM358 ส่วนแรงดันที่ขา 3 (Inverting) จะมีแรงดันอ้างอิงอยู่ค่า ๆ หนึ่งเช่นกัน โดยปกติแรงดันที่ ขา 2 จะมากกว่าขาที่ 3 ทำให้ที่ Output ของ Op-Amp มีค่าเป็น 0 โวลต์ แต่เมื่อมีการตัดผ่านลำแสง แรงดันที่ขา 2 จะน้อยกว่า แรงดันที่ขา 3 ทำให้ Output ของ Op-Amp เปลี่ยนเป็น 5 โวลต์ หรือ Logic "1" ส่งให้Rabbit 3000 ต่อไป



รูปที่ 3.11 โครงสร้างภายในของ IC LM358

3.4.2 การเขียนโปรแกรมตรวจจับผู้บุกรุก โดยการตัดผ่านลำแสงอินฟาเรด Server (Rabbit 3000)

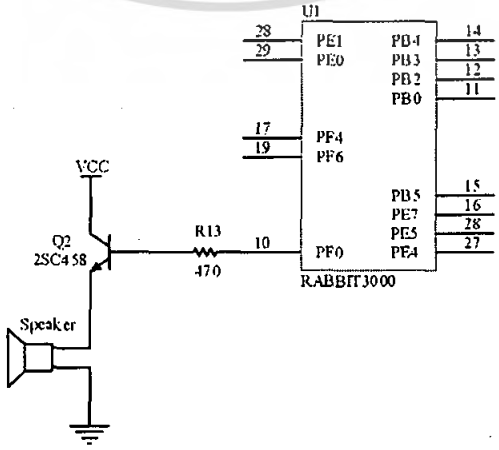


รูปที่ 3.12 Flowchart ของโปรแกรมตรวจจับการตัดผ่านลำแสงอินฟาเรด

จากวงจรในรูปที่ 3.10 Port PE1, PE0, จะได้รับ Logic"1" เมื่อมีคนตัดผ่านลำแสงอินฟาเรด โปรแกรมจะสั่งเปิดไฟทั้งสามหลอด แล้วส่งค่า 224 หรือ 225 ให้กับ Client เมื่อมีการ Connect จากนั้น ก็จะเข้าสู่ฟังก์ชั่น ของการกำเนิดเสียง เพื่อเตือนภัยให้กับเจ้าของบ้านได้ทราบว่าขณะนี้มีผู้บุกรุกคำสั่งที่ใช้ในการตรวจสอบการเปลี่ยนแปลงระดับบิท คือ

BitRdPortI (PEDR, 0)

เป็นฟังก์ชั่นของการอ่านค่า PE0 ซึ่งค่าที่ Return ออกมาคือสถานะของ PE0



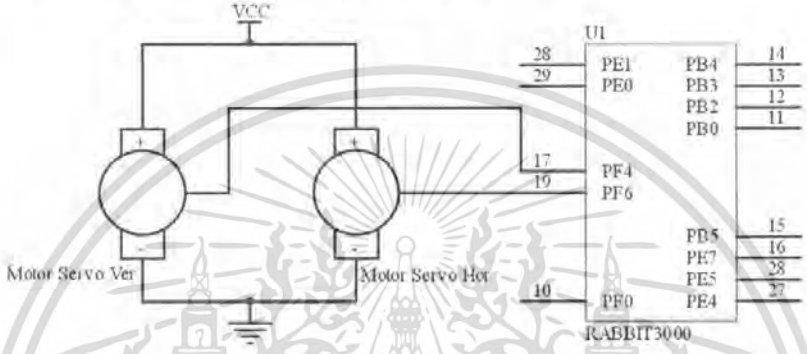
รูปที่ 3.13 วงจรกำเนิดเสียงเตือนภัย

3.4.3 การเขียนโปรแกรมแสดงสถานะการตัดผ่านลำแสงอินฟาเรด Client

ในการเขียนโปรแกรมเพื่อแสดงสถานการณ์ตัดผ่านลำแสงอินฟาเรดที่หน้าต่าง ถูกแสดงและอธิบายในหัวข้อที่ 3.3.3 แล้ว

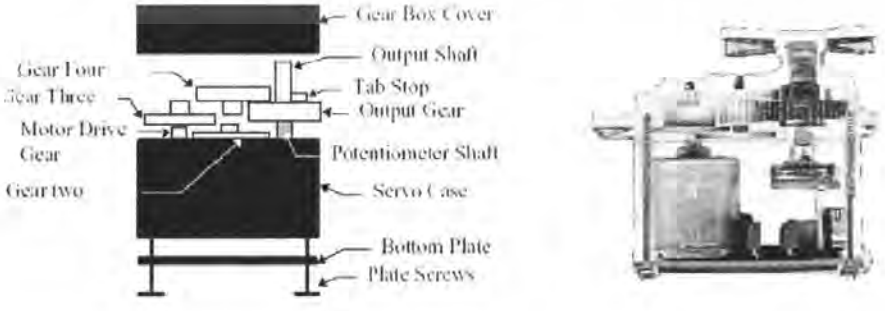
3.5 การควบคุม Servo Motor

3.5.1 วงจรควบคุมการหมุน Servo Motor 2 แกน



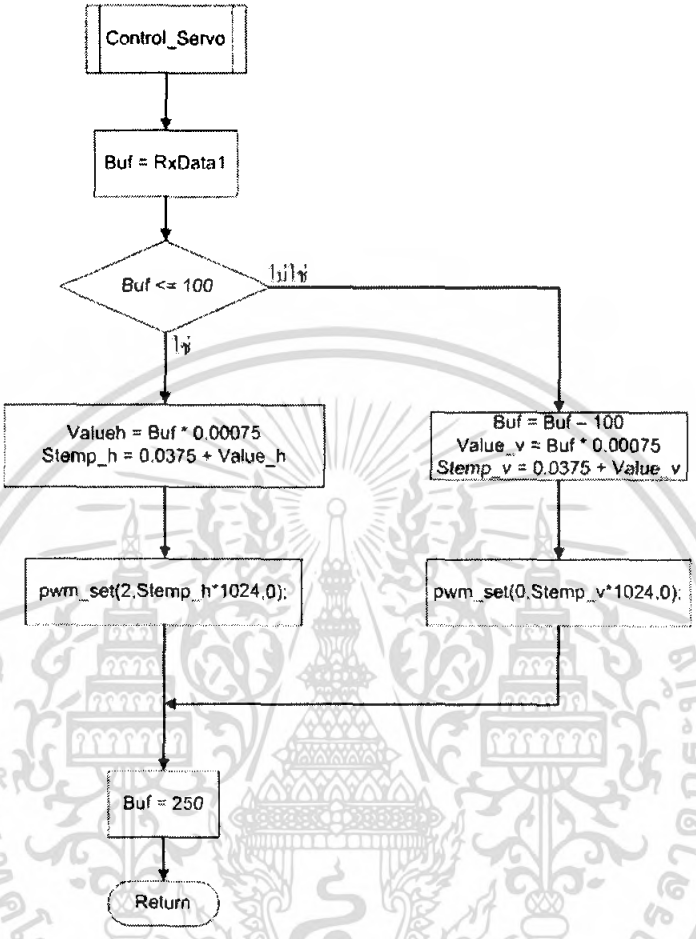
รูปที่ 3.14 วงจรควบคุมการหมุนของ Servo Motor

จากวงจร Rabbit 3000 จะสร้างสัญญาณ พัลส์ ความถี่ 50 Hz ออกมาที่ PE4, PE6, เพื่อควบคุมการหมุน Motor Servo ทั้งสองตัว โดยใช้โมดูล Pulse Width Modulation ภายในRabbit 3000 สร้างสัญญาณ Pulse ที่มี Duty Cycle เปลี่ยนแปลงแต่ว่าความถี่ยังคงเดิมได้ซึ่งเป็นสิ่งที่ดี ที่ไม่ต้องใช้งาน CPU มาก ตามทฤษฎีของการหมุน Motor Servo เราต้องสร้างพัลส์ ความกว้าง 1 ms เมื่อต้องการหมุนไปที่ตำแหน่ง -90 องศา ป้อนพัลส์ความกว้าง 2 ms เมื่อต้องการให้หมุนไปที่ตำแหน่ง +90 องศา แต่ถ้าต้องการให้อยู่ที่ตำแหน่ง 0 องศา ก็ป้อนพัลส์ที่มีความกว้าง 1.5 ms เนื่องจากว่าภายใน Motor Servo มีวงจร Drive อยู่ภายในแล้ว ดังนั้นเราจึงไม่จำเป็นที่จะต้องต่อวงจรภาค Drive ให้กับ Motor Servo อีก สามารถป้อนสัญญาณพัลส์ควบคุมได้โดยตรงเลย



รูปที่ 3.15 โครงสร้างภายใน Motor Servo

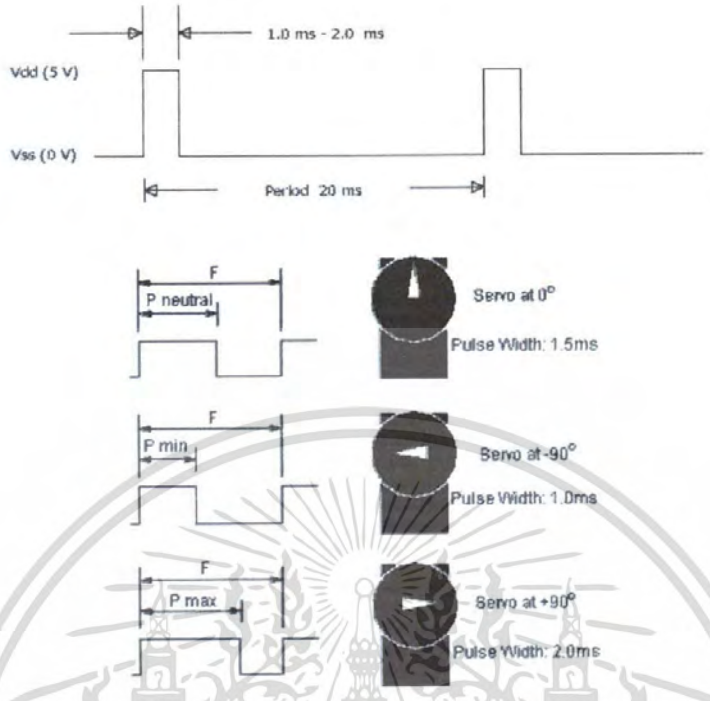
3.5.2 การเขียนโปรแกรมควบคุมการหมุนของ Motor Servo (Rabbit 3000)



รูปที่ 3.16 Flowchart ของโปรแกรมควบคุมการหมุนของ Motor Servo

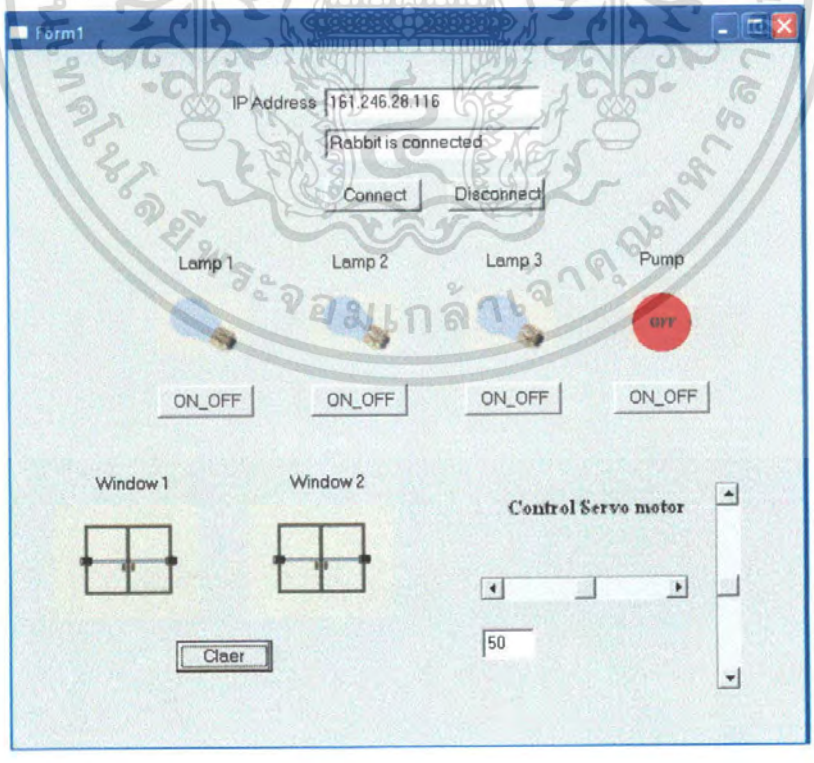
ตัวแปรที่ชื่อ Buf เป็นตัวแปรที่ประกาศขึ้นเพื่อ เก็บค่าของข้อมูลที่ถูกส่งมาจาก Client แล้วนำมาประมวลผลเพื่อสร้างเป็นสัญญาณความกว้างพัลส์ค่าต่าง ๆ ที่ใช้ควบคุมการทำงานของ Motor Servo ซึ่งโดยทั่วไปแล้ว จะมีตำแหน่งอ้างอิงความกว้างของพัลส์อยู่ 3 จุด ดังแสดงในรูปที่ 3.17

ดังนั้นข้อมูลที่มีค่าน้อยกว่า 100 จะใช้ควบคุมตำแหน่งของ Motor Servo ในแกนนอนส่วน ข้อมูลที่มากกว่า 100 จนถึง 200 จะใช้ควบคุมตำแหน่งของ Motor Servo ในแกนตั้ง เนื่องจาก Servo หมุนได้ 180 องศา ข้อมูลที่ใช้ควบคุมจึงมีความละเอียด 1.8 องศาต่อ 1 ค่า และการคูณ ค่า Buf ด้วยค่า 0.00075 นั้นก็เพราะต้องคิดความละเอียดของพัลส์ออกมาเป็นร้อยละ ของความกว้างของคาบเวลา แล้วบวกกับค่า 0.0375 ซึ่งคือค่าเริ่มต้นของตำแหน่งที่ 0 องศา แล้วนำไปกำหนดในฟังก์ชันของ PWM ที่มีความละเอียด 10 บิท



รูปที่ 3.17 ตำแหน่งของ Motor Servoกับความกว้างของสัญญาณพัลส์

3.5.3 การเขียนโปรแกรมควบคุมการหมุนของ Servo Motor จากระบบ Network



รูปที่ 3.18 แสดงหน้าต่างควบคุมที่เพิ่มส่วนการควบคุมตำแหน่ง Motor Servo ทั้งสองตัว

ใช้ Scroll Bar เป็นตัวส่งค่าตำแหน่งในการควบคุม Motor Servo ซึ่งในแต่ละแกนจะสามารถส่งค่าได้ตั้งแต่ 0 จนถึง 100 แล้วทางด้าน Server ก็จะนำค่าที่ได้ไปคำนวณเพื่อสร้างเป็นความกว้างสัญญาณพัลส์ต่าง ๆ แล้วส่งไปควบคุมการหมุนของ Servo อีกทีหนึ่ง

บทที่ 4

วิธีการทดลองและผลการทดลอง

4.1 อุปกรณ์ที่ใช้ในการทดลอง

- Router	1 ตัว
- สาย LAN	2 เส้น
- คอมพิวเตอร์	1 เครื่อง
- Rabbit 3000 Core Modules RCM 3720	1 เครื่อง

4.2 การควบคุมอุปกรณ์ไฟฟ้าด้วยสวิตช์ที่บ้าน

4.2.1 วิธีการทดลอง

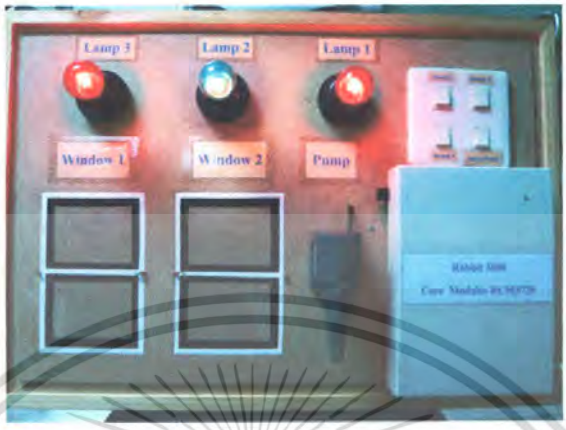
1) ต่ออุปกรณ์ต่างๆดังรูป 4.1

รูปที่ 4.1 แสดงการต่อวงจรใช้งาน

2) เสียบปลั๊ก Rabbit3000 แล้วเปิด Switch หลังจากนั้นทำการกดสวิตช์เพื่อเปิดหลอดไฟ คือ Lamp1, Lamp2, Lamp3 และ Pump สังเกตผลที่เกิดขึ้น

3) ปิดสวิตช์

4.2.2 ผลการทดลอง



รูปที่ 4.2 ผลการทดลองเปิด – ปิด Switch จากที่บ้าน

เมื่อทำการเปิด Switch ทั้ง 4 ตัวผลปรากฏว่าหลอดไฟและปั๊มน้ำทำงานทั้งหมดแล้วเมื่อทำการปิด Switch หลอดไฟและปั๊มน้ำก็หยุดทำงานซึ่งจะเห็นได้จากรูปที่ 4.2

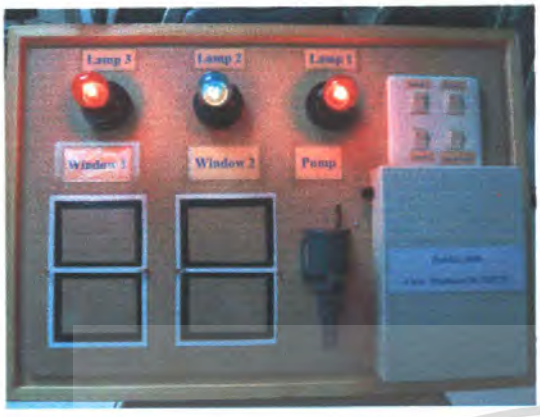
4.3 การควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายคอมพิวเตอร์

4.3.1 วิธีการทดลอง

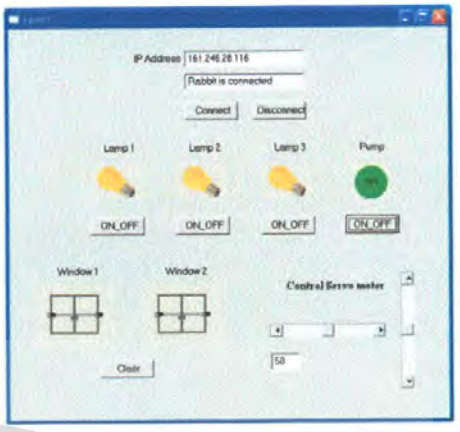
1) เปิดคอมพิวเตอร์ และ RABBIT 3000 Core Modules RCM 3720 แล้วทำการเชื่อมต่ออินเตอร์เน็ตจากคอมพิวเตอร์ และเปิดโปรแกรมควบคุมอุปกรณ์ไฟฟ้าผ่าน เครือข่ายคอมพิวเตอร์ (โปรแกรมที่เขียนขึ้นด้วย C++ Builder 6)
2) กดปุ่ม Connect หน้าจอจะแสดงผล “Rabbit is connecte ” ทดสอบการทำงานโดยการกดปุ่ม ON_OFF ของ Lamp1, Lamp2, Lamp3 และ Pump สังเกตุผลจากแบบจำลองและหน้าจอแสดงผลของโปรแกรม

4.3.2 ผลการทดลอง

เมื่อกำหนดหมายเลข IP Address แล้วกด Connect จะสังเกตเห็นว่าที่หน้าต่างในการคอลโทรลผ่านระบบ Network ในรูปที่ 4.3 (ข) จะแสดงคำว่า “Rabbit is connect” หมายความว่าการติดต่อสำเร็จแล้ว ต่อไปเมื่อทำการกดปุ่ม ON_OFF ของ Lamp1, Lamp2, Lamp3, และ Pump ผลปรากฏว่าหลอดไฟทั้ง 3 หลอดและปั๊มน้ำนั้นทำงานทั้งหมด ดังจะเห็นได้จากในรูปที่ 4.3 (ก) ในขณะเดียวกันนั้นที่ หน้าต่างควบคุมระยะไกลนั้นก็แสดงผลเป็น ON ทั้งหมดเช่นกัน



(ก)



(ข)

รูปที่ 4.3 แสดงผลการควบคุมจากระบบ Network

4.4 การควบคุมอุปกรณ์ไฟฟ้าทั้งสองระบบพร้อมกันและ การรักษาความปลอดภัยภายในบ้านผ่านเครือข่ายคอมพิวเตอร์

4.4.1 วิธีการทดลอง

1) จากการทดลองในหัวข้อที่ 4.3 เมื่อทำการกดปุ่ม Disconnect หน้าจอจะแสดงผล “Rabbit is Disconnect”

2) เปิดสวิตช์ไฟจากแบบจำลอง โดยกดสวิตช์ Lamp1, Lamp 2

3) กดปุ่ม Connect สังเกตุผลที่เกิดขึ้นบนหน้าจอแสดงผล

4) กดปุ่ม ON_OFF ของ Lamp1 และ Lamp2 ที่ หน้าจอแสดงผลสังเกตุการเปลี่ยนแปลงของหลอดไฟและหน้าต่างควบคุม

5) จำลองการผ่านเซ็นเซอร์ที่ติดตั้งไว้ที่หน้าต่างจำลอง (Window1) โดยการใช้มือตัดผ่านหรือบังเซ็นเซอร์สังเกตุผลที่แบบจำลองและหน้าจอแสดงผลของโปรแกรม

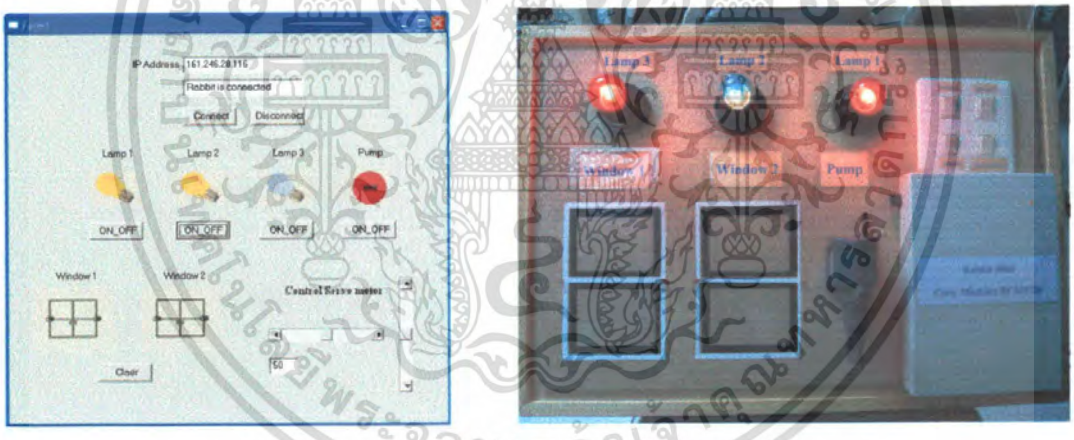
6) ทำการ Disconnect และจำลองการผ่านเซ็นเซอร์ที่ติดตั้งไว้ที่หน้าต่างจำลอง (Window2) โดยการใช้มือตัดผ่านหรือบังเซ็นเซอร์ หลังจากนั้นทำการ Connect อีกครั้งสังเกตุผลที่แบบจำลองและหน้าจอแสดงผลของโปรแกรม

7) กดปุ่ม Clear สังเกตุการเปลี่ยนแปลง

4.4.2 ผลการทดลอง

(ก) (ข)

รูปที่ 4.4 เมื่อเปิดไฟจากสวิตช์ปกติขณะที่ยังไม่ Connect

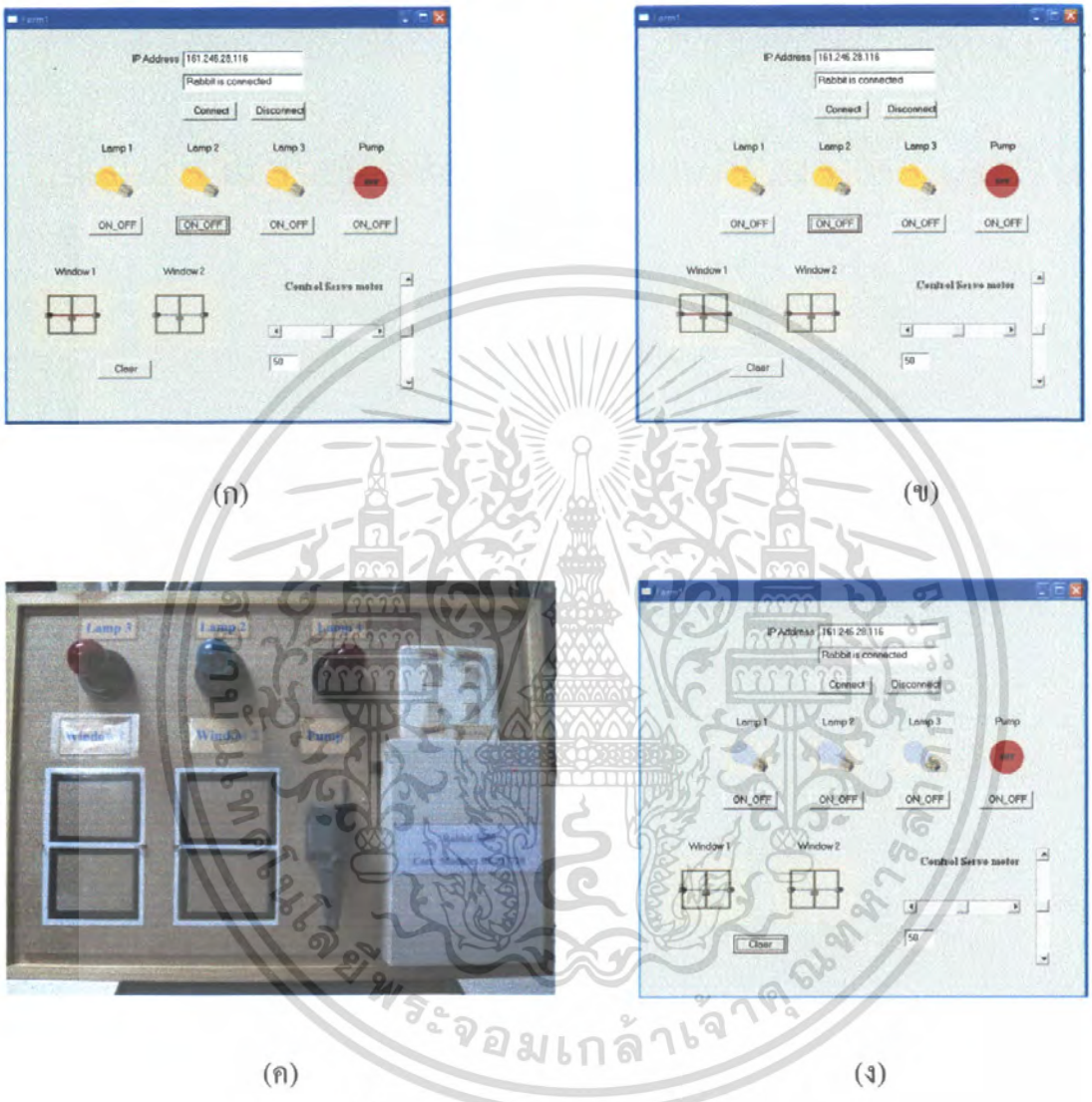


(ก) (ข)

รูปที่ 4.5 แสดงผลการทดลองเมื่อทำการ Connect อีกครั้ง

จากการทดลองในข้อที่ 1 เมื่อเรากดปุ่ม Disconnect จะเห็นข้อความแสดงในช่องด้านล่างช่องหมายเลข IP Address ว่า “Rabbit is Disconnect” แสดงได้ดังในรูปที่ 4.4 (ก) จากนั้นเมื่อเปิด Switch Lamp1, Lamp2 ซึ่งจะเห็นว่าหลอดไฟก็ติด 2 หลอด ดังแสดงในรูปที่ 4.4 (ข) รูปที่ 4.5 (ก) แสดงให้เห็นว่าเมื่อเรา Connect เข้ามาดูอีกครั้ง ก็จะเห็นว่าหลอดไฟ Lamp1, Lamp2 ติดอยู่นอกจากนั้นเมื่อทำการ กดปุ่ม ON_OFF Lamp3 ผลปรากฏว่า Lamp3 ที่บอร์ดแสดงผลในรูปที่ 4.5 (ข) ON จากนั้นจำลองว่ามีผู้บุกรุกทาง Window1 เราจะเห็นว่าที่ Window1 จะมีแถบสีแดงตรงกลางโชว์ขึ้นมา ดังในรูปที่ 4.6 (ก) เช่นเดียวกันเมื่อมีผู้บุกรุกทาง Window2 มันก็จะแสดงเหมือนในรูปที่ 4.6

(ข) เมื่อกดปุ่ม Clear จะเห็นว่าหลอดไฟทุกหลอดดับหมดและที่หน้าต่างแสดงผลก็ OFFหมด เช่นกันดังแสดงในรูปที่ 4.6 (ค) และ(ง) ตามลำดับ



(ก) (ข)

(ค) (ง)

รูปที่ 4.6 แสดงผลการทดลองเมื่อมีผู้บุกรุก

4.5 การควบคุม Motor Servo ผ่าน Network

4.5.1 วิธีการทดลอง

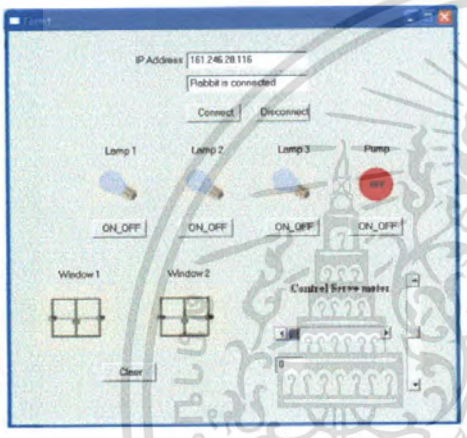
1) เปิดหน้าต่าง ควบคุมแล้วกำหนดหมายเลข IP Address ลงในช่องที่กำหนด แล้วกด Connect รอจนกว่าจะขึ้นคำว่า “Rabbit is Connect”
2) ทดลองปรับ Scroll Bar ของส่วนควบคุม Servo แกนนอนไปที่ตำแหน่ง ซ้ายสุดแล้วสังเกตผลที่เกิดขึ้น
3) ทดลองปรับ Scroll Bar ของส่วนควบคุม Servo แกนนอนไปที่ตำแหน่ง

ขวาสุดแล้วสังเก็ตผลที่เกิดขึ้น

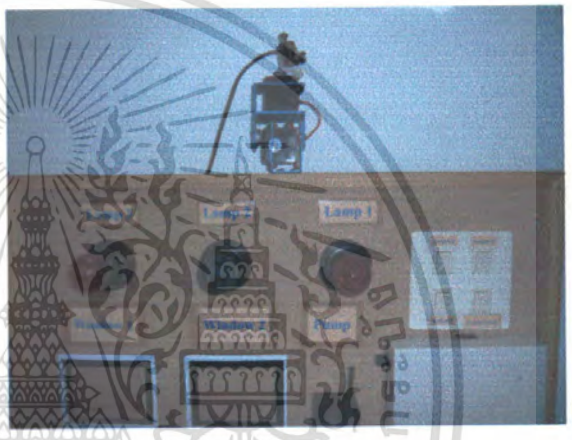
4) จากนั้นทดลองปรับ Scroll Bar ของส่วนควบคุม Servo แกนตั้งไปที่ตำแหน่งบนสุดแล้วสังเก็ตผลที่เกิดขึ้น

5) จากนั้นทดลองปรับ Scroll Bar ของส่วนควบคุม Servo แกนตั้งไปที่ตำแหน่งล่างสุดแล้วสังเก็ตผลที่เกิดขึ้น

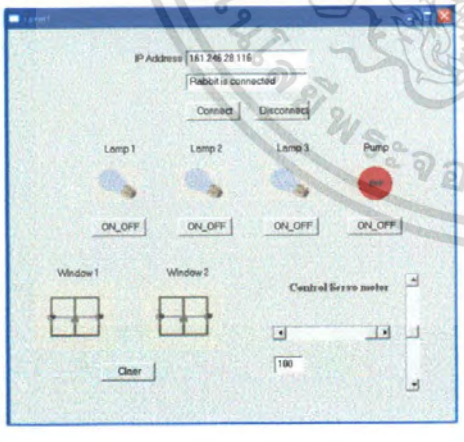
4.5.2 ผลการทดลอง



(ก)

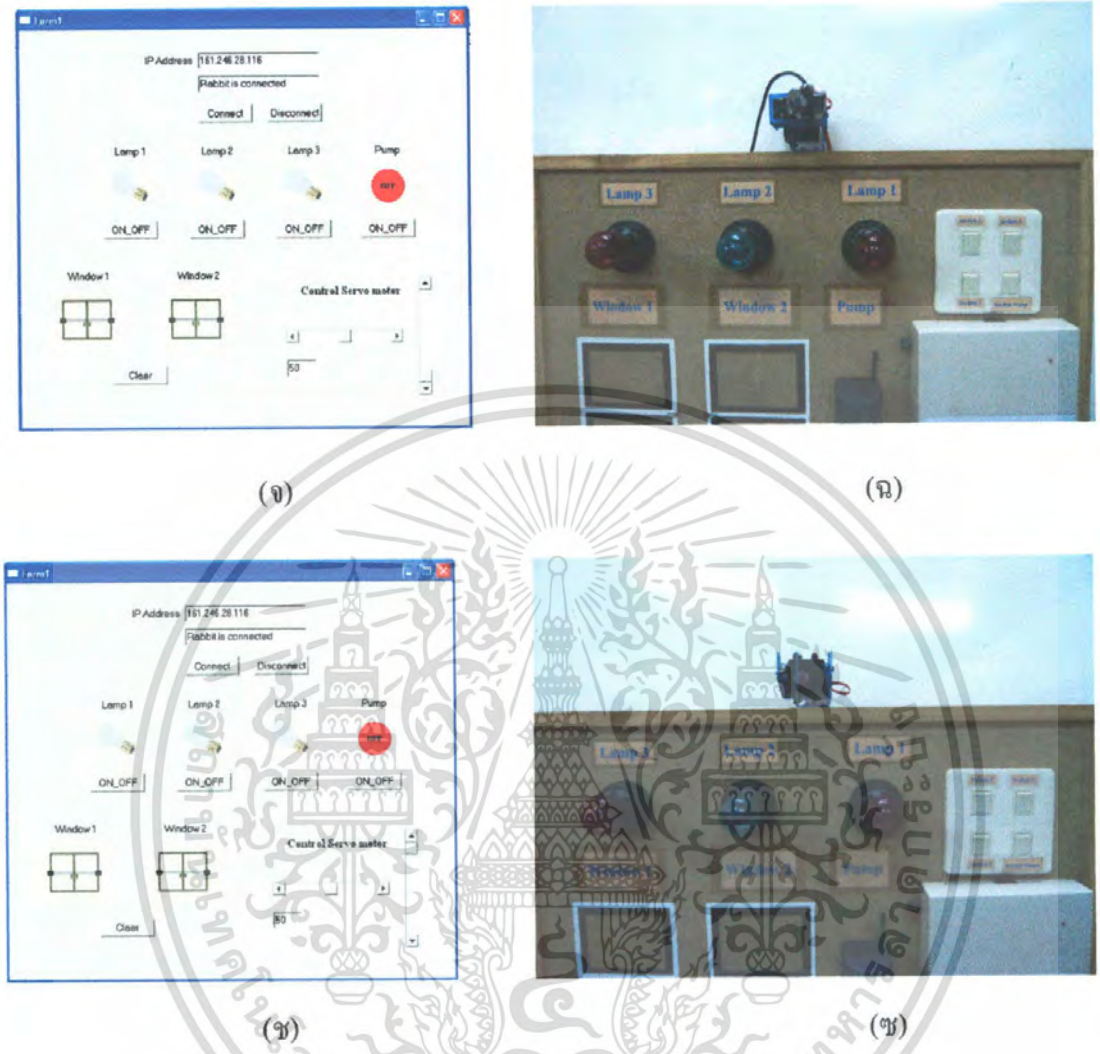


(ข)



(ค)

(ง)



(ข) (ฉ)

(ช) (ซ)

รูปที่ 4.7 แสดงผลการควบคุม Motor Servo จากหน้า Control

จากผลการทดลองจะเห็นว่าการปรับมุมมองของกล้องสามารถทำได้ตั้งแต่ 0 องศา ไปจนถึง 180 องศา ตามคุณสมบัติของ Motor Servo ทั้งแกนนอนและแกนตั้งทำให้ได้มุมมองที่ครอบคลุม พื้นที่ได้ครึ่งวงกลม สามารถที่จะนำไปประยุกต์ใช้ประโยชน์ได้มากมายไม่จำเป็นต้องเกี่ยวกับงานรักษาความปลอดภัยเพียงอย่างเดียวก็ได้

บทที่ 5
สรุปผลการทดลอง

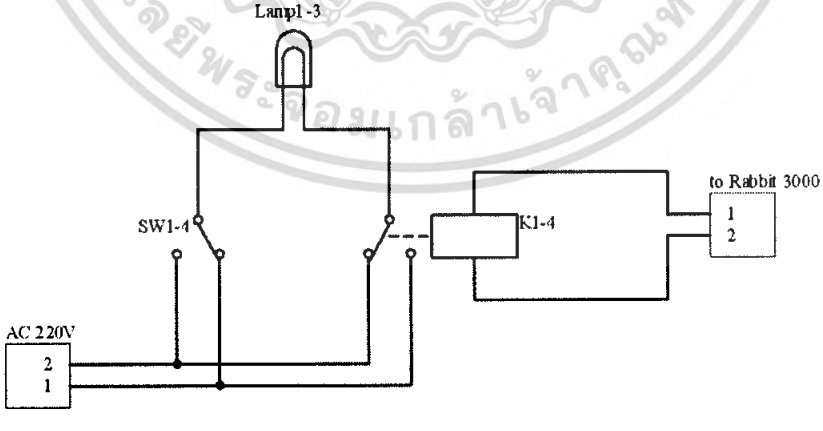
5.1 สรุปผลการทดลอง

การควบคุมอุปกรณ์ไฟฟ้าด้วยสวิตช์ที่บ้าน

เมื่อเปิดสวิตช์ที่แบบจำลองหรือเทียบได้กับการเปิดสวิตช์ที่บ้าน lamp1,lamp2,lamp3และ pump สามารถทำงานได้เปรียบเสมือนเราสามารถควบคุมอุปกรณ์ไฟฟ้าต่าง ๆ ได้ด้วยสวิตช์ โดยไม่ต้องเปิด Rabbit 3000 RCM 3720 เลยซึ่งจะทำให้เราสามารถใช้สวิตช์ควบคุมอุปกรณ์ไฟฟ้าได้ตามเดิม ถึงแม้ว่าRabbit 3000 RCM3720 จะเสียหรือชำรุดก็ตาม

การควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายคอมพิวเตอร์

เป็นการควบคุมอุปกรณ์ไฟฟ้าต่าง ๆ ผ่านระบบเครือข่ายคอมพิวเตอร์โดยใช้โปรแกรมที่เขียนขึ้นมาด้วย C++Builder6 ดังจะเห็นได้จากรูปที่4.6 (ง) เมื่อกดปุ่มConnect ระบบคอมพิวเตอร์จะทำการเชื่อมต่อกับRabbit 3000 RCM 3720 แล้วจะแสดงคำว่า "Rabbit is Connect" เมื่อการเชื่อมต่อระบบสำเร็จ จากนั้นเราสามารถที่จะควบคุมอุปกรณ์ไฟฟ้าได้จากหน้าจอแสดงผลที่คอมพิวเตอร์ปลายทางโดยสามารถควบคุมการ เปิด - ปิด lamp1, lamp2, lamp3 และ pump ได้โดยสะดวกและยังแสดงผลการเปิด – ปิด หลอดไฟได้ด้วยรูป หลอดไฟ ทำให้เราทราบถึงสถานะ การทำงานของอุปกรณ์ไฟฟ้าที่อยู่ที่บ้านได้ด้วย



รูปที่ 5.1 วงจรสวิตช์ 2 ทางระว่างการควบคุมด้วย Rabbit3000 และ SW1-4

เนื่องจากว่าเราได้ออกแบบให้สวิตช์ที่ใช้ควบคุม อุปกรณ์เครื่องใช้ไฟฟ้านั้นเป็นสวิตช์สองทางจึงสามารถที่จะแยกการควบคุมได้อย่างเป็นอิสระตามรูปที่ 5.1

การควบคุมอุปกรณ์ไฟฟ้าทั้งสองระบบพร้อมกันและ การรักษาความปลอดภัย ภายในบ้านผ่านเครือข่ายคอมพิวเตอร์

เมื่อกดปุ่ม Disconnect ระบบคอมพิวเตอร์จะตัดการเชื่อมต่อกับ RCM 3720 หลังจากนั้นเมื่อเปิดสวิตช์ที่แบบจำลองโดยเปิด lamp1, lamp2 หลอดไฟจะติดสว่าง เมื่อเราทำการกดปุ่ม Connect อีกครั้งหน้าจอแสดงผลจะแสดงให้เห็นว่าหลอดไฟ lamp1, lamp2 มีการเปิดอยู่ทำให้เราสามารถทราบถึงสถานะ การเปิดหลอดไฟจากที่แบบจำลองและเมื่อกดปุ่มควบคุมอุปกรณ์ไฟฟ้าที่คอมพิวเตอร์ปลายทางก็สามารถควบคุมหลอดไฟให้เปิด-ปิดได้ตามที่เราต้องการอีกด้วย

เมื่อจำลองการตัดผ่านเซ็นเซอร์ที่หน้าต่างจำลอง(Window1) สัญญาณกันขโมยจะดังขึ้นนานตามเวลาที่กำหนดไว้ตามโปรแกรมการทำงานและหลอดไฟทั้งหมดที่แบบจำลองจะติดสว่างเพื่อป้องกันผู้บุกรุกที่เข้ามาโดยผ่านเซ็นเซอร์และเมื่อสังเกตที่หน้าจอแสดงผลของคอมพิวเตอร์ปลายทางจะแสดงผลมีขีดสีแดงเกิดขึ้นที่Window1และแสดงสถานะหลอดไฟติดสว่างทั้งหมดทำให้ผู้ที่อยู่ปลายทางทราบว่ามีผู้บุกรุกเข้าที่Window1 เมื่อทำการ Disconnect โปรแกรมที่คอมพิวเตอร์ปลายทางแล้วทำการตัดผ่านเซ็นเซอร์Window2 ผลที่เกิดขึ้นคือสัญญาณกันขโมยจะดังขึ้นนานตามเวลาที่กำหนดไว้ตามโปรแกรมการทำงานและหลอดไฟทั้งหมดที่แบบจำลองจะติดสว่างเพื่อป้องกันผู้บุกรุกที่เข้ามาโดยผ่านเซ็นเซอร์อีกครั้ง และเมื่อทำการConnectเข้ามาอีกครั้งที่หน้าจอแสดงผลจะแสดงเส้นขีดสีแดงที่Window2 และหลอดไฟจะติดสว่างทั้งหมดทำให้เราทราบว่ามีผู้บุกรุกเข้าที่Window2

สรุปการทำงานทั้งหมดไม่ว่าจะ Connect หรือ Disconnect RCM 3720 สามารถที่จะแสดงสถานะการทำงานต่าง ๆ ของอุปกรณ์ไฟฟ้าปลายทางได้ และเปลี่ยนแปลงได้อย่างอิสระ หรือ พูดได้ว่าการควบคุมอุปกรณ์ไฟฟ้าไม่ว่าจะควบคุมด้วยสวิตช์ที่แบบจำลองหรือควบคุมที่คอมพิวเตอร์ปลายทางผ่านเครือข่ายคอมพิวเตอร์นั้นเป็นอิสระต่อกัน ทำให้ผู้ใช้งานง่ายสะดวกและยังป้องกันการบุกรุกได้อย่างมีประสิทธิภาพ

การควบคุม Motor Servo ผ่าน Network

เมื่อเราทำการควบคุม Motor Servo ผ่านระบบ Network นั้นเราจะสังเกตเห็นว่าสามารถที่จะควบคุมได้ตั้งแต่ 0 องศา จนถึง 180 องศา จาก Scroll Bar ทั้งในแนวตั้งและในแนวนอน แล้วความเร็วที่ทำการปรับก็เป็นไปตามการลากเมาท์ด้วยทำให้สามารถมองภาพจากจอได้อย่างไม่กระตุกและสามารถมองได้หลายมุมมองมากยิ่งขึ้นอีกด้วย

5.2 ปัญหาและวิธีการแก้ไข

1. เมื่อทำการเปิด-ปิดอุปกรณ์ไฟฟ้าไม่ว่าจะด้วยสวิตช์ปกติที่แบบจำลองหรือผ่านเครือข่ายคอมพิวเตอร์ บางครั้งทำให้สัญญาณกันขโมยดังขึ้นโดยที่เราไม่ต้องการ เนื่องจากเมื่อเปิดปิดไฟทำให้มีการกระเพื่อมของแหล่งจ่ายไฟเล็กน้อยทำให้ส่วนของวงจรอินฟาเรดทำงานส่งค่าที่ผิดพลาดให้กับ Rabbit 3000 จึงมีการประมวลผลผิดพลาดตาม

การแก้ไขก็คือ ใส่คำสั่ง Delay ไว้ก่อนหน้า loop การทำงานของการเช็คสถานะอินฟาเรด

2. การส่งภาพแสดงสถานะหลอดไฟติดที่หน้าจอแสดงผลมีการกระพริบ เนื่องจากการต่อวงจรตรวจสอบสถานะของอุปกรณ์ โดยใช้ไฟ AC ที่ต่อขนานกับอุปกรณ์แล้วทำการแปลงแรงดันเป็น DC เพื่อขับ OPTO แล้วส่งสถานะการทำงานให้กับ RCM 3720 ไม่นิ่งทำให้การส่งค่าให้กับ Rabbit 3000 ผิดพลาด

การแก้ไข ก็คือใส่คำสั่ง Delay ไว้ก่อนหน้า loop การทำงานของการเช็คสถานะหลอดไฟ

3. ตอนที่ทำการ Connect เข้าไปที่ Rabbit 3000 Motor Servo จะทำการรีเซ็ตค่ากลับมาอยู่ที่ตำแหน่ง 0 องศา ทั้งในแกนนอนและแกนตั้ง ทำให้ช่วงที่ Motor Servo ทำงานนั้นกินกระแสมากส่งผลให้ Rabbit 3000 เกิดอาการรีเซ็ตเนื่องจากไฟต่ำกว่าค่าที่จะสามารถทำงานได้

แนวทางในการแก้ไขคือ ทำการ Delay ทางโปรแกรม โดยให้ทำงานทีละแกนเพื่อป้องกันการกินกระแสมากเกินไป แล้วทำการแยกไฟเลี้ยงในส่วนของ Motor Servo กับส่วนของ Rabbit 3000 ออกจากกันซึ่งก็สามารถช่วยได้ในระดับหนึ่ง

5.3 แนวทางการนำไปประยุกต์ใช้งานอื่น ๆ

1. สามารถนำไปควบคุมอุปกรณ์ไฟฟ้าเช่นการ เปิด-ปิด แสงสว่าง ในสถานที่อันตราย
2. ควบคุมการ เปิด – ปิด ปั้มน้ำเพื่อรดน้ำต้นไม้ในสวนได้จากที่บ้าน ถึงแม้ว่าเราจะ
3. ใช้เพื่อตรวจสอบว่าลืม ปิด อุปกรณ์เครื่องใช้ไฟฟ้าตอนออกจากบ้านหรือไม่ถ้าลืมก็สามารถสั่งปิดได้

4. สามารถนำไปประยุกต์ใช้ในงานรักษาความปลอดภัยอื่น ๆ ได้

บรรณานุกรม

1. จักกริช พฤษการ, “การสื่อสารข้อมูลและเครือข่ายคอมพิวเตอร์”: บริษัท สำนักพิมพ์ท้อป จำกัด, 2549

2. ทีมงานสมาร์ทเลิร์นนิ่ง,”การออกแบบลายวงจรพิมพ์ด้วย Protel DXP”: ห้างหุ้นส่วนสามัญ สมารทเลิร์นนิ่ง, 2549

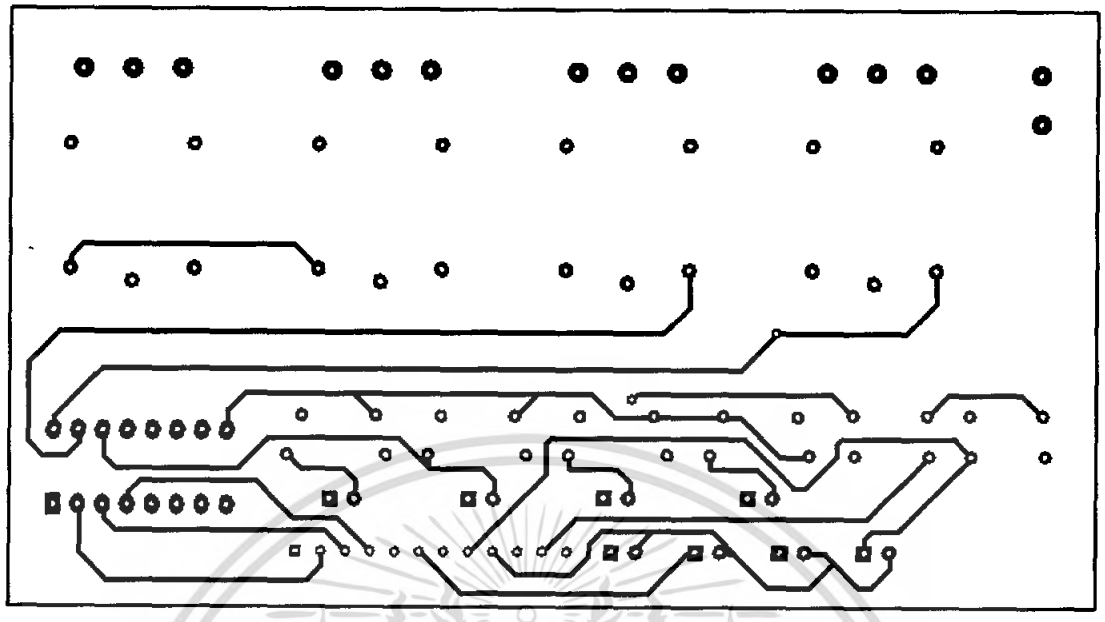
3. นุกูล กระจาย “C++Builder 5” พิมพ์ครั้งที่ 1 กรุงเทพฯ: สุวีริยาสาสน์, 2544

4. สุวัฒน์ ปุณณะชัยยะ, “เปิดโลกของTCP/IP และโปรโตคอลของอินเตอร์เนท” ,พิมพ์ครั้งที่1 กรุงเทพ: โปรวิชั่น, 2543

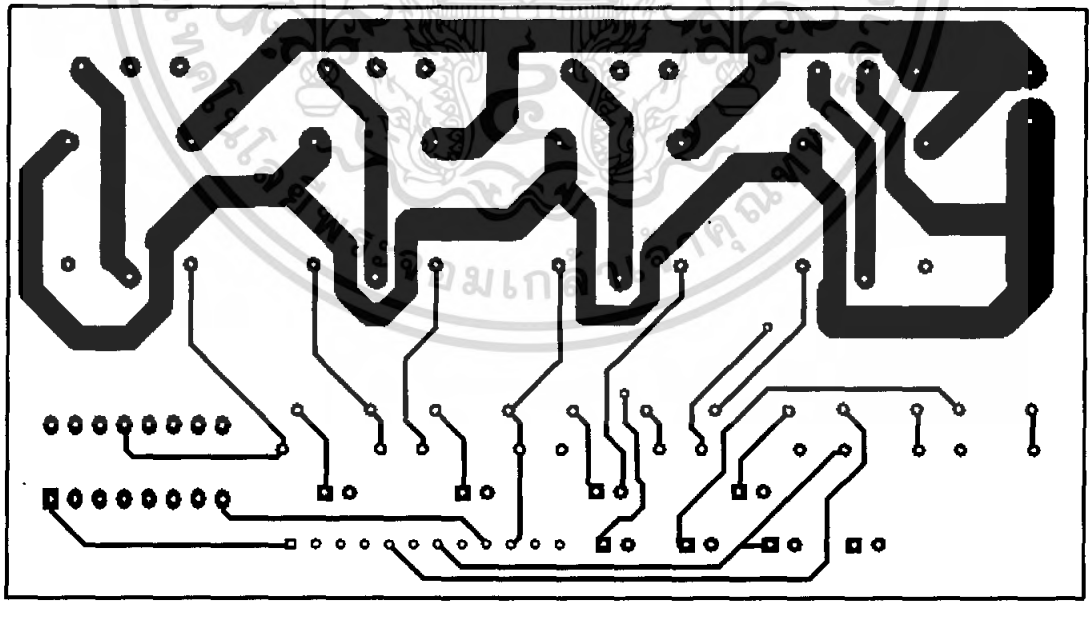
5. อุดม รานอก, “ภาษา C สำหรับงานควบคุมไมโครคอนโทรลเลอร์”, พิมพ์ครั้งที่1นนทบุรี: ไอดีซี ฯ , 2548

6. Z-World, “Dynamic C” [CD-ROM], Z-World, 1999.

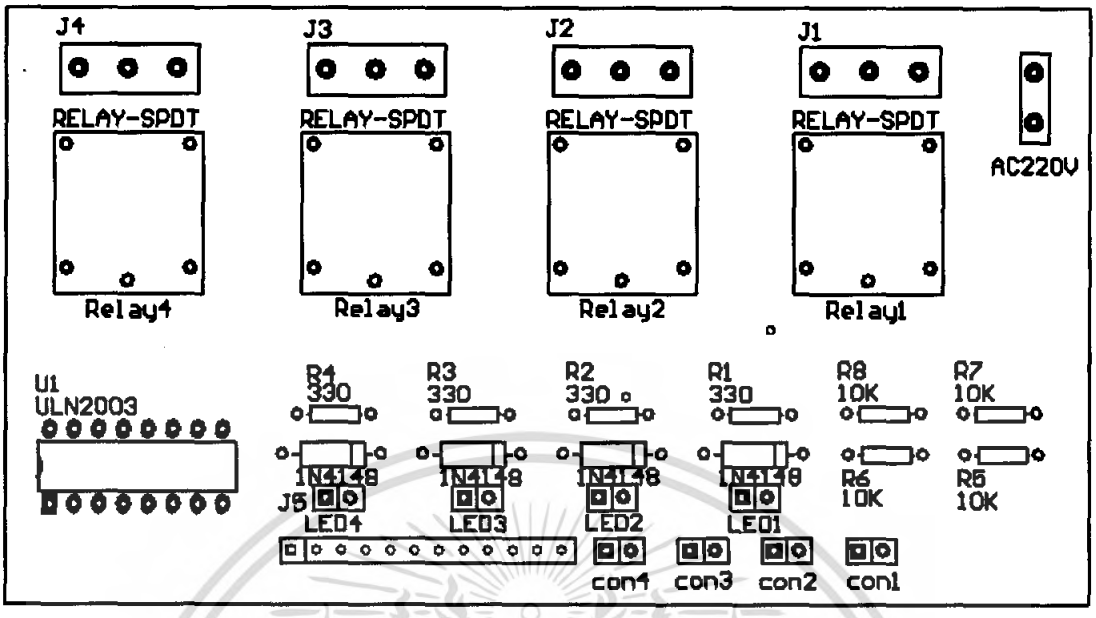
ภาคผนวก



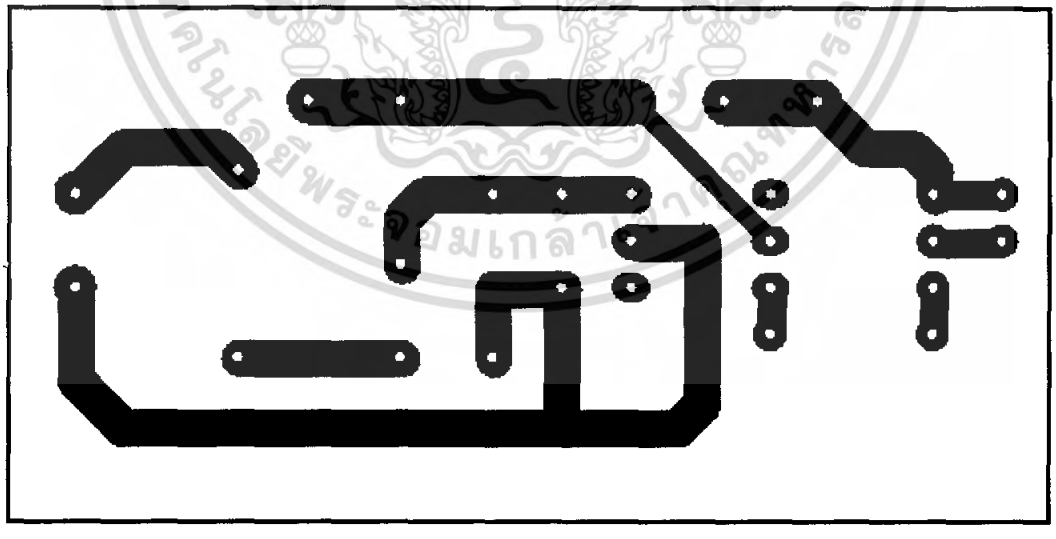
แสดงภาพ PCB Top layer ของวงจรควบคุมการ เปิด-ปิด หลอดไฟและปั๊มน้ำ



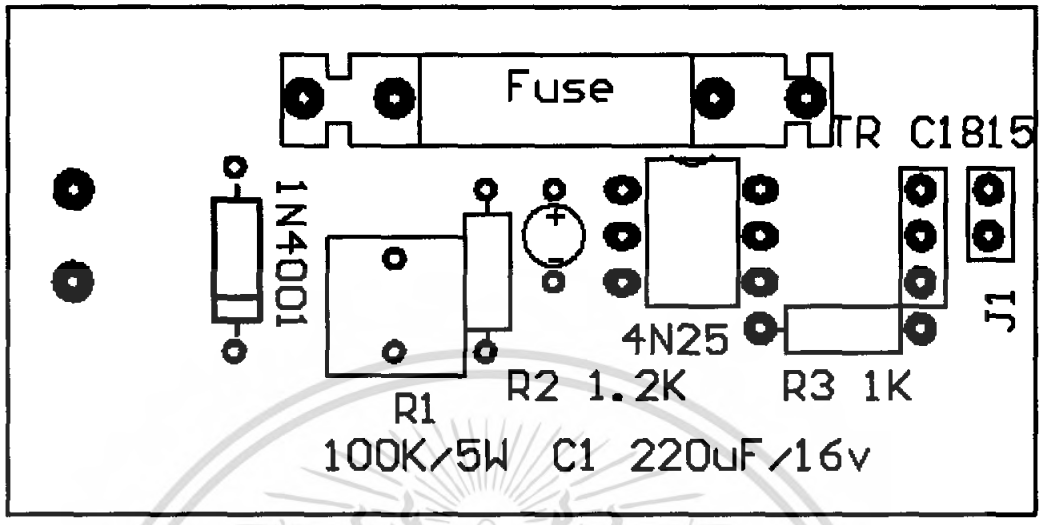
แสดงภาพ PCB Bottom layer ของวงจรควบคุมการ เปิด-ปิด หลอดไฟและปั๊มน้ำ



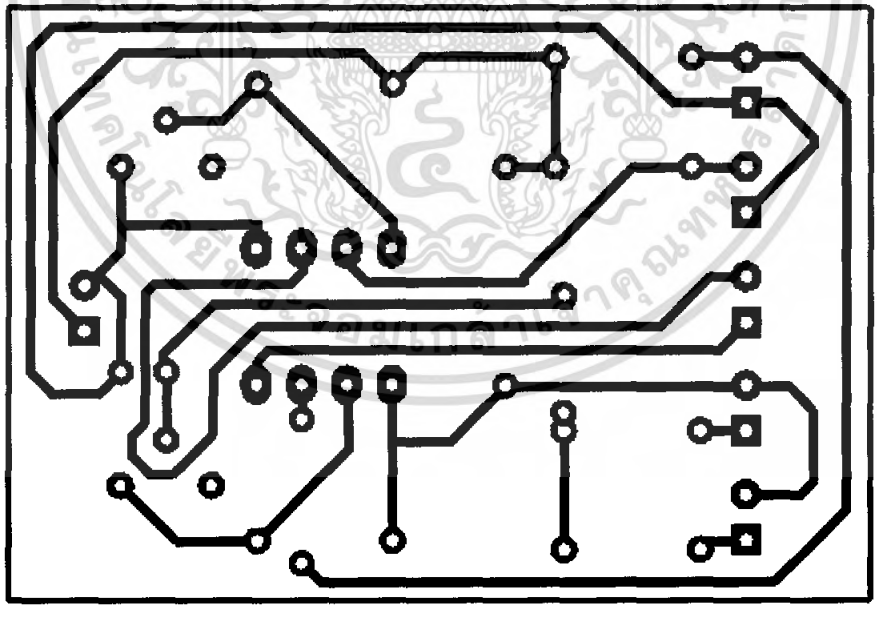
แสดงการวางอุปกรณ์บน PCB วงจรควบคุมการเปิด-ปิด หลอดไฟและปั้มน้ำ



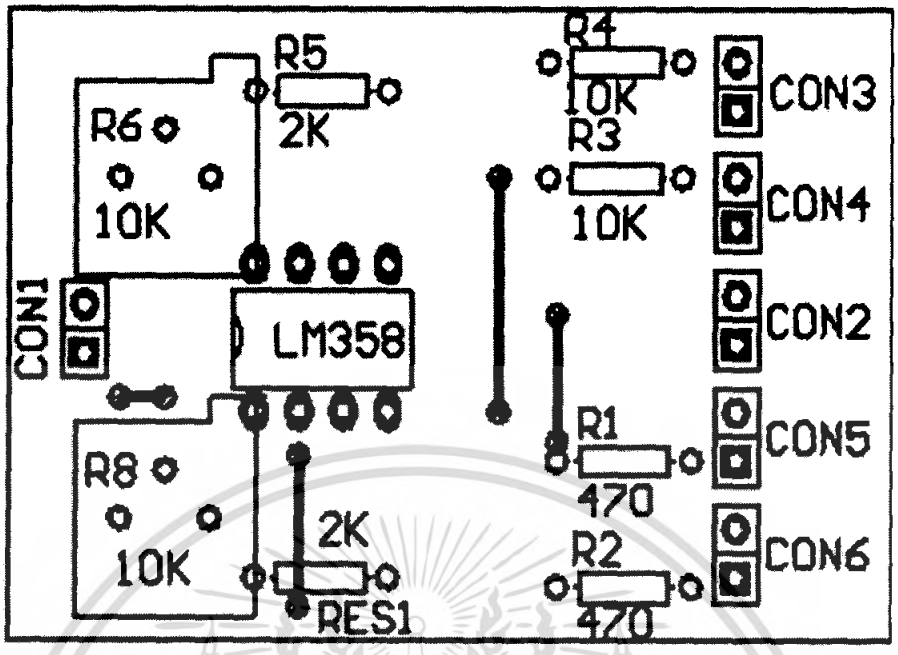
แสดงภาพ PCB Bottom layer ของวงจรตรวจจับสถานะหลอดไฟและปั้มน้ำ



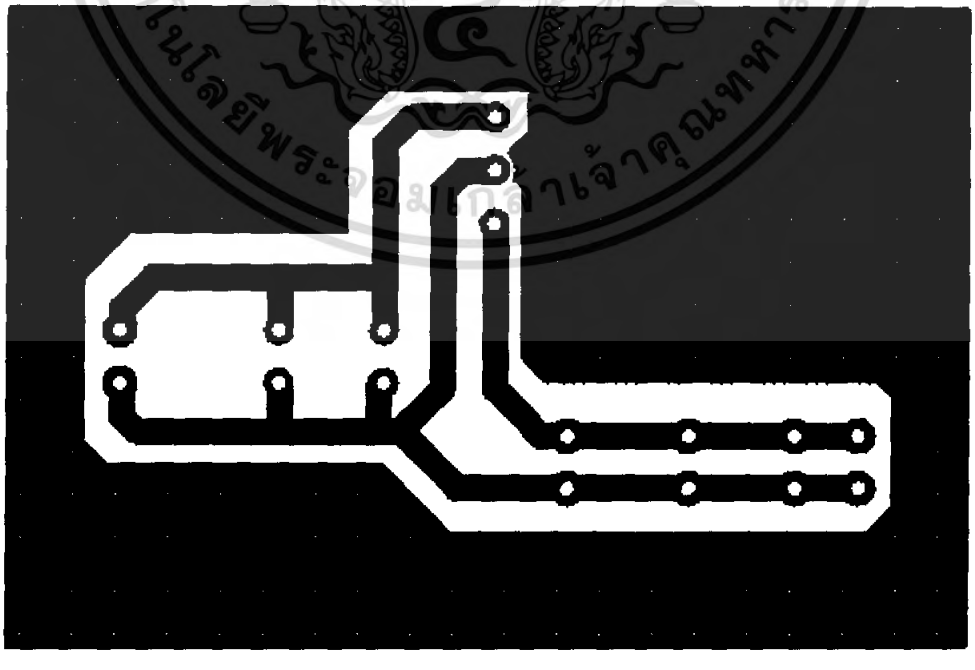
แสดงภาพการวางอุปกรณ์บน PCB ของวงจรตรวจจับสถานะ หลอดไฟและปั้มน้ำ



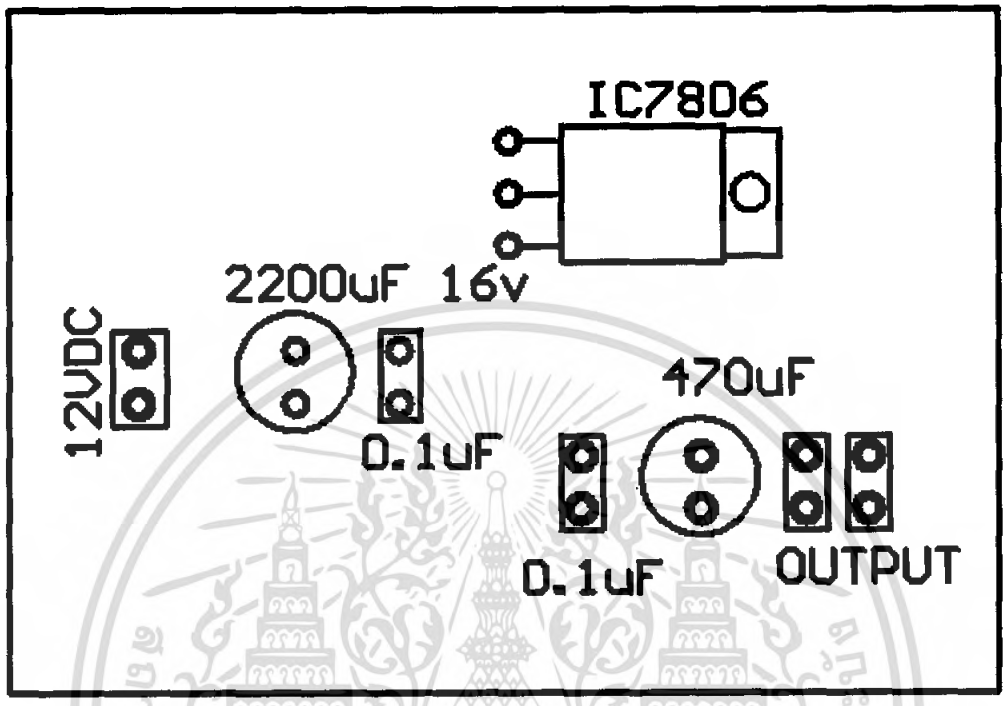
แสดงภาพ PCB Bottom layer ของวงจรตรวจจับการตัดผ่านเซ็นเซอร์อินฟาเรด



แสดงภาพการวางอุปกรณ์บน PCB ของวงจรตรวจจับการผ่านเซ็นเซอร์อินฟาเรด



แสดงภาพ PCB Bottom layer วงจรรักษาระดับแรงดัน 6V-DC



แสดงภาพการวางอุปกรณ์บน PCB ของวงจรรักษาระดับแรงดัน 6V-DC