ระบบควบคุมอุปกรณ์ไฟฟ้าในโรงงานด้วยคอมพิวเตอร์

Electrical Device Controlling System in Factory by PC

โดย

นายวิเชษฐ์ แหวนวิเศษ

รหัส 43067090

อาจารย์ที่ปรึกษา

ดร. จันทร์บูรณ์ สถิตวิริยวงศ์

รายงานนี้เป็นส่วนหนึ่งของวิชาโครงการพัฒนาระบบงาน

หลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

ภาคเรียนที่ 1 ปีการศึกษา 2545

คณะเทคโนโลยีสารสนเทศ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ชื่อหัวข้อ	ระบบควบคุมอุปกรณ์ไฟฟ้าในโรงงานด้วยคอมพิวเตอร์
นักศึกษา	นายวิเชษฐ์ แหวนวิเศษ
อาจารย์ที่ปรึกษา	ดร. จันทร์บูรณ์ สถิตวิริยวงศ์
ระดับการศึกษา	วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
แขนงวิชา	วิทยาการสารสนเทศ
ปีการศึกษา	2545

บทคัดย่อ

ในปัจจุบันนี้ คอมพิวเตอร์ได้มีบทบาทในชีวิตประจำวันของมนุษย์เราเป็นอย่างมาก และต่อไปในอนาคตก็คาดว่าจะมีการใช้งานคอมพิวเตอร์มากยิ่งขึ้น จึงมีความจำเป็นที่จะต้องมีการเรียนรู้การใช้คอมพิวเตอร์เพื่อที่จะได้ใช้งานได้อย่างมีประสิทธิภาพ ในโครงงานนี้จะเป็นการศึกษาการนำคอมพิวเตอร์มาประยุกต์ใช้งานในการควบคุมการปิดเปิดอุปกรณ์ไฟฟ้า ในโรงงานอุตสาหกรรมซึ่งจะช่วยลดค่าใช้จ่ายค่าไฟฟ้าจากการที่ลืมปิดไฟ เนื่องจากการควบคุมจะใช้คอมพิวเตอร์ควบคุม สวิทช์ไฟฟ้าตามจุดต่างๆ จึงสามารถควบคุมได้ทั่วถึงที่จุดๆ เดียว

Title	Electrical Device Controlling System in Factory by PC
Student	Mr. Wichest Waenwiset
Advisor	Dr. Chantaboon Sathidwiriyawong
Level of Study	Master of Science in Information Technology
Major	Information Science
Academic Year	2002

ABSTRACT

Computer has influenced for our life in many ways not only for nowadays but also in the future it will be used much more. So then it is necessary to learn how to use computer to maximize efficiency of usage. This project is learning how to apply computer to control electrical devices in factory that will be useful to reduce the cost of electricity from forgetting to turn-off switches sometimes, because the controlling will control in many parts of the factory by using the same computer.

กิตติกรรมประกาศ

โครงการพัฒนาระบบนี้สำเร็จได้เพราะได้รับการสนับสนุนจากบุคคลหลายฝ่าย กระผมจึงใคร่ขอกราบขอบพระคุณ

- บิดา – มารดา ที่ได้อบรมสั่งสอนและสนับสนุนให้ได้เล่าเรียนจนประสบความสำเร็จในการศึกษา
- อาจารย์ จันทร์บูรณ์ สถิตวิริยวงศ์ ที่ได้กรุณาให้คำปรึกษาแนะนำสิ่งต่างๆ ในโครงการพัฒนาระบบงาน
- อาจารย์ทุกท่านที่ได้ประสิทธิ์ประสาทวิชาความรู้หลักวิชาการต่างๆ เพื่อเป็นพื้นฐานในการทำงาน
- เจ้าหน้าที่คณะเทคโนโลยีสารสนเทศทุกท่านที่คอยอำนวยความสะดวก
- สุดท้ายคือ เพื่อน และญาติมิตรที่คอยให้กำลังใจและคำแนะนำด้วยดีเสมอมา

นายวิเชษฐ์ แหวนวิเศษ

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

สารบัญตาราง

หน้า

ตารางที่

สารบัญรูปภาพ

หน้า

รูปที่

บทที่ 1
บทนำ

1.1 ความเป็นมา

เนื่องจากในปัจจุบันคอมพิวเตอร์ได้เข้ามามีบทบาทกับชีวิตประจำวันของเราเป็นอย่างมาก ไม่ว่าจะเป็นสถานศึกษา โรงพยาบาล สถานที่ราชการ สำนักงานเอกชน ต่างๆ จึงมีความจำเป็นที่จะต้องมีการเรียนรู้การใช้งานคอมพิวเตอร์ และจะมีประโยชน์มากยิ่งขึ้น ถ้าสามารถประยุกต์ใช้งานคอมพิวเตอร์กับงานต่างๆได้มากยิ่งขึ้น อันอาจจะเป็นการช่วยลดค่าใช้จ่าย ต่องานบางอย่างลงได้ หรืออาจจะทำให้การทำงานบางอย่างสะดวกรวดเร็วยิ่งขึ้น ในโครงงานนี้จะทำการศึกษา การใช้คอมพิวเตอร์มาควบคุมอุปกรณ์ไฟฟ้าในโรงงานอุตสาหกรรม เพื่อความสะดวกและจะเป็นการช่วยลดค่าใช้จ่ายค่าไฟฟ้า จากการลืมปิด สวิทช์ ไฟฟ้าได้เนื่องจากการควบคุมจะกระทำที่เครื่องคอมพิวเตอร์เพียงเครื่องเดียวแต่สามารถควบคุมได้หลายๆ ส่วนพร้อมๆกัน

และเนื่องจาก การที่ภาษา Microsoft Visual Basic 6.0 เป็นภาษาที่ศึกษาง่ายและสามารถใช้งานได้หลากหลายจึงถูกนำมาใช้กับโครงงานนี้โดยใช้ Microsoft Visual Basic เขียนโปรแกรมติดต่อกับอุปกรณ์ภายนอกเพื่อควบคุมอุปกรณ์ไฟฟ้า โดยการสร้างโปรแกรมด้วย VB นั้นจะเป็นการเลือกเครื่องมือต่าง ๆ มาออกแบบหน้าจอของโปรแกรมที่เราจะสร้าง ซึ่งเรียกว่า Visual Programming ในการเขียนโปรแกรมแบบนี้ เราไม่จำเป็นต้องใช้คำสั่งมากนักก็สามารถโปรแกรมได้อย่างรวดเร็ว

โครงงานที่ได้ทำการศึกษาและพัฒนาขึ้นมานี้มีความคล้ายคลึงทางด้านแนวคิดของโครงงานของนักศึกษารุ่นก่อนแต่เนื่องจากยังมีบางสิ่งที่ยังไม่สมบูรณ์จึงได้มีการเพิ่มเติมบางส่วนในแนวคิดนั้นให้สมบูรณ์ยิ่งขึ้นดังนี้

โครงงาน “ระบบควบคุมอุปกรณ์ไฟฟ้าผ่านอินเตอร์เน็ต” ควบคุมโดยตั้งเวลาเปิดปิดแบบอัตโนมัติอย่างเดียวโดยเทียบเวลาที่ตั้งค่าไว้ในฐานข้อมูลกับเวลาของเครื่องคอมพิวเตอร์ที่ควบคุม ไม่มีการสั่งงานแบบแมนนวล และไม่มีการตรวจสอบสถานการณ์ทำงานจริงของอุปกรณ์ที่ควบคุม เนื่องจากไม่มี Sensor ดักจับสัญญาณจากอุปกรณ์ที่ควบคุม

โครงงาน “ระบบตั้งเวลาควบคุมการใช้งานเครื่องใช้ไฟฟ้า” มีลักษณะเช่นเดียวกันกับโครงงานก่อนคือควบคุมโดยตั้งเวลาเปิดปิดแบบอัตโนมัติ โดยเทียบเวลาที่ตั้งค่าไว้ในฐานข้อมูลกับ

เวลาของเครื่องคอมพิวเตอร์ที่ควบคุม ไม่มีการสั่งงานแบบแมนนวล และตรวจสอบสถานะของอุปกรณ์ที่ควบคุมไม่ได้เนื่องจากไม่มีการใช้ Sensor ตรวจจับสัญญาณ

โครงงาน "ระบบรักษาความปลอดภัยภายในบ้านผ่านอินเตอร์เน็ต" เป็นโครงงานที่แสดงผลข้อมูลของเครื่องแสดงผลข้อมูลที่ส่งมาจากกล้องที่จับภาพ และ ตัวตรวจจับอุณหภูมิอย่างเดียว โดยดึงข้อมูลที่ได้นั้นจากฐานข้อมูลมาแสดงผลเพียงอย่างเดียว ไม่มีการควบคุมอย่างอื่น

ส่วนโครงงานที่ได้ทำการศึกษาและพัฒนาขึ้นมานี้เป็นการใช้คอมพิวเตอร์สั่งงานการเปิดปิดอุปกรณ์ไฟฟ้าภายในโรงงานอุตสาหกรรมแบบรวมศูนย์การควบคุมไว้ที่แห่งเดียวกันโดยใช้คอมพิวเตอร์สั่งงานควบคุม การควบคุมมี 2 ลักษณะคือ การเปิดปิดสวิทช์แบบแมนนวล และแบบอัตโนมัติ พร้อมทั้งมีการเก็บสถานการณ์ทำงานของอุปกรณ์ที่ควบคุม และยังใช้ Sensor จับสัญญาณการทำงานของอุปกรณ์จริงป้อนเพื่อจะได้ทราบสถานะของอุปกรณ์จริงนั้นว่ายังใช้งานได้ดีอยู่หรือไม่

1.2 วัตถุประสงค์ของการพัฒนาระบบงาน

1. เพื่อให้เข้าใจหลักการทำงานของระบบบัส I^2C
2. เพื่อสร้างเครื่องควบคุมอุปกรณ์เครื่องใช้ไฟฟ้าภายในโรงงาน ที่สั่งโดย PC ได้
3. เพื่อศึกษาการเขียนโปรแกรมควบคุมอุปกรณ์โดยใช้ Microsoft Visual Basic

1.3 หลักการที่เกี่ยวข้องในการพัฒนาระบบงาน

เนื่องจากระบบงานนี้จะใช้การสื่อสารระบบบัส I^2C เป็นหลัก ในการติดต่อระหว่างอุปกรณ์ควบคุมกับคอมพิวเตอร์ ส่วนการติดต่อกับโปรแกรมควบคุมจะใช้อินเตอร์เน็ต ดังนั้นจึงต้องอาศัยหลักการพื้นฐานความรู้ในด้านต่างๆ ดังต่อไปนี้คือ

1. หลักการสื่อสารแบบอนุกรม
2. หลักการเชื่อมต่อพอร์ตอนุกรมกับระบบบัส I^2C
3. หลักการใช้งาน Microsoft Visual Basic
4. หลักการในการสร้างฐานข้อมูล

1.4 รูปแบบของการพัฒนาระบบงาน

ระบบงานที่พัฒนามีลักษณะเป็น Standalone โดยใช้คอมพิวเตอร์สั่งงานปิดเปิดสวิทช์ โดยส่งรหัสควบคุมไปยังวงจรควบคุมและรับข้อมูลสถานะการติดหรือดับของหลอดไฟจากวงจรควบคุมมาแสดงผลที่หน้าจอ

1.5 ขอบเขตของการพัฒนาระบบงาน

1. การเขียนโปรแกรมควบคุมอุปกรณ์ไฟฟ้าผ่านทางพอร์ตอนุกรมโดยใช้ Visual Basic 6.0
2. การสร้างวงจรควบคุมเพื่อควบคุมอุปกรณ์ไฟฟ้าโดยใช้ LED แสดงสถานะการสั่งงานของสวิทช์ และใช้ Dip switch แสดงสัญญาณ การติดหรือดับของอุปกรณ์ที่ควบคุม

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1. เข้าใจหลักการการใช้คอมพิวเตอร์ควบคุมอุปกรณ์ไฟฟ้า
2. สามารถนำมาประยุกต์ใช้งานได้จริงกับโรงงาน อาคาร หรือ ที่พักอาศัย สำนักงานได้

1.7 ขั้นตอนดำเนินงาน

1. ศึกษาหลักการที่เกี่ยวข้อง
2. ออกแบบระบบระบบ
3. ออกแบบและสร้างอุปกรณ์ควบคุม
4. เขียนโปรแกรมควบคุมอุปกรณ์ควบคุม
5. ทดลองระบบงานและสรุปผล

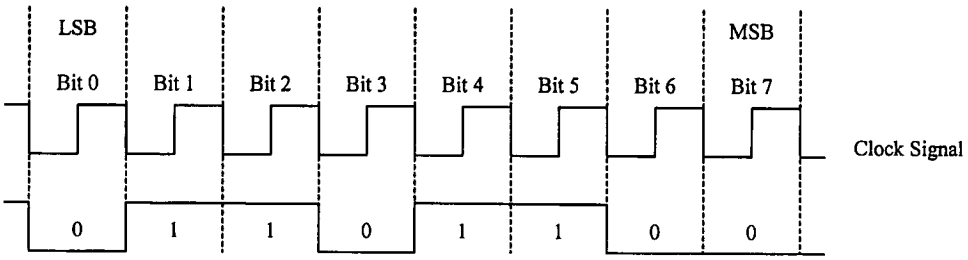
บทที่ 2
พอร์ตอนุกรม

มีทางเลือกอยู่ 2 ทางในการที่จะเคลื่อนย้ายข้อมูลจากคอมพิวเตอร์ไปยังอุปกรณ์ต่อพ่วงอื่น ๆ หรือคอมพิวเตอร์ด้วยกัน นั่นคือการรับส่งข้อมูลแบบขนานและการรับส่งข้อมูลแบบอนุกรม การรับส่งข้อมูลแบบขนาน จะเป็นการรับหรือส่งข้อมูลคราวละ 4 หรือ 8 บิต ในเวลาเดียวกัน ซึ่งจะทำให้การรับและส่งข้อมูลทำได้ที่ความเร็วสูง ซึ่งก็หมายความว่าจำนวนของสายที่ใช้ในการส่งจะต้องมีมากเท่ากับจำนวนบิตของข้อมูลที่จะส่งด้วย นอกจากนี้ยังจะต้องใช้สายมากเป็น 2 เท่าของจำนวนบิตข้อมูลที่จะส่งก็ได้ ซึ่งก็เป็นปัญหาในเรื่องราคาของสายที่ใช้ในการเชื่อมต่อแบบขนานมักจะมีราคาแพง

ในขณะที่การรับส่งข้อมูลแบบอนุกรมเป็นการรับส่งข้อมูลครั้งละ 1 บิต แต่ก็สามารถรับส่งข้อมูลได้คราวละหลาย ๆ บิตได้ หากแต่จะต้องมีการตกลงกันระหว่างตัวส่งและตัวรับว่า จะรับส่งข้อมูลคราวละกี่บิต ตัวรับจะต้องรอข้อมูลมาให้ครบทุกบิตเสียก่อนจึงทำการประมวลผล ส่งผลให้การสื่อสารข้อมูลอนุกรมอาจมีความเร็วต่ำกว่าแบบขนาน ในด้านจำนวนสายสัญญาณการรับส่งข้อมูลแบบอนุกรมจะใช้จำนวนสายที่น้อยกว่ามาก อย่างน้อยที่สุดใช้เพียง 2-3 เส้นเท่านั้น แต่อัตราเร็วในการรับส่งข้อมูลอาจต่ำกว่าแบบขนาน อย่างไรก็ตามการรับส่งข้อมูลแบบอนุกรมสามารถใช้สายสัญญาณที่มีความยาวมากกว่าแบบขนาน ทำให้ระยะทางในการสื่อสารข้อมูลแบบอนุกรมสามารถทำได้มากกว่า

2.1 การสื่อสารแบบอนุกรม

การสื่อสารแบบอนุกรมนั้นจะแบ่งออกได้เป็น 2 แบบคือการสื่อสารอนุกรมแบบซิงโครนัสและการสื่อสารอนุกรมแบบอะซิงโครนัส การสื่อสารแบบซิงโครนัสจะมีสัญญาณนาฬิการ่วมอยู่กับการรับและส่งสัญญาณด้วย ตัวอย่างการส่งข้อมูลแบบซิงโครนัสก็คือคีย์บอร์ดของคอมพิวเตอร์ ซึ่งสายเส้นหนึ่งจะเป็นสายของสัญญาณนาฬิกา ส่วนสายอีกเส้นจะเป็นสายของข้อมูล ดังนั้นการติดต่อกันแบบซิงโครนัสนี้จะต้องใช้สายในการเชื่อมต่ออย่างน้อยที่สุด 3 เส้นคือ สัญญาณนาฬิกา, ข้อมูลและกราวด์ รูปที่ 2.1 แสดงให้เห็นถึงไทมิ่งไดอะแกรมของการส่งข้อมูลแบบซิงโครนัส

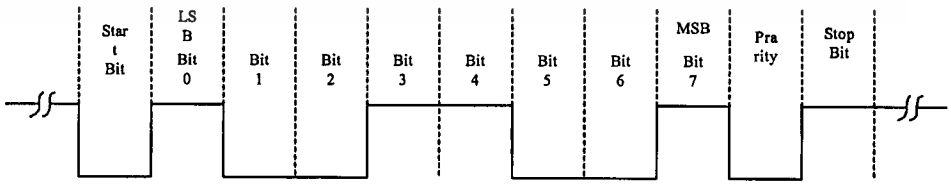


รูปที่ 2.1 รูปแบบอย่างง่ายที่สุดของข้อมูลอนุกรมแบบซิงโครนัส

การสื่อสารข้อมูลแบบอะซิงโครนัสคือการรับและส่งข้อมูลไปในสายโดยไม่จำเป็นต้องมีสัญญาณนาฬิการ่วมด้วยเหมือนกับการรับส่งข้อมูลแบบซิงโครนัส แต่จะใช้การกำหนดค่าสัญญาณนาฬิกาทั้งภาครับและภาคส่งให้มีค่าเท่ากัน ซึ่งเรียกสัญญาณนาฬิกาที่ใช้ในการกำหนดค่าให้ภาครับและภาคส่งนี้ว่า **อัตราการถ่ายทอดข้อมูล** หรือ **บอดเรต (baudrate)** มีหน่วยเป็น บิตต่อวินาที (bit per second : bps)

รูปแบบของข้อมูลที่ใช้ในการรับส่งแบบอะซิงโครนัสประกอบด้วย 4 ส่วนด้วยกันคือ

1. บิตเริ่มต้น (Start Bit) ซึ่งจะมีขนาด 1 บิต
2. บิตข้อมูลแบบอนุกรมจะมีขนาด 5,6,7 หรือ 8 บิต
3. บิตตรวจสอบพาริตี้ (Parity Bit) จะมีขนาด 1 บิตหรือไม่มี
4. บิตปิดท้าย (Stop Bit) จะมีขนาด 1,1.5, หรือ 2 บิต



รูปที่ 2.2 แสดงรูปแบบอย่างง่ายที่สุดของข้อมูลอนุกรมแบบอะซิงโครนัส

รูปที่ 2.2 แสดงรูปแบบของข้อมูลอนุกรมแบบอะซิงโครนัส ซึ่งเมื่อไม่มีข้อมูลที่จะส่ง ขา DATA จะมีลอจิก “1” ซึ่งจะเรียกสถานะนี้ว่าสถานะหยุดรอ (waiting state) การเริ่มต้นส่งข้อมูลจะเริ่มจากการให้ขา DATA มีลอจิก “0” ด้วยช่วงระยะเวลา 1 บิต ซึ่งจะเรียกบิตนี้ว่าบิตเริ่มต้น

จากนั้นบิตข้อมูลจะถูกส่งออกไป โดยเริ่มจากบิตที่มีนัยสำคัญต่ำสุด (LSB) ก่อน ซึ่งข้อมูลในไบต์ที่จะส่งอาจจะมีจำนวนบิต 5,6,7 หรือ 8 บิตก็ได้ จากนั้นจะตามด้วยบิตพาริตี้ ซึ่งใช้เพื่อตรวจสอบความผิดพลาดที่เกิดขึ้นจากการส่งข้อมูล บิตสุดท้ายที่ส่งคือบิตปิดท้าย ซึ่งจะให้ขาดาต้ามีสถานะลอจิก 1 อีกครั้งด้วยระยะเวลาอย่างน้อย 1 บิต, 1.5 บิต หรือ 2 บิต เพื่อเป็นการแสดงว่าสิ้นสุดข้อมูลแล้ว

อุปกรณ์พิเศษที่ได้รับการออกแบบมาสำหรับการรับและส่งข้อมูลแบบอะซิงโครนัสเรียกว่า Universal Asynchronous Receiver/Transmitter หรือ UART อัตราความเร็วในการรับและส่งข้อมูลของการรับส่งข้อมูลแบบอะซิงโครนัสคือ **ค่าบอดเรต** ซึ่งก็คือค่าจำนวนบิตต่อวินาทีที่ใช้ในการรับและส่งข้อมูล บอดเรตมาตรฐานที่ใช้สำหรับพอร์ตอนุกรม RS-232 ได้แก่ 110, 150, 300, 600, 1200, 2400, 4800, 9600 และ 19200 บิตต่อวินาที และมีค่าเพิ่มมากขึ้นตามเทคโนโลยีของคอมพิวเตอร์ซึ่งการรับส่งแบบอนุกรมโดยไม่ผ่านโมเด็มอาจจะสามารถกำหนดค่าบอรเรตได้สูงถึง 115200 บิตต่อวินาที เนื่องจากบอดเรตคือจำนวนบิตของข้อมูลที่สามารถถ่ายทอดได้ภายใน 1 วินาที ยกตัวอย่างข้อมูลอนุกรมถูกส่งในลักษณะ 8 บิต ไม่มีการตรวจสอบพาริตี้ มีบิตเริ่มต้น 1 บิต และบิตปิดท้าย 1 บิต ความยาวของข้อมูลที่รับส่งนี้เท่ากับ 10 บิต ถ้าใช้บอดเรตในการส่งข้อมูลเท่ากับ 9600 บิตต่อวินาที ก็จะสามารถรับส่งข้อมูลได้ด้วยความเร็ว 960 ไบต์ต่อวินาที และถ้ามีการใช้พาริตี้ความเร็วในการรับส่งข้อมูลจะเหลือเป็น 872 ไบต์ต่อวินาที

การตรวจสอบพาริตี้สามารถกำหนดให้เป็นแบบคี่(odd) แบบคู่(even) หรือไม่มีการตรวจสอบพาริตี้ก็ได้ การตรวจสอบพาริตร้เป็นการตรวจสอบจำนวนรวมของบิตที่เป็นลอจิก “1” ภายในข้อมูลที่ส่งไป 1 ไบต์ว่ามีจำนวนรวมเป็นเลขคู่หรือเลขคี่โดยต้องรวมบิตพาริตี้เข้าไปด้วย ยกตัวอย่างข้อมูลที่จะทำการส่งมีขนาด 8 บิต และมีค่าเท่ากับ 99 ฐานสิบหก หรือ 10011001 ฐานสอง จะเห็นว่าข้อมูลในไบต์นี้มีจำนวนลอจิก “1” จำนวน 4 ตัวซึ่งเป็นเลขคู่ ดังนั้นถ้ากำหนดค่าพาริตี้เป็นคู่ค่าในบิตพาริตี้ จะต้องมีลอจิกเป็น “0” แต่ถ้าพาริตี้เป็นคี่ ค่าที่บิตพาริตี้จะต้องเป็น “1” เพื่อให้ข้อมูล 1 ไบต์รวมทั้งบิตพาริตี้มีจำนวนบิตที่เป็นลอจิก “1” มีจำนวนรวมกันเป็นเลขคี่ ในตารางที่ 2.1 แสดงตัวอย่างของบิตพาริตี้ในการรับส่งข้อมูล

ตารางที่ 2.1 แสดงบิตพาริตี้ของข้อมูล

ข้อมูล	บิตพาริตี้คู่	บิตพาริตี้คี่
00000000	0	1
00000001	1	0
00000010	1	0
00000011	0	1
00000100	1	0
11111110	1	0
11111111	0	1

บิตพาริตี้ถูกสร้างขึ้นจากภาคส่งข้อมูลของ UART ซึ่งทางภาครับจะต้องทำการกำหนดคุณสมบัติการตรวจสอบพาริตี้ให้ตรงกันว่าจะตรวจสอบพาริตี้คี่หรือพาริตี้คู่ จากนั้นภาครับของ UART จะทำการตรวจสอบค่าพาริตี้ที่เกิดขึ้นว่าเป็นคู่หรือเป็นคี่ โดยการนับจำนวนลอจิก "1" ทั้งหมดรวมทั้งบิตพาริตี้ด้วย ถ้ากำหนดพาริตี้ไว้เป็นคู่แต่อ่านค่าตัวเลขในการนับออกมาได้ตัวเลขเป็นคี่ทางภาครับจะแสดงข้อผิดพลาดออกมาให้ผู้ใช้ทราบ นับเป็นการตรวจสอบความผิดพลาดที่เกิดขึ้นในการถ่ายทอดข้อมูลที่ง่ายที่สุด แต่จะเชื่อถือได้เมื่อมีบิตข้อมูลที่ทำการส่งผิดพลาดเพียงบิตเดียวเท่านั้น ถ้าข้อมูลที่ทำการส่งมีบิตที่ผิดพลาดมากกว่า 1 บิต การตรวจสอบด้วยวิธีนี้จะไม่ได้ผล สำหรับการตั้งพาริตี้บิตเป็น NONE นั้นทั้งภาครับและภาคส่ง จะไม่มีการตรวจสอบพาริตี้

คอมพิวเตอร์ในรุ่น AT เกือบทั้งหมดจะใช้ UART เบอร์ 16450 และ 16550 ส่วนคอมพิวเตอร์ในรุ่น XT ใช้ UART เบอร์ 8250 UART ชิปเหล่านี้มีระดับแรงดันเป็นแบบทีทีแอล (0 และ+5) แต่เพื่อให้มีแรงดันเป็นไปตามมาตรฐาน RS-232 และเพื่อให้การรับส่งข้อมูลสามารถทำได้ที่ระยะทางไกลมากขึ้น ระดับแรงดันทีทีแอลจะถูกแปลงเป็นระดับแรงดันที่สูงขึ้น โดยลอจิก "0" มีระดับแรงดัน +3V ถึง +12V ในขณะที่ลอจิก "1" มีระดับแรงดัน —3V จนถึง —12V

2.2 มาตรฐานพอร์ตอนุกรมแบบ RS-232

มาตรฐานการเชื่อมต่อแบบอนุกรม RS-232 เป็นมาตรฐานอุตสาหกรรมที่ออกแบบมาเพื่อใช้ในการส่งข้อมูลอนุกรมแบบอะซิงโครนัส 2 ทิศทาง โดยมาตรฐาน RS-232 ในอดีตนั้นถูกออกแบบมาเพื่อการส่งผ่านข้อมูลจากคอมพิวเตอร์ไปยังโมเด็มเพียงอย่างเดียว เพื่อที่จะนำข้อมูลจากโมเด็มนี้สื่อสารผ่านสายโทรศัพท์ไปยังคอมพิวเตอร์อีกชุดซึ่งอยู่ห่างไกลกัน โดยคณะกรรมการวาง

มาตรฐานที่มีชื่อเรียกกันว่า EIA RS-232 มาตรฐานนี้ในช่วงแรกจะใช้คอนเน็กเตอร์เป็นแบบ DB-25 โดยกำหนดความยาวสูงสุดของสายสัญญาณไว้ที่ 50 ฟุต มีระดับสัญญาณตั้งแต่ —3V จนถึง -12V

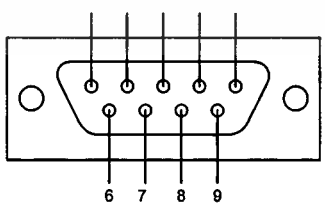
มาตรฐาน RS-232 ได้กำหนดรูแบบของอุปกรณ์เชื่อมต่อข้อมูล (Data Terminal Equipment: DTE) กับวงจรข้อมูลปลายทาง (Data Circuit Terminating : DCE) ไว้ว่า อุปกรณ์ DTE จะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัวเช่น ไมโครคอนโทรลเลอร์หรือไมโครคอมพิวเตอร์ ซึ่งมีความสามารถในการสร้างบิตข้อมูลแบบอนุกรมได้ ส่วนอุปกรณ์ DCE จะทำหน้าที่เป็นเพียงตัวรับข้อมูลที่ส่งมาจาก DTE เท่านั้น โดยการรับส่งข้อมูลระหว่างอุปกรณ์ทั้งสองจะกระทำผ่านมาตรฐาน RS-232

ข้อแตกต่างของอุปกรณ์ DTE และ อุปกรณ์ DCE อย่างหนึ่งที่เห็นได้ชัดคือ คอนเน็กเตอร์ของ DTE จะเป็นตัวผู้ ส่วนคอนเน็กเตอร์ของ DCE จะเป็นตัวเมีย ซึ่งพอร์ตนอุกรมของคอมพิวเตอร์ที่ใช้กันอยู่ทั่วไปจะเป็นแบบ DTE ส่วนคอนเน็กเตอร์ที่อยู่ที่โมเด็มจะเป็นแบบ DCE

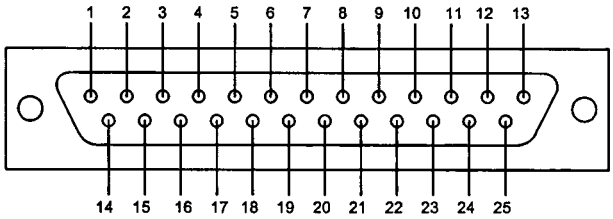
สำหรับการใช้งานบนคอมพิวเตอร์ พอร์ตอนุกรม RS-232 มักถูกใช้เชื่อมต่อกับโมเด็มหรือเมาส์ โดยสามารถรับส่งข้อมูลได้ที่ความยาวของสายสัญญาณสูงสุดถึง 20 เมตร

2.3 คอนเน็กเตอร์สำหรับพอร์ต RS-232 และการเชื่อมต่อ

มาตรฐานการเชื่อมต่อแบบ RS-232 จะใช้คอนเน็กเตอร์แบบ DB-25 ตัวผู้หรือ DB-9 ตัวผู้ ซึ่งคอนเน็กเตอร์แบบ DB-25 จะมีขาต่อใช้งานเพียง 9 เส้นเช่นเดียวกับคอนเน็กเตอร์แบบ DB-9 เนื่องจากขาอื่น ๆ ที่เคยใช้งานในอดีต ปัจจุบันมีการใช้งานไม่มากนัก จึงถูกยกเลิกไป โดยแสดงรูปร่างและตำแหน่งขาในรูปที่ 2.3



รูปที่ 2.3 (ก) คอนเน็กเตอร์อนุกรม 9 ขาหรือแบบ DB-9 (มองจากด้านหลังคอมพิวเตอร์)



รูปที่ 2.4 (ข) คอนเน็กเตอร์อนุกรม 25 ขาหรือแบบ DB-25 (มองจากด้านหลังคอมพิวเตอร์)

ตารางที่ 2.2 การจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS-232 ทั้งแบบ DB-9 และDB-25

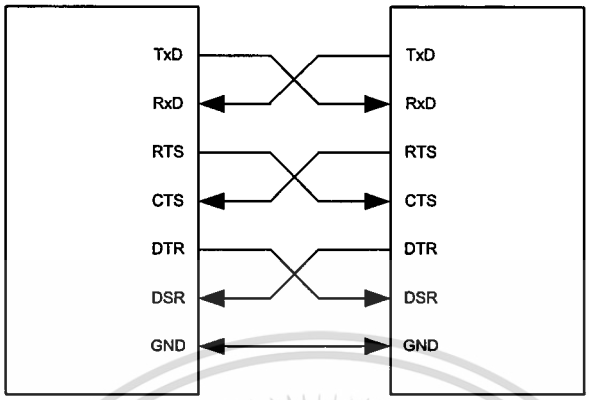
คอนเน็กเตอร์ DB-9	คอนเน็กเตอร์ DB-25	ชื่อของสายสัญญาณ	ชนิดของสายสัญญาณ
1	8	Data Carrier Detect : DCD	อินพุต
2	3	Received Data : RxD	อินพุต
3	2	Transmitted Data : TxD	เอาต์พุต
4	20	Data Terminal Ready : DTR	เอาต์พุต
5	7	Signal Ground : GND	-
6	6	Data Set Ready : DSR	อินพุต
7	4	Request To Send : RTS	เอาต์พุต
8	5	Clear To Send : CTS	อินพุต
9	22	Ring Indicator : RI	อินพุต

- **Transmitted Data (TD) :** เป็นวงจรที่สร้างสัญญาณ Transmitted Data ซึ่งถูกส่งจาก DTE ไปยัง DCE ซึ่งสัญญาณที่ส่งออกมาอาจจะเป็นโค้ดคำสั่งของพอร์ตหรือข้อมูลก็ได้
- **Received Data (RD) :** เป็นวงจรที่สร้างสัญญาณ Received Data ซึ่งถูกส่งจาก DCE ไปยัง DTE ซึ่งสัญญาณที่ส่งออกมาอาจจะเป็นโค้ดคำสั่งของพอร์ตหรือข้อมูลก็ได้ ซึ่งเป็นสัญญาณที่มีทิศทางการส่งตรงข้ามกับสัญญาณ Transmitted Data
- **Data Terminal Ready (DTR) :** สัญญาณ DTR จะถูกส่งจาก DTE ไปยัง DCE เพื่อเป็นการแจ้งความพร้อมในการสื่อสารให้อุปกรณ์ได้ทราบ โดยถ้าหากพอร์ตนั้น (อุปกรณ์ DCE) มี

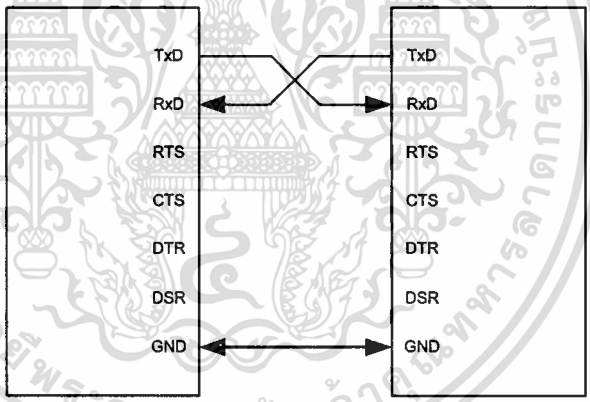
ความสามารถในการตอบรับแบบอัตโนมัติ (automatically answer) อุปกรณ์นั้นก็จะสามารถตอบรับได้เฉพาะเมื่อสัญญาณ DTR อยู่ในสถานะ on เท่านั้น

- **Carrier Detect (CD) :** สัญญาณ CD จะถูกส่งจาก DCE ไปยัง DTE เพื่อเป็นการแจ้งว่าพอร์ตอยู่ในสถานะกำลังติดต่อกับพอร์ตตัวอื่น หรือพอร์ตกำลังได้รับสัญญาณที่พร้อมสำหรับการติดต่อสื่อสาร สำหรับสัญญาณ Carrier Detect นี้ สามารถเรียกอีกชื่อได้ว่า Received Line Signal Detector
- **Request To Send (RTS) :** สัญญาณ RTS จะถูกส่งจาก DTE ไปยัง DCE โดยเมื่อสัญญาณ RTS อยู่ในสถานะ on ก็หมายถึงเครื่องคอมพิวเตอร์พร้อมที่จะรับข้อมูลจากพอร์ต และในทางกลับกันถ้าหากสัญญาณ RTS อยู่ในสถานะ off ก็หมายถึง เครื่องคอมพิวเตอร์ไม่พร้อมที่จะรับข้อมูลจากพอร์ต
- **Clear To Send (CTS) :** สัญญาณ CTS จะถูกส่งจาก DCE ไปยัง DTE ซึ่งเป็นสัญญาณที่ทำหน้าที่ตรงข้ามกับสัญญาณ RTS โดยเมื่อสัญญาณ CTS อยู่ในสถานะ on ก็หมายถึงพอร์ตพร้อมที่จะรับข้อมูลจากเครื่องคอมพิวเตอร์และในทางกลับกันถ้าหากสัญญาณ CTS อยู่ในสถานะ off ก็หมายถึง พอร์ตไม่พร้อมที่จะรับข้อมูลจากเครื่องคอมพิวเตอร์
- **Data Set Ready (DSR) :** สัญญาณ DSR จะถูกส่งจาก DCE ไปยัง DTE เพื่อเป็นการแจ้งความพร้อมในการสื่อสารจากพอร์ตให้กับเครื่องคอมพิวเตอร์ได้ทราบ โดยสัญญาณ DSR จะอยู่ในสถานะ on ก็ต่อเมื่อพอร์ตได้รับสัญญาณ DTR เท่านั้น
- **Ring Indicator (RI) :** สัญญาณ RI จะถูกส่งจาก DCE ไปยัง DTE เพื่อเป็นการแจ้งให้เครื่องคอมพิวเตอร์ทราบว่า พอร์ตกำลังได้รับสัญญาณกระดิ่ง (ring signal) จากอุปกรณ์ตัวอื่น โดยที่สัญญาณ RI จะอยู่ในสถานะ on พอร์ตได้รับสัญญาณกระดิ่ง และ off เมื่อพอร์ตไม่ได้รับสัญญาณกระดิ่ง เนื่องจากอุปกรณ์รุ่นใหม่ ๆ ในปัจจุบันมักจะมีความสามารถในการตอบรับแบบอัตโนมัติ (automatically answer) ดังนั้นจึงไม่มีความจำเป็นต้องใช้สัญญาณ RI อีกต่อไป
- **Signal Ground (GND) :** ขากราวด์ของระบบ

สำหรับการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกแสดงดังในรูปที่ 2.4 ลูกศรในรูปแสดงถึงทิศทางของข้อมูล ในรูปที่ 2.4 (ก) เป็นการเชื่อมต่อแบบ Null modem หรือการเชื่อมต่อโดยตรงโดยไม่ต้องผ่านโมเด็ม โดยมีการตรวจสอบหรือแฮนด์เช็กเต็มรูปแบบ ส่วนในรูปที่ 2.4 (ข) เป็นการเชื่อมต่อแบบ Null modem ในลักษณะที่ใช้สายสัญญาณเพียง 3 เส้น โดยเส้นหนึ่งสำหรับส่งข้อมูล อีกเส้นหนึ่งสำหรับข้อมูล และเส้นสุดท้ายเป็นกราวด์



(ก) การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ Null modem



(ข) การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ RS-232 โดยใช้สายสัญญาณเพียง 3 เส้น

รูปที่ 2.5 การต่ออุปกรณ์ภายนอกกับพอร์ทอนุกรมคอมพิวเตอร์ในลักษณะต่างๆ

2.4 UART

UART มาจากคำว่า Universal Asynchronous Receiver Transmitter ซึ่งหมายถึงอุปกรณ์ที่ทำหน้าที่รับและส่งข้อมูลแบบอะซิงโครนัสนั่นเอง สำหรับการสื่อสารอนุกรมบนคอมพิวเตอร์แล้ว UART ถือว่าเป็นหัวใจสำคัญของการสื่อสารอนุกรม

หน้าที่หลัก UART คือทำหน้าที่แปลงข้อมูลที่อยู่ในรูปแบบขนานจากคอมพิวเตอร์ให้อยู่ในรูปแบบอนุกรมแบบอะซิงโครนัส แล้วส่งออกไป และทำหน้าที่แปลงสัญญาณอนุกรมแบบอะ

ซิงโครนัสที่ป้อนเข้ามายัง UART ให้เป็นแบบขนานก่อนที่จะส่งเข้าสู่คอมพิวเตอร์ ซึ่งนอกจาก UART จะส่งข้อมูลไปยังคอมพิวเตอร์แล้ว ยังแจ้งข้อมูลอื่น ๆ ให้คอมพิวเตอร์รับทราบด้วย เช่น อัตราเร็วในการรับส่งข้อมูล (บอดเรต), รูปแบบการส่งข้อมูล, ความผิดพลาดที่เกิดขึ้นระหว่างการถ่ายทอดข้อมูล (ผิดพลาดจากพาริตี้, เฟรมข้อมูล, โอเวอร์รัน) เป็นต้น

ภายใน UART จะมีส่วนของวงจรสร้างบอดเรตแบบโปรแกรมได้ (programmable buadrate generator) โดยการกำหนดค่าตัวหารให้กับสัญญาณนาฬิกาของ UART โดยตัวหารนี้มีขนาด 16 บิต ดังนั้นจึงสามารถกำหนดตัวหารอยู่ในช่วง 1 — 65,535 UART สามารถรับส่งข้อมูลได้ทั้งแบบฮาล์ฟดูเพล็กซ์ (half duplex) และฟูลดูเพล็กซ์ (full duplex) โดยการส่งแบบฮาล์ฟดูเพล็กซ์เป็นการส่งแบบทิศทางเดียว ส่วนการส่งแบบฟูลดูเพล็กซ์นั้นสามารถรับและส่งข้อมูลได้ในคราวเดียวกัน

2.4.1 ชนิดของ UART

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

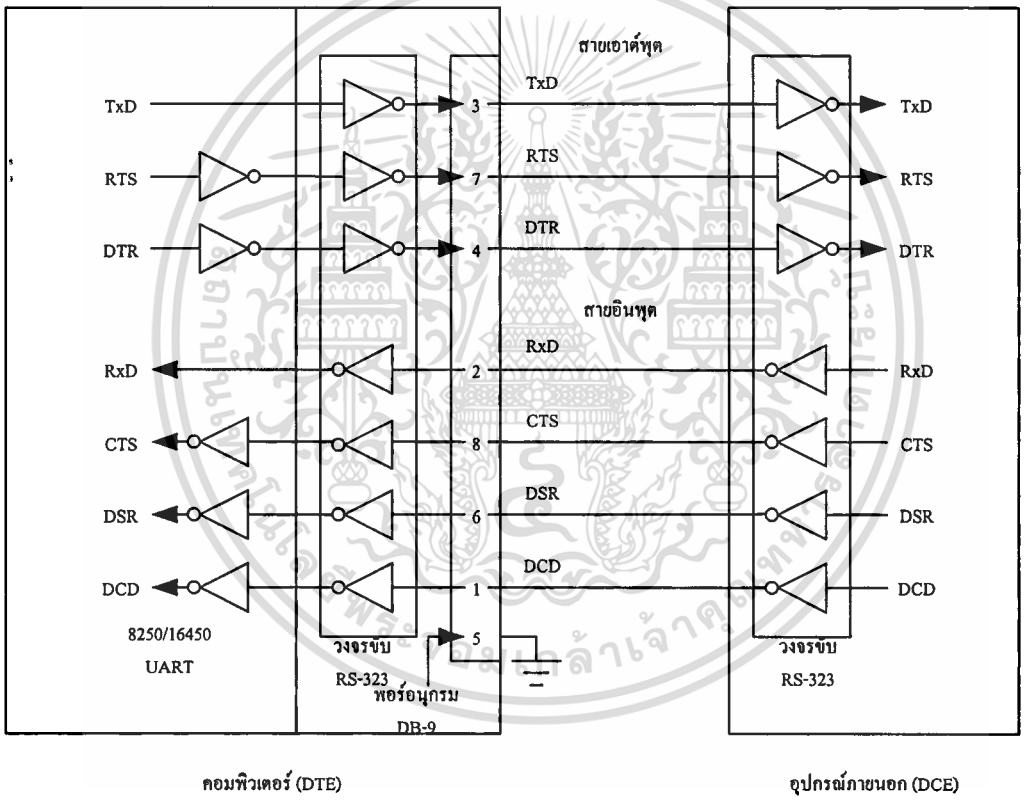
UART อีกเบอร์หนึ่งคือ 16450 มีความสามารถรับส่งข้อมูลได้ที่ความเร็ว 115,200 บิตต่อวินาที และเพิ่มรีจิสเตอร์สำหรับพักข้อมูลสำหรับ UART นอกจากนั้นยังเพิ่มส่วนองชิปต์รีจิสเตอร์แบบ FIFO (First In First Out) ขนาด 16 ไบต์เข้าไป ทำให้สามารถสนับสนุนความเร็วในการรับส่งข้อมูลที่ 256 กิโลบิตต่อวินาทีได้ โดยคอมพิวเตอร์ในปัจจุบันใช้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

อย่างไรก็ตาม ความเร็วในการส่งข้อมูลที่มากมายของ UART เบอร์ใหม่ ๆ ก็ไม่ได้ช่วยให้การรับส่งข้อมูลของคอมพิวเตอร์เร็วขึ้น เนื่องจากว่าคอมพิวเตอร์ยังใช้ความถี่ของสัญญาณนาฬิกาในการแปลงข้อมูลเพียง 1.8432 MHz เท่านั้น

2.5 ลักษณะสัญญาณอินพุตและเอาต์พุตของพอร์ต RS-232

สัญญาณเอาต์พุตที่ใช้ควบคุม (RTS และ DTR) และสัญญาณแสดงสถานะอินพุต (CTS, DSR และ DCD) ของพอร์ตอนุกรม RS-323 จะถูกกลับสถานะภายในตัว UART ส่วนสัญญาณ

ข้อมูลทั้งภาคส่งและรับจะไม่ถูกกลับสถานะ UART จะให้ระดับสัญญาณเอาต์พุตออกมาเป็นแบบทีทีแอลเท่านั้น ดังนั้นเมื่อสัญญาณถูกส่งออกมาจาก พอร์ต UART จึงต้องส่งเข้าสู่วงจรขับเพื่อปรับระดับแรงดันให้ได้ระดับสัญญาณเป็นไปตามมาตรฐาน RS-323 ก่อนส่งออกไปจากคอมพิวเตอร์ สำหรับอุปกรณ์ต่อเชื่อมปลายทางก็จะต้องมีวงจรขับในลักษณะนี้เช่นเดียวกัน เพื่อให้ได้ระดับสัญญาณในระดับเดียวกัน แต่วงจรขับที่ใช้ทั้งภายในคอมพิวเตอร์และอุปกรณ์ต่อเชื่อมปลายทางนั้นจะถูกกลับสถานะ ดังแสดงเป็นบล็อกไดอะแกรมในรูปที่ 2.6



รูปที่ 2.6 ไดอะแกรมแสดงโครงสร้างทางฮาร์ดแวร์ของพอร์ตอนุกรม

2.6 แอดเดรสของพอร์ตอนุกรม

แอดเดรสพื้นฐานของพอร์ตอนุกรมมี 4 ตำแหน่งดังนี้คือ

COM1 : 3F8H

COM2 : 2F8H

COM3 : 3E8H

COM4 : 2E8H

เมื่อเริ่มเปิดเครื่องเพื่อใช้งานคอมพิวเตอร์ ไบออสภายในคอมพิวเตอร์จะทำการตรวจสอบแอดเดรสของพอร์ตอนุกรมทั้งหมด ถ้าไบออสตรวจพบแอดเดรสของพอร์ตอนุกรม ไบออสจะนำแอดเดรสที่ตรวจพบไปเก็บไว้ในหน่วยความจำขนาด 2 ไบต์ สำหรับพอร์ตอนุกรม COM1 จะเก็บไว้ที่แอดเดรส 0000:0400H และ 0000:0401H ส่วนตำแหน่งอื่น ๆ มีรายละเอียดดังนี้

COM2 = 0000:0402H - 0000:0403H

COM3 = 0000:0404H - 0000:0405H

COM4 = 0000:0406H - 0000:0407H

นอกจากนี้ที่หน่วยความจำแอดเดรส 0000:0411H ยังใช้สำหรับแสดงจำนวนของพอร์ตอนุกรมที่มีใช้อยู่ในคอมพิวเตอร์อีกด้วย โดยมีรายละเอียดดังแสดงในตารางที่ 2.3

ตารางที่ 2.3 แสดงข้อมูลในแอดเดรส 0000:0411H ที่ใช้แจ้งจำนวนพอร์ตอนุกรม

บิต 3	บิต 2	บิต 1	จำนวนพอร์ต
0	0	0	ไม่มีพอร์ตอนุกรม
0	0	1	มีพอร์ตอนุกรม 1 พอร์ต
0	1	0	มีพอร์ตอนุกรม 2 พอร์ต
0	1	1	มีพอร์ตอนุกรม 3 พอร์ต
1	0	0	มีพอร์ตอนุกรม 4 พอร์ต

เนื่องจากในปัจจุบันมีการใช้งานตามมาตรฐานการเชื่อมต่อแบบ RS-232-C กันอย่างแพร่หลาย ดังนั้นจึงขอกล่าวรายละเอียดเฉพาะมาตรฐานการเชื่อมต่อแบบ RS-232-C เท่านั้น ซึ่งเป็นมาตรฐานที่ถูกกำหนดโดย EIA ซึ่งเป็นองค์กรอุตสาหกรรมอิเล็กทรอนิกส์ของอเมริกา โดยแบ่งการเชื่อมต่อเป็น 2 ลักษณะคือ DTE (Data Terminal Equipment) และ DCE (Data Communication Equipment) ซึ่งโดยปกติ DTE จะต้องต่อเข้ากับ DCE เสมอ เช่น การต่อเครื่องคอมพิวเตอร์(อุปกรณ์ DTE) เข้ากับพอร์ต(อุปกรณ์ DCE) เป็นต้น

พอร์ตอนุกรม RS-232-C จะเป็นพอร์ตของเครื่องคอมพิวเตอร์ที่มีขาต่อ (connector) ทั้งประเภท 9 และ 25 ขา และเราเรียกกันว่าพอร์ต COM1: และ COM2: นั่นเอง ในความเป็นจริงพอร์ตอนุกรมไม่ได้ถูกควบคุมโดยตรงจาก CPU บนเมนบอร์ด แต่การสื่อสารทั้งหมดจะถูกจัดการโดย

ชิป UART (Universal Asynchronous Receiver/Transmitter) อีกทีหนึ่ง ซึ่งในปัจจุบันเบอร์ที่ใช้กันมากที่สุดคือเบอร์ 16550C ซึ่งเป็นเวอร์ชั่นที่ได้รับการแก้ไขข้อผิดพลาดแล้ว ซึ่งชิป UART นี้จะทำหน้าที่ในการรับและส่งข้อมูลดังต่อไปนี้

1. การส่งข้อมูล (Data Transmission)
   - แปลงตัวอักษรให้เป็นสายข้อมูลแบบบิต(เราเรียกว่าขบวนการ serialization)
   - สร้างเฟรมข้อมูลโดยการเพิ่มบิตที่จำเป็นสำหรับการสื่อสารและการตรวจสอบ เช่น บิต START, STOP และ PARITY เป็นต้น
   - ส่งผ่านเพรมข้อมูลที่สร้างขึ้นมาแล้วจากขั้นตอนที่ผ่านมา ด้วยความเร็วของพอร์ตอนุกรม (baud rate)
   - แสดงสถานะความพร้อมที่จะรับข้อมูลตัวอักษรถัดไปให้กับเครื่องคอมพิวเตอร์
2. การรับข้อมูล (Data Receiver)
   - รับตัวอักษรจากอินเตอร์เฟซ
   - ตรวจสอบความถูกต้องของเฟรมข้อมูล ตามมาตรฐานเฟรมที่กำหนด โดยถ้าหากเฟรมข้อมูลมีรูปแบบที่ไม่ถูกต้องก็จะมีการแจ้งข้อผิดพลาดทันที
   - ตรวจสอบความถูกต้องของพาริตี้
   - แปลงสายข้อมูลแบบบิตให้เป็นตัวอักษร
   - ส่งตัวอักษรให้กับเครื่องคอมพิวเตอร์
   - แสดงสถานะความพร้อมที่จะรับข้อมูลตัวอักษรถัดไปให้กับอินเตอร์เฟซ

2.7 รูปแบบของเฟรมข้อมูล (Data Format)

ในการสื่อสารระหว่างอุปกรณ์ 2 ตัวนั้น อุปกรณ์ทั้งสองจะต้องมีความเข้าใจถึงรูปแบบของเฟรมข้อมูลที่ตรงกัน ซึ่งการส่งข้อมูลแบบ Asynchronous นั้น นอกจากข้อมูลที่จะต้องส่งผ่านแล้ว อุปกรณ์นั้นยังต้องจัดการช่วงจังหวะในการส่งหรือรับข้อมูลระหว่างกันอีกด้วย โดยที่เฟรมข้อมูลในการส่งหรือรับจะประกอบด้วยบิตข้อมูลที่มีความหมายดังต่อไปนี้

- บิตข้อมูล (Data Bit)

  เมื่อชิป UART ได้รับตัวอักษรที่ส่งมาจากเครื่องคอมพิวเตอร์แล้ว ก็ต้องทำการแปลงตัวอักษรดังกล่าวให้เป็นสายข้อมูลชนิดบิตที่มีความยาวตั้งแต่ 5 ถึง 8 บิต ซึ่งเราเรียกขบวนการการเปลี่ยนแปลงตัวอักษรให้เป็นสายข้อมูลชนิดบิตนี้ว่า ขบวนการ Serialization จากนั้นพอร์ตก็จะทำการส่งแต่ละบิตไปยังอุปกรณ์ปลายทาง โดยจะเริ่มต้นส่งจากบิตที่มีนัยสำคัญต่ำสุด (least significant bit) ไปยังบิตที่มีนัยสำคัญสูงสุด (most significant bit)

บิตเริ่มต้นข้อมูล (Sart Bit)

ในการส่งข้อมูลแบบ Asynchronous นั้น เราจะต้องมีวิธีการบอกพอร์ตให้ทราบถึงจุดเริ่มต้นของข้อมูลที่ต้องการส่ง ดังนั้นก่อนหน้าข้อมูลในทุก ๆ เฟรมจะต้องถูกนำหน้าด้วยบิตเริ่มต้นข้อมูล Start Bit เสมอ

บิตสิ้นสุดข้อมูล (Stop Bit)

ในการส่งข้อมูลแบบ Asynchronous นั้น ในกรณีที่บิตเริ่มต้นข้อมูลเกิดการสูญหายในระหว่างการส่ง อุปกรณ์ปลายทางก็จะไม่สามารถทราบถึงจุดสิ้นสุดของสายข้อมูลบิตได้เลย เว้นแต่ว่าอุปกรณ์นั้นจะตรวจพบบิตเริ่มต้นข้อมูลใหม่อีกครั้งเท่านั้น ดังนั้นจึงมีการเพิ่มบิตสิ้นสุดข้อมูลต่อท้ายทุก ๆ ข้อมูลของแต่ละตัวอักษรเพื่อเป็นการแจ้งการสิ้นสุดของสายข้อมูลบิต โดยที่การเรียงกันของบิตเริ่มต้นข้อมูล สายข้อมูล และบิตสิ้นสุดข้อมูลเรียกว่าเฟรมของข้อมูล (data frame)

บิตพาริตี้ (Parity Bit)

เนื่องจากการส่งผ่านข้อมูลทางสายโทรศัพท์นั้น สามารถเกิดสัญญาณรบกวนได้ง่าย ด้วยเหตุนี้จึงเป็นไปได้มากที่สถานะของแต่ละบิตของข้อมูลที่ถูกส่งจะมีการเปลี่ยนแปลง เช่น จากบิต 0 เป็นบิต 1 เป็นต้น ในระหว่างการส่งผ่านข้อมูล ด้วยเหตุนี้จึงมีการคิดค้นวิธีที่ง่ายที่สุดที่ช่วยในการตรวจสอบความถูกต้องของสถานะของบิตข้อมูล ซึ่งเรียกกันว่า การตรวจสอบค่าพาริตี้ (parity check) โดยในการส่งผ่านข้อมูลด้วยโปรโตคอล START/STOP นั้น ในการส่งตัวอักษรข้อความทั่วไปจะใช้เพียง 7 บิตข้อมูลเท่านั้น ดังนั้นจึงมีการเพิ่มบิตพาริตี้ต่อสายเฟรมข้อมูลที่ถูกสร้างขึ้นมา ตามที่กล่าวมาแล้วในส่วนของบิตสิ้นสุดข้อมูล เพื่อตรวจสอบสถานะของผลบวกของบิตที่เป็น 1 ของสายบิตข้อมูลของแต่ละตัวอักษร สำหรับหลักการในการคำนวณค่าของบิตพาริตี้มีดังนี้

1. ถ้าหากจำนวนของบิตที่มีค่าเท่ากับ 1 ของสายบิตข้อมูลมีค่าเป็นเลขคู่ (even number) บิตพาริตี้จะมีค่าเท่ากับ 0
2. ถ้าหากจำนวนของบิตที่มีค่าเท่ากับ 1 ของสายบิตข้อมูลมีค่าเป็นเลขคี่ (odd number) บิตพาริตี้จะมีค่าเท่ากับ 1

บทที่ 3
ทฤษฎีของ I^2C

3.1 การเชื่อมต่อกับอุปกรณ์แบบ I^2C และการอินเทอร์รัปต์

ในระบบไมโครคอนโทรลเลอร์บางครั้งจะต้องใช้พอร์ตจำนวนมาก ซึ่งพอร์ตภายในของไมโครคอนโทรลเลอร์เองมีจำนวนจำกัด ถ้าหากต้องการขยายพอร์ตภายนอกหรือใช้อุปกรณ์ต่างๆ ภายนอกเพิ่มเติมจะต้องนำชิปไอซีมาต่อกับไมโครคอนโทรลเลอร์ เช่น ใช้ 74LS244 เป็นพอร์ตอินพุต ใช้ 74LS373 เป็นพอร์ตเอาต์พุต หรือใช้ชิป 8555 ซึ่งเป็นพอร์ตขนานที่สามารถโปรแกรมได้มาต่อ แต่การต่ออุปกรณ์ต่างๆ จะต้องใช้สายสัญญาณของไมโครคอนโทรลเลอร์จำนวนมาก จึงมีการออกแบบระบบบัสแบบใหม่ที่ใช้สายสัญญาณในการเชื่อมต่อน้อยลง

3.2 ระบบบัสแบบ I^2C

ระบบบัสแบบ I^2C ย่อมาจาก Inter-IC Communication ซึ่งพัฒนาโดยห้องวิจัยของ Phillips ในประเทศ Netherlands เมื่อปี ค.ศ. 1980 สำหรับเชื่อมต่อชิปต่างๆ กับระบบไมโครคอนโทรลเลอร์แบบอนุกรมโดยใช้สายสัญญาณเพียงสองเส้น เส้นหนึ่งเป็นสายข้อมูล อีกเส้นหนึ่งเป็นสายสัญญาณนาฬิกาสำหรับกำหนดจังหวะการสื่อสารข้อมูล ปัจจุบันได้มีชิปสนับสนุนการทำงานของไมโครคอนโทรลเลอร์หลายตัวที่ใช้การเชื่อมต่อระบบบัสแบบนี้

สายข้อมูลที่ใช้รับส่งข้อมูลแบบอนุกรมมีชื่อว่า Serial Data Line (SDA) ส่วนสายสัญญาณนาฬิกาควบคุมมีชื่อว่า Serial Clock Line (SCL) สายทั้งสองนี้จะรับส่งข้อมูลได้สองทิศทาง วงจรทางเอาต์พุตของอุปกรณ์ที่ใช้บัสแบบนี้จะเป็นแบบวงจรเดรนเปิด (open-drain) หรือคอลเล็คเตอร์เปิด (open-collector) ดังนั้นการต่อบัสแบบนี้จะต้องมีตัวต้านทานต่อพูลอัพกับแรงดันไฟ +5 โวลท์

ระบบบัสแบบนี้สามารถรับส่งข้อมูลได้สองทิศทางด้วยความเร็วสูงถึง 100 กิโลบิตต่อวินาที และสามารถใช้กับไอซีที่ใช้แรงดันไฟฟ้าต่างกันได้ ระบบบัสแบบนี้สามารถเชื่อมต่อกับอุปกรณ์ได้หลายตัวโดยที่อุปกรณ์แต่ละตัวที่ต่ออยู่กับระบบบัสแบบนี้สามารถส่งข้อมูลถึงกันได้ โดยใช้รูปแบบการรับส่งข้อมูลหรือโพรโตคอล (protocol) ที่อุปกรณ์ทุกตัวรู้จัก อุปกรณ์แต่ละชนิดจะมีค่าแอดเดรสประจำตัวมัน ถ้าหากต้องการให้อุปกรณ์ตัวใดรับข้อมูล ตัวส่งจะส่งแอดเดรสของอุปกรณ์ตัวนั้นออกไปก่อน ถ้าหากอุปกรณ์ตัวใดมีแอดเดรสตรงกันก็จะรับข้อมูลนั้นเข้าไป สำหรับ

อุปกรณ์ที่ต้องการส่งข้อมูลอุปกรณ์ตัวนั้นจะเรียกว่ามาสเตอร์(master) โดยจะเป็นตัวที่สร้างจังหวะสัญญาณต่าง ๆ บนระบบบัส ส่วนอุปกรณ์ที่ถูกควบคุมหรือเป็นตัวรับข้อมูลจะเรียกว่าสเลฟ(slave)

ถ้าหากทำการเชื่อมต่อไมโครคอนโทรลเลอร์กับอุปกรณ์ที่ใช้บัสแบบ I^2C หลายตัว ถ้าไมโครคอนโทรลเลอร์ต้องการส่งข้อมูล ตัวมันจะทำหน้าที่เป็นมาสเตอร์ และจะส่งสัญญาณไปบนบัส โดยส่งค่าแอดเดรสออกไปก่อน ถ้าหากค่าแอดเดรสนี้ตรงกับแอดเดรสของชิปตัวใด ชิปตัวนั้นจะส่งสัญญาณตอบรับ(ACK) ออกมา และชิปตัวนั้นก็จะถูกไมโครคอนโทรลเลอร์ควบคุม แต่ถ้าค่าแอดเดรสไม่ตรงกับชิปตัวใดก็จะไม่เกิดอะไรขึ้น

การส่งข้อมูลต่าง ๆ จะต้องเกิดขึ้นเมื่อระบบบัสว่างเท่านั้น โดยทั่วไปแล้วสภาวะที่มีบนระบบบัสแบบนี้จะมี 5 สภาวะ ดังรูปที่ 2.3 คือ

1.บัสว่าง (bus not busy) สภาวะนี้ค่าลอจิกบนสาย SDA และ SCL จะเป็นลอจิกสูงทั้งคู่

2.เริ่มส่งข้อมูล (start data transfer) สภาวะนี้สาย SCL จะเป็นลอจิกสูง แต่สาย SDA จะเปลี่ยนจากลอจิกสูงไปเป็นลอจิกต่ำ เรียกว่าสภาวะเริ่มต้น (start)

3.สภาวะหยุด (stop) สภาวะนี้สาย SCL จะเป็นลอจิกสูง แต่สาย SDA จะเปลี่ยนจากลอจิกต่ำไปเป็นลอจิกสูง

4.สภาวะมีข้อมูล (data valid) สภาวะนี้จะอยู่ระหว่างสภาวะเริ่มต้นและสภาวะหยุด โดยการรับส่งข้อมูลต่าง ๆ จะเกิดในสภาวะนี้ การรับส่งข้อมูลแต่ละบิตจะใช้สัญญาณนาฬิกาหนึ่งลูก โดยข้อมูลบน SDA จะต้องคงที่ ขณะที่ SCL เป็นลอจิกสูง และบิตข้อมูลใน SDA จะเปลี่ยนแปลงได้ขณะที่ SCL เป็นลอจิกต่ำ ถ้าหากบิตบน SDA มีการเปลี่ยนแปลงขณะที่ SCL เป็นลอจิกสูง ระบบจะตีความว่าเป็นสภาวะเริ่มต้นส่งข้อมูล หรือสภาวะหยุดแทน

5.สภาวะตอบรับ (acknowledge) เมื่ออุปกรณ์มาสเตอร์ส่งข้อมูลออกมาครบหนึ่งไบต์แล้ว ในช่วงสัญญาณ SCL ลูปที่ 9 มาสเตอร์จะส่งข้อมูลลอจิกสูงออกมา และถ้าตัวรับได้รับข้อมูลครบแล้วมันจะส่งสัญญาณตอบรับ ACK โดยทำให้ระดับลอจิกสูงบนสัญญาณ SDA ให้กลับเป็นระดับลอจิกต่ำ แต่ถ้าตัวรับได้รับข้อมูลไม่ถูกต้องตัวรับจะบังคับให้ตัวส่งหยุดในสภาวะรอ

ในการโอนถ่ายข้อมูลระหว่างสภาวะเริ่มและสภาวะหยุดสามารถโอนถ่ายข้อมูลได้ไม่จำกัด แต่เมื่อสเลฟได้รับข้อมูลแต่ละไบต์แล้ว มันจะส่งสัญญาณกลับมาทุกครั้ง

การเชื่อต่ออุปกรณ์ต่าง ๆ ที่ใช้ระบบบัสแบบ I^2C นั้นเราสามารถใช้อุปกรณ์หลาย ๆ ตัวมาต่อกับไมโครคอนโทรลเลอร์ได้ แต่ใช้สายสัญญาณทั้งหมดเพียงสองเส้นเท่านั้น ดังนั้นการติดต่อกับอุปกรณ์แต่ละตัวจะต้องระบุด้วยว่าจะต้องการติดต่อกับอุปกรณ์ตัวใด ขั้นตอนการเขียนข้อมูลที่ใช้กับระบบบัสแบบ I^2C มีสามขั้นตอนดังนี้

1.เขียนข้อมูลอ้างแอดเดรส (addressing) ข้อมูลไบต์แรกที่มาสเตอร์จะส่งออกไปคือข้อมูลที่ใช้อ้างแอดเดรสของอุปกรณ์ที่ต้องการติดต่อ โดยทั่วไปแล้วข้อมูลที่กำหนดแอดเดรสจะมีจำนวน 7 บิต

จากรูปจะเห็นว่าบิตต่ำสุดจะเป็นตัวระบุว่าอุปกรณ์ที่ต้องการติดต่อนั้นเราจะใช้อ่านหรือเขียนข้อมูลลงไป ถ้าบิตนี้เป็นลอจิก "0" หมายความว่าจะใช้เขียนข้อมูลลงไป ถ้าเป็นลอจิก "1" หมายความว่าจะใช้อ่านข้อมูล ส่วน 7 บิตบนจะเป็นบิตแอดเดรส ซึ่งแบ่งออกเป็นบิตแอดเดรสคงที่ (fixed address bit) จำนวน 4 บิตบน ซึ่งจะถูกโปรแกรมมาจากโรงงานที่ผลิตชิปแต่ละประเภท ส่วนอีก 3 บิตต่อมาจะเป็นบิตแอดเดรสที่ผู้ใช้สามารถโปรแกรมได้ (programmable address bit)

2.การเขียนไบต์ควบคุม (control byte) ข้อมูลไบต์นี้จะขึ้นกับอุปกรณ์แต่ละประเภท เพื่อกำหนดการทำงานต่าง ๆ ของตัวมัน อุปกรณ์บางประเภทอาจจะไม่ต้องมีการเขียนไบต์นี้ก็ได้

3.การเขียนไบต์ข้อมูล (data byte) เป็นข้อมูลที่จะโอนถ่ายในระบบ ระหว่างสภาวะเริ่มต้นและสภาวะหยุดจะมีการอ่านเขียนข้อมูลจำนวนกี่ไบต์ก็ได้

3.3 การเขียนโปรแกรมควบคุมบัส I^2C ด้วย Visual Basic

ตามที่ได้กล่าวไปแล้วว่าการโอนถ่ายข้อมูลจะต้องเกิดขึ้นเมื่อระบบบัสว่างเท่านั้น และการอ่านเขียนข้อมูลจะต้องมีการสร้างสภาวะต่าง ๆ ขึ้นมาด้วยดังไดอะแกรมเวลาในรูปที่ 2.3

จากส่วนของฮาร์ดแวร์จะเห็นว่า ขาที่ใช้งานในการติดต่ออุปกรณ์บนระบบบัส I^2C นั้นใช้ขาของพอร์ตอนุกรมทั้งหมด 3 ขา คือ DTR,RTS และ DCD ขาทั้งสามนี้สามารถควบคุมด้วยคอนโทรล MSComm1 ของ Visual Basic ซึ่งมีรูปแบบการใช้งานดังนี้

การกำหนดให้ขา SDA เป็น "1" ต้องเขียนคำสั่งเป็น MSComm1.RTSEnable = True

การกำหนดให้ขา SDA เป็น "0" ต้องเขียนคำสั่งเป็น MSComm1.RTSEnable = False

การกำหนดให้ขา SCL เป็น "1" ต้องเขียนคำสั่งเป็น MSComm1.DTREnable = True

การกำหนดให้ขา SCL เป็น "0" ต้องเขียนคำสั่งเป็น MSComm1.DTREnable = False

การอ่านค่าอินพุทของระบบบัส I^2C นั้นจะใช้คำสั่งเพื่อตรวจสอบค่าจาก CDHolding โดยจะต้องตรวจสอบว่าค่าที่ได้เป็น True หรือ False

3.3.1 โปรแกรมย่อยสร้างสัญญาณเริ่มต้นหรือ START

จากทฤษฎีในตอนต้นกำหนดไว้ว่า เมื่อต้องการติดต่อกับอุปกรณ์บนระบบบัส I^2C ในครั้งแรกจะต้องส่งสัญญาณเริ่มต้นหรือ START ออกไปก่อน โดยมีขั้นตอนการสร้างสัญญาณ START ดังนี้

1. กำหนดให้ขา SCL และ SDA มีลอจิกเป็น “1” ก่อนเพื่อให้อยู่ในสภาวะบัสว่าง

2. กำหนดให้ขา SDA มีลอจิก “0”

3. กำหนดให้ขา SCL มีลอจิก “0” ตาม

ซึ่งสามารถเขียนเป็นโปรแกรมด้วย Visual Basic ได้ดังนี้

```
Private Sub I2Cstart()
        MSComm1.RTSEnable = True    ' SDA=1
        MSComm1.DTREnable = True    ' SCL=1
        MSComm1.RTSEnable = False   ' SDA=0
        MSComm1.DTREnable = False   ' SCL=0
End Sub
```

3.3.2 โปรแกรมย่อยการสร้างสัญญาณหยุดหรือ Stop

การทำให้เกิดสถานะหยุดหรือสิ้นสุดการถ่ายทอดข้อมูลบนบัส I^2C หรือสถานะหยุด (Stop) ก็คือการทำให้บัสของ I^2C เข้าสู่สภาวะบัสว่างมีขั้นตอนดังนี้

1. กำหนดให้ขา SDA มีลอจิก “0” ก่อน

2. จากนั้นกำหนดให้ขา SCL มีลอจิกเป็น “1”

3. ทำให้ขา SDA มีลอจิกเป็น “1”

ซึ่งสามารถเขียนเป็นโปรแกรมด้วย Visual Basic ได้ดังนี้

```
Private Sub I2Cstop()
        MSComm1.RTSEnable = False   ' SDA=0
        MSComm1.DTREnable = True    ' SCL=1
        MSComm1.RTSEnable = True    ' SDA=1
End Sub
```

3.3.3 โปรแกรมย่อยรอสัญญาณรับรู้หรือ Acknowledge

เมื่อมีการส่งข้อมูลจากอุปกรณ์มาสเตอร์ไปยังอุปกรณ์ สเลฟ อุปกรณ์สเลฟจะตอบกลับสัญญาณที่ส่งออกไปนี้ด้วยสัญญาณ รับรู้หรือ Acknowledge (ACK) เพื่อเป็นการบอกให้รู้ว่าได้รับข้อมูลเรียบร้อยแล้ว แต่เมื่อมาเขียนโปรแกรมบน Visual Basic การทำงานที่มีลักษณะเป็นมัลติทาสกิ้ง(multi-tasking) การเขียนโปรแกรมเพื่อให้รอรับสัญญาณ ACK นั้นจะทำให้โปรแกรม

หยุดชะงัก จึงต้องเขียนโปรแกรมเสมือนว่าได้รับสัญญาณ ACK แทน โดยทำให้บัสว่างแล้วส่ง SCL เป็น “0” ออกไปเพื่อแสดงว่าได้รับ ACK แล้ว สามารถเขียนโปรแกรมได้ดังนี้

```
Private Sub Ack()
        MSComm1.RTSEnable = True    ‘ SDA=1
        MSComm1.DTREnable = True    ‘ SCL=1
        MSComm1.DTREnable = False   ‘ SCL=0
End Sub
```

3.3.4 โปรแกรมย่อยส่งข้อมูล

การส่งข้อมูลลอจิก “0” ก็คือการทำให้ขา SDA มีลอจิกเป็น “0” แล้วทำการส่งสัญญาณนาฬิกาออกไป โดยมีขั้นตอนดังนี้

1. ทำให้ขา SDA เป็น “0” สำหรับการส่งข้อมูลลอจิก “0”
2. ให้ขา SCL เป็น “1” สำหรับการป้อนสัญญาณนาฬิกาในขณะที่ SDA ยังคงเป็น “0”
3. ให้ขา SCL กลับสู่สถานะเดิมคือลอจิก “0”

สามารถเขียนโปรแกรมได้ดังนี้

```
Private Sub send0()
        MSComm1.RTSEnable = False   ‘ SDA=0
        MSComm1.DTREnable = True    ‘ SCL=1
        MSComm1.DTREnable = False   ‘ SCL=0
End Sub
```

สำหรับการส่งข้อมูลลอจิก “1” มีขั้นตอนดังนี้

1. ทำให้ขา SDA มีลอจิก “1” สำหรับการส่งข้อมูลลอจิก “1”
2. ทำให้ขา SCL เป็น “1” ในขณะ SDA ยังคงเป็น “1”
3. ทำให้ขา SCL มีลอจิก “0” เหมือนเดิม

สามารถเขียนโปรแกรมได้ดังนี้

```
Private Sub send1()
        MSComm1.RTSEnable = True    ‘ SDA=1
        MSComm1.DTREnable = True    ‘ SCL=1
```

```
    MSComm1.DTREnable = False   ' SCL=0
End Sub
```

3.3.5 โปรแกรมย่อยอ่านและส่งข้อมูลขนาด 8 บิต

ในการอ่านข้อมูลจากอุปกรณ์ในระบบบัส I^2C ส่วนใหญ่ข้อมูลที่อ่านได้จะมีขนาด 8 บิต โดยขั้นตอนการอ่านค่าข้อมูลมีดังนี้

1. กำหนดลูปในการรับค่าอินพุตไว้ 8 รอบสำหรับการอ่านค่าอินพิต 8 บิต
2. ให้บัสเป็นบัสว่างเพื่อรอรับสัญญาณจากอุปกรณ์ สเลฟ โดยกำหนดให้ SDA และ SCL มีลอจิก "1"
3. ทำการอ่านค่าข้อมูลจากขา SDA ผ่านคำสั่ง CDHolding
4. ถ้าอ่านค่าบิตเข้ามาแล้วมีค่าเป็น "1" ให้นำค่าบิตนั้น OR กับตัวแปรที่ทำการเก็บค่า แต่ถ้าค่าบิตที่อ่านได้เป็น "0" ไม่ต้องเปลี่ยนแปลงใด ๆ
5. กำหนดให้ขา SCL มีลอจิก "0" เพื่อเป็นการตอบรับข้อมูล
6. ถ้าข้อมูลที่ทำการอ่านค่ายังไม่ครบ 8 บิตให้ย้อนกลับไปอ่านค่าในบิตต่อ ๆ ไปจนครบ
7. นำค่าตัวแปรคืนค่าให้กับฟังก์ชัน

สามารถเขียนโปรแกรมได้ดังนี้

```
Private Function dat()
        Dim DAT1 As Byte
        Dim i as Integer
        For i = 7 To 0 Step -1
                MSComm1.RTSEnable = True    ' SDA=1
                MSComm1.DTREnable = True    ' SCL=1
                If (MSComm1.CDHolding And &H80) <> &H80 Then
                        DAT1 = 2 ^ i Or DAT1
                End If
                MSComm1.DTREnable = False   ' SCL=0
        Next I
        Dat = DAT1
End Function
```

การส่งค่าข้อมูลไปยังอุปกรณ์บนระบบบัส I^2C ปกติจะเป็นการส่งข้อมูลขนาด 8 บิตหรือ 1 ไบต์โปรแกรมย่อยต่อไปนี้จะเป็นการนำเอาข้อมูล 8 บิตมาทำการตรวจสอบว่าเป็นลอจิก “0” หรือ “1” แล้วทำการส่งข้อมูลเหล่านั้นออกไป

```
Private Sub Send8BIT(Add As Byte)
    Dim i as Integer
    For i = 7 To 0 Step –1                  ‘ Loop 8 Cycle
        If (Add And 2 ^ i) = 2 ^ i Then     ‘ Test Bit 0 OR 1
            Call Send1
        Else
            Call Send0
        End If
    Next i
End Sub
```

สำหรับขั้นตอนการติดต่อกับระบบบัสแบบ I^2C ของชิปทุกตัวจะมีจังหวะเวลาเหมือนกัน เราสามารถสร้างเป็นโปรแกรมย่อยขึ้นมาได้ ถ้าหากใช้ชิปตัวอื่นก็สามารถเรียกมาใช้ได้

บทที่ 4

การใช้ Microsoft Visual Basic 6.0

โปรแกรมภาษา Visual Basic (VB) นั้นเป็นภาษาที่พัฒนาขึ้นโดยบริษัทไมโครซอฟต์ เพื่อให้เป็นภาษาที่ใช้ในการพัฒนาโปรแกรมสำหรับ Windows และ Windows NT ซึ่งเป็นระบบปฏิบัติการ (Operating System) ของบริษัทดังกล่าวนั้นเช่นกัน VB สามารถพัฒนาโปรแกรมได้หลายอย่างด้วยกัน ตั้งแต่โปรแกรมธรรมดาทั่วไป โปรแกรมเกี่ยวกับฐานข้อมูล หรือโปรแกรมทางด้านอินเตอร์เน็ต เป็นต้น หลักการสำคัญของ VB คือการที่โปรแกรมเมอร์ ออกแบบหน้าจอสำหรับการติดต่อกับผู้ใช้ โดยใช้การวาดภาพ และวางองค์ประกอบต่าง ๆ ลงบนหน้าจอได้ตามความต้องการของโปรแกรมเมอร์ ซึ่งจะทำให้โปรแกรมเมอร์เห็นและแก้ไขหน้าจอได้ทันที ด้วยการเห็นสิ่งที่โปรแกรมเมอร์ต้องการให้ระบบงานประยุกต์เป็นไปตามนั้นหรือเรียกว่า What you see is what you get (WYSIWYG) เมื่อวางองค์ประกอบต่าง ๆ ของหน้าจอเสร็จ จะเป็นขั้นตอนการเขียนโปรแกรมซึ่งภาษาที่ใช้เขียนโปรแกรมนั้นเรียกว่า ภาษา Visual Basic (VB) เป็นโปรแกรมภาษาที่มีโครงสร้างของภาษา BASIC (Beginner's All-Purpose Symbolic Instruction Code) ที่จะสร้างความสัมพันธ์ระหว่างองค์ประกอบต่าง ๆ ของหน้าจอ เช่น เมนู ฟอร์ม รายงาน เข้าด้วยกันโดการพัฒนาแบบแยกส่วนนี้ทำให้การพัฒนาโปรแกรมด้วยเครื่องมือและภาษานี้เป็นไปอย่างรวดเร็ว

หลักการพัฒนาโปรแกรม VB อย่างหนึ่งที่ทำให้ VB เป็นภาษาที่มีความคล่องตัวและได้รับความนิยมอย่างสูงคือ Modularity ซึ่งแบ่งการทำงานของโปรแกรมออกเป็นส่วนย่อย ๆ แล้วนำมาต่อกัน ตัวอย่างเช่น Form ที่ถูกออกแบบและโปรแกรมให้ใช้กับระบบหนึ่งสามารถจะนำ Module นั้นไปใช้กับอีกระบบหนึ่งได้ทั้งหมด รวมถึงหน้าจอและโค้ด (Code) ของโปรแกรมต่าง ๆ ที่เขียนไว้แล้วอีกทั้งสามารถจะนำมาปรับเปลี่ยนเพื่อให้เหมาะสมกับงานนั้น ๆ มากขึ้น เป็นการช่วยประหยัดเวลาในการพัฒนาโปรแกรมได้

ในปัจจุบันผู้คนส่วนใหญ่จะทำงานกับระบบปฏิบัติการ Windows โดยมีภาพเป็นตัวสื่อสารระหว่างผู้ใช้กับเครื่องคอมพิวเตอร์ และโปรแกรมต่าง ๆ ดังนั้นการทำงานส่วนใหญ่จึงไม่ต้องการพิมพ์คำสั่ง เพียงแต่ใช้ Mouse เลือกคำสั่งหรือสัญลักษณ์รูป (Icon) ที่ต้องการ การใช้แนวความคิดเดียวกันนี้ในการเขียนโปรแกรมแบบ Object-Oriented คือการมองทุก ๆ องค์ประกอบในโปรแกรมเป็นวัตถุ (Object) หนึ่งชิ้น ซึ่งวัตถุแต่ละชิ้นจะมีคุณสมบัติแตกต่างกัน เช่น ปุ่มสามารถเลือกได้โดยการใช้ Mouse กรอบข้อมูล (Textbox) สามารถรับค่าจากแป้นพิมพ์ได้โดยตรง โดยการทำงานจะ

แตกต่างกันออกไปตามแต่เหตุการณ์ในขณะนั้น ดังนั้นภาษาโปรแกรมรุ่นใหม่จึงจำเป็นต้องพัฒนาไปตามแนวคิดแบบ Object-Oriented และเขียนโปรแกรมแบบ Event-Driven คือใช้เหตุการณ์และสภาพแวดล้อมในขณะนั้นเป็นตัวกำหนดความต้องการที่จะให้ Object แต่ละตัวมีการปฏิบัติอย่างไร และภาษา VB ก็เป็นภาษาที่พัฒนาภายใต้แนวคิดแบบ Event-driven เช่นกัน

4.1 ความต้องการของระบบสำหรับติดตั้งใช้งาน VB6

สำหรับเครื่องคอมพิวเตอร์ที่ใช้งาน VB6 จะต้องเป็นเครื่องคอมพิวเตอร์ที่มีรายละเอียดดังต่อไปนี้เป็นอย่าง

- ต้องมีระบบปฏิบัติการ Windows 95 หรือสูงกว่า หรือระบบปฏิบัติการ Windows NT4.0 หรือสูงกว่า
- หน่วยประมวลผลกลางรุ่น Pentium 90 MHz หรือสูงกว่า
- พื้นที่ในฮาร์ดดิสก์อย่างน้อย 50 เมกกะไบต์
- การ์ดจอ VGA 640x480 หรือสูงกว่าที่ Windows สนับสนุน
- RAM ไม่น้อยกว่า 24 MB สำหรับ Windows 95 และ 32 MB สำหรับ Windows NT4.0

4.2 หลักการในการเขียนโปรแกรมด้วย VB6

หลักการพื้นฐานที่ต้องทราบในการพัฒนาโปรแกรมด้วย VB6 คือ คอนโทรล คุณสมบัติ และการเขียนโปรแกรมแบบ Event-Driven เราสามารถแบ่งขั้นตอนการสร้างโปรแกรมใน Visual Basic ได้เป็น 2 ขั้นตอนหลักได้แก่

1. การออกแบบหน้าจอของโปรแกรม ซึ่งเป็นส่วนที่ทำหน้าที่ติดต่อกับผู้ใช้ (เรียกว่ายูสเซอร์อินเตอร์เฟส : User Interface)
2. การเขียนโปรแกรม ซึ่งใน Visual Basic เป็นการกำหนดคุณสมบัติของคอนโทรลบนฟอร์มให้เหมาะสม และการเขียนคำสั่งตอบสนองต่ออีเว็นต์

4.3 ไฟล์ประเภทต่าง ๆ ที่มีในโปรเจ็กต์ของ VB6

โปรเจ็กต์เป็นไฟล์ใช้เก็บฟอร์ม และโมดูลต่าง ๆ เช่น คลาสโมดูล โมดูลของ ActiveX Controls ซึ่งในโปรเจ็กต์หนึ่งจะมีไฟล์โมดูลต่าง ๆ อยู่ได้หลายไฟล์ โดยโปรเจ็กต์ที่เราสร้างขึ้นมาจะมีไฟล์ในรูปแบบต่าง ๆ ที่เป็นไปได้ดังต่อไปนี้

ไฟล์กลุ่มโปรเจ็กต์	คำอธิบาย	ส่วนขยายไฟล์
ไฟล์กลุ่มโปรเจ็กต์	เป็นไฟล์ที่ใช้เก็บว่ามีโปรเจ็กต์อะไรเก็บอยู่บ้าง (ต้องมากกว่า 1 โปรเจ็กต์ขึ้นไป)	.vbg
ไฟล์โปรเจ็กต์	เป็นไฟล์หลักโปรเจ็กต์ต่าง ๆ ของแอพพลิเคชั่นของเรา	.vbp
ไฟล์ของฟอร์ม	เป็นไฟล์ที่เก็บข้อมูลเกี่ยวกับฟอร์ม เช่น คอนโทรลต่าง ๆ และคำสั่งจัดการอีเวนต์ สำหรับฟอร์มนั้น ๆ เป็นต้น	.frm
ไฟล์ไบนารีของฟอร์ม	เป็นไฟล์ที่เก็บคุณสมบัติที่เป็นไบนารีของฟอร์ม เช่น รูปภาพ หรือ ไอคอน เป็นต้น	.frx
ไฟล์โมดูลมาตรฐาน	ส่วนใหญ่จะใช้เก็บค่าคงที่ ตัวแปร โปรแกรมย่อย ที่ให้โมดูลอื่น ๆ สามารถเรียกใช้งาน	.bas
ไฟล์คลาสโมดูล	ใช้ในการสร้างออบเจ็กต์ที่มีลักษณะต่าง ๆ ตามที่เราต้องการเองได้	.cls
ไฟล์ ActiveX	จะเป็นไฟล์ของคอนโทรล ActiveX ซึ่งเป็นคอนโทรลที่สร้างขึ้นมาเองได้ และสามารถนำไปใช้ในแอพพลิเคชั่นทั่วไปที่สร้างขึ้นมาใหม่ได้	.ctl
ไฟล์ ActiveX Documents	จะเป็นไฟล์ของแอพพลิเคชั่น ที่สามารถนำไปแสดงในโปรแกรมเว็บบราวน์เซอร์ได้	.dob
ไฟล์ Property Pages	เป็นไฟล์ของ Property Page ที่ใช้แสดงคุณสมบัติของคอนโทรลของเรา	.pag

4.4 การรับและส่งข้อมูลแบบอนุกรม

มีหลากหลายวิธีในการรับและส่งข้อมูลแบบอนุกรมผ่านพอร์ตอนุกรม RS-232 เช่นใช้คำสั่งพิมพ์ออกทางเครื่องพิมพ์ เรียกอินเตอร์รัปต์ของไบออสหรือของดอส การเขียนหรืออ่านไปยังแอดเดรสของพอร์ตโดยตรง วิธีสุดท้ายเป็นวิธีที่มีความยืดหยุ่นในการใช้งานที่สุด

แต่ในระบบปฏิบัติการวินโดว์ ได้เข้าฝังตัวพอร์ตอนุกรมเข้าเป็นส่วนหนึ่งของระบบปฏิบัติการแล้ว ดังนั้นการเรียกใช้งานจึงจำเป็นต้องเรียกผ่านเครื่องมือที่ติดต่อผ่านระบบปฏิบัติการ เช่นการใช้คอนโทรล MSCOMM32.OCX ของโปรแกรม Visual Basic

4.5 คอนโทรล MSComm

สำหรับการใช้งาน Visual Basic ตั้งแต่เวอร์ชัน 2 เป็นต้นมา ใน Visual Basic จะมี Custom Control สำหรับการสื่อสารอนุกรมผ่านทางพอร์ตอนุกรมของคอมพิวเตอร์มาให้ โดยใน Visual Basic เวอร์ชัน 2 และเวอร์ชัน 3 จะใช้ชื่อว่า MSCOMM.VBX ส่วนเวอร์ชัน 4 ใช้ชื่อว่า MSCOMM16.OCX สำหรับการทำงานกับระบบปฏิบัติการ 16 บิต และ MSCOMM32.OCX สำหรับการทำงานกับระบบปฏิบัติการ 32 บิต สำหรับใน Visual Basic เวอร์ชัน 5 จะมีเพียง MSCOMM32.OCX เท่านั้นเพราะถูกออกแบบมาให้ใช้งานกับระบบปฏิบัติการ 32 บิต

คอนโทรล MSComm (Communications) เป็นคอนโทรลตัวหนึ่งที่ช่วยในการติดต่อกับพอร์ทอนุกรม (serial port) ซึ่งสามารพทำการส่งข้อมูลผ่านทางพอร์ทอนุกรมได้ด้วยคอนโทรลนี้ เช่น การติดต่อโดยตรงกับบอร์ดอิเล็กทรอนิกส์ ซึ่งคอนโทรล MSComm ที่มากับ Visual Basic จะเป็นคอนโทรลที่ทำงานโดยมีการตอบสนองต่อเหตุการณ์แบบ event-driven นั่นก็คือคอนโทรลจะทำหน้าที่ตรวจสอบการเกิดขึ้นหรือการต้องขอให้เกิดเหตุการณ์เช่นเดียวกับคอนโทรลทั่วไปของ Visual Basic นั้นเอง ดังนั้นในการเขียนโค้ดจึงไม่จำเป็นต้องสร้างโพรซีเยอร์ที่ทำหน้าที่คอยตรวจสอบเหตุการณ์ต่าง ๆ ของพอร์ตอนุกรม ซึ่งจะทำให้ง่ายต่อการใช้งานเป็นอย่างมาก

MSComm จัดเตรียมทางเลือกเอาไว้ 2 ทางเพื่อความสะดวกในการสื่อสารข้อมูล ทางแรกคือ การสื่อสารข้อมูลที่กระตุ้นด้วยเหตุการณ์(event-driven communications) เป็นรูปแบบการใช้งานที่มีประสิทธิภาพมากสำหรับการตอบสนองแบบทันทีทันใด เช่น เมื่อตัวอักษรถูกส่งมาที่พอร์ตอนุกรมหรือเกิดการเปลี่ยนแปลงที่ขา Data Carrier Detect (DCD) หรือขา Request To Send (RTS) เหตุการณ์ Oncomm ของ MSComm จะสามารถตรวจจับสัญญาณนั้นได้ทันที ส่วนทางเลือกที่สองเป็นการคอยตรวจสอบค่าเหตุการณ์และความผิดพลาดที่เกิดขึ้นด้วยการดูค่าที่เปลี่ยนแปลงภายในคุณสมบัติ CommEvent หลังจากให้โปรแกรมทำงานในฟังก์ชั่นต่าง ๆ ไปเรียบร้อยแล้ว ซึ่งวิธีนี้ใช้งานได้ดีในกรณีที่โปรแกรมมีขนาดเล็ก

คอนโทรล MSComm 1 ตัวสามารถควบคุมการทำงานของพอร์ตอนุกรมได้ 1 พอร์ต ถ้าในโปรแกรมที่ใช้งานต้องการติดต่อกับพอร์ตอนุกรมมากกว่า 1 พอร์ตต้องใช้คอนโทรล MSComm มากกว่า 1 ตัวเพื่อควบคุมพอร์ตอนุกรมในแต่ละพอร์ต แอดเดรสของพอร์ตอนุกรมและแอดเดรสของการเกิดอิจเตอร์รัปต์สามารถเปลี่ยนแปลงได้จากการแก้ไขค่าที่ Control Panel

การเพิ่มคอนโทรล Microsoft Comm Control ทำได้โดย เรียกหน้าต่าง Components แล้วเลือกที่ Microsoft Comm Control 6.0

คอนโทรล MSComm จะมีหน้าที่มาตรฐานหลัก ๆ สำหรับการสื่อสารผ่านพอร์ตอนุกรม 3 ประการ ดังต่อไปนี้

- หมุนหมายเลขติดต่อกับโทรศัพท์ปลายทางที่กำหนด
- ตรวจสอบการเข้ามาของข้อมูลยังพอร์ตอนุกรมโดยอัตโนมัติ
- ส่งข้อมูลตามที่กำหนดจากโปรแกรมไปยังพอร์ตอนุกรม

ในความเป็นจริงคอนโทรล MSComm ไม่ได้ทำหน้าที่ติดต่อกับพอร์ตอนุกรมโดยตรง แต่มันจะทำหน้าที่เรียกใช้ฟังก์ชันวินโดว์ API ซึ่งวินโดว์จะทำการส่งหรือรับข้อมูลผ่านทางพอร์ตอนุกรมโดยอาศัยไดรเวอร์ Comm.drv อีกทอดหนึ่ง ดังนั้นจึงสามารถสรุปสั้น ๆ ได้ว่าทุกครั้งที่มีการเรียกใช้คอนโทรล MSComm ก็หมายถึงเรียกใช้ฟังก์ชันวินโดว์ API ซึ่งจะถูกตีความอีกทอดหนึ่งโดยไดรเวอร์ Comm.drv จากนั้นก็จะส่งผ่านข้อมูลที่ถูกจัดรูปแบบตามมาตรฐานสื่อสาร(ทั้งนี้ก็ขึ้นกับอุปกรณ์ที่ต่อเข้ากับพอร์ตอนุกรม) ให้กับดีไวซ์ไดรเวอร์อรกทอดหนึ่งนั่นเอง

เหตุการณ์ที่สำคัญของคอนโทรล MSComm คือ CommEvent จะรายงานเหตุกาณ์ทุกครั้งที่เกิดข้อผิดพลาดหรือมีการสื่อสาร ซึ่งสามารถอ่านค่าได้เฉพาะในขณะทำงานเท่านั้น รูปแบบคำสั่งคือ object.CommEvent

คอนโทรล MSComm จะมีการเรียกโพรซีเยอร์เหตุการณ์ OnComm ทุกครั้งที่เกิดข้อผิดพลาดหรือมีการสื่อสารเกิดขึ้น ซึ่งค่าตัวเลขที่จำนวนเต็ม แสดงถึงขอ้ผิดพลาดหรือเหตุการณ์ที่มีการสื่อสารดังกล่าว ก็จะถูกตัดเก็บเอาไว้ในคุณสมบัติ CommEvent เสมอ ดังนั้นถ้าหากต้องการตรวจสอบข้อผิดพลาดหรือเหตุการณ์ที่มีการสื่อสารภายในโพรซีเยอร์เหตุการณ์ OnComm ก็ควรที่จะใช้ค่าตัวเลขจากคุณสมบัติ CommEvent ในการตรวจสอบเสมอ ซึ่งค่าตัวเลขที่รายงานโดยคุณสมบัติ CommEvent มีดังต่อไปนี้

ค่าคงที่สำหรับคุณสมบัติ Error

ค่าคงที่	ค่า	รายละเอียด
ComEventBreak	1001	ได้รับสัญญาณการหยุด(Break signal)
ComEventCTSTO	1002	สายสัญญาณ Clear To Send อยู่ในสถานะ low (timeout) ในขณะที่พยายามจะส่งออกตัวอักษร
comEventDSRTO	1003	สายสัญญาณ Data Set Ready อยู่ในสถานะ

		low (timeout) ในขณะที่พยายามจะส่งออกตัวอักษร
comEventFrame	1004	เฟรมของข้อมูลไม่ถูกต้องซึ่งถูกตรวจพบโดยฮาร์ดแวร์
comEventOverrun	1006	เกิด Port Overrun หมายถึง มีการรับตัวอักษรตัวใหม่เข้ามาในขณะที่ตัวอักษรก่อนหน้านี้ยังไม่ถูกอ่านจากฮาร์ดแวร์ ดังนั้นจึงเกิดการสูญหายของตัวอักษร
comEventCDTO	1007	สายสัญญาณ Carrier Detect อยู่ในสถานะ low (timeout)ในขณะที่พยายามจะส่งออกตัวอักษร
comEventRxOver	1008	Receive Buffer Overflow หมายถึงขนาดของบัฟเฟอร์ด้านรับเข้าข้อมูล (receive buffer) ไม่เพียงพอกับขนาดของข้อมูลที่รับเข้ามา
comEventRxParity	1009	Parity Error ซึ่งถูกตรวจพบโดยฮาร์ดแวร์
comEventTxFull	1010	Transmit Buffer Overflow หมายถึงขนาดของบัฟเฟอร์ด้านส่งออกข้อมูล (transmit buffer) เต็ม ในขณะที่พยายาม จัดเก็บข้อมูลใหม่ลงบัฟเฟอร์
comEventDCB	1011	Unexpected error หมายถึง เกิดข้อผิดพลาดที่ไม่ได้ถูกนิยามเอาไว้ในขณะอ่าน Device Control Block (DCB) จากพอร์ตอนุกรม

ค่าคงที่สำหรับคุณสมบัติ OnComm

ค่าคงที่	ค่า	รายละเอียด
comEvSend	1	มีจำนวนตัวอักษรในบัฟเฟอร์ด้านส่งออกข้อมูลน้อยกว่าจำนวนตัว อักษรที่กำหนดในคุณสมบัติ
comEvReceive RThreshold	2	การรับเข้าจำนวนตัวอักษรที่ถูกกำหนดในคุณสมบัติ ซึ่งจะเกิดขึ้นอย่างต่อเนื่องจนกว่าผู้อ่านจะใช้คุณสมบัติ Input ในการ อ่านข้อมูลจากบัฟเฟอร์สำหรับรับเข้าข้อมูล
comEvCTS	3	มีการเปลี่ยนสถานะของสายสัญญาณ Clear TO Send
comEvDSR	4	มีการเปลี่ยนสถานะของสายสัญญาณ Data Set Ready ซึ่งเหตุ

		การณ์นี้จะเกิดขึ้นเมื่อมีการเปลี่ยนสถานะของสายสัญญาณ Data Set Ready จาก 1 เป็น 0 เท่านั้น
ComEvCD	5	มีการเปลี่ยนสถานะของสายสัญญาณ Carrier Data
ComEvRing	6	มีการตรวจพบการเรียกหมายเลข(สัญญาณเสียงกริ่ง) ซึ่ง UART บาง ตัวอาจจะไม่สนับสนุนคุณสมบัตินี้
comEvEOF	7	มีการรับตัวอักษรรหัสจุดสิ้นสุดของไฟล์ (EOF, ASCII26)

คุณสมบัติที่สำคัญ คือ

CommPort เป็น รายงานหรือกำหนดหมายเลขของพอร์ตอนุกรมของเครื่องคอมพิวเตอณีที่ต้องการติดต่อ สำหรับหมายเลขพอร์ตอนุกรมสามารถมีค่าได้ตั้งแต่ 1 ถึง 16 (ค่าปกติจะเท่ากับ 1) ซึ่งก่อนที่จะเปิดพอร์ตด้วยคุณสมบัติ Port Open ต้องกำหนดหมายเลขพอร์ตอนุกรมให้กับคุณสมบัติ CommPort เสียก่อน โดยถ้าหากหมายเลขของพอร์ตอนุกรมที่กำหนดให้กับคุณสมบัติ CommPort ไม่เป็จความจริง ก็จะเกิดข้อผิดพลาด 68 (Device unavailable) ทันที ซึ่งในกรณีนี้สามารถแก้ไขได้โดยการกำหนดหมายเลขของพอร์ตอนุกรมที่ถูกต้องเสียใหม่ แล้วจึงทำการเปิดพอร์ตอนุกรมอีกครั้งด้วยคุณสมบัติ Port Open

InBufferSize เป็น รายงานหรือกำหนดขนาดของบัฟเฟอร์ด้านรับเข้า ซึ่งมีหน่วยเป็นไบต์(โดยปกติ 1 ไบต์จะเท่ากับ 1 ตัวอักษร) ในการเลือกขนาดของบัฟเฟอร์ด้านรับเข้าที่เหมาะสมนั้นในทางปฏิบัติเป็นสิ่งที่ยากมาก ทั้งนี้ขึ้นอยู่กับความหนาแน่นของข้อมูลที่มีการส่งไปมา ในแต่ละครั้งและความเร็วของการสื่อสาร(transmission rate) ของพอร์ต ซึ่งโดยปกติโปรแกรมเมอร์โดยทั่วไปจะกำหนดให้มีค่าเท่ากับ 1,024 ไบต์ (1 KB) โดยถ้าหากเกิดข้อผิดพลาด overflow ในขณะรันแอพพลิเคชัน ก็ให้ทำการเพิ่มขนาดของบัฟเฟอร์ด้านรับเข้าต่อไป

InputMode คือ การกำหนดชนิดของข้อมูลที่จะถูกอ่าน โดยคุณสมบัติ Input จากบัฟเฟอร์ด้านรับเข้า โดย comInputModeText เป็นข้อมูลชนิดข้อความทั่วไป (default) และ comInputModeBinary เป็นข้อมูลชนิดไบนารี

การกำหนดชนิดของข้อมูลที่จะถูกอ่านโดยคุณสมบัติ Input ก็ขึ้นกับลักษณะของงานที่กำลังควบคุม โดยปกติถ้าหากข้อมูลชนิดตัวอักษร ANSI ทั่วไป ก็จะกำหนดให้เป็นข้อมูลชนิดข้อความ (comInputModeText) แต่ถ้าหากเป็นข้อมูลที่ประกอบด้วยตัวอักษรควบคุม(แอสกีตั้งแต่ 0 ถึง 31) ก็จะกำหนดให้เป็นข้อมูลชนิดไบนารี(comInputModeBinary)

OutBufferSize เป็นการกำหนดขนาดของบัฟเฟอร์ด้านส่งออกซึ่งมีหน่วยเป็นไบต์(โดยปกติ 1 ไบต์จะเท่ากับ 1 ตัวอักษร) ปกติซอฟต์แวร์จะกำหนดให้มีค่าเท่ากับ 512 ไบต์

ในการเลือกขนาดของบัฟเฟอร์ด้านส่งออกที่เหมาะสมนั้น ในทางปฏิบัติเป็นสิ่งที่ยากมากเช่นกัน ทั้งนี้ก็ขึ้นอยู่กับความหนาแน่นของข้อมูลที่มีการส่งไปมา ในแต่ละครั้งและวามเร็วของการสื่อสารของพอร์ต ซึ่งโดยปกติโปรแกรมเมอร์โดยทั่วไปจะกำหนดให้มีค่าเท่ากับ 512 ไบต์ โดยถ้าหากเกิดข้อผิดพลาด overflow ในขณะรันแอพพลิเคชัน ก็ให้ทำการเพิ่มขนาดของบัฟเฟอร์ด้านรับเข้าต่อไป

RThreshold คือการกำหนดจำนวนตัวอักษรที่จะรับเข้าก่อนที่คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเท่ากับ commEvReceive และมีการเรียกโพรซีเยอร์เหตุการณ์ OnComm

ถ้าหากคุณสมบัติ Rthreshold มีค่าเท่ากับ 0 (default) ก็จะเป็นการยกเลิกการเรียกโพรซีเยอร์เหตุการณ์ OnComm เมื่อมีการรับเข้าตัวอักษรมายังบัฟเฟอร์ด้านรับเข้า และในทางกลับกันถ้าหากคุณสมบัติ Rthreshold มีค่า 1 ก็จะมีการเรียกโพรซีเยอร์เหตุการณ์ OnComm ทุกครั้งที่มีการรับเข้าตัวอักษรมายังบัฟเฟอร์ด้านรับเข้า

Setting คือการกำหนดพารามิเตอร์ในการสื่อสารผ่านพอร์ตอนุกรม การกำหนดพารามิเตอร์ในการสื่อสารผ่านพอร์ตอนุกรมดังต่อไปนี้

รูปแบบของการกำหนดลำดับของพารามิเตอร์ในการสื่อสารผ่านพอร์ตอนุกรม สำหรับคุณสมบัติ Setting จะต้องเรียงลำดับดังนี้ “BBBB,P,D,S” (โดยปกติทั่ว ๆ ไป จะมีค่าเท่ากับ “9600,N,8,1”) เพราะถ้าหากลำดับไม่ถูกต้องก็จะทำให้เกิดข้อผิดพลาดหมายเลข 380 (Invalid property value) ทันที ซึ่งสัญลักษณ์แต่ละตัวมีความหมายดังนี้ BBBB คือความเร็วของการส่งถ่ายข้อมูลในหน่วยของ baud rate ซึ่งในทางปฏิบัติ 1 baud rate อาจจะมีค่าเท่ากับ 1 bps (bit per second) หรือมากกว่าก็ได้ สำหรับค่าของ baud rate ที่คอนโทรล MSComm สามารถรับได้มีค่าดังต่อไปนี้ 110, 300, 600, 1200, 2400, 9600(default), 14400, 19200, 28800, 38400(reserved), 56000(reserved), 128000(reserved) หรือ 256000(reserved)

P คือ บิตพาริตี้(parity bit) สำหรับใช้ในการตรวจสอบความถูกต้องของข้อมูล โดยกำหนด E คือ Even Parity, M คือ Mark Parity, N คือ None (Default), O คือ Odd Parity และ S คือ Space

D คือ ขนาดของบิตข้อมูล ซึ่งสามารถกำหนดได้เป็น 4,5,6,7 หรือ 8 (default)

S คือ ขนาดของบิตหยุด(stop bit) ซึ่งสามารถกำหนดได้เป็น 1 (default), 1.5 หรือ 2

SThreshold เป็นการกำหนดจำนวนตัวอักษรที่น้อยที่สุดที่ถูกจัดเก็บในบัฟเฟอร์ด้านส่งออก ก่อนที่คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเท่ากับ comEvSend และมีการเรียกโพรซีเยอร์เหตุการณ์ OnComm

ถ้าหากคุณสมบัติ SThreshold มีค่าเท่ากับ 0 (default) ก็จะเป็นการยกเลิกการเรียกโพรซีเยอร์เหตุการณ์ OnComm เมื่อมีการส่งออกตัวอักษรไปยังบัฟเฟอร์ด้านส่งออก และในทางกลับกัน ถ้าหากคุณสมบัติ SThreshold มีค่าเท่ากับ 1 ก็จะมีการเรียกโพรซีเยอร์เหตุการณ์ OnComm เมื่อบัฟเฟอร์ด้านส่งออกว่าง

โดยถ้าหากจำนวนของตัวอักษรในบัฟเฟอร์ด้านส่งออกน้อยกว่าค่าตัวเลขที่กำหนดคุณสมบัติ CommEvent ก็จะมีค่าเท่ากับ comEvSend และพร้อมทั้งเกิดการเรียกโพรซีเยอร์เหตุการณ์ OnComm ทันที เช่น สมมติให้คุณสมบัติ SThreshold มีค่าเท่ากับ 10 และถ้าหากจำนวนตัวอักษรในบัฟเฟอร์ด้านส่งออกลดลงจาก 10 เป็น 9 ก็จะเกิดเหตุการณ์ OnComm ทันที เป็นต้น

คำสั่งที่สำคัญคือ

PortOpen คือการกำหนดสถานะการเปิด(open) หรือปิด(close) ของพอร์ตอนุกรมของเครื่องคอมพิวเตอร์ รูปแบบการใช้งานคือ

object.PortOpen[=boolean]

Boolean หมายถึง ข้อมูลชนิดบูลีน ที่กำหนดสถานะของพอร์ตอนุกรม ดังต่อไปนี้

True หมายถึง พอร์ตอนุกรมถูกเปิด

False หมายถึง พอร์ตอนุกรมถูกปิด(default)

พอร์ตอนุกรมถูกปิดโดยอัตโนมัติโดยคอนโทรล MSComm เมื่อแอพพลิเคชันสิ้นสุดการทำงาน โดยถ้าหากหมายเลขของพอร์ตอนุกรมที่กำหนดให้เปิดไม่มีการติดตั้งอยู่จริง ก็จะเกิดข้อผิดพลาดหมายเลข 68 (Device unavailable) ทันที

Input เป็นการอ่านข้อมูลจากบัฟเฟอร์ พร้อมทั้งลบข้อมูลในบัฟเฟอร์ด้านรับเข้าทิ้ง ซึ่งสามารถอ่านค่าได้เฉพาะในขณะทำงานเท่านั้น รูปแบบการใช้งานคือ object.Input

ทุกครั้งที่มีการใช้คุณสมบัติ Input ในการอ่านข้อมูลจากบัฟเฟอร์ด้านรับเข้านั้น จำนวนของตัวอักษรที่อ่านได้จะถูกกำหนดลงในคุณสมบัติ InputLen1 ทันทีซึ่งถ้าหากกำหนดให้คุณสมบัติ InputLen มีค่าเท่ากับ 0 ก็จะหมายถึงการกำหนดให้คุณสมบัติ Input อ่านข้อมูลทั้งหมดจากบัฟเฟอร์ด้านรับเข้านั่นเองซึ่งชนิดของข้อมูลที่อ่านโดยคุณสมบัติ Input จะเป็นข้อมูลแบบข้อความหรือไบนารี ก็ขึ้นกับการกำหนดค่าของคุณสมบัติ InputMode

Output คือการส่งข้อมูลไปยังบัฟเฟอร์ด้านส่งออก ซึ่งสามารถกำหนดค่าได้เฉพาะในขณะทำงานเท่านั้น รูปแบบการใช้งานคือ object.Output[=value]

สำหรับชนิดของข้อมูลที่ถูกส่งโดยคุณสมบัติ Output จะเป็นข้อมูลแบบข้อความหรือไบนารี่ ก็ขึ้นกับการกำหนดค่าของคุณสมบัติ InputMode ดังที่กล่าวมาแล้วข้างต้น

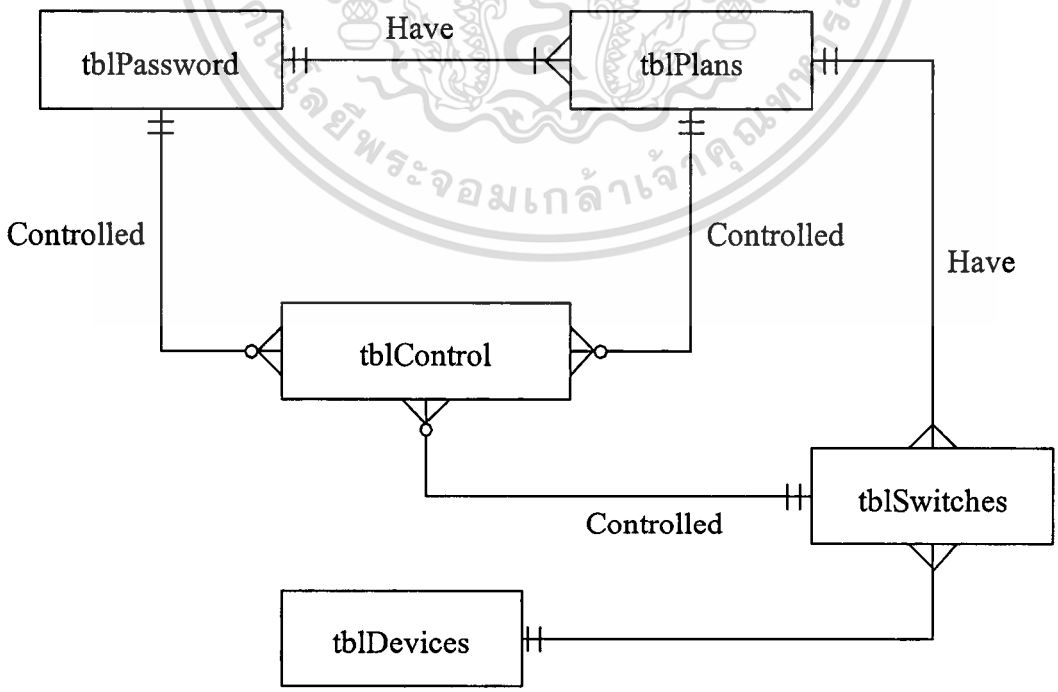
บทที่ 5

การวิเคราะห์และออกแบบระบบงาน

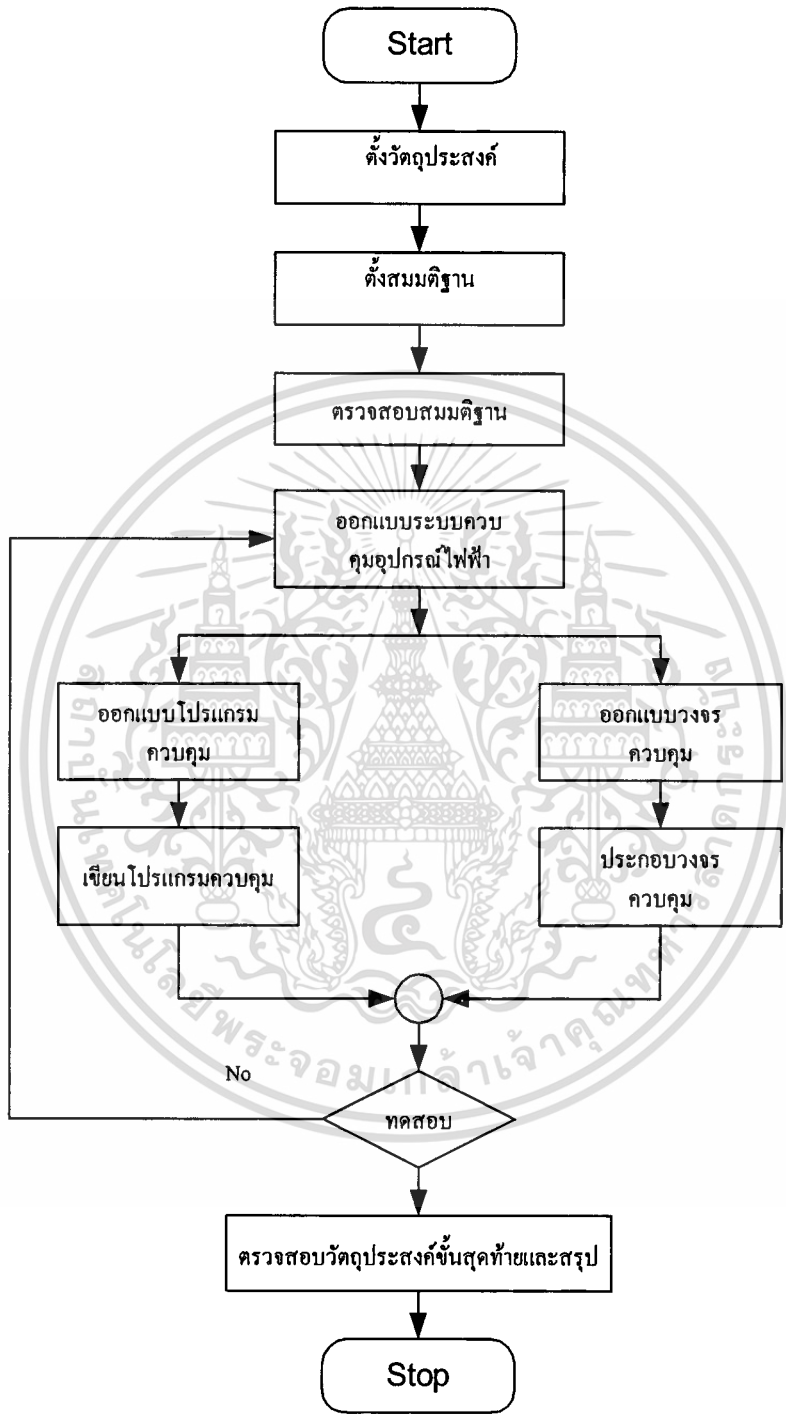
5.1 การวิเคราะห์ระบบงาน

เราสามารถดูภาพรวมของระบบได้จาก Context Diagram และ ER Diagram รวมถึง Flowchart การดำเนินงาน ดังรูปที่ 5.1, 5.2 และ 5.3 ตามลำดับ

รูปที่ 5.1 แสดง Context Diagram



รูปที่ 5.2 แสดง ER Diagram



รูปที่ 5.3 แสดง Flowchart การดำเนินงาน

5.2 Data Dictionary

ตารางที่อธิบายถึงรายละเอียดข้อมูลจากการวิเคราะห์ทำให้ได้ตารางในฐานข้อมูล ซึ่งมีรายละเอียดตังตารางต่อไปนี้

ตารางที่ 5.1 แสดงตาราง USER

แฟ้มข้อมูล :	tblPassword		
คำอธิบาย :	แฟ้มข้อมูลที่เก็บรายละเอียดของผู้ใช้งาน		
ฟิลด์	คีย์	ประเภท	หมายเหตุ
User_ID	PK	Char(4)	รหัสผู้ใช้งาน
Name_Surname		Char(30)	ชื่อนามสกุล
Username		Char(10)	ชื่อผู้ใช้งาน
Password		Char(10)	รหัสผ่าน
UserLeval		Char(10)	Access Level

ตารางที่ 5.2 แสดงตาราง PLAN

แฟ้มข้อมูล :	tblPlans		
คำอธิบาย :	แฟ้มข้อมูลที่เก็บรายละเอียดของผู้ใช้งาน		
ฟิลด์	คีย์	ประเภท	หมายเหตุ
Plan_ID	PK	Char(4)	รหัสแบบแปลน
Planname		Char(30)	ชื่อแบบแปลน
Planfilename		Char(10)	ชื่อไฟล์แบบแปลน
User_ID	FK	Char(4)	รหัสผู้ใช้

ตารางที่ 5.3 แสดงตาราง SWITCH

แฟ้มข้อมูล :	tblSwitches		
คำอธิบาย :	แฟ้มข้อมูลที่เก็บรายละเอียดของผู้ใช้งาน		
ฟิลด์	คีย์	ประเภท	หมายเหตุ
Switch_ID	PK	Char(2)	รหัสอุปกรณ์ไฟฟ้า
Control_IC		Char(20)	ชื่ออุปกรณ์ไฟฟ้า
SwitchValue		Datetime()	วันที่ติดตั้ง
SwitchLocation		Datetime()	เวลาที่ติดตั้ง
Device_ID	FK	Char(1)	สถานะการทำงาน
Switchstatus		Char(1)	สถานะสวิทช์
Pland_ID	FK	Char(4)	รหัสแบบแปลน

ตารางที่ 5.4 แสดงตาราง CONTROLLING

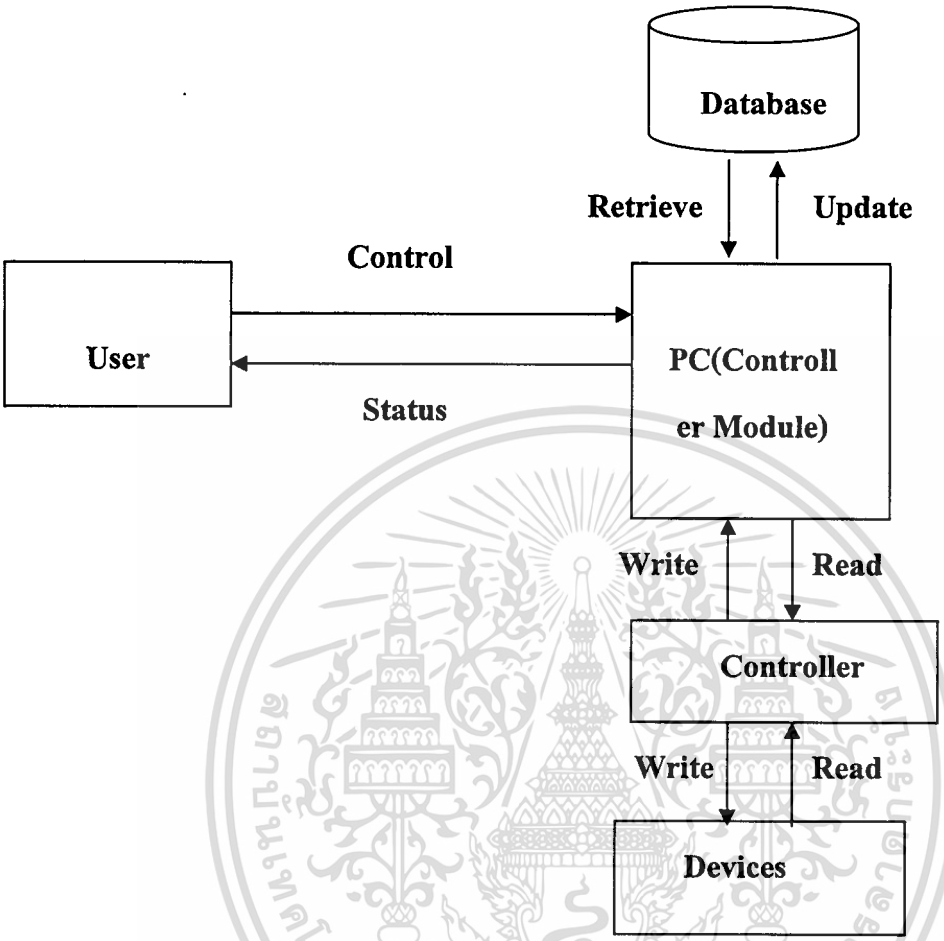
แฟ้มข้อมูล :	CONTROLLING		
คำอธิบาย :	แฟ้มข้อมูลที่เก็บรายละเอียดของผู้ใช้งาน		
ฟิลด์	คีย์	ประเภท	หมายเหตุ
User_ID	FK	Char(4)	รหัสผู้ใช้งาน
Plan_ID	FK	Char(4)	รหัสแบบแปลน
Switch_ID	FK	Char(4)	รหัสอุปกรณ์ไฟฟ้า
OpenDate		Datetime()	วันที่เปิด
OpenTime		Datetime()	เวลาที่เปิด
CloseDate		Datetime()	วันที่ปิด
CloseTime		Datetime()	เวลาที่ปิด

ตารางที่ 5.5 แสดงตาราง DEVICE

แฟ้มข้อมูล :	tblDevices		
คำอธิบาย :	แฟ้มข้อมูลที่เก็บรายละเอียดของผู้ใช้งาน		
ฟิลด์	คีย์	ประเภท	หมายเหตุ
Device_ID	PK	Char(4)	รหัสแบบแปลน
Devicename		Char(30)	ชื่อแบบแปลน
Devicelocation		Char(10)	ชื่อไฟล์แบบแปลน
Devicestatus		Char(1)	สถานะอุปกรณ์
PositionX		Number	ระยะห่างจากซ้าย
PositionY		Number	ระยะห่างจากบน

5.3 โครงสร้างภายในของระบบ

ในระบบควบคุมนี้จะต้องอาศัยส่วนต่าง ทำงานร่วมกัน มีการประสานการทำงานซึ่งกันและกัน โดยมีลักษณะเป็นลำดับขั้นตอน ซึ่งมีรูปแบบแสดงถึงผังงานการติดต่อภายในระบบ ดังรูปที่ 5.4



รูปที่ 5.4 แสดงผังงานลักษณะการติดต่อภายในระบบ

จากรูปที่ 5.4 ได้แสดงถึงแผนผังโครงสร้างของระบบสามารถอธิบายได้ดังนี้

- User

ในส่วนนี้เป็นส่วนที่สร้างขึ้นมาให้สำหรับผู้ใช้งานโดยผู้ใช้งานจะทำการกำหนดการเปิด-ปิดอุปกรณ์ไฟฟ้า ณ เวลาใดหรือตำแหน่งใด เป็นต้น โดยค่าของข้อมูลที่ผู้ใช้งานได้กำหนดลงไปนั้นจะถูกนำไปเก็บไว้ในฐานข้อมูล และสถานะของการสั่งงานจะปรากฏที่จอภาพ

- Database

ในส่วนนี้จะเป็นฐานข้อมูลของระบบควบคุมเพื่อเก็บข้อมูลจากการใช้งานต่าง ๆ ทั้งหมด โดยในโครงงานนี้จะใช้ Microsoft Access ช่วยในการจัดการด้านฐานข้อมูลของระบบนี้ ซึ่งอาจจะแยกเก็บรายระเอียดเป็นลักษณะตารางหลาย ๆ ตาราง โดยมีความสัมพันธ์ต่อกัน อย่างเช่น ตาราง Control อาจจะเก็บรายละเอียดในการควบคุมอุปกรณ์ไฟฟ้า เช่น รายการตั้งเวลาในการควบคุมอุปกรณ์ไฟฟ้าในระบบเป็นต้

- Controller Module(PC)

เป็นส่วนที่ทำหน้าที่เป็นตัวกลางในการติดต่อกันระหว่างซอฟต์แวร์กับฮาร์ดแวร์ ส่วนนี้ จะทำหน้าที่ดึงข้อมูลจากฐานข้อมูลที่มีการกำหนดค่าไว้ โดยแปลงค่าของข้อมูลเหล่านั้นไปยังวงจรควบคุม โดยจะติดต่อกันผ่านพอร์ทอนุกรมโดยใช้ I^2C บัส

- Controller

อุปกรณ์ควบคุมเป็นอุปกรณ์ทางด้านฮาร์ดแวร์ หน้าที่ของอุปกรณ์นี้ก็คือ แปลงค่าข้อมูลจากส่วน Controller Module ไปเป็นสัญญาณดิจิตอลคือ 0, 1 เพื่อนำไปควบคุมอุปกรณ์ไฟฟ้าในการเปิด-ปิด อุปกรณ์ไฟฟ้าต่าง ๆ และ อ่านค่าอินพุทจากอุปกรณ์ภายนอกเพื่อแสดงสถานะการติดหรือดับของอุปกรณ์นั้นๆ

- Devices

เป็นอุปกรณ์ไฟฟ้าที่จะทำกรควบคุม เช่น หลอดไฟ พัดลม เป็นต้น จะมีเพียงสถานะการทำงานคือ เปิดและปิด เท่านั้น

5.4 แนวคิดในการออกแบบระบบควบคุม

ระบบนี้สามารถแยกการทำงานได้เป็น 2 ส่วนดังนี้

- ส่วนการเก็บข้อมูล

เป็นส่วนที่เกี่ยวกับการเก็บข้อมูลระบบเพื่อการควบคุม ได้แก่ การเริ่ม Import แบบบ้านหรืออาคารเข้ามาในโปรแกรม ซึ่งจะทำการใส่รายละเอียดของอุปกรณ์ไฟฟ้าหรือดวงไฟต่างๆ โดยจัดเก็บไว้ในรูปแบบของฐานข้อมูล รวมถึงการแสดงผลของโปรแกรม นอกจากนั้นส่วนนี้ยังมีหน้าที่ติดต่อกับผู้ใช้งาน เมื่อมีการใช้งานโดยผู้ใช้สามารถที่จะกำหนดหรือเลือกแบบอาคารหรือบ้านเข้ามาในโปรแกรมจากนั้นก็จะทำการกำหนดตำแหน่งดวงไฟลงไปโดยอาศัยความสามารถของ Microsoft Visual Basic ในการสร้าง Control ในขณะ Runtime เพื่อใช้เป็นตำแหน่งดวงไฟดังกล่าว และเนื่องจากการที่มีคุณสมบัติเป็น Control จึงทำให้สามารถที่จะจัดเก็บ Properties ต่างๆ สำหรับ Control นั้น ๆ ไว้ในฐานข้อมูลเพื่อที่จะสามารถเรียกใช้ใหม่ภายหลังได้ เช่น ตำแหน่งของดวงไฟที่อยู่บนแบบบ้าน ชนิดของดวงไฟว่าเป็นประเภทไหน หรือแม้แต่ระยะเวลาที่จะต้องการให้เปิดหรือปิดไฟโดยระบบนี้สามารถที่จะกำหนดตำแหน่งโคมไฟภายในบ้านได้เอง และมีข้อมูลที่เป็น Properties ของไฟล์เหล่านั้นนำไปเก็บอยู่ในรูปของฐานข้อมูลที่สร้างขึ้นขณะโปรแกรมทำงานได้ และยังสามารถที่จะกำหนดคำช่วยเหลือพิเศษ เพื่อใช้ในการสื่อสารหรือการกำหนดภาพที่ใช้ในการแสดงผล กำหนดตำแหน่งในการแสดงผล ส่วนเรื่องข้อมูลในระบบนี้จะเก็บข้อมูลที่เกี่ยวข้องกับการควบคุมตำแหน่งของการแสดงผลเมื่อโปรแกรมทำงาน เช่น ตำแหน่งแสดงผลบนหน้าจอ

ประเภทอุปกรณ์ที่จะควบคุมค่าช่องสัญญาณที่ใช้ในการควบคุมอุปกรณ์ไฟฟ้า และสถานะของการใช้อุปกรณ์ต่าง ๆ ภายในอาคารหรือที่พักอาศัย

- ส่วนการควบคุม

ในส่วนนี้จะเป็นการควบคุมอุปกรณ์ไฟฟ้าโดยการดึงข้อมูลจากฐานข้อมูลที่ได้มีการกำหนดไว้ โดยการส่งและแปลงข้อมูลไปยังแบบบ้านโมเดลจำลองซึ่งมีแผ่นวงจรไฟฟ้าหรืออุปกรณ์ไมโครคอนโทรลเลอร์ที่ต่อออกจาก Communication Port เพื่อทำการเปิดหรือปิดไฟตามที่ต้องการ ซึ่งในระบบนี้จะเลือกใช้ Serial Port RS-232 และใช้สัญญาณ 4 เส้นดังนี้ RTS, DTR, DCD และ GND ผ่าน I^2C บัส โดยข้อมูลที่ส่งจะเป็นข้อมูลที่ใช้ในการบอกให้เปิดปิดไฟ เช่น กำหนดตั้งเวลาเปิดไฟในห้องรับแขก เวลา 8.00 น. หรือ ทำการปิดไฟในห้องครัวหลังจากเวลา 20.00 น. เป็นต้น การควบคุมระบบจะเป็นการควบคุมแบบกลุ่มและแบบย่อยในแต่ละตำแหน่งอุปกรณ์ โดยในแบบกลุ่มเราจะกำหนดการควบคุมในลักษณะที่เรียกว่าเป็น Circuit โดยการควบคุมแบบย่อยจะเป็นการควบคุมผ่านทางช่องสัญญาณ ซึ่งการแบ่งการควบคุมดังกล่าวทำให้ระบบนี้สามารถควบคุมได้ทีละกลุ่มของอุปกรณ์ โดยการสั่งงานเพียงหนึ่งครั้ง หรืออุปกรณ์แต่ละชิ้นตามต้องการได้ โดยจะทำการส่งรหัสข้อมูลที่เป็นตัวเลขไปให้โปรแกรมย่อยเพื่อใช้ในการควบคุมอุปกรณ์

5.5 ความต้องการของระบบ

ความต้องการของระบบแยกออกได้เป็น 2 ส่วนคือ ส่วน Hardware และส่วนที่ Software ดังนี้

5.5.1 Hardware

- PC : มีคุณสมบัติไม่ต่ำกว่า CPU Pentium Celeron 266 MHz RAM 64 MB HardDisk 4.3 GB
- Client : มีคุณสมบัติไม่ต่ำกว่า CPU Pentium 166 MHz Ram 32 MB
- Controller : บอร์ดเชื่อมต่อพอร์ตอนุกรมและควบคุมการเปิดปิดอุปกรณ์ไฟฟ้า
- UPS

5.5.2 Software

- Microsoft Windows 98 เป็นระบบปฏิบัติการ
- Microsoft Access 97 ใช้ในการจัดเก็บรายละเอียดข้อมูลและจัดการควบคุมความถูกต้องของข้อมูล
- Microsoft Visual Basic 6.0 เป็นโปรแกรม Front-End ที่ใช้ในการออกแบบจัดการข้อมูลในระบบ

บทที่ 6
การพัฒนาระบบงาน

ขั้นตอนต่อไปหลังจากที่เสร็จการออกแบบระบบงานแล้ว ก็คือนำสิ่งที่ได้ออกแบบไว้มาพัฒนาเป็นระบบงานจริง

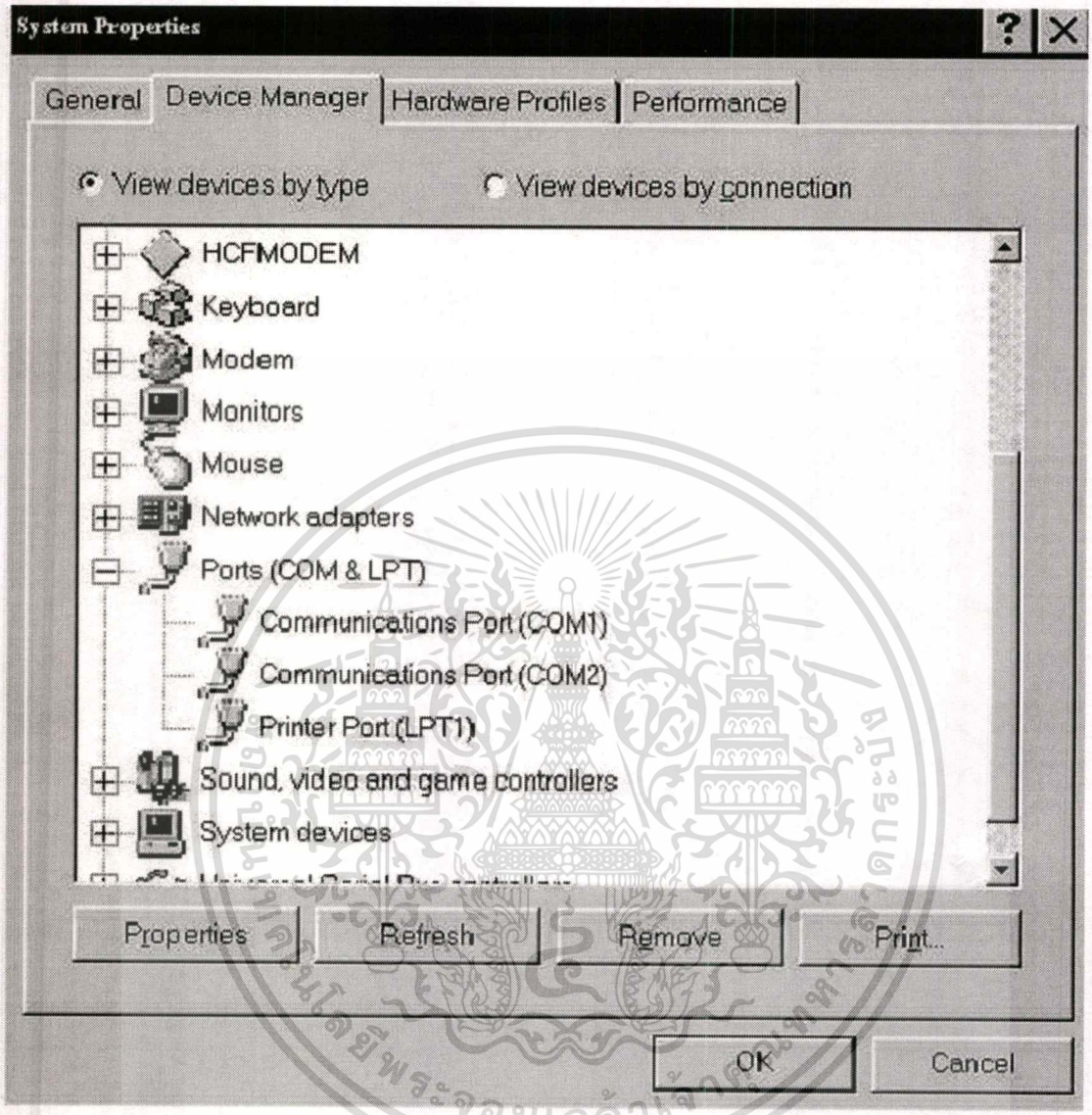
เนื่องจากการควบคุมตัวคอนโทรลเลอร์ ในที่นี้จะใช้การติดต่อผ่านเว็บ โดยคอนโทรลเลอร์ต่อตรงเข้ากับเครื่องคอมพิวเตอร์ ใช้แอพพลิเคชันที่พัฒนาโดย Visual Basic 6.0 เป็นตัวควบคุมคอนโทรลเลอร์ ส่วนการเรียกใช้โปรแกรม แอพพลิเคชันนี้ สามารถเรียกใช้โดยตรงจากเครื่อง PC

6.1 การติดตั้งและใช้งานโปรแกรม

เครื่องที่นำมาใช้งานจะต้องมีพอร์ตอนุกรมว่าง 1 พอร์ตสำหรับติดต่อกับคอนโทรลเลอร์ ระบบปฏิบัติการที่ใช้เป็น Windows 95/98 และต้องติดตั้ง Microsoft Visual Basic 6.0 ไว้ด้วย ส่วน Windows Application ที่ใช้ควบคุมอุปกรณ์ไฟฟ้าจะติดตั้งไว้ที่ไดเร็คตอรีใดก็ได้ตามต้องการ

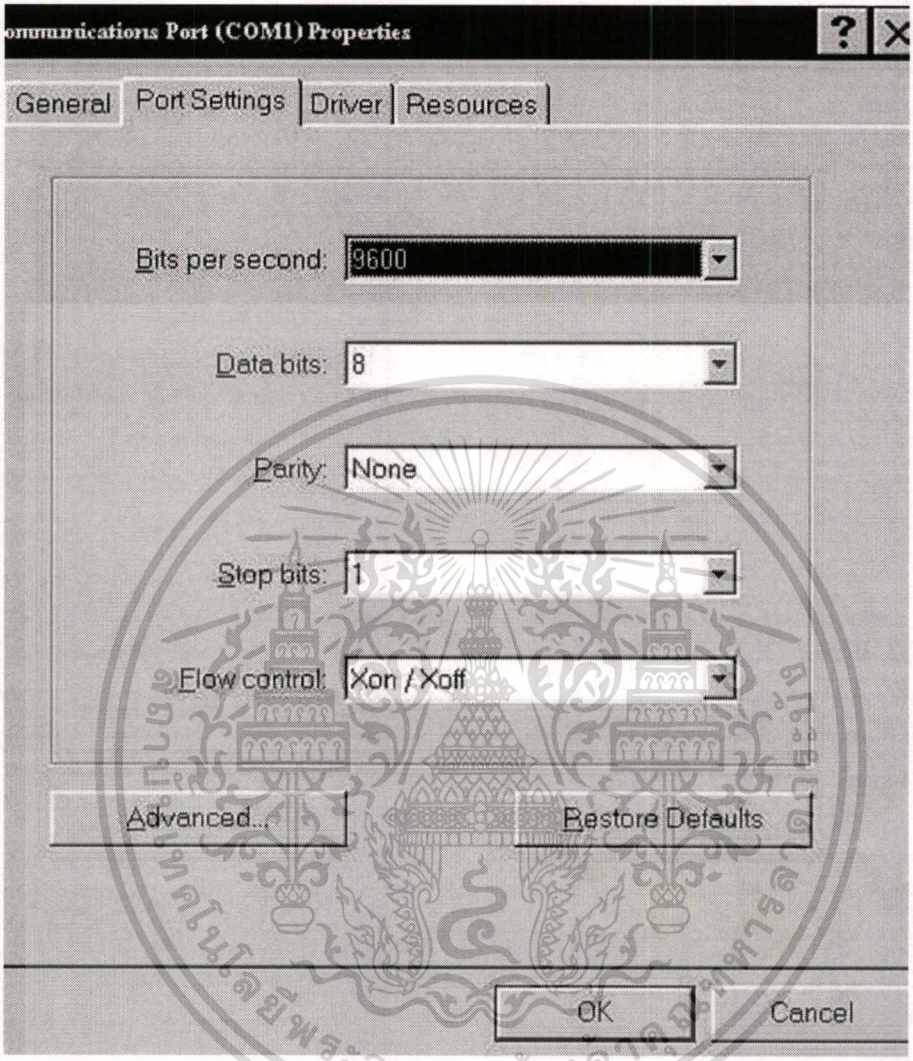
6.2 การกำหนดค่าให้กับพอร์ตอนุกรม

ในการทำงานของระบบนี้จะอาศัยพอร์ตอนุกรมเป็นเส้นทางการส่งคำสั่งจากระบบไปควบคุมคอนโทรลเลอร์ เพื่อควบคุมสวิทช์ปิดเปิดไฟต่างๆ ซึ่งในการกำหนดค่าให้กับพอร์ตอนุกรมนั้น จะอยู่ในส่วนของ Control Panel ใน System ดังรูปที่ 6.1



รูปที่ 6.1 แสดงการตรวจสอบพอร์ตอนุกรมที่จะใช้งาน

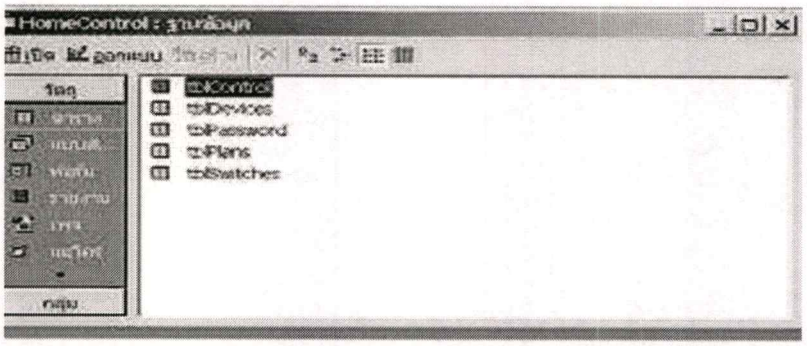
หลังจากที่ได้ตรวจสอบแล้วว่ามีพอร์ตอนุกรมใดที่สามารถใช้งานได้ ก็ให้เลือกพอร์ตอนุกรมนั้นโดยดับเบิ้ลคลิกที่พอร์ตนั้นเพื่อเข้าไปกำหนดค่าต่างๆ ที่จำเป็น ในที่นี้เลือกใช้ พอร์ตคอม COM1 ดังรูปที่ 6.2



รูปที่ 6.2 แสดงการกำหนดค่าของพอร์ตอนุกรม COM1

6.3 การสร้างฐานข้อมูลของระบบ

การสร้างฐานข้อมูลของระบบควบคุมนี้เพื่อเก็บข้อมูลการใช้งาน สถานะของการเปิดเปิดสวิทช์ รวมถึงข้อมูลของอุปกรณ์ดวงไฟแต่ละตัว ข้อมูลรูป แผนภาพของบ้านที่จะควบคุม โดยในระบบนี้จะใช้ Microsoft Access ช่วยในการจัดการฐานข้อมูล ซึ่งในฐานข้อมูลนี้จะมี 5 ตารางดังรูปที่ 6.3

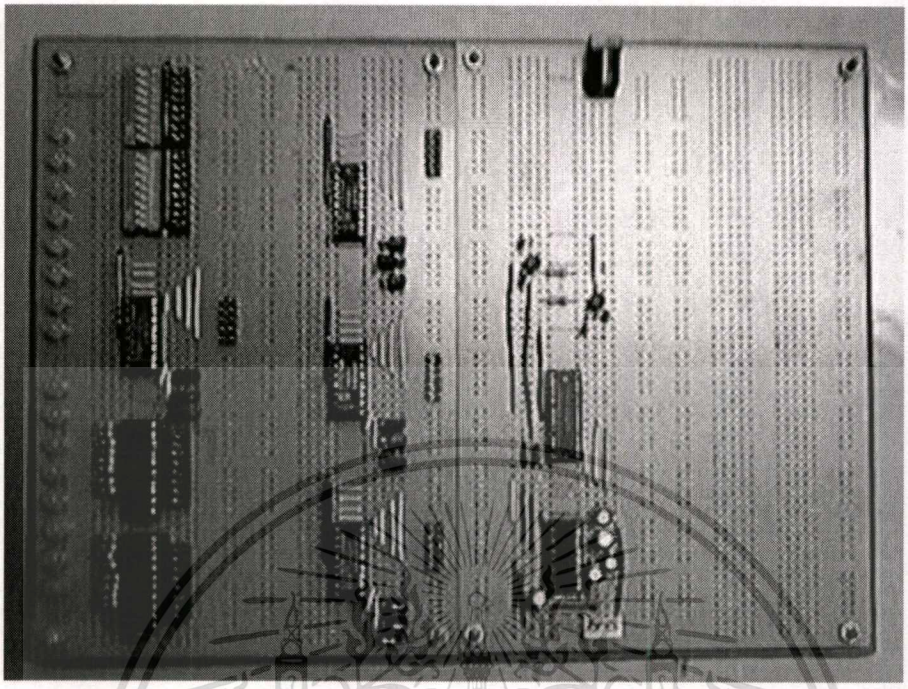


รูปที่ 6.3 แสดงตารางในฐานข้อมูลที่สร้างด้วย Microsoft Access

6.4 การติดตั้งอุปกรณ์ด้านฮาร์ดแวร์

การควบคุมคอนโทรลเลอร์ให้ปิดเปิดสวิทช์นั้น โปรแกรมที่ควบคุมคอนโทรลเลอร์จะทำการสร้างรหัสข้อมูลเป็นข้อมูลระดับบิต 0 และ 1 หรือปิดและ เปิด ตามลำดับ ไปให้คอนโทรลเลอร์ ซึ่งเป็นข้อมูลที่คอนโทรลเลอร์เข้าใจ ซึ่งคอนโทรลเลอร์ในที่นี้จะเกี่ยวข้องกับวงจรอิเล็คทรอนิคส์ ซึ่งอาศัยหลักการการส่งสัญญาณสองเส้นคือ สัญญาณข้อมูล และสัญญาณนาฬิกา ผ่านพอร์ตอนุกรม RS-232 โดยการควบคุมจะอ้างถึง ตำแหน่งและรหัสควบคุมแก่ไอซีแต่ละตัวที่ควบคุมสวิทช์ หรือ ไอซีที่อ่านสัญญาณจาก Sensor

ข้อมูลจำเพาะของวงจรนี้สามารถควบคุมสวิทช์ได้ 16 ตัวแสดงสถานะของสวิทช์ด้วย LED และยังมีวงจรสำหรับอ่านข้อมูลจากตัว Sensor อีก 16 ช่อง เอาไว้แสดงสถานะของ อุปกรณ์ปลายทาง ซึ่งถ้าอุปกรณ์มีการเสียหายเกิดขึ้นไม่สามารถทำงานได้ หลังจากที่ได้สั่งให้ปิดเปิดแล้ว เราจะทราบได้จากสถานะของอินพุต ที่อ่านได้ เปรียบเทียบกับ สถานะของสวิทช์ แสดงอยู่ ถ้าไม่ตรงกันแสดงว่า อุปกรณ์นั้นๆ มีปัญหาอยู่



รูปที่ 6.4 แสดงบอร์ดของคอนโทรลเลอร์ที่ใช้ในการทดลอง

6.5 การสร้างส่วนติดต่อกับผู้ใช้(User Interface)

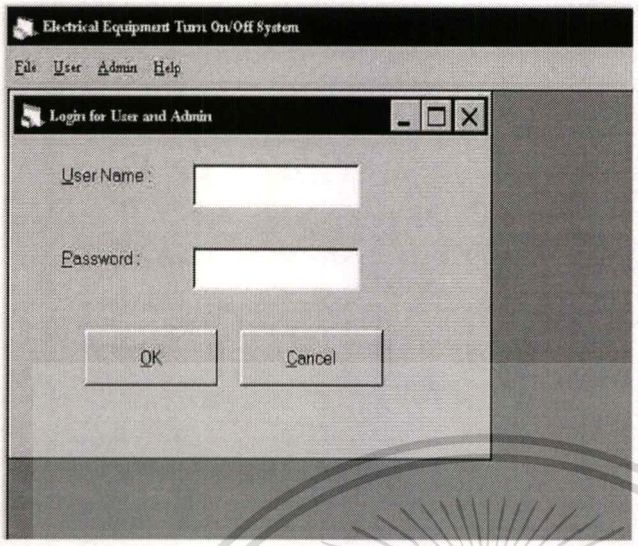
เป็นการสร้างฟอร์มต่างๆ ให้ผู้ใช้สามารถใช้งานได้สะดวก ในที่นี้เลือกใช้ Microsoft Visual Basic 6.0 ซึ่งง่ายในการทำการศึกษา อีกทั้งความสามารถในการใช้งานของ Microsoft Visual Basic 6.0 นี้ก็มีมากมาย ไม่ว่าจะเป็นงานทางด้านควบคุม อุปกรณ์ฮาร์ดแวร์ งานด้านฐานข้อมูล การใช้งานก็ง่าย

ระดับการใช้งานแบ่งเป็น 2 ระดับ คือ

- ผู้ใช้ที่เข้าไปกำหนดเวลาปิดเปิดสวิทช์ไฟได้และดูสถานะได้
- ผู้ใช้ที่สามารถเข้าไปแก้ไขการติดตั้งอุปกรณ์รวมทั้งกำหนดผู้ใช้งานได้ทั้งหมด

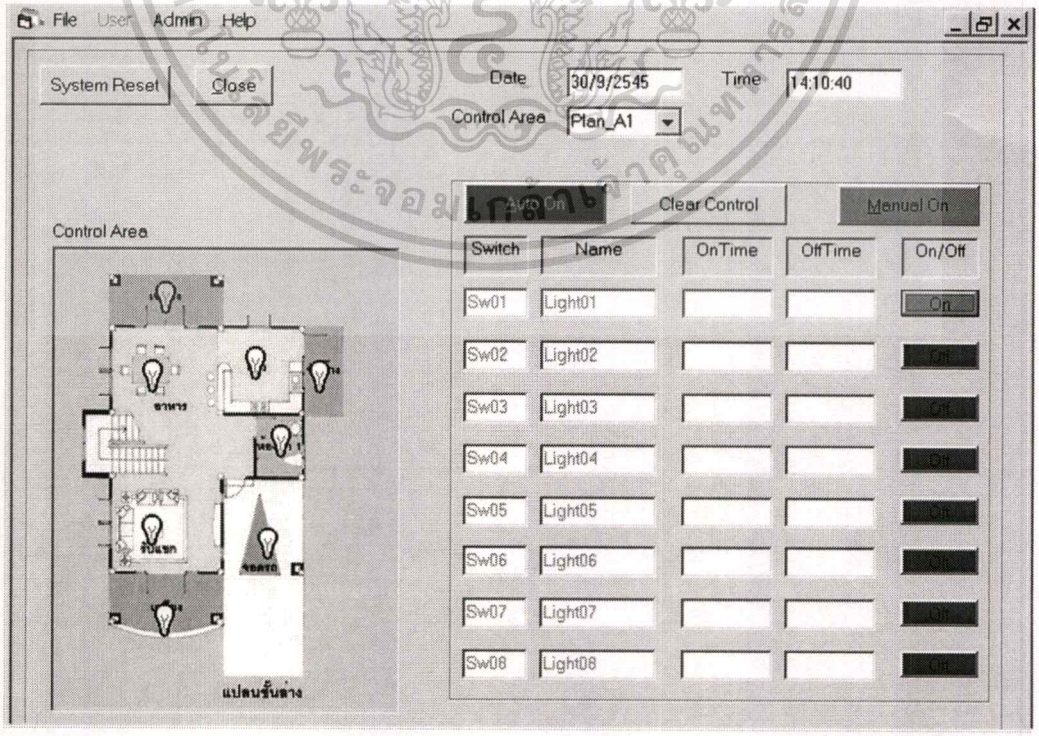
6.5.1 การ Login เข้าสู่ระบบผ่าน แอพพลิเคชันของ VB

ส่วนนี้สำหรับผู้ใช้งานที่จะเข้าไปทำการกำหนดเวลาในการปิดเปิดสวิทช์ และ การแก้ไขเปลี่ยนแปลงรายละเอียดของอุปกรณ์ เพื่อเก็บลงฐานข้อมูล และรวมทั้งการลบเพิ่มผู้ใช้งาน ซึ่งจะใช้หน้าจอ Login เดียวกัน



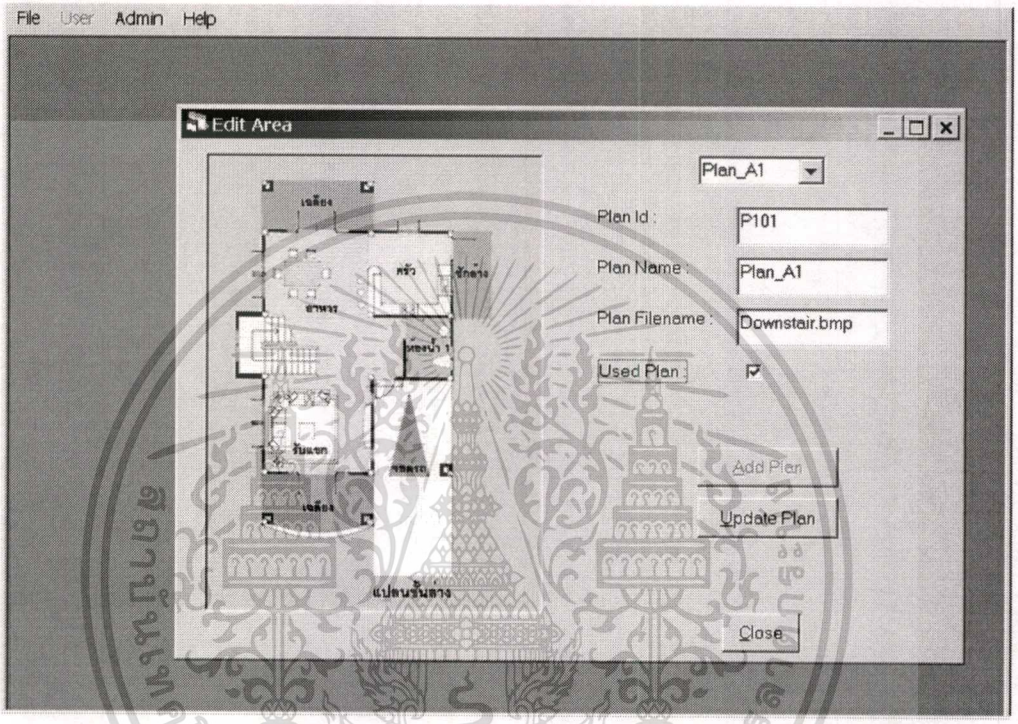
รูปที่ 6.5 แสดงหน้าจอการ Login เข้าสู่ระบบผ่าน แอพพลิเคชันของ VB

6.5.2 หน้าจอหลักของโปรแกรมควบคุมการปิดเปิดอุปกรณ์ไฟฟ้า โดยมีส่วนแสดงแผนผังของบ้าน และส่วนของการควบคุม ซึ่งสามารถควบคุมการปิดเปิดได้ พร้อมทั้งมีส่วนแสดงสถานะของอุปกรณ์ไฟฟ้าด้วย



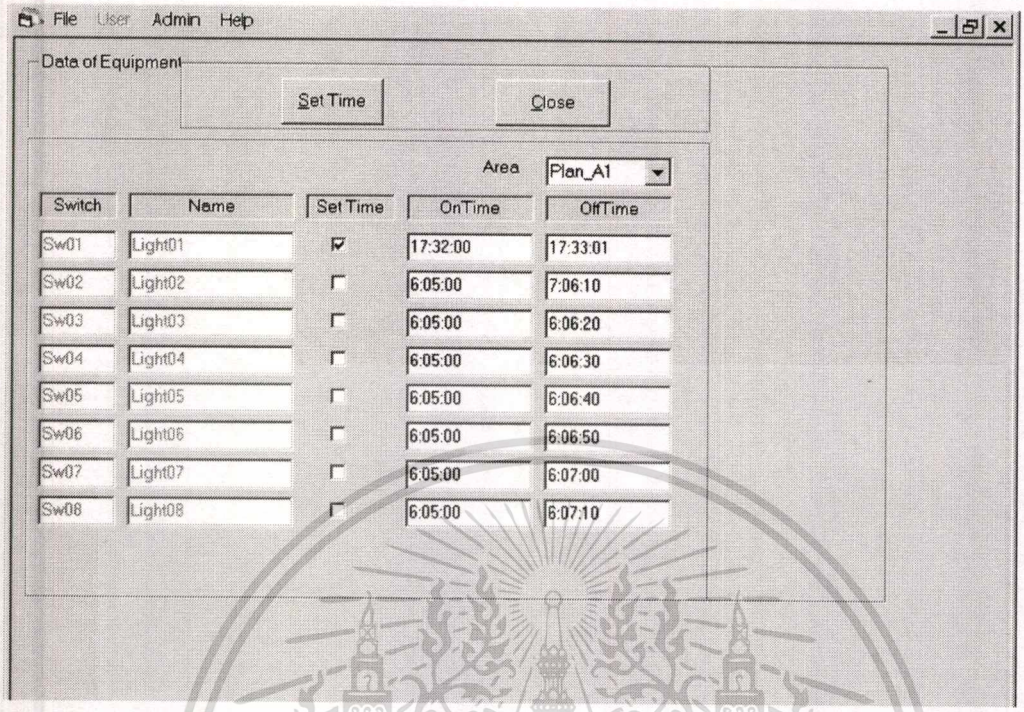
รูปที่ 6.6 แสดงหน้าจอหลักของโปรแกรมควบคุมการเปิดปิดอุปกรณ์ไฟฟ้า

6.5.3 หน้าจอสำหรับการเพิ่มลบ แก้ไข ตำแหน่งของอุปกรณ์ไฟฟ้า
โดยที่ผู้ใช้ต้องเป็น Administrator เท่านั้นจึงจะสามารถเข้าใช้งานหน้าจอนี้ได้



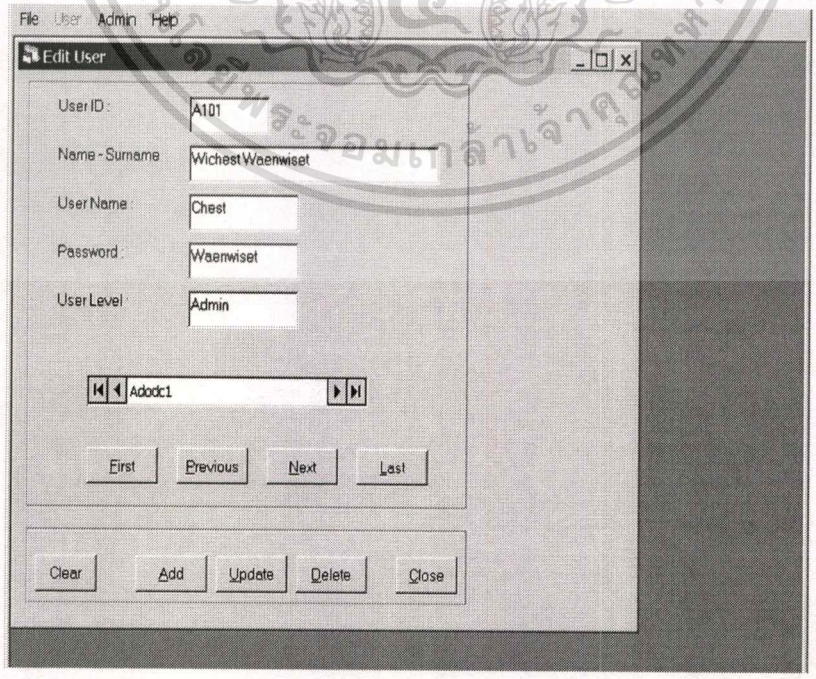
รูปที่ 6.7 แสดงหน้าจอการเพิ่ม ลบ และแก้ไขตำแหน่งของอุปกรณ์ไฟฟ้า

6.5.4 หน้าจอสำหรับการเพิ่มลบ แก้ไข ข้อมูลของอุปกรณ์ไฟฟ้า
โดยที่ผู้ใช้ต้องเป็น Administrator จะสามารถใช้ได้ทุกๆ ส่วน แต่ผู้ใช้ที่เป็น User จะสามารถใช้ได้เฉพาะส่วนของการตั้งเวลาเท่านั้น



รูปที่ 6.8 แสดงหน้าจอการตั้งเวลาการทำงานของอุปกรณ์ไฟฟ้า

6.5.5 หน้าจอสำหรับการเปลี่ยนแปลงแก้ไขข้อมูลผู้ใช้
โดยที่ผู้ใช้ต้องเป็น Administrator เท่านั้น



รูปที่ 6.9 แสดงหน้าจอการเปลี่ยนแปลงแก้ไขข้อมูลผู้ใช้

บทที่ 7
สรุปผลการดำเนินงาน

7.1 ผลการพัฒนาระบบ

- ส่วนจัดการในด้านแบบแปลน

เพื่อให้เห็นภาพเสมือนจริงของการควบคุมอุปกรณ์ไฟฟ้าจึงได้มีการนำแผนภาพของอาคารหรือที่พักอาศัยนั้นเข้ามาในระบบ ขั้นตอนนี้จึงเป็นส่วนสำคัญเพื่อที่จะให้ผู้ใช้ได้เห็นอุปกรณ์ที่จะควบคุมว่าอยู่ในส่วนที่ต้องการหรือไม่

- ส่วนการจัดการอุปกรณ์ทางด้านฮาร์ดแวร์

ฮาร์ดแวร์ในการทดลองโครงงานนี้เป็นอุปกรณ์อ่านและเขียนสัญญาณได้ 2 ทิศทางขึ้นอยู่กับการสั่งงานว่าต้องการให้ทำงานเป็นอุปกรณ์อ่านหรือเขียน บอร์ดที่ใช้ในการทดลองนี้มี 32 ช่องสัญญาณ โดยในการทดลองใช้ 16 ช่องสัญญาณ เป็นอุปกรณ์เขียนหรือควบคุมการปิดเปิดสวิทช์ และ อีก 16 ช่องสัญญาณเป็นอุปกรณ์อ่านหรือรับสัญญาณอินพุต จาก Sensor การนำไปใช้งานจริงสามารถทำได้โดยใช้ ช่องสัญญาณที่ต่อออกมาเป็นคอนเน็คเตอร์บนบอร์ด แต่สัญญาณที่ได้ไม่แรงพอที่จะไปขับ รีเลย์สวิทช์ จะต้องมีการนำไปขยายสัญญาณเสียก่อน อาจจะใช้ ทรานซิสเตอร์เป็นต้น

- ส่วนการนำเข้าและจัดการข้อมูล

ส่วนนี้เป็นส่วนสำคัญที่จะใช้เก็บข้อมูลที่จำเป็นต่อการทำงานของระบบ ข้อมูลที่จะต้องนำเข้าไปเก็บในฐานข้อมูลและดึงเอาออกมาใช้ก็มี ข้อมูลแบบแปลนของอาคาร ข้อมูลของอุปกรณ์ข้อมูล ของผู้ใช้งาน เป็นต้น

- ส่วนการควบคุมระบบ

เป็นส่วนหลักของระบบนี้ ซึ่งถ้าผู้ใช้งานกำหนดข้อมูลต่างๆ ถูกต้อง รวมถึงการวางอุปกรณ์ไฟฟ้าต่างๆ ให้ตรงกับแผนผังที่ผู้ใช้งานได้ออกแบบไว้ ก็จะสามารถควบคุมได้จากเครื่องคอมพิวเตอร์ ที่ต่ออยู่กับอุปกรณ์ฮาร์ดแวร์ได้

7.2 ปัญหาในด้านการแสดงผล

เนื่องจากวงจรที่ใช้เป็นวงจรควบคุมการปิดเปิดสวิทช์ และการรับค่าของ Sensor จากอุปกรณ์ไฟฟ้า ดังนั้นการทำงานจึงไม่รู้สึกเสมือนจริง เท่าใดนัก ถ้าต้องการให้เห็นจริงจะต้องมีการ

ต่อสัญญาณไปควบคุมกับอุปกรณ์จริงและต่อ Sensor จากอุปกรณ์จริงกลับเข้ามาที่ คอนโทรลเลอร์ การแสดงผลของสถานะของสวิทช์จะช้ากว่าการแสดงผลของ LED ที่แทนสวิทช์แต่ละจุด เนื่องจาก LED ที่ส่งมาจากคอนโทรลเลอร์แล้วทำงานทันที แต่สถานนะของสวิทช์จะต้องไปนำข้อมูลจากฐานข้อมูลที่เก็บสถานะของการปิดเปิดแต่ละครั้งไว้ มาแสดงผลจึงทำให้แสดงผลได้ช้ากว่า LED ถ้าต้องการให้แสดงผลได้พร้อมกัน ก็ไม่ต้องไปใช้ข้อมูลสถานะจากฐานข้อมูลให้ใช้ข้อมูลชุดเดียวกับที่สั่งให้ LED ทำงานแทน แต่ผลเสียที่เกิดขึ้นก็คือเมื่อมีการปิดเครื่องเซิร์ฟเวอร์ หรือเครื่องคอมพิวเตอร์เกิดแฮงก์ขึ้นมาแล้วเปิดขึ้นมาใหม่ สถานะของสวิทช์ที่ได้จะไม่ตรงกับสถานะจริงของสวิทช์ที่คอนโทรลเลอร์สั่งงานอยู่

บรรณานุกรม

กฤษฎา ใจเย็น และ ชัยวัฒน์ ลิ้มพรจิตรวิไล. 2541. **การเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกผ่านพอร์ทอนุกรม.** กรุงเทพฯ : อินโนเวตีฟ เอ็กเพอริเมนต์.

ธาริน สิทธิธรรมชารี. **คู่มือการเขียนโปรแกรม Microsoft Visual Basic 6.0 ฉบับเพื่อใช้งานจริง.** Success Media Co.,Ltd.

ธีรวัฒณ์ ประกอบผล. 2545. **การพัฒนาไมโครคอนโทรลเลอร์ด้วยภาษาซี.** กรุงเทพฯ : สมาคมส่งเสริมเทคโนโลยี(ไทย-ญี่ปุ่น)

สุชาย ธนวเสถียร และ นรินทร์ อัครพิเชษฐ. 2541. **Fundamental of Visual Basic Client-Server Programming.** กรุงเทพฯ: ซีเอ็ดยูเคชัน.

ณัฐพล สถิรเจริญกุล. 2543. "ระบบควบคุมอุปกรณ์ไฟฟ้าผ่านเครือข่ายอินเตอร์เน็ต". วิชาโครงการพัฒนาระบบงาน. คณะเทคโนโลยีสารสนเทศ. สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

สมิทธิ สุนทรนิทัศน์. 2544. "ระบบตั้งเวลาควบคุมการใช้งานเครื่องใช้ไฟฟ้า". วิชาโครงการพัฒนาระบบงาน. คณะเทคโนโลยีสารสนเทศ. สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.

ศักดิ์ชัย กิตติจิรายุ. 2544. "ระบบรักษาความปลอดภัยในบ้านผ่านอินเตอร์เน็ต". วิชาโครงการพัฒนาระบบงาน. คณะเทคโนโลยีสารสนเทศ. สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.